D5.6 – MARIO Reading and Listening Component

Project Acronym:	**MARIO**
Project Title:	**Managing active and healthy aging with use of caring service robots**
Project Number:	**643808**
Call:	**H2020-PHC-2014-single-stage**
Topic:	**PHC-19-2014**
Type of Action:	**RIA**

D5.6

Work Package:	WP5
Due Date:	M30
Submission Date:	31/07/2017
Start Date of Project:	01/02/2015
Duration of Project:	36 months
Organisation Responsible of Deliverable:	CNR
Version:	3.0
Status:	final
Author name(s):	Valentina Presutti, Luigi Asprino, Andrea Giovanni Nuzzolese, Aldo Gangemi, Domenico Pisanelli, Alessandro Russo (CNR) Massimiliano Raciti (R2M) André Freitas (PASSAU) Lazaros Penteridis, Alexandros Gkiokas (ORTELIO)
Reviewer(s):	Mark Burgin (RUR) Diego Reforgiato Recupero (R2M)
Nature:	☐ R – Report ☐ P – Prototype ☐ D – Demonstrator ☒ O – Other
Dissemination level:	☒ P – Public ☐ CO – Confidential, only for members of the consortium (including the Commission) ☐ RE – Restricted to a group specified by the consortium (including the Commission Services)
Project co-funded by the European Commission within the Horizon 2020 Programme (2014-2020)	

Revision history			
Version	**Date**	**Modified by**	**Comments**
0.1	16/06/2017	Valentina Presutti (CNR)	First draft starting from D5.5.
0.2	21/06/2017	Alessandro Russo (CNR)	Revision and changes on the document structure; inclusion of conversational strategy guidelines.
0.3	23/06/2017	Alessandro Russo (CNR)	Additional content on NLU services; added examples.
0.3.1	27/06/2017	Luigi Asprino, Andrea Giovanni Nuzzolese, Alessandro Russo (CNR)	Content outline for use case scenarios and frame semantics.
0.4	03/07/2017	Luigi Asprino (CNR)	Integrated contribution on CGA app; revision of content on research activities.
0.5	05/07/2017	Andrea Giovanni Nuzzolese (CNR)	Content review and additional contribution on frame semantics and FRED.
0.5.1	06/07/2017	André Freitas (PASSAU)	Revision of contribution on Correction component.
0.6	07/07/2017	Massimiliano Raciti (R2M) Alessandro Russo (CNR)	Revision of contribution on Speech to Text component. Integrated contribution on My Memories app.
1.0	12/07/2017	Alessandro Russo (CNR)	Revision of document outline and overview; draft version for internal review.
1.0	19/07/2017	Adam Santorelli (NUIG)	Comments and feedback on v1.0.
1.0	21/07/2017	Mark Burgin (RUR), Diego Reforgiato Recupero (R2M)	Reviewer's feedback on v1.0.
1.1	24/07/2017	Alessandro Russo (CNR)	Revision and minor updates on the Vocal Interface section according to reviewer's feedback.

continued . . .

continued . . .

Version	Date	Modified by	Comments
1.2	25/07/2017	Luigi Asprino (CNR)	Revision and readability improvements of the slides on the Paraphrase Corpus and Research Activities, according to reviewer's feedback.
1.3	26/07/2017	Alessandro Russo (CNR)	List of acronyms and abbreviation, additional content to improve readability of section on My Memories app, and additional references according to reviewer's feedback.
2.0	27/07/2017	Alessandro Russo, Luigi Asprino, Andrea Giovanni Nuzzolese (CNR)	Revision of the overall document wrt reviewers' comments; added abbreviations list; minor formatting edits; version for coordinator.
2.0	28/07/2017	Aisling Dolan (NUIG)	Comments and feedback on v2.0.
2.1	28/07/2017	Alessandro Russo (CNR)	Revision with additional edits; final version.
2.2	07/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Document refactoring to address comments in final review report.
2.3	08/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Added definition of approach and methodology.
2.4	12/06/2018	Alessandro Russo (CNR)	Added architectural overview.
2.5	13/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Review and extension of content on NLU services.
2.5.1	14/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Added examples on NLU services.
2.6	15/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Review of content on speech to text; review of content on research activities.

continued . . .

continued . . .

Version	Date	Modified by	Comments
2.6	18/06/2018	Luigi Asprino, Valentina Presutti, Alessandro Russo (CNR)	Added draft on discussion and lessons learned.
2.7	18/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Review and improvements of service descriptions.
3.0	19/06/2018	Luigi Asprino, Alessandro Russo (CNR)	Overall document revision and improvements; update of Executive Summary and introductory sections; version for submission.

Executive Summary

Language understanding is a key requirement for social robots like MARIO aiming at supporting speech-based human-robot interactions. To enable this capability, user's utterances have to be first converted into a textual representation using an automatic speech recogniser providing speech to text capabilities. The ability to process and understand user utterances is then responsibility of a language understanding component, which supports the ability of client applications to manage a speech-based interaction with the user.

Deliverable 5.6 describes the software components that were designed and developed in the context of Task 5.2 (WP5) and that contribute to the overall MARIO system architecture and software framework. This version represents the final, consolidated version of Deliverable 5.2 and its corresponding intermediate version provided in Deliverable 5.5.

Specifically, the components presented here constitute MARIO's Speech to Text and Natural Language Understanding subsystem. The role and capabilities of these components are also illustrated through concrete examples and representative use case scenarios based on applications developed in the project. The deliverable also reports on research activities carried out in the context of Task 5.2.

In addition, the document includes a discussion on the experience gained and lessons learned from the use of semantic technologies in the context of the project.

Table of Contents

List of Figures

List of Tables

List of Listings

List of Acronyms and Abbreviations

API	Application Programming Interface
ASR	Automatic Speech Recognition
CCG	Combinatory Categorial Grammar
CGA	Comprehensive Geriatric Assessment
CSK	Common Sense Knowledge
DRSs	Discourse Representation Structures
DRT	Discourse Representation Theory
GUI	Graphical User Interface
HRI	Human-Robot Interaction
JSON	JavaScript Object Notation
KB	Knowledge Base
MON	MARIO Ontology Network
NEL	Named Entity Linking
NER	Named Entity Recognition
NL	Natural Language
NLP	Natural Language Processing
NLU	Natural Language Understanding
OWL	Web Ontology Language
PWD	Person/People with Dementia
Q&A	Question & Answer
RDF	Resource Description Framework
REST	Representational State Transfer
SDK	Software Development Kit
SPARQL	SPARQL Protocol and RDF Query Language
SRL	Semantic Role Labelling
S2T	Speech to Text
WP	Work Package
WFD	Word Frame Disambiguation
WSD	Word Sense Disambiguation

1 Introduction

Natural language is the primary means through which people with dementia (PWD) can communicate and interact with MARIO robots.

This deliverable presents the approaches, methodologies and software components that enable MARIO to convert spoken natural language input into a textual representation (i.e., Speech to Text), which is then translated into a formal representation so as to enable Natural Language Understanding (NLU) capabilities.

MARIO's NLU subsystem has been designed as a modular component consisting of an extensible set of reusable and composable modules, each implementing and providing a language processing and understanding capability made available as a service. This document provides a description of these services, together with architectural choices made to integrate them into the overall MARIO framework. Concrete examples are provided as well, to explain how MARIO's applications have used language processing capabilities to support their logic and user interaction strategies.

The document also reports on research activities that were carried out in the context of Task 5.2 and includes a discussion on lessons learned concerning the use of the semantic technologies.

1.1 Work Package 5 Objectives

WP5 aims at developing the framework and tools that allow MARIO robots to interact with humans and understand their needs expressed through spoken natural language. As fundamental building blocks, understanding capabilities exploit machine reading/listening components and RDF/OWL ontologies to first produce and then process a formal encoding of the textual representation of natural language.

As such, the main objectives of WP5 are:

- to design and develop the MARIO Ontology Network (MON) and Knowledge Base;
- to provide MARIO with the ability of transforming natural language into a formal representation, to enable reading and listening capabilities;
- to provide MARIO with the capability of recognising, storing and reusing sentiment information, on the basis of semantic sentiment analysis techniques.

1.2 Purpose and Target Group of the Deliverable

This deliverable aims at describing the software components designed and developed in Task 5.2. These components provide the robot with natural language understanding

capabilities, relying on state of the art speech recognition technology and natural language processing techniques, used in order to automatically translate spoken natural language to a textual and formal representation, respectively.

Due to its technical nature, this deliverable is mainly targeting researches, practitioners and developers interested in NLP techniques and in the role of NLP and machine reading tools for service companion robots. In addition to the technical aspects, the adopted methodological approach and main design choices are discussed as well, together with concrete use cases, with the aim of providing health experts with an understanding on how these techniques can support the interaction between PWD and companion robots.

This deliverable directly relies on the results of Task 5.1 (reported in Deliverable 5.1) as far as the background knowledge and knowledge models that it uses are concerned, and it is strongly related to the activities carried out in Task 5.3 concerning Sentiment Analysis (as reported in Deliverable 5.7 [2]).

The overall WP5 receives as input the user and functional requirements and the system architecture from WP1, while WP2 provides the Kompai robot and platform where the software components are deployed. These components are exploited by the applications and modules developed in WPs 3, 4 and 6 (as far as natural language-based interaction is concerned), in line with the integration procedures defined in WP7. Validation activities in WP8 provide feedback to the iterative design and development process of the software components, and contribute to their assessment, evolution and refinement.

1.3 Document Outline

The rest of the document is structured as follows. Section 2 gives an overview of: (*i*) the methodology for identifying, organising and carrying out the activities of the Task 5.2; (*ii*) the architectural reference, the core components and services related to the natural language understanding. Section 3 describes the component providing the automatic speech recognition capabilities. The MARIO's Natural Language Understanding Subsystem is presented in Section 4. Section 5 presents two applications, namely the CGA app and the My Memories app, that are representative examples of MARIO's applications that rely on NLU services as part of their logic and interaction/dialogue management strategy. The activities aimed at advancing the state-of-the-art in research fields related to Task 5.2 are summarized in Section 6. Finally, Section 7 discusses open problems and lessons learned.

1.4 About MARIO

MARIO addresses the difficult challenges of loneliness, isolation and dementia in older persons through innovative and multi-faceted inventions delivered by service robots. The effects of these conditions are severe and life-limiting. They burden individuals and so-

cietal support systems. Human intervention is costly but the severity can be prevented and/or mitigated by simple changes in self-perception and brain stimulation mediated by robots.

From this unique combination, clear advances are made in the use of semantic data analytics, personal interaction, and unique applications tailored to better connect older persons to their care providers, community, own social circle and also to their personal interests. Each objective is developed with a focus on loneliness, isolation and dementia. The impact centres on deep progress toward EU scientific and market leadership in service robots and a user driven solution for this major societal challenge. The competitive advantage is the ability to treat tough challenges appropriately. In addition, a clear path has been developed on how to bring MARIO solutions to the end users through market deployment.

2 Approach, Methodology and Architectural Overview

Speech is largely considered as the most powerful and effective communication modality for an assistive social robot to interact with its users, and spoken dialogue is generally regarded as the most natural way for human-robot interaction (HRI) [3]. The ability to communicate using natural language is thus a fundamental requirement for a social robot that aims at providing support for people with dementia (PWD). Recent technological developments and research results are contributing to solving the challenges that characterise the design and implementation of language understanding capabilities for human-robot interaction.

The activities carried out in Task 5.2 primarily aim at providing MARIO robots with the ability to acquire, process and understand natural language sentences (i.e., user's *utterances*) that PWD use when interacting with the robot. This section introduces the overall approach adopted throughout the entire lifespan of this task, summarises the general methodology that has been followed, and provides an overview of the core software components that were designed, developed and integrated in the architecture of the MARIO platform.

2.1 General Approach

The structuring and organisation of the core activities have been following two complementary paths, though interlinked and interleaved among each other. This approach allowed us on the one hand to concretely contribute to the development of the MARIO software framework and, on the other hand, to investigate challenging research problems and contribute to advance the state of the art. The two activity paths are summarised hereafter.

Iterative design and development of targeted approaches and solutions

The work carried out along this path has been focusing on the design, development and deployment of the software components (presented in Sections 3 and 4) implementing targeted approaches and working solutions that provide MARIO with natural language acquisition and understanding capabilities.

In line with the overall principles and methodology adopted in the project, we have been following an incremental and iterative design and development approach, inspired by Agile principles. As a consequence, the implemented approaches and solutions have been:

- designed following a requirements-driven and user-centered approach, taking into account pilot sites' needs and scenarios;
- incrementally integrated, tested and validated during trial activities with PWD;
- gradually refined and improved on the basis of trials feedback.

The iterative process of testing and revision was aimed at gradually adapting, extending

and improving the available features and capabilities on the basis of a continuous assessment process driven by trial results, to best meet the requirements of PWD and carers in each pilot site.

Research activities and solutions targeting open problems

The work carried out along this path has been focusing on research activities aimed at the identification of solutions targeting open problems in the broad field of knowledge representation and semantic language understanding. These research problems are either inspired by and abstracted from concrete project use cases, or derive from general challenges that can be specialised in the context of socially assistive robots.

Research activities, summarised in Section 6, concretely led to:

- the definition of resources, methods and techniques to advance the state of the art in the field of knowledge engineering and semantic NLP;
- the production of scientific publications and proof-of-concept implementations.

Although the level of maturity of these results and proof-of-concept implementations prevented an on-the-field deployment during pilot trials, they represent an integral part of the overall contribution of Task 5.2.

2.2 Methodology and Main Activities

As part of the iterative user-centered approach for enabling robot reading and listening capabilities, the identification of the fundamental needs and requirements plays an important role. Technical and non-technical requirements were thus derived from the reference scenarios and use cases, as initially described in Deliverable 1.1 [4] and then further refined in relation to the set of applications that constitute the 4-Connect Community Module (Deliverable 3.1 [5]), the 4-Connect My Social Network Module (Deliverable 3.3 [6]) and the Comprehensive Geriatric Assessment (CGA) module (Deliverable 4.3 [7]).

In order to support the design and implementation of MARIO's language acquisition and understanding capabilities:

- caregivers and domain experts across all pilot sites have been continually involved and provided input and feedback to the development of MARIO's understanding capabilities, in terms of dialogue and use case scenarios; specifically:
  - concrete examples of prototypical speech-based interactions between PWD and MARIO were provided in the form of dialogue scripts;
  - additional input was provided as a result of the on-the-field observation of PWD interacting with MARIO during the initial acceptance and validation trials (e.g., identifying and reporting on typical questions that PWD pose to MARIO);
- audio/video recordings (and their corresponding transcripts) were collected from the

three pilot sites, focusing on speech-based interactions between PWD and caregivers.[1]

These elements, together with the domain- and application-specific scenarios related to the suite of MARIO's applications, served as a basis for:

- evaluating state of the art speech to text solutions, with the goal of identifying a speech recognition tool to be integrated in the platform (as discussed in Section 3);
- identifying the core natural language processing and understanding capabilities to be implemented and made available to the other components and applications;
- driving the architectural decisions on the structuring and integration of NLP/NLU features in the context of the platform (as introduced in the next subsection and then presented in Section 4).

2.3 Architectural Reference, Core Components and Services

In the MARIO architectural framework, the main capabilities of the robot are provided by a set of software components or *applications*, each implementing a specific task- and goal-oriented functional or behavioural skill. Concrete examples include, among the others, the ability to perform the CGA, to play music, to reminisce with the PWD, and so on. These abilities clearly correspond to the so-called MARIO applications (My Music, My Memories, CGA, etc.), whose rationale, design and implementation are extensively discussed in Deliverables 3.1, 3.3 and 4.3 [5, 6, 7].

Each application is responsible for managing the multimodal interaction with the user (combining speech-based and touch-based interaction modalities) to provide the intended functionality, performs potentially complex processing and reasoning steps according to its internal logic, maintains and updates an internal state, and operates on the basis of user input and domain-specific knowledge. For example, the My Music application is responsible for guiding the user through the process of selecting and playing her favourite music, as a sequence of multimodal interactions where natural language input and User Interface screens are combined. Applications have thus different needs concerning both language understanding and the way they manage a dialogue-based interaction with PWD.

While each application is directly responsible for undertaking a dialogue-based interaction with the user according to the application's scope and goal, other speech-based interactions between the user and robot take place outside the scope of a specific application. This corresponds to those speech-based interaction scenarios where the user issues specific commands or requests to the robot (e.g., to trigger and activate an application), or where the user and MARIO engage in so-called "small talk" interactions, such as greetings, questions issues by the user about the robot (e.g., *"What's your name?"*, *"How are*

[1] where direct involvement of PWD was not possible, speech-based interactions were recorded by caregivers reproducing realistic settings

you?", etc.) or to get information about the current date or time, and so on. Supporting these interactions, which are part of MARIO's social skills and abilities, can be seen as part of the responsibilities of MARIO's Task Manager, as further clarified in this section.

In term of language processing and understanding, we can thus identify heterogeneous needs related to MARIO's applications and components, including:

- supporting "small talk" and basic conversational interactions;
- recognising user's intent to trigger an application or an action within an application;
- understanding and interpreting user's answers to closed and open-ended questions (as in the case, for example, of the questionnaire-based assessment managed by the CGA application);
- characterise user's reaction to open-ended prompts (as in the case, for example, of the My Memories application supporting reminiscence).

On the one hand, this heterogeneity prevents to commit to a particular language processing and understanding approach, or to adopt a system-wide dialogue representation and management strategy. This would represent a constrain for the different applications and limit the potential extensibility of the framework with new skills, abilities and applications. On the other hand, regardless of this heterogeneity, the identified needs can be abstracted and mapped to specific natural language processing and understanding tasks.

These observations and evaluations motivate the choice of conceiving and designing MARIO's NLU subsystem as a modular component consisting of an extensible set of reusable and composable modules, each implementing and providing a language processing and understanding capability made available as a service.

The goal of the NLU subsystem is thus to make available multiple NLP/NLU services that can be used by MARIO's applications and components. Specifically, the core components that enable MARIO to get user's speech, provide language processing capabilities and use the available services are shown in the architectural model of Figure 1.

The *Speech to Text component* (presented in Section 3) is responsible for converting user speech into a textual representation, while the *Natural Language Understanding Subsystem* implements and provides the NLU/NLP services, as presented in Section 4. These service are complemented by *Sentiment Analysis* capabilities, presented in Deliverable 5.7 [2] and also made available following a service-oriented approach. As shown in the diagram, there is a dependency between language understanding modules and MARIO's knowledge management framework [8], as some of the text analysis services also rely on and exploit local domain knowledge stored in the MARIO Knowledge Base.

In line with the discussion above, language processing and understanding services are accessed by MARIO's *Applications* and by the *NLU Manager*. The applications rely on the services to support their speech-based user interaction processes part of the overall application logic; concrete examples are provided in Section 5. The *NLU Manager* (con-

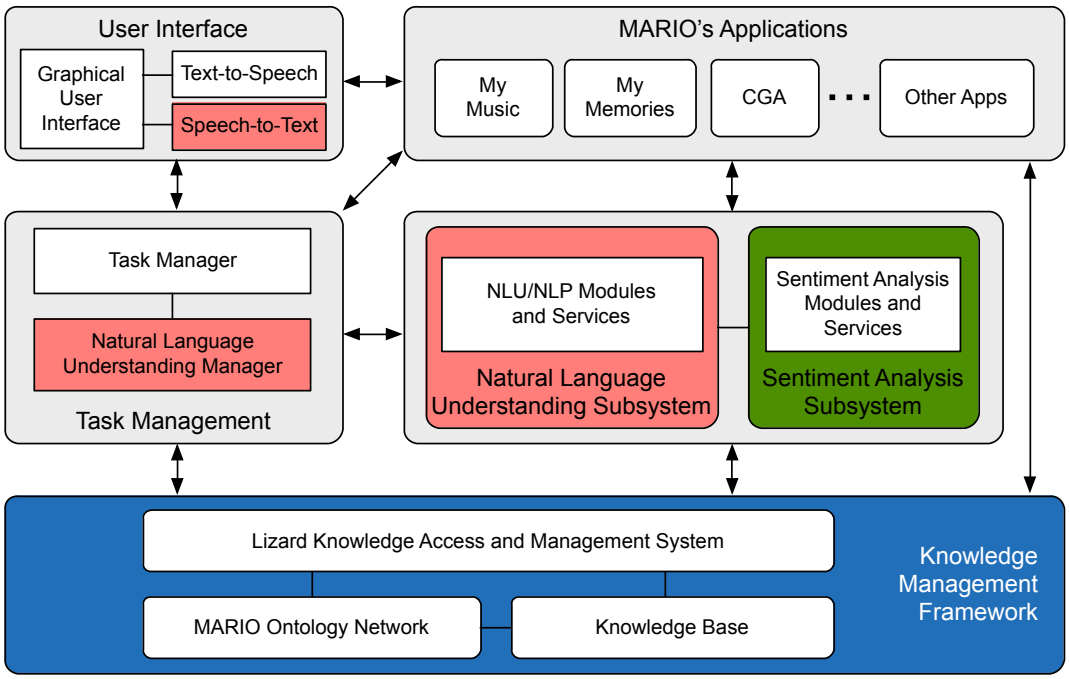


Figure 1: Overview of core components and their dependencies

ceptually shown as part of MARIO's Task Management capabilities) is the component that receives the recognised text from the Speech to Text component and is responsible for managing the speech-based interactions between the user and robot that take place outside the scope of a specific application. To this end, it relies on the language processing and understanding services to implement speech-based conversational interactions covering, e.g., small talk and recognition of user's intent to trigger an application. Additionally, the NLU Manager provides a general-purpose mechanism that allows relating User Interface control elements (e.g., buttons and selection choices shown on MARIO's screen) to specific actions to be triggered. The triggering of an action is constrained by the invocation of one or more NLP/NLU services, whose output and result allow deciding on the possibility to execute the action. Basically, given the recognised text as input, the NLU Manager check whether it can manage and process it with its internal modules or using the general approach outlined before. Otherwise, if there is an active application the input text is forwarded to it for processing.

The decision of adopting a service-oriented approach for providing language processing and understanding capabilities (as well as sentiment analysis) is further motivated by considering that a service-oriented approach:

- does not force to commit to a specific programming language for implementing the language processing and understanding services; this favours the potential extensibility of the offered services and allows reusing state of the art libraries supporting specific text processing tasks;
- does not force the client components to use a specific programming language for

accessing the services; this is important in the MARIO framework, where applications and components were developed using multiple programming languages;

- allows further integration of text processing capabilities already available as a service; this is the case of FRED [9], whose semantic machine reading capabilities have been integrated in the platform by exploiting its service-oriented API.

3 Vocal Input Interface and Speech to Text Component

A speech recogniser or Speech to Text (S2T) engine is a software component providing automatic speech recognition (ASR) capabilities, i.e., able to convert an input speech utterance into a textual representation. It receives as input the user's speech (typically captured by a microphone or provided as an audio file) and produces as output a sequence of words that most likely corresponds to what the user said, according to its internal acoustic and linguistic models. As part of this process, a speech recogniser also provides basic noise filtering and segmentation capabilities, to reduce the impact of background noise and detect utterances' boundaries, respectively.

In MARIO, Nuance Dragon NaturallySpeaking[2] was selected as S2T engine and integrated in the software platform as part of the vocal interaction interface, according to the process reported in the following subsections.

3.1 Identification of MARIO's Speech to Text Engine

As a basis for enabling robot reading and listening capabilities, an empirical analysis and comparative evaluation of state of the art speech recognition engines was performed at the early stages of Task 5.2 (second half of 2015), in order to identify the most appropriate and possibly best performing "off the shelf" speech recognition engine to be integrated in the MARIO software framework. To this end, we considered the leading state of the art speech recognition frameworks and services, and *(i)* we assessed them against the specific requirements reported below, and *(ii)* we empirically evaluated their recognition accuracy with respect to a reference dataset.

The fundamental requirements identified for the integration of a S2T engine in MARIO are the following:

1. multilingual support, with a focus on English and Italian (as languages required to cover MARIO's three pilot sites);

2. the ability to fully operate on-board the robot, with no dependency on external services and network connectivity, as a reliable Internet connection is not always available at the pilot sites and privacy concerns were raised concerning the transmission and processing of data/speech by third party services.

Although the second requirement would exclude any speech recognition framework or API made available online as-a-service, cloud-based speech to text APIs and services were considered as part of the initial set of candidate solutions. Specifically, these speech recognition engines and services were initially considered:

- CMU Pocketsphinx and CMU Sphinx4, which are open-source native C and Java

[2] https://www.nuance.com/dragon/dragon-for-pc/premium-edition.html

speech recognition libraries[3], respectively;

- Nuance Dragon NaturallySpeaking[4], a standalone software package for speech recognition;
- IBM Bluemix/Watson speech recogniser (now available as Watson Speech to Text[5]), Speechmatics[6], Project Oxford Speech to Text (now part of Microsoft's Cognitive Services suite[7]) and Google Cloud Speech-to-Text[8] as cloud-based online speech recognition services.

Among the online speech recognition services, IBM Bluemix/Watson speech recogniser, despite not providing S2T for Italian, was selected for further analysis: Speechmatics did not support the Italian language and preliminary empirical tests showed that it was too slow (i.e., had a high response time); Project Oxford Speech to Text did not support Italian and access to the API was tied to the usage of Microsoft's .NET Framework; Google's API, while providing S2T for Italian, requires a subscription fee for the Cloud Platform, with a free usage plan limited to 60 minutes/month (whereas IBM's service, while still requiring a subscription fee, offers a free usage plan for 1000 minutes/month).

The performance of CMU Pocketsphinx, Sphinx4, IBM Bluemix/Watson speech recogniser and Nuance Dragon NaturallySpeaking was evaluated on an open dictionary extracted from from anonymised patient dialogues collected from the three pilot sites and provided as audio recordings (with the corresponding transcripts). Specifically, the audio recordings were segmented and provided as input to the speech to text engines, and the recognition output was then compared to the available transcripts in order to compute the accuracy in terms of F1 score, as summarised in Figure 2.[9]

Combining these results with the lack of support for Italian of IBM Bluemix/Watson, Dragon NaturallySpeaking emerged as the best choice with respect to MARIO's requirements.

Nuance Dragon NaturallySpeaking. Dragon Naturally Speaking (DNS) 13 Premium is among the best off-the-shelf S2T engines, with full support for English and Italian. It can be installed and deployed as a standalone solution, without any dependency on Internet connectivity and external services. Moreover, the availability of a version for PC deployable on machines running Windows allows directly using the Kinect's microphone array (available on MARIO Kompai robots) as input device. The engine enables the definition of user-specific vocal profiles, through a voice training step that has to be performed once for each user to ensure recognition accuracy. The setup of a vocal profile takes only a few minutes and has thus no significant impact on the overall user experience. DNS then

[3] https://cmusphinx.github.io/wiki/download/
[4] https://www.nuance.com/dragon/dragon-for-pc/premium-edition.html
[5] https://www.ibm.com/watson/services/speech-to-text/
[6] https://www.speechmatics.com/
[7] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/
[8] https://cloud.google.com/speech-to-text/
[9] These results refer to an evaluation performed in the second half of 2015; the capabilities of the engines and services may have changed at the time of writing.

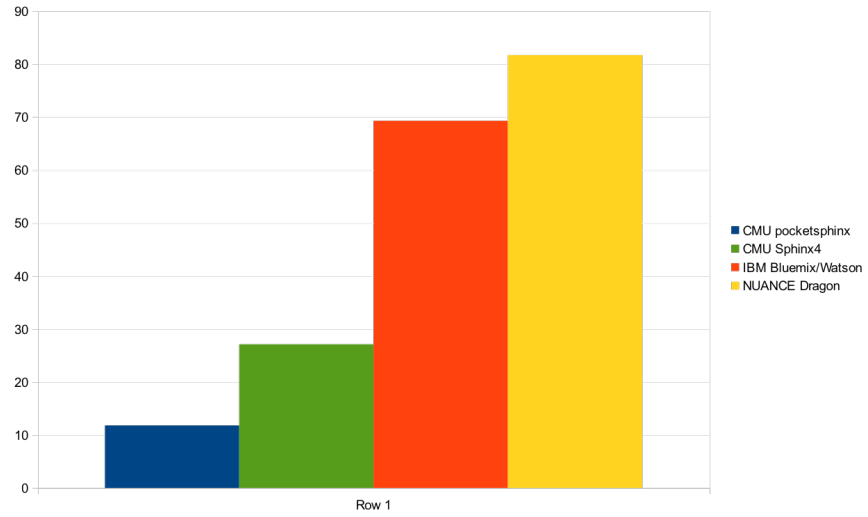


Figure 2: F1 score for open dictionary of the speech recognition services and engines tested

implements a continuous voice training process for maintaining users's vocal profiles and improving recognition accuracy over time. We initially bought license rights covering up to three installations. Additional licenses were then acquired for the other Kompai robots delivered to pilot sites once they were ready to be used for trial activities.

3.2 S2T Component Design, Implementation and Integration

As a standalone software package, Dragon NaturallySpeaking is not specifically designed to be programmatically integrated with and controlled by external applications. It does provide a scripting language (in Visual Basic/C#), but its usage is limited to the creation of new voice commands. To integrate DNS in the MARIO architecture and software framework, we leveraged on NatLink[10], an open source extension module for Dragon written in Python that allows accessing speech recognition output text and developing command and control macros to control the S2T engine.

MARIO's Speech to Text component is thus designed and implemented as a Python software package that wraps Dragon's recognition engine and is responsible for: *(i)* making available to the other components the textual representation of user's utterances provided by Dragon's engine; and *(ii)* implementing and providing an interface for controlling the status of the recogniser, enabling other components to activate and deactivate the acquisition of speech from the microphone. As shown in Figure 3, the S2T component integrates the Dragon recogniser and consists of two modules: the *Speech to text Manager* and the *Speech to text Controller*.

[10] http://qh.antenna.nl/unimacro/

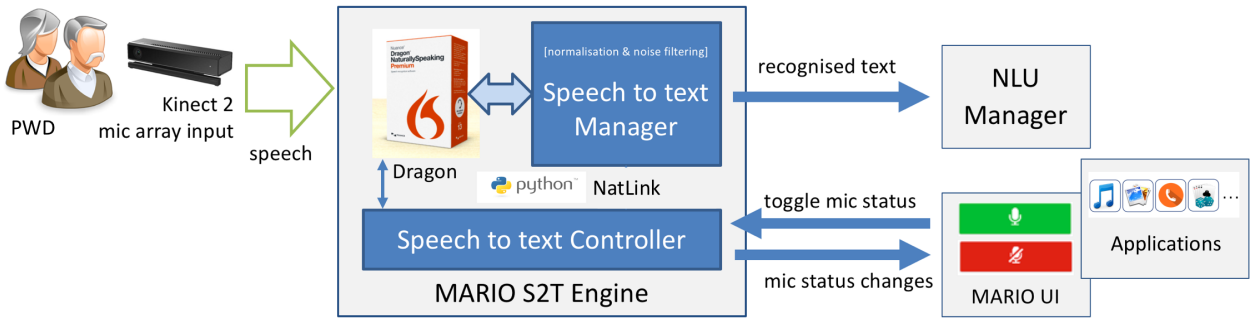


Figure 3: MARIO Speech to Text component

3.2.1 Speech to text Manager

The Speech to text Manager interacts with Dragon through the API of the NatLink module and is responsible for:

- intercepting the output text produced by the Dragon engine;
- filtering out output sequences produced by Dragon when the recognition fails because of background noise, cleaning the text by replacing specific character sequences produced by Dragon for accented characters (mainly in the case of Italian), and encoding the text in UTF-8 format;
- making the recognised text available to the NLU Manager to trigger the language processing and understanding process.

In the MARIO software platform, the communication between software components is mainly mediated by MARIO's *Event Bus*, a message-oriented middleware supporting topic-based publish-subscribe interaction patterns. In line with this paradigm, the Speech to text Manager is also responsible for creating and managing a dedicated topic (named `speech2text`) that is then used to make available the S2T output by publishing messages containing the recognised text. Similarly, the NLU Manager interested in receiving the S2T output subscribes to the named topic and receives from the Event Bus the messages containing the recognised text to be processed. An example of an event bus message containing as body the text produced by the S2T component is shown in Listing 1.

```
{
  "messageId":"speech2text-16",
  "timestamp":"2017-05-16T17:20:07.143",
  "properties":{},
  "body":"I would like to listen to some music"
}
```

Listing 1: Example of a message containing the S2T output text

3.2.2 Speech to text Controller

The Speech to text Controller interacts with Dragon through the API of the NatLink module and controls the status of the speech acquisition process. It is responsible for:

- processing the commands sent by other components to enable or disable the acquisition of the input speech from the microphone;
- notifying status changes of the acquisition process, i.e., informing other components that the microphone has been switched on or off.

As in the case of Speech to text Manager, the interaction between the Speech to text Controller and other components is mediated by the Event Bus. A dedicated topic (named `s2tstatecontroller`) is provided to the other components in order to send commands to the Speech to text Controller that subscribes to the topic to receive the commands. Similarly, a dedicated topic (named `s2tstate`) is available to the Speech to text Controller to publish status changes so that subscribed components are notified. When it receives a command sent on the `s2tstatecontroller` topic to switch on/off the recognition, the Speech to text Controller: *(i)* enables/disables Dragon's microphone (i.e., the ability to acquire and process the input speech), and *(ii)* notifies the status change by publishing a message to the `s2tstate` topic to inform interested components. In line with well-established Enterprise Integration Patterns [10], pairs of correlated messages are thus used to implement a two-way request-reply interaction pattern, allowing components to ask the S2T component to switch on/off the microphone and get back a response with the status change. A concrete example of request-reply message pair is provided in Listing 2.

```
// request to switch on the microphone (s2tstatecontroller topic)
{
  "messageId":"s2tstatecontroller-13",
  "timestamp":"2017-05-16T19:23:53.387",
  "properties":{},
  "body":"{
    \"status\":\"on\",
    \"source\":\"gui\"
  }"
}

// correlated response with status change (s2tstate topic)
{
  "messageId":"s2tstate13",
  "timestamp":"2017-05-16T19:23:53.393",
  "correlationId":"s2tstatecontroller-13",
  "properties":{},
  "body":"{
    \"status\":\"on\",
    \"source\":\"gui\"
  }"
}
```

Listing 2: Example of a request-reply message pair to activate the speech recogniser

Concretely, the status of the S2T component is controlled through the described mechanism by the following components.

- *MARIO User Interface*: all screens of MARIO's GUI show the current status of the S2T engine, with a red/green microphone icon shown at the top right corner, which also enables users to manually switch on/off the S2T engine.
- *MARIO Text to Speech* (integrated in MARIO User Interface): to avoid that robot's utterances are undesirably considered as input speech, the S2T engine is switched off before any utterance produced by the robot is spoken out, and is then switched back on.
- *MARIO Applications*: some applications provide features that need to ensure the S2T engine is not active during the interaction. This is the case of: *(i)* the *My Music* application that deactivates the S2T engine before playing music, to avoid that songs' lyrics are undesirably processed as input speech; *(ii)* the *My Games* application that deactivates the S2T engine before starting a game, to avoid that any background music or sound is undesirably processed as input speech; *(iii)* the *My Chat* application that deactivates the S2T engine during a call, to avoid that the call is considered as input speech.

4 Natural Language Understanding Subsystem

As introduced in Section 2.3, MARIO's components and applications have heterogeneous needs in terms of NLP and speech-based interaction management. For example, task management capabilities require to recognise user's intention to trigger an application, the CGA application relies on Q&A interaction patterns based on open- and closed-ended questions, and the My Memories application supports reminiscence by engaging the user with closed-ended questions or open-ended prompts.

To overcome this heterogeneity of needs and the impossibility to adopt a "one-size-fits-all" approach, MARIO's NLU subsystem provides and makes available a set of reusable and composable language processing and understanding services. MARIO's NLU Manager and applications can use these services and, on top of them, build their domain-specific user interaction logic and internal dialogue management strategy.

In the following we introduce the core language processing and understanding capabilities that were identified and then concretely implemented and made available.

Pattern Matching. The ability of recognising specific keywords (i.e., keywords spotting) or more complex patterns in the input text is a basic form of shallow language processing. Despite its relative simplicity, pattern matching becomes a powerful general-purpose tool to address well-defined recognition tasks, as in the case of yes-no questions. As it emerges from the review of dialogue systems and conversational agents for PWD reported in Deliverable 4.1 [11], keyword-spotting techniques are often at the heart of the interaction capabilities of these systems. In addition, pattern matching can be used as building block to support more advanced processing capabilities.

Named Entity Recognition and Linking. The ability of recognising entity mentions in the input text and associate them with the corresponding type (such as persons, places, etc.) is fundamental in language processing. The recognition of specific entities is a requirement of different MARIO applications: in the CGA process, for example, the answers the PWD is supposed to provide to some of the assessment questions correspond to specific entities such as locations, dates, numbers and persons. The recognition of entities can then be extended with the ability of linking the recognised entity mentions to specific entities in a knowledge base. Entity linking performed with respect to MARIO's user-specific knowledge base (KB) supports those use cases where applications (such as the My Memories and My Chat apps) have to understand that the PWD is mentioning, for example, a specific family member or friend. Entity recognition and linking introduce *semantic* features, with the ability of identifying the type of an entity mention and link it to a specific entity in the KB, respectively.

Word Sense Disambiguation. Word Sense Disambiguation refers to the ability of identifying the sense/meaning a word (among its potential word senses) in the context of

a sentence it is used in. This language understanding capability is a prerequisite for building other semantic interpretation and machine reading approaches, as reported in this section.

Frame Recognition. Frame Recognition refers to the ability of identifying the semantic frames evoked by words in a sentence. Frame recognition introduces frame semantics in MARIO's understanding capabilities and can be used by applications to relate the recognised frames to user's intent and link them to specific actions.

Semantic Role Labelling. Semantic Role Labelling has the goal of identifying the semantic role of the elements in a sentence. When coupled with frame recognition, semantic role labelling allows assigning to the constituents in a sentence their role as frame elements with respect to the recognised frame occurrence. This provides MARIO with deep frame-based semantic parsing and machine reading capabilities.

The modules implementing these services rely on different strategies and have different degrees of complexity, from shallow parsing to deep semantics-oriented natural language processing and interpretation.

Each module can provide a general-purpose service (e.g., for keywords detection, pattern matching, or frame recognition) or focus on a specific interpretation domain or task, such as the ability to recognise specific entities (e.g., persons, dates, numbers, etc.) and link them to the local, user-specific knowledge base.

Moreover, some modules provide their NLP/NLU services by reusing or composing services provided by other modules. For example, pattern matching is used as a basis for recognising specific entities, and (Named) Entity Linking (NEL) relies on the capabilities of the module providing (Named) Entity Recognition (NER) (Section 4.1.3). Similarly, frame-based semantic parsing, as provided by the FRED machine reader [9], combines multiple processing abilities in a complex pipeline that includes word sense disambiguation, frame recognition, semantic role labelling and entity linking.

Services also vary in terms of semantic features: while basic shallow parsing and processing techniques (such as pattern matching and named entity recognition) do not rely on advanced semantic features, other services providing (named) entity linking or deep frame-based semantic text analysis build on resources providing linguistic knowledge and background knowledge, and can also rely on and exploit local domain knowledge stored in the MARIO Knowledge Base.

Figure 4 specialises the diagram provided in Figure 1 and shows the core components and services that constitute MARIO's Natural Language Understanding subsystem. Each service and its capabilities are then presented in the next subsection.

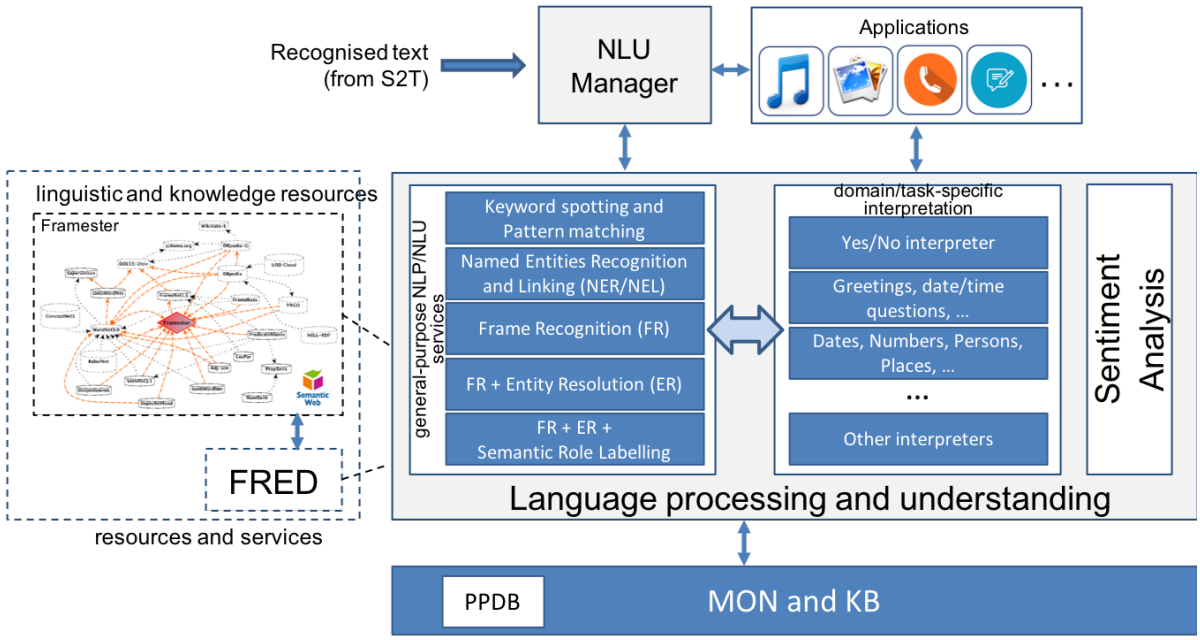


Figure 4: Overview of the NLU subsystem

4.1 Natural Language Understanding Services

This section provides a description of the Natural Language Understanding services that have been implemented, integrated into the MARIO platform, and made available to the other components and applications. Although sentiment analysis capabilities are reported in Deliverable [2], the corresponding service is summarised as part of this section.

4.1.1 Pattern Matching

This service aims at matching user's utterances against provided patterns, defined as regular expressions. This shallow parsing technique provides client applications with a general-purpose mechanism for detecting predefined sequences in the input text. In its simplest form, pattern matching can be used for simple keywords detection.

Evaluating regular expressions against strings is a common feature of every modern programming language. On top of the Java regular expression engine we have developed a service for facilitating the definition of regular expressions and their evaluation against natural language. The service takes as input a text and requires either a list of chunks or keywords, or an expression to be evaluated.

When operating on an input list of strings, the service performs a number of preprocessing and normalisation tasks for building a regular expression from the given chunks or keywords, and then it evaluates the resulting expression on the input text. The preprocessing and normalisation steps for composing the regular expression include lowering the case

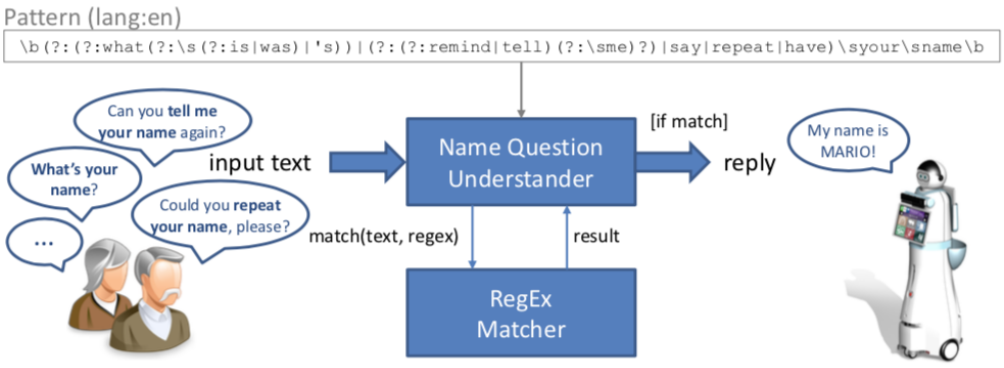


Figure 5: Example of NLU module that relies on the pattern matching service for recognizing when the PWD asks for the name of the robot

of the keywords (and input text), character substitutions (e.g., replacing white spaces with `\s`) and OR-ing the keyword strings. The service can be invoked to either match the entire input against the keywords or expression, or to identify the subsequences of the input text that match the pattern. In the first case a boolean value is returned (i.e. matched or not matched); in the latter case it returns a list of spans in the input text matching the regular expression.

Despite its simplicity, pattern matching *(i)* can be useful for identifying predictable input sequences or structured entities (e.g., dates or days of week); *(ii)* has a better performance and lower computational complexity with respect to deep NLP processing and complex syntactic/semantic parsing algorithms; *(iii)* provides a relatively simple yet powerful language to define pattern expressions.

However, among potential disadvantages we have: *(i)* the need to design and adjust the patterns so as to balance between too restrictive and too permissive expressions; *(ii)* the robustness against language variations has to be explicitly encoded in the patterns, and this can be hard to scale up to cover all possible inputs; *(iii)* no information is provided on the syntactic and/or semantic structure of the input text.

Beyond providing a general-purpose pattern matching capability, this service supports the development of higher-level understanding modules and can be used to manage interaction scenarios where the potential variability of user input can be encoded in a pattern expressions. An example is illustrated in Figure 5, where a regular expression is used to build an understanding module that recognises when the user asks for the name of the robot.

4.1.2 Enhancing Pattern Matching with Paraphrases

When defining patterns to be matched against the input text, the effort of encoding possible language variations is left to the one who build the expression. In some scenarios, this corresponds to defining possible variants of a restricted number of terms, as in the case of recognising the answer to a yes-no question (which, for example, cover most of the items in the CGA questionnaires).

To simplify this effort, we identified the possibility of further extending the pattern matching service with paraphrases. To this end, an additional service is provided, which is based on the service presented in Section 4.1.1. This service basically increases the coverage of a regular expression by OR-ing words with their paraphrases.

In this way, the initial pattern defined in the regular expression is expanded by including lexical, phrasal and syntactic paraphrases retrieved from the Paraphrase Database (PPDB)[11]. Therefore, the resulting regular expression is an alternation of paraphrases that extend the provided tokens. As an example, this service is concretely used for interpreting yes-no answers of the Comprehensive Geriatric Assessment Questionnaire (see Deliverable 4.3 [7] and Section 5.1). Starting from manually defined seeds of positive and negative expressions, the PPDB was queried to include the corresponding paraphrases and increase the pattern coverage for interpreting positive and negative answers.

Re-engineering PPDB. The Paraphrase Database (PPDB) [12] is an enormous collection of lexical, phrasal, and syntactic paraphrases. The database is released in six sizes (from S to XXXL) ranging from highest precision/lowest recall to lowest average precision/highest recall. PPDB is an automatically extracted database containing millions of paraphrases in 16 different languages. The goal of PPBD is to improve language processing by making systems more robust to language variability and unseen words. The entire PPDB resource is freely available under the Creative Commons Attribution 3.0 United States License. PPDB is distributed as a set of plain text files, with one paraphrase rule per line. For the English language, each line is formatted as follows

LHS ||| PHRASE ||| PARAPHRASE ||| (FEATURE=VALUE)* ||| ALIGNMENT ||| ENTAILMENT

where:

- PHRASE is a multiword expression;
- PARAPHRASE is its paraphrase;
- LHS is the constituent or CCG-style slashed constituent label (in the parse tree) for the paraphrase pair;
- ALIGNMENT indicates the type of alignment between phrase and its paraphrase (e.g., one-to-one, one-to-may etc.);

[11] http://paraphrase.org

- FEATURE is a list of scores calculated among the paraphrase pair;
- ENTAILMENT is an automatically assigned entailment relation holding between the PHRASE and PARAPHRASE (e.g., Equivalence for pairs like couch/sofa, or ForwardEntailment for pairs like dog/animal).

In order to make the PPDB corpus available to the understanding modules and MARIO's applications, we have transformed the paraphrase corpus in RDF (Resource Description Framework) according to a specific PPDB ontology module we designed. The PPDB ontology is available online at https://w3id.org/ppdb/ontology.

From the paraphrase corpus we extracted only the paraphrases for the English and Italian languages with an high confidence value. Access to the PPDB RDF dataset is also made available as a service, which enable the possibility to dynamically query the resource to retrieve paraphrases for a given term or expression.

4.1.3 Named Entity Recognition and Linking, and Word Sense Disambiguation

Named-entity recognition (NER) is the task of locating named entities mentioned in text and classifying them into a set of pre-defined semantic categories such as persons, organizations, locations, numbers, dates, etc. For example, given the sentence *"I was born in Rome, the capital of Italy"* the goal is to recognize that "Rome" and "Italy" are locations. Named-entity Linking (NEL) is the task of linking recognised entities mentioned in text to specific entities in a knowledge base. For example, in the sentence used before the goal is to determine that "Rome" refers the specific entity described in the Wikipedia article available at a specific URL[12]. Word Sense Disambiguation (WSD) denotes the task of determining the senses of words in a text, considering their context. For example, given the sentence used before, WSD has the goal of recognising that the word "capital" has the meaning of (i.e., the sense) "a seat of government" (rather that "wealth in the form of money or property").

State of the art tools and services providing general-purpose NER, NEL and WSD capabilities are available and are used as building blocks in the NLP pipeline of FRED (Section 4.1.5) and in the Word Frame Disambiguation service (Section 4.1.4).

Entity Recognition. To provide NER capabilities as a standalone service, we rely on and integrated the NER component provided by the Stanford's CoreNLP framework [13] (supporting the English language) and the NER component provided by the Apache OpenNLP library[14] (for which models supporting Italian are available). Both can be deployed off-line without the need of a network connection. However, due to the lack of support for the Italian language of NER capabilities for recognising mentions that refer to date elements (calendar days, months, days of week) and numbers, specific modules and services were

[12] https://en.wikipedia.org/wiki/Rome in this case the reference knowledge base is DBpedia
[13] https://stanfordnlp.github.io/CoreNLP/
[14] https://opennlp.apache.org/

implemented, on the basis of the pattern matching service, to provide this ability. This is particularly relevant for the CGA application, as some of the questions posed to PWD require to identify these elements to interpret user's answers. The NER service is also used, for example, by the My Memories application for recognising dates, places and persons mentioned by PWD during the reminiscence process.

Entity Linking. Concerning entity linking, existing general purpose NEL frameworks require a considerable computational power and working memory, making impractical their deployment on a robotic platform. Examples of this kind of services are Babelfy [13] (which jointly perform NEL and WSD), TagMe [14] and Apache Stanbol[15]. These tools can only be used as remote services and are integrated within FRED [9]. Babelfy and TagMe perform entity linking for both Italian and English.

General purpose NEL tools typically perform the linking with respect to "encyclopaedic" knowledge bases, such as DBpedia. In the context of MARIO, the need to perform entity linking with respect to the local user-specific knowledge base has emerged as an important requirement. For example, PWD may refer to relatives and friends in their speech, and those mentions have to be recognised and linked to the specific entities representing those people in the KB as part of user profile. A general purpose NEL is not able to identify that in a sentence like *"That's Paul in the picture"* the mention of the named entity *Paul* refers, for example, to PWD's son. Ad-hoc NEL solutions, made available as lightweigh NEL services working off-line, have thus been developed for supporting MARIO's applications.

Specifically, NEL services have been implemented for both for English and Italian to support:

1. the recognition of mentions that refer to PWD's relatives and friends; in particular the recognition and linking is not limited to persons' names and also considers the social relationship they have with the PWD, as defined in the KB: for example, in a sentence like *"That's my wife in the photo"* the service links *"wife"* with the entity of MARIO's KB representing PWD's wife;

2. the recognition of mentions of pre-defined entities representing persons in the knowledge base: this is the case of the entities representing, e.g., the current and former pope and president (representing the expected answers to specific questions of the CGA);

3. the recognition of mentions of pre-defined entities representing specific places, such as PWD's home address and the place where MARIO operates (e.g., the hospital or nursing home).

Word Sense Disambiguation. As in the case of NEL tool, the deployment of existing WSD services is impractical on robotic platforms. For instance, Babelfy, an application that jointly performs NEL and WSD, requires 25GB of RAM. Babelfy can be used only

[15] https://stanbol.apache.org/

as and external service, as exploited by FRED [9] and the Word Frame Disambiguation service [15] described in the next section. UKB [16] is a collection of programs for performing graph-based WSD. UKB can be configured so to consume an acceptable amount of resources hence allowing the deployment on a robotic platform. Although WSD is not explicitly used as a standalone service in the MARIO platform, to make available a WSD service to any application running on MARIO, we have developed a REST service exposing an interface for UKB. The service takes as input a text and provides as output the text tagged with word senses.

4.1.4 Word Frame Disambiguation

Word Frame Disambiguation is the task of recognizing frames evoked by words in a given sentence. For example, in the sentence ”Rome is the capital of Italy” the word ”capital” evokes the frame ”Political Locales”[16]. A service relying for Word Frame Disambiguation relying on Framester and Babelfy has been developed for the MARIO project and is available online at[17]. A description of the WFD is given in [15]. This service has also been integrated within the FRED NLP pipeline. An on-line demo of this service can be accessed at[18] and the API are documented at[19]. The service takes as input a text and a Framester profile (e.g. Base, Transitive etc.) and provides as output a JSON. An example of request and corresponding response is the following example 3. The service recognizes the occurrence of the Framester's Frames ”Desiring” and ”Reading” which are evoked by the words ”want” and ”read” respectively. The service also provides the result of the Word Sense Disambiguation performed through Babelfy (field ”bnSynset”).

```
// Request input: "I want to read the news, T"
{
  "text":"I want to read the news",
  "profile":"T",
  "annotations":[
    ...
    {
      "word":"want",
      "begin":"2",
      "end":"6",
      "bnSynset":"http://babelnet.org/rdf/s00086682v",
      "frames":["https://w3id.org/framester/framenet/abox/frame/Desiring"]
    },
    ...
    {
      "word":"read",
```

[16] Framester's Frame for Political Locales https://w3id.org/framester/framenet/abox/frame/Political_locales

[17] https://w3id.org/framester

[18] https://lipn.univ-paris13.fr/framester/en/wfd_json/sentence

[19] https://github.com/framester/Framester/wiki/Framester-Documentation#CURL_commands_for_Accessing_Word_Frame_Disambiguation_API

```
      "begin":"10",
      "end":"14",
      "bnSynset":"http://babelnet.org/rdf/s00092424v",
      "frames":["https://w3id.org/framester/framenet/abox/frame/Reading"]
    },
    ...
    {
      "word":"news",
      "begin":"19",
      "end":"23",
      "bnSynset":"http://babelnet.org/rdf/s00057546n",
      "frames":[]
    }
  ]
}
```


Listing 3: Example of output produced by the Word Frame Disambiguation Service

4.1.5 Frame-based Semantic Processing

A machine reader is a tool able to transform natural language text to formal structured knowledge so as the latter can be interpreted by machines, according to a shared semantics. FRED [17] is a machine reader for the semantic web: its output is a RDF/OWL graph. FRED performs a deep semantic analysis of text which is based on the frame semantics.

Frame Semantics. Frame semantics is a theory of linguistic meaning that relates linguistic semantics to encyclopedic knowledge. The basic idea is that one cannot understand the meaning of a single word without access to the knowledge related to that word. For example, one would not be able to understand the word ”sell” without knowing the situation of commercial transfer, which involves a seller, a buyer, goods and money. A word evokes a frame of semantic knowledge relating to the specific concept to which it refers. Frames are data structures representing stereotyped situations. A frame defines the types of the entities that participate to the situation and the roles that the entities play within that situation. For example, the frame ”Commerce” evoked by the word sell defines a situation where an ”Agent”, i.e. the ”Buyer”, gives to another ”Agent”, i.e. the ”Seller”, an amount of ”Money”, for some ”Goods”. In ”Buyer”, ”Seller”, ”Money” and ”Goods” are the roles (or frame elements in FrameNet terminology) of the ”Commerce” frame. ”Agent” is the class (or Semantic Type in FrameNet terminology) of entities that can play the roles ”Buyer” and ”Seller” within the frame ”Commerce”. FrameNet[20] is an ongoing project that aims at building a lexical database of English Frames, based on annotating examples of how words are used in actual texts.

Within the context of the MARIO project, FRED has been extended [9] for: (i) supporting adjective semantics; (ii) integrating Babelfy [13] for performing word sense disambiguation

[20] FrameNet, https://framenet.icsi.berkeley.edu

in both Italian and English[21]; (iii) and, integrating Word Frame Disambiguation within the pipeline [15]. A demo is available on-line at[22]. The default language for the demo is English, for Italian use <BING_LANG:it> before the sentence. Always finish a sentence with a dot.

FRED is a web-based system for automatic frame-based extraction of Linked Data and ontologies from natural language text. The approach implemented by FRED falls into the machine reading paradigm [18], which aims to transform (part of) a natural language text into data. FRED adds to that paradigm the ability to generate knowledge graphs that can be interpreted by machines, according to a shared formal semantics, and is linked to available background knowledge. It leverages the results of many NLP components by reengineering and unifying them in a unique RDF/OWL graph designed by following Semantic Web ontology design practices (e.g., ODPs - Ontology Design Patterns).

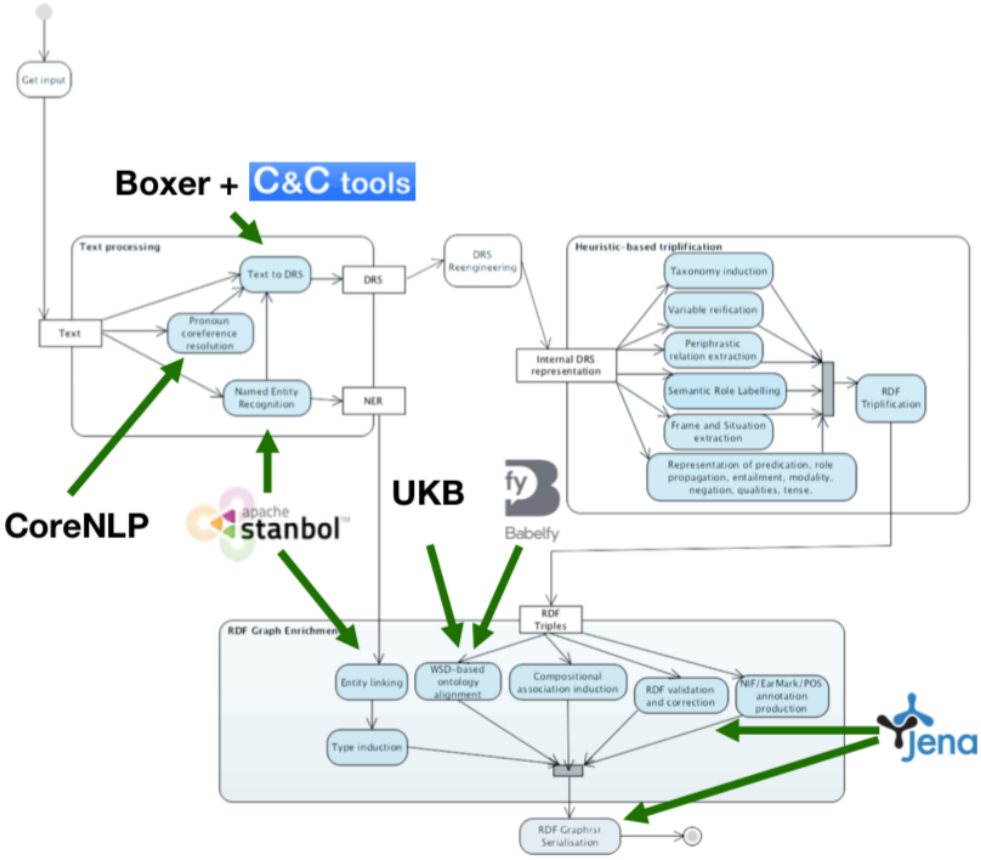


Figure 6: FRED workflow and architecture

The Figure 6 gives an overview of the FRED's architecture and the workflow carried out in order to produce the machine-reading representation of text. The core of FRED takes as input Discourse Representation Structures (DRSs), based on Hans Kamp's Discourse

[21] We got a free license for research purpose.
[22] http://wit.istc.cnr.it/stlab-tools/fred

Representation Theory (DRT) [19]. The DRSs taken by FRED as input are produced by Boxer [20], which performs deep parsing out of Combinatory Categorial Grammar (CCG) parse trees [21]. It also makes use of both VerbNet [22] and FrameNet [23] for frame labelling and semantic role labelling.

Semantic Role Labelling. Semantic Role Labelling (SRL) is the task of assigning labels to words or phrases in a sentence that indicate their semantic role in the sentence, such as that of an agent, goal, or result. For example, in the sentence "The box holds three hundred pictures." can be recognized two semantic roles "the container", i.e. "the box", and "the content", i.e. "three hundred pictures".

Additionally, FRED:

- represents modality, tense and negation in its unified OWL/RDF graph,by identifying the corresponding patterns in Boxer output;
- enriches the OWL/RDF representation of the sentence with compositional semantics, taxonomy induction and quality representation;
- integrates the results of Entity Linking (EL) performed on the input text for enriching its output graph with `owl:sameAs` axioms;
- exploits word sense disambiguation (WSD) in order to provide a public identity to these classes by identifying equivalent or more general concepts into WordNet [24] and BabelNet [25], and by creating alignments, where appropriate.

FRED is thus able to process and structure input text so as to produce an output that:

- consists of linked-data-ready data and ontologies, with a formal representation encoded in RDF/OWL;
- is aligned with public Semantic Web ontologies and public entity names in the Linked Open Data cloud;
- is further aligned with the MARIO Ontology Network (MON) and MARIO's knowledge base.

The Figure 7 shows the RDF/OWL graph returned by FRED on the input "I want to read news". This FRED graphs also includes the result of the Word Frame Disambiguation service (cf. Section 4.1.4). FRED recognizes the occurrence (i.e. `fred:want_1`) of the frame "Desiring" (evoked by the word "want") and the occurrence (i.e. `fred:read_1`) of the frame "Read" (evoked by the word "read"). Moreover, FRED recognizes the roles of the entities of the input sentence. The roles for the frame occurrence `fred:want_1` are:

- the *Experiencer*, i.e., the perceiver of the action "want", which is the entity identified as `fred:person_1`;
- the *Theme*, i.e. what the "Experiencer" want to happen, which is the frame occurrence identified as `fred:read_1`.

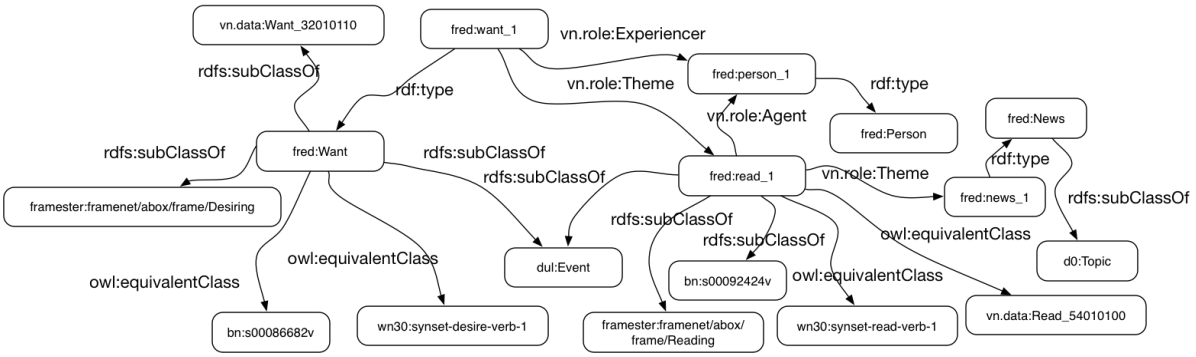


Figure 7: The picture shows the RDF/OWL graph that is the result of the deep semantic analysis performed by FRED on the sentence ”I want to read news”.

The roles for the frame occurrence `fred:read_1`:

- the *Agent*, i.e. the ”Agent” who reads, which is the the person identified as `fred:person_1` (it is worth noticing that it is the same entity that plays the role *Experiecer* within the frame ”Want”);
- the *Theme*, i.e. what is being read.

It is worth noticing that the knowledge graph produced by FRED is aligned with (i) Framester, thus enriching the graph with factual knowledge; (ii) BabelNet, WordNet and VerbNet that provide the senses of the words within the input sentence; (iii) Dolce Ultralite, that provide the foundational types of the entities mentioned in the input sentence.

The FRED API can be accessed on-line at[23]. The API specification in Swagger language is provided at[24].

4.1.6 Sentiment Analysis

Sentiment analysis refers to the ability of automatically extracting and categorising sentiment information from the textual representation of natural language sentences. MARIO's sentiment analysis subsystem, presented in Deliverable 5.7 [2], complements the capabilities provided by the NLU Subsystem with two additional services, based on sentiment polarity analysis and semantic sentiment analysis, respectively.

The sentence-based polarity detection service relies on the sentiment analysis module of the Stanford CoreNLP framework[25] and takes as input the textual representation of user's utterance and classifies the input sentence according to a five-value scale of sentiment. The output represents the overall tonality or sentiment expressed in the sentence, on a scale that includes *very negative* (-2), *negative* (-1), *neutral* (0), *positive* (1), and *very*

[23] http://wit.istc.cnr.it/stlab-tools/fred
[24] http://wit.istc.cnr.it/stlab-tools/fred/swagger.json
[25] https://nlp.stanford.edu/sentiment/

positive (2).

Semantic sentiment analysis capabilities are built as an extension of FRED and thus exploit a frame-base formal representation of the textual input. The service[26] reuses existing affective-based linguistic resources, such as SentiWordNet [26], to go beyond polarity detection and to identify in a sentence the sentiment expressed by an opinion holder on a certain entity or topic.

[26] http://wit.istc.cnr.it/stlab-tools/sentilo/

5 Representative Use Case Scenarios

Listening, Reading and Understanding services presented in Section 4 constitute the basis of the MARIO's dialoguing capabilities. The Comprehensive Geriatric Assessment (CGA) application and the My Memories application are representative examples of MARIO's applications that rely on NLP services as part of their logic and interaction/-dialogue management strategy. CGA and My Memories application requirements, characteristics and design are detailed in Deliverables 4.3 [7] and 3.3 [6], respectively. In the following, we describe the peculiarities of these applications in terms of NLP and dialogue management approaches. Research outcomes related to these applications have been presented at the 1st International Workshop on Application of Semantic Web technologies in Robotics (AnSWeR), co-located with the 14th Extended Semantic Web Conference (ESWC 2017) in Portoroz, Slovenia [M2, M3].

5.1 Comprehensive Geriatric Assessment Application

The Comprehensive Geriatric Assessment application enables the robot to perform a multidimensional assessment of the PWD through a conversational approach on the basis of standardised clinical questionnaires. Specifically:

- MARIO undertakes a dialogue-based interaction with the PWD, who is required to answer specific questions (e.g., about his/her daily life and ability to autonomously perform specific activities);
- user's answers are interpreted to assign a clinical score;
- the application extensively use the MARIO KB for retrieving user profiles, clinical tests, related multilingual questions and scores;
- the dialogues performed by the robot are defined through scripts which act as blueprints for the execution of evaluation questionnaires.

CGA's questions. The questions defined in the assessment questionnaires are either closed-ended or assume a specific answer known to the system.

- **YES/NO questions:** e.g. "Do you need any help when getting dressed?"
- **Multiple choice:** e.g. How many full meals do you eat a day? One, two or three?
- **Wh-questions** whose answers maps to entities/properties in the knowledge base (persons, places, dates, etc.) e.g. "What was your mother's maiden name?", "What is your street address?", "When were you born?"

Figure 8 shows the CGA's dialoguing architecture. The dialogue flow is driven by the robot (i.e., the interaction is system-initiated) and unfolds on the basis of question-answer

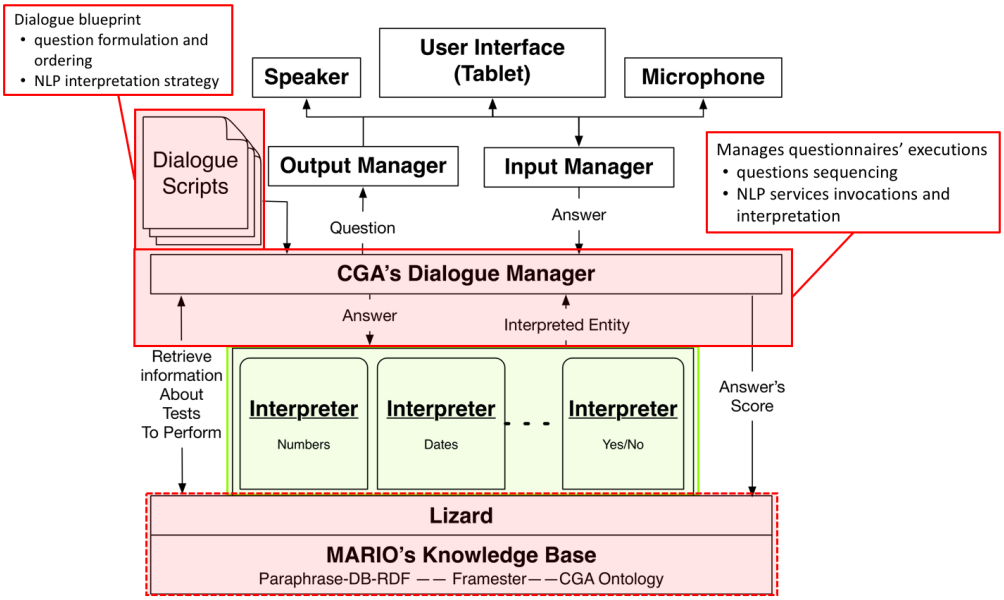


Figure 8: The CGA's dialoguing architecture.

interaction patterns defined in a *dialogue script*. The script can be used to define multiple dialogues and includes one dialogue for each clinical questionnaire to be performed. A dialogue specifies the questions to be asked, as a set of utterances, and their execution sequence. For each question/utterance the script defines:

- an identifier, as a URI used to retrieve from the KB additional information associated with the question;
- the type of the expected answer (e.g. yes/no, a number, a date, etc.);
- the NLP understanding service to be invoked in order to interpret user's reply;
- additional arguments for the NLP service invocation (e.g., the entities that the linguistic game has to recognise within the user's answer);
- a set of condition-action rules (i.e., conditional effects), defining the actions to undertake on the basis of the result of the interpretation (e.g., score assignments) and the identifier of the next question/utterance in the dialogue flow.

The CGA's dialogue can be easily configured through a JSON file. An example of the structure of the script is shown in the Figure 9 below. On the basis of the dialogue script, the Dialogue Manager retrieves from the KB (via Lizard's API) the information needed to formulate a question. Questions are spoken out by the robot and contextually shown on screen (with possible answers, where applicable) through MARIO's multimodal user interface. Responses provided through the touch screen are easily processed and the corresponding action (i.e., the score assignment) is directly executed. Vocal user's replies are processed according to the interpretation rules defined in the script, by invoking the NLP service in charge of interpreting the answer.

Figure 9: An example of dialogue script.

Direct observation of users' behaviour during trial activities shows that PWD tend to reply to MARIO's questions in a concise and focused way. The CGA's dialogue manager thus relies on the interpretation capabilities of the following NLP understanding services:

- "Yes/No/Don't know" interpreter, strengthened by the use of the paraphrases from PPDB; given a sentence, it is able to classify the answer as affirmative, negative or uncertain;
- "Number" understanding service, which recognises and extracts numbers from the answer;
- "Dates" understanding service, able to recognise and extract dates in user's answers;
- "Entity recognition" service, which recognises entity mentions (beyond numbers and dates) in user's answers (considering the entities stored in the knowledge base).

The interpretation result or recognised entities are then used by the Dialogue Manager to assign a score and move to the next question, as defined in the condition-action rules of the dialogue script. Specifically:

- conditions are boolean statements over possible results of the interpretation, actions define score assignments;
- e.g., for a question whose possible answers are yes (with a score of 1) and no (with a score of 0):
  - if <result_of_yes/no_understander>==YES, then assign 1 to the user's answer;
  - if <result_of_yes/no_understander>==NO, then assign 0 to the user's answer.

An example of interaction of the PWD with MARIO during a CGA session is provided in Figure 10. In this example, MARIO asks to Alex (i.e. the PWD) what is his mother's

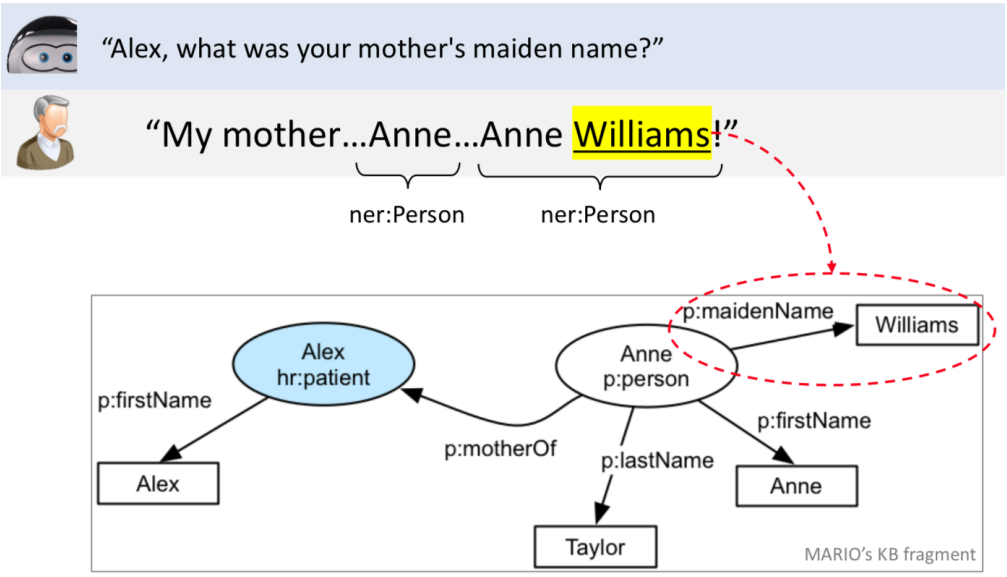


Figure 10: An of interaction of the PWD with MARIO during a CGA session.

maiden name. This question is included in the Short Portal Mental Status Questionnaire (SPMSQ). In order to interpret and assess what the PWD says, MARIO performs the following steps:

1. it performs the Name Entity Recognition on the PWD's utterance;
2. it check if NER service recognizes that a person is mentioned;
3. it verify if the mother of the PWD is mentioned with its maiden name.

In order to perform the last step, MARIO retrieves from its Knowledge Base the entity the representing the PWD's mother and all the information related with her.

5.2 My Memories - Reminiscence Application

The My Memories application enables the robot to undertake interactive and personalised reminiscence sessions through a conversational approach based on user-specific knowledge and materials. This application extensively accesses to the Knowledge Base in order to retrieve the user profile, the PWD family/social relationships, the events of the PWD life, the media objects (e.g., photographs) in which the PWD or one of its relatives appears in. The information retrieved from the KB are used to instantiate the interaction patterns. The interaction patterns are defined by the caregivers and aim at triggering PWD memories with verbal prompts and photographs. Figure 11 shows the architecture of the My Memories application. The interaction with the user during a reminiscence session can follow two different conversational approaches, both based on system-initiated dialogue fragments defined in the form of interaction patterns.

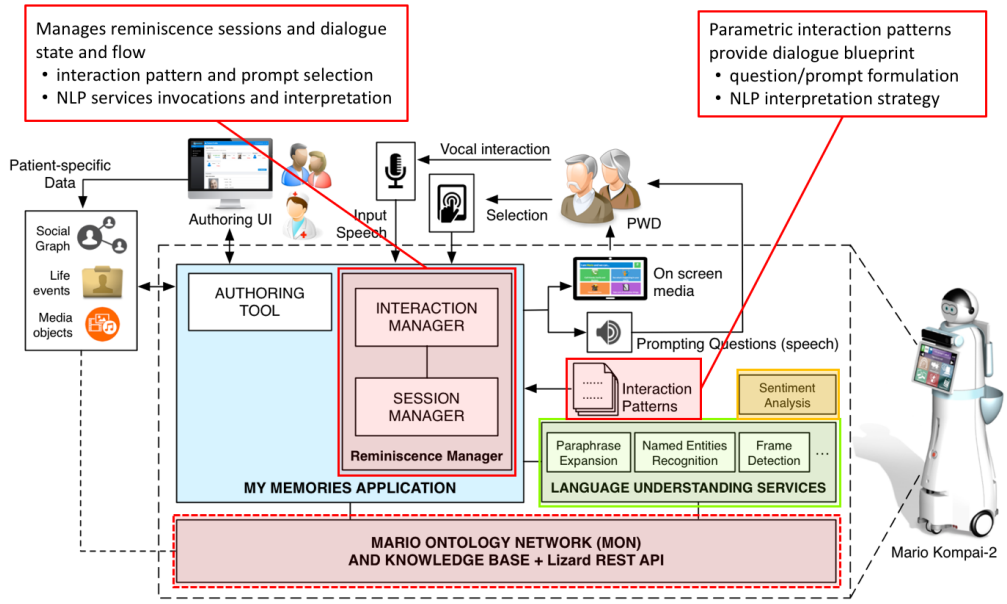


Figure 11: The architecture of the My Memories application.

Question-based. MARIO asks the user focused closed-ended questions related to the image contextually shown as memory trigger. The memory trigger can concern: (i) who appears in the picture; (ii) where/when the picture was taken; (iii) details about a person (e.g., birthplace); (iv) or life event (e.g., marriage date). The expected answer is known and it range from a simple positive/negative answer, to specific persons, places, dates or events that map to entities/properties in the knowledge base. NLP services support the process of evaluating user's answers with respect to the expected ones. The expected answers constrain the language interpretation domain and evaluation maps to entity recognition and linking NLP tasks.

Prompt-based. MARIO prompts the user with

- open-ended prompts (e.g., of the form "tell me more about..."");
- questions (e.g., of the form "what was it like to...", "what was s/he like...") related to the image.

In the prompt-based interaction sentiment analysis performed on the PWD's utterances plays a fundamental role (cf. Deliverable 5.7 [2]). When dealing with this type of prompts, the interpretation of user's replies adopts a different strategy and relies on sentiment analysis capabilities. Basically, the application attempts identify the polarity of user's utterances, to recognize whether the visual and verbal prompt is eliciting a positive, neutral or negative mood or reaction from the person. The interaction patterns are extended in this case by defining utterance templates for the different polarities, so that the robot can, e.g., encourage the user to tell him more about the subject if the reaction is positive, or otherwise propose to move to another picture.

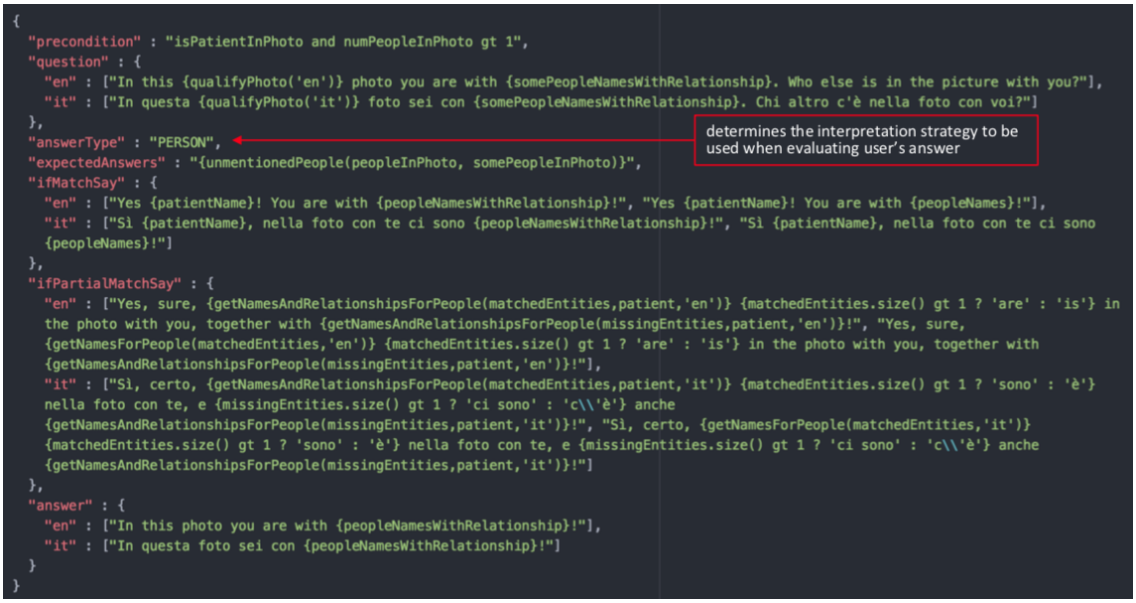


Figure 12: An example of the interaction pattern.

The interaction patterns can be configured through a JSON file, Figure 12 shows an example of such a file:

- The *precondition* defines under which conditions the prompting question can be used. The conditions are expressed as queries over the KB.
- The *question* is a multilingual parametric prompting question template to be instantiated with KB data (entities and their property values).
- The *answerType* is the entity type of the expected answer (e.g., yes/no, person, location, date, etc.).
- The *expectedAnswer* is the entity or the entities representing the expected answer, referencing KB entities and their property values.
- The *ifMatchSay* is a multilingual parametric system utterance template to be instantiated if user's answer matches with the expected one.
- The *ifPartialMatchSay* is a multilingual parametric utterance template instantiated if user's answer partially matches the expected one (if applicable).
- The *answer* is a multilingual parametric utterance templates instantiated if user asks for help or her answer does not match the expected one.

Interaction process. The selection of the interaction patterns is a dynamic process, driven by patient's replies and reactions, and by traversing the links in the knowledge graph on the basis of the dialogue context and history. So, for example, a question about when a photo was taken can be followed by a question concerning a person that appears

in the picture, and then move to a life event where the person participated in, and so on, exploiting the properties of and links between the entities in the knowledge base. Similarly, sentiment data can influence the selection process as well: for example, a negative reaction to a picture concerning an event or showing a specific person may lead to avoid subsequent prompts with images about the same event or with that person. Moreover, sentiment data emerging from the interactions can be associated with the concerned entities (pictures, people, events, etc.) and stored in the knowledge base. This knowledge is then used in subsequent reminiscence sessions so that, for example, photos that generated a positive reaction are favored in the selection process, whereas those causing negative reactions are less likely to be reproposed.

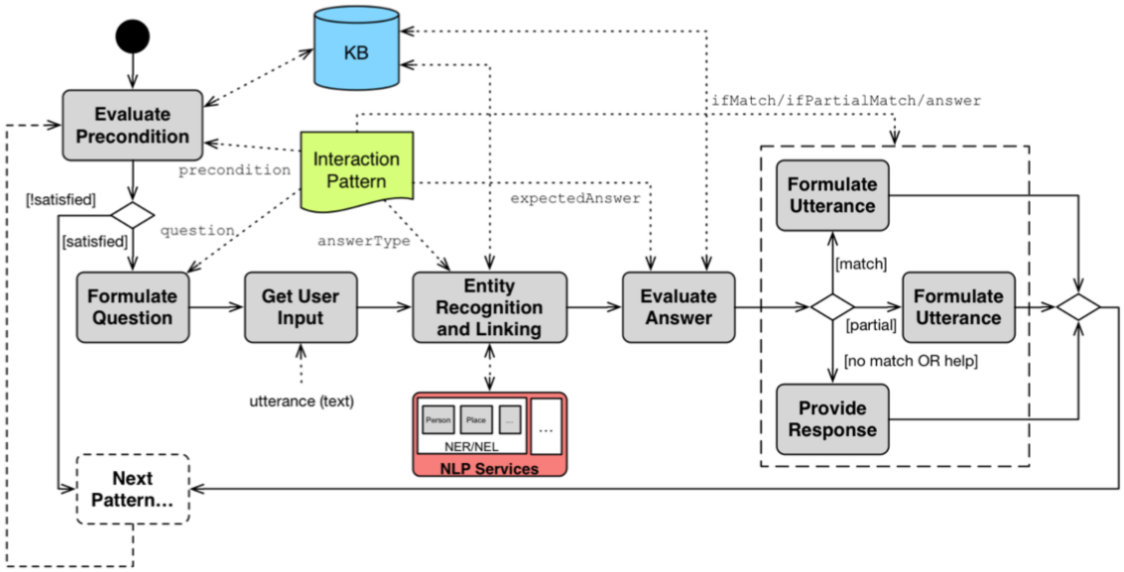


Figure 13: The overall process managing the dialogue of the My Memories application.

For an interaction pattern whose applicability preconditions are satisfied (with respect to the Knowledge Base, and in particular for a specific photograph), the dialogue is managed according to the following main steps:

- the corresponding question template is instantiated and the question is posed to the PWD;
- the textual representation of PWD's vocal input is processed according to the expected answer type, relying on entity recognition and linking capabilities of the NLP subsystem;
- depending on the outcome of PWD's answer evaluation step (the answer matches with the expected one, it partially matches, etc.), the corresponding utterance is issued by Mario, as defined in the interaction pattern.

The overall process is graphically summarised in the Figure 13 and then illustrated with a concrete example in the Figures 14 and 15.

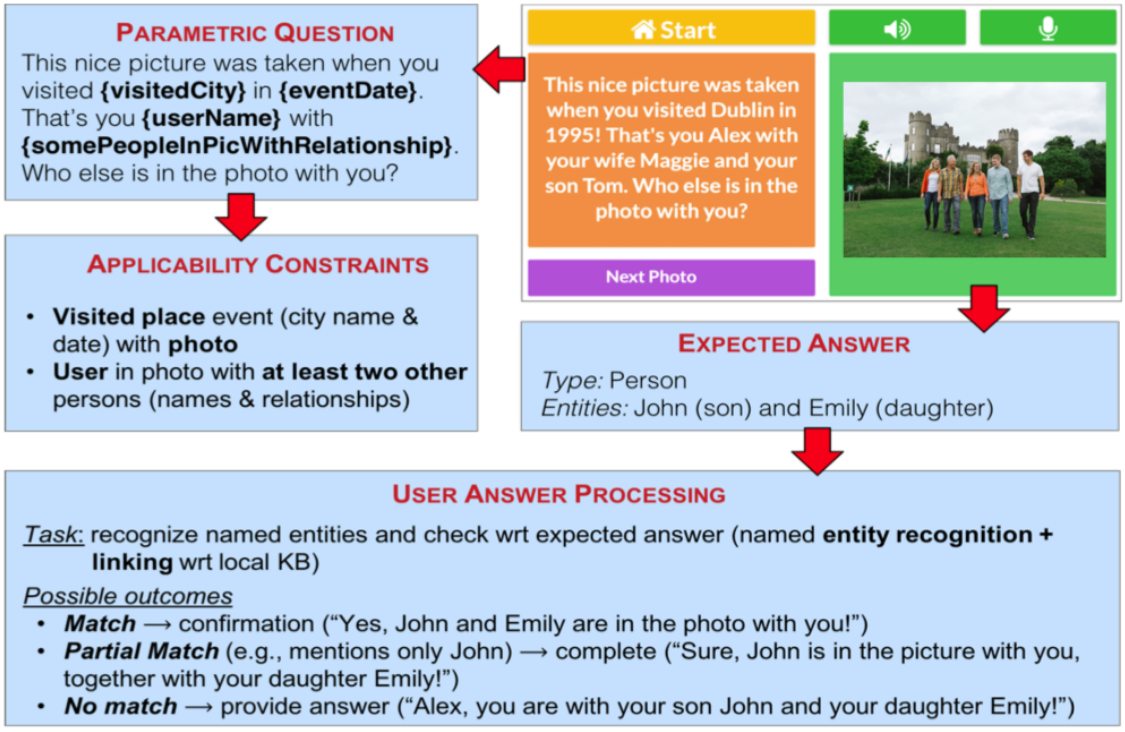


Figure 14: An example of interaction with the PWD during a reminiscence session.

The Figure 14 illustrates what is shown to the user during an interaction, whereas Figure 15 shows a PWD-robot interaction and the knowledge that the robot need in order to interpret the PWD's answer.

The answer is interpreted relying on the named entity recognition service. The interpreter also verifies if the persons recognized by the NER service match with expected ones.

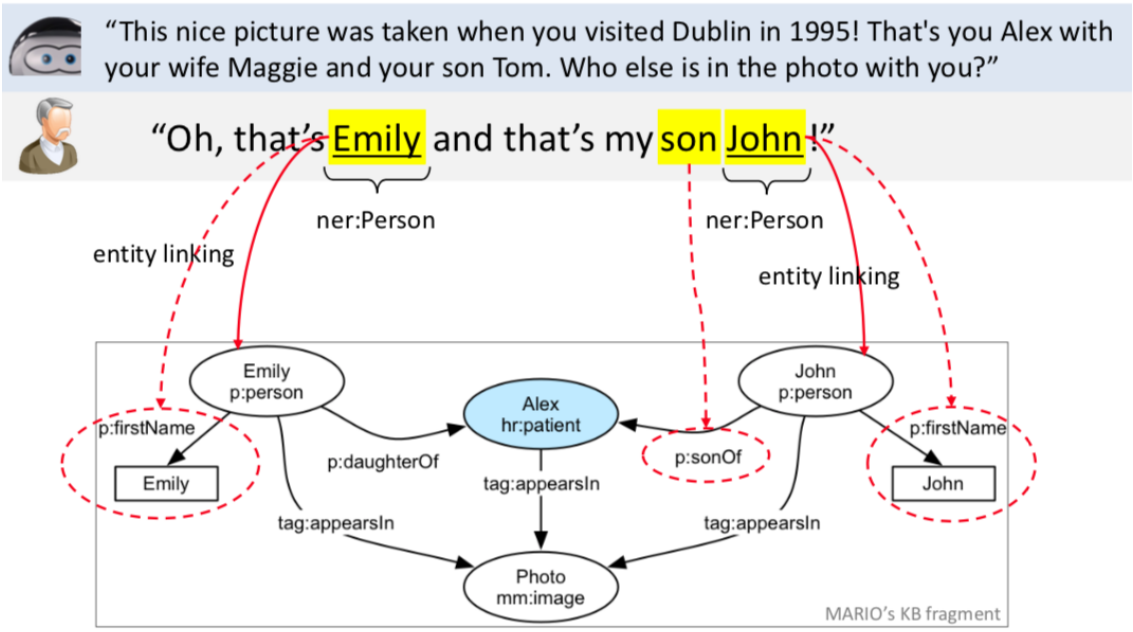


Figure 15: An example of interaction with the PWD during a reminiscence session and the interpretation tasks performed on PWD's answers.

6 Advanced Research Activities

This section outlines and summarises the research activities carried out within the context of the Task 5.2. These activities aimed at the identification of solutions targeting open problems in Natural Language Processing and Understanding and Knowledge Representation. These research problems are either inspired by and abstracted from concrete project use cases, or derive from general challenges that can be specialised in the context of socially assistive robots.

The research paths established in the context of Task 5.2 go beyond the time-frame of WP5 and represent ongoing work. While the level of maturity of the work presented in this section prevents an on-the-field deployment of the components based on the research activities, we summarise our main contributions as an integral part of the outcomes of Task 5.2.

6.1 Frame-based Ontology Matching

Every robot that uses natural language as a way to interact with humans needs of a method for bridging the information extracted from user speech and the system structured knowledge. This method should act into two ways. On the one hand, information extracted from users speech should be matched and integrated with system structured knowledge. On the other hand, when the robot needs to talk, the system structured knowledge should be verbalized.

In [27, 28], we have devised a method for aligning frame and ontologies. This method allows frame-based representation of the meaning of the users speech (e.g. a representation obtained using FRED [9]) to be integrated and stored in the knowledge base. The aim is to use frames as common interpretation key for text via alignment to the ontologies used for representing knowledge in the MARIO Ontology Network – MON (see D5.1), or any other ontology. This module is to be considered a research proof-of-concept, whose level of maturity prevents an on-the-field deployment during pilot trials.

Following [29], the approach devised for frame-ontology matching considers frames as "unit of meaning" for ontologies and exploits them as means for representing the intensional meaning of the ontology entities. Our strategy consists of three steps, summarized as follows.

Selecting frames evoked by annotations. In order to associate ontological entities with frames we analyze the textual annotation associated with them. Annotations provide humans with insights of the intensional meaning the designer wants to represent with a certain entity. The main idea of this approach is that words used in annotations *evoke* frames that are representative of the intensional meaning of the entity. In associating entity with frames, the ambiguity of words has to be taken into account. For instance, the verb *bind* evokes either the FrameNet's frame *Imposing obligation* or *Becoming attached*.

Therefore, to associate entities with the most appropriate frames, we have: (i) to associate words in the entities' annotation with the most appropriate sense (WSD by using UKB [16] and Babelfy [13]); (ii) and then, to select evoked frames by exploiting the Framester's mapping between WordNet's synsets and FrameNet's frames [15]. This approach is able to associate ontology entities to frames even if its annotations use specialized terminology. In this case it is exploited the Framester's mapping from Babelnet synsets and DBPedia resources[27] to frames. At the end of this step ontology entities are associated with a set of frames. For instance the object property `isParticipantIn` of the ODP Participation[28] is associated with the frames: *Participation*, *Collaboration*, *People* and *Evaluative comparison*.

Mapping frames and ontologies. This step creates an effective mapping between ontology entities and frames evoked by its textual annotations. An example of mapping is provided by FrameBase's integration rules [30]. FrameBase's rules allow to transform class to frame and properties to frame elements, or properties in binary projection of frames, and classes in their valences. These assumptions are too restrictive. The choice of certain ontological type for representing a concept depends on requirements that are external from the domain that is being represented. Therefore, we claim that the mapping ontologies-frames has to be done without assuming any fixed correspondence between the ontological types of the two models (e.g. without assuming that object properties always correspond to binary projections of frames). In order to identify the effective mapping between ontologies and frames, for each entity we compute any possible mapping between the entity and the frames selected in the previous step (i.e. those evoked by its annotations). In frame semantics, a frame is characterized by its roles (also called frame elements) and each element possibly define the semantic type of the individual that can play that role in the frame. Frames, frame elements and semantic types have a name and a description. For each ontology entity we compute the semantic text similarity (by means of ADW [31]) between the textual annotations of the ontology entity and those associated with the evoked frames, its elements, and its semantic types. We map the ontology entity to the top-scoring frame entity in semantic text similarity. For instance, it easy to see that the top-scoring alignment for `isParticipantIn` is that mapping it on the frame *Participation*[29], its domain/range (i.e. `Object` and `Event`) on the frame elements *Participant* and *Event*, respectively.

Frame-based ontology matching. Once input ontologies and frames are aligned, each ontology entity is associated with a formal specification of its intensional meaning (that we call *frame-based specification*). As pointed out in [32] the properties *subclass of* and *sub-property of* are not enough to explicit complex relation between entities. In light of this consideration we express the relation between frames and ontology entities by interpreting both as *predicates*. A formalization of frames as *multigrade predicates* is provided by [15]. A straightforward interpretation of ontology entities as predicates represents classes as n-ary predicates (the arguments of the n-ary predicate are

[27] Both Babelfy and UKB are able to perform entity linking over text.
[28] http://ontologydesignpatterns.org/wiki/Submissions:Participation
[29] FrameNet Frame Participation https://goo.gl/IMdAwA

the entities in its neighborhood) and properties as binary predicates. For instance, the class `TimeIndexedPartipation`[30] can be represented as a ternary predicate with arguments provided by `Event`, `TemporalEntity` and `Object`. Interpreting frames and ontology entities in predicates allows us to express complex relationship which cannot be formalized by only using OWL/RDFs vocabularies. Framester ontology [15] defines a set relationship holding between predicates. Using the Framester vocabulary the class `TimeIndexedPartipation` can be specified as `projectionOf` the frame *Participation*, with members `involveEvent`, `atTime` and `includesObject` (which can be interpreted as subroles of *Event*, *Time* and *Participant*). Also the property `isParticipantIn` of the ODP *Participation* can be specified as `projectionOf` the frame *Participation*, with members `Object` and `Event`. Therefore, the class `TimeIndexedParticipation` and the object property `isParticipantIn` are "aligned" to the same frame and a complex correspondence between `TimeIndexedParticipation` and `isParticipantIn` can be derived. In this case `isParticipantIn` is a `subframeOf TimeIndexedParticipation`. The subframe relation might be used for creating a `CONSTRUCT` SPARQL query or an inference rule[31] transforming instances of the class in instances of the property.

Discussion. This method exploits the frame semantics as cognitive model for representing the intensional meaning of ontology entities. The frame-based representation enables at finding correspondences between ontology entities abstracting from their logical type thus leading a step ahead the state of the art of ontology matching. The method allows information coming from potentially any application to be integrated within the MARIO's knowledge base. Moreover, the frame ontology alignment bridges the gap between structured knowledge and natural language allowing information extracted from user speech (e.g. using FRED [9]) to be integrated in the knowledge base.

The proposed approach has been implemented in a software module that is currently being evaluated. This module is to be considered a research proof-of-concept, whose level of maturity prevents an on-the-field deployment during pilot trials. We are evaluating the resulting alignments in a both direct and indirect way. The benchmarks used for assessing ontology matching systems are not able to evaluate the capability of finding correspondences among ontology entities with different logical types. In order to accomplish this purpose we are extending the existing benchmarks for ontology matching. On the other hand, we are using the proposed approach in a question answering system for selecting relevant resources answering a given question. The frame occurrences in a question together with the frame-ontology alignment help in formulate the query over the linked data, hence identifying resources that answer the given question.

[30] Time Indexed Participation ODP https://goo.gl/qX3DDr
[31] Refer to [30] for examples of these kinds of rules.

6.2 Equipping MARIO with Common Sense Knowledge

Common Sense Knowledge (CSK) is knowledge about the world, shared by all people. It is rarely expressed explicitly, e.g. in written or spoken communication, because of its very nature to be common and shared. CSK is essential for humans to understand different situations they encounter: the recognition of a scene in a picture, reported in a video, expressed in a spoken/written sentence, or experienced in the real world. Similarly, CSK is essential for Artificial Intelligence (AI) systems (e.g. robots) in order to enable their intelligent behavior [33].

Although recent research has produced multiple and diverse resources encoding forms of CSK (e.g. NELL [34] and ConceptNet [35]), existing resources mainly cover encyclopedic knowledge, which makes them suitable to answer questions about e.g. "the capital of a country" but it is useless for questions such as "what can be done with an object" that require some form of common sense reasoning. As a matter of fact, there is a type of CSK missing or hardly usable from existing resources, due to several reasons: (i) they are developed in isolation for specific tasks, e.g. action recognition; (ii) they use different tagging models and formats; (iii) they lack empirical validation e.g. CyC [36]; (iv) they suffer from sparsity and ambiguity of data, e.g. NELL. Furthermore, most CSK is hidden in unstructured content, e.g. images, videos and text, or in human competence.

As part of the research activities carried out during the MARIO project we have developed a set of methodologies for generating, collecting and integrating common sense knowledge within the MARIO's knowledge base. Framester is an example of lexical and factual common sense knowledge base included in the MARIO's knowledge base that strengths MARIO's understanding capabilities which are also improved by contextual knowledge. An example of contextual knowledge is the information about physical objects and the locations where it is likely to find them (e.g. it is usual to find the dishwasher in the kitchen and in the utility room). Besides strengthening the understanding capabilities, this knowledge is useful for other tasks such as stimulating memory or take the PWD to objects etc. Section 6.3 outlines the methodology that has been developed in order to automatically produce prototypical knowledge about physical objects and their common locations.

Another kind of common sense knowledge for a robot in an assistive context is the procedural knowledge. For example, to make a pancake someone needs a set of ingredients (Eggs, Milk and flour) and need to perform a series of steps (e.g., Mix, Cook etc.). This information could be useful on the one hand to strengthen the understanding capabilities with contextual information (if the user is preparing a pancake, it is more likely to listen words such as "cook", "mix", "milk" etc.). On the other hand, MARIO could use this knowledge to stimulate and guide the user in doing something. We have selected and integrated in the MARIO knowledge base the Human Activities Dataset [37]. The procedure for integrating this dataset into the MARIO knowledge base is described in the Section 6.4.

6.3 Prototypical Object-Location Relation Extraction Using Distributional Semantics

The methodology for automatically building a background knowledge of the prototypical Object-Location relation for the MARIO robot exploits both distributional and foundational semantics. This methodology is inspired by a work by Basile et al. [38] which has been furtherly extended in order to benefit of some recent results in automatic entity classification [39].

The methodology uses the distributional semantics for extracting this relation form a text corpus and the foundational semantics for typing the entities extracted from the text. A relation is a tuple t = (e_1, ..., e_n) where e_i are entities in a predefined relation r within a document D. Relation extraction is the task of extracting the tuples t from a document D. Basile et al. [38] suggested that the relatedness relation encoded in distributional vector representations can be made more precise based on the type of the entities involved in the relation, i.e., if two entities are distributionally related, the natural relation that comes from their respective types is highly likely to occur. For example, the location relation that holds between an object and a room is represented in a distributional space if the entities representing the object and the room are highly associated according to the distributional space's metric.

The procedure proposed by Basile et al. [38] relies on the manually annotated foundational types of the entities contained in the distributional space, thus hindering the use of this method on a larger scope. In [39] we have proposed a methodology for overcoming this issue. This methodology leverages supervised machine learning and crowdsourcing to automatically assess foundational distinctions over linked open data entities.

The resulting procedure is summarized in the following steps: (i) Obtaining a word vector space model of the entities of a given corpus (cf. Section 6.3.1); (ii) Selecting vectors representing objects and locations (cf. Section 6.3.2); (iii) Computing the similarity between vectors representing objects and locations entities (cf. Section 6.3.3).

6.3.1 Obtaining a word vector space model of the entities of a given corpus

Word space vectors are abstract representations of the meaning of words, encoded as vectors in a high-dimensional space. A word vector space is constructed by counting co-occurrences of pairs of words in a text corpus, building a large square n-by-n matrix where n is the size of the vocabulary and the cell i,j contains the number of times the word i has been observed in co-occurrence with the word j. The i-th row of the matrix represents the distributional representation of the corresponding word in the corpus. Words that appear in similar contexts often have similar representations in the vector space; this similarity is geometrically measurable with a distance metric such as cosine similarity, defined as the cosine of the angle between two vectors. Alternatively, a pre-computed vector space representation can be used. We used NASARI [40] which is a vector space representation

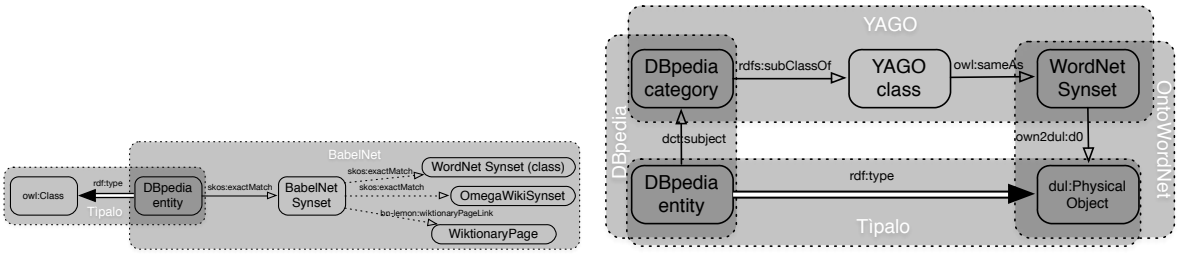
for BabelNet synsets (which include WordNet synsets and Wikipedia entities).

6.3.2 Selecting vectors representing objects and locations

In [38], Basile et. al. used a small subset of objects (only those falling under the Wikipedia category Domestic_implements). By the methodology presented in [39], we were able to select a larger set of Wikipedia entities representing physical objects. We approached this problem as a classification task, using two classification approaches: *alignment-based* and *machine learning-based*. These two approaches have been used to perform two very basic but diverse distinctions, which need to be addressed before approaching all the others: whether a LOD entity e.g. `dbr:Rome`[32], (i) inherently refers to a class or an instance, and whether it (ii) refers to a physical object or not. The first distinction (class vs. instance) is fundamental in formal ontology, as evidenced by upper-level ontologies (e.g. SUMO and DOLCE), and showed its practical importance in modelling and meta-modelling approaches in computer science, e.g. the class/classifier distinction in Meta Object Facility[33]. It is also at the basis of LOD knowledge representation formalisms (RDF and OWL) for supporting taxonomic reasoning (e.g. inheritance). Automatically learning whether a LOD entity is a class or an instance – from a common sense perspective – impacts on the behaviour of practical applications relying on LOD as common sense background knowledge. Examples include: question answering, knowledge extraction, and more broadly human-machine interaction. In fact, many LOD datasets that are commonly used for supporting these tasks (especially general purpose datasets e.g. DBpedia, Wikidata, BabelNet) only partially, and often incorrectly, assert whether their entities are classes or instances, and this has been proved to be a source of many inconsistencies and error patterns [41]. Since no established procedure exists, we tested different families of methods in an exploratory way. This led us to reuse – or compare to – existing work, which provides us with a baseline, which includes Tìpalo [42] as well as other relevant alignments between DBpedia and lexical resources (in particular those provided by Framester [15]).

Alignment-based Classification. Alignment-based methods exploit the linking structure of LOD, in particular the alignments between DBpedia, foundational ontologies, and lexical linked data, i.e. LOD datasets that encode lexical/linguistic knowledge. The advantage of these methods is their inherent unsupervised nature. Their main disadvantages are the need of studying the data models for designing suitable queries, and the potential limited coverage and errors that may accompany the alignments. We have developed **SENECA** (Selecting Entities Exploiting Linguistic Alignments), which relies on existing alignments in LOD, to make an automatic assessment of the foundational distinctions asserted over DBpedia entities. A graphical description of SENECA is depicted in Figure 16.

[32] `dbr:` stands for `http://dbpedia.org/resource/`
[33] `https://www.omg.org/spec/MOF/`



(a) The alignment paths followed by SENECA for selecting candidate classes among DBpedia entities. It identifies as classes all DBpedia entities aligned via BabelNet to a WordNet synset, an OmegaWiki synset or a Wiktionary page, and all DBpedia entities typed as `owl:Class` in Tìpalo.

(b) The alignment paths used by SENECA for identifying candidate Physical Objects among DBpedia entities. It navigates the YAGO taxonomy that via OntoWordNet links DBpedia entities to `dul:PhysicalObject` or Tìpalo that links DBpedia entities to `dul:PhysicalObject`.

Figure 16: SENECA approach for assessing whether a DBpedia entity is a class or an instance (Figure 16a) and whether it is a physical object or not (Figure 16b).

Class vs. Instance. As far as this distinction is concerned, SENECA works based on the hypothesis that common nouns are mainly classes and they are expected to be found in dictionaries, while it is less the case for proper nouns, that usually denote instances. This hypothesis was suggested by [43], who manually annotated instances in WordNet, information that SENECA reuses when available. A good quality alignment between the main LOD lexical resources and DBpedia is provided by BabelNet [25][34]. SENECA exploits these alignments and selects all the DBpedia entities that are linked to an entity in WordNet[35], Wiktionary[36] or OmegaWiki[37]. With this approach, 63,620 candidate classes have been identified, as opposed to WordNet annotations that only provide 38,701 classes. In order to further increase the potential coverage, SENECA leverages the typing axioms of Tipalo [42], broadening it to 431,254 total candidate classes. All the other DBpedia entities are assumed to be candidate instances. SENECA criteria for selecting candidate classes among DBpedia entities are depicted in Figure 16a.

Physical Object. Almost 600,000 DBpedia entities are only typed as `owl:Thing` or not typed at all. However, each DBpedia entity belongs to at least one Wikipedia category. Wikipedia categories have been formalised as a taxonomy of classes (i.e. by means of `rdfs:subClassOf`) and aligned to WordNet synsets in YAGO [44][38]. WordNet synsets are in turn formalised as an OWL ontology in OntoWordNet [45][39]. OntoWordNet is based on DUL, hence it is possible to navigate the taxonomy up to the DUL class for Physical

[34] We use BabelNet 3.6, which is aligned to WordNet 3.1

[35] `http://wordnet-rdf.princeton.edu/`, we use WordNet 3.0 and its alignments to WordNet 3.1, to ensure interoperability with the other resources

[36] `https://www.wiktionary.org/`

[37] `http://www.omegawiki.org/`

[38] We use YAGO 3, aligned to WordNet 3.1

[39] OntoWordNet is aligned to WordNet 3.0

Object. SENECA looks up the Wikipedia category of a DBpedia entity and follows these alignments. Additionally, it uses Tìpalo, which includes type axioms of DBpedia entities based on DUL classes. SENECA uses these paths of alignments and taxonomical relations, as well as the automated inferences that enable to assess whether a DBpedia entity is a Physical Object or not. With this approach, graphically summarised in Figure 16b, 67,005 entities were selected as candidate physical objects.

Machine learning-based Classification. Within machine learning, *classification* is the problem of predicting which category an entity belongs to, given a set of examples, i.e. a training set. The training set is processed by an algorithm in order to learn a predictive model based on the observation of a number of *features*, which can be categorical, ordinal, integer-valued or real-valued. We have designed our target distinctions in the form of two binary classifications. We have experimented with eight classification algorithms: J48, Random Forest, REPTree, Naive Bayes, Multinomial Naive Bayes, Support Vector Machines, Logistic Regression, and K-nearest neighbours classifier. We have used WEKA[40] for their implementation.
Features. The classifiers were trained using the following four features.

Abstract. Considering that DBpedia entities are all associated with an abstract providing a definitional text, our assumption is that these texts encode useful distinctive patterns. Hence, we retrieve DBpedia entity abstracts, and represent them as 0-1 vectors (bags of words). We built a dictionary containing the 1000 most frequent tokens found in all the abstracts of the dataset. The dictionary is case-sensitive, since the tokens are not normalised. The resulting vector has a value 1 for each token mentioned in the abstract, 0 for the others. By inspecting a good amount of abstracts, we noticed that very frequent words, such as conjunctions and determiners, are used in a way that can be informative for this type of classifications. For example, most of class definitions begin with “A” (“A knife is a tool...”). For this reason, we did not remove stop-words.

URI. We notice that the ID part of URIs is often as informative as a label, and often follows conventions that may be discriminating especially for the class vs. instance classification. In DBpedia, the ID of a URI reflects an entity name (it is common practice in order to make the URI more human-readable), and it always starts with an upper case letter. If the entity’s name is a compound term and the entity denotes an instance, each of its components starts with a capital letter. We have also noticed that DBpedia entity names are always mentioned at the beginning of their abstract and, for most of the instance entities, they have the same capitalisation pattern as the URI ID. Moreover, instances tend to have more terms in their ID than classes. These observations were captured by three numerical features: (i) number of terms in the ID starting with a capital letter, (ii) number of terms in the ID that are also found in the abstract, and (iii) number of terms in the ID.

Incoming and Outgoing Properties. As part of our exploratory approach, we want to test the ability of LOD to show relevant patterns leading to foundational distinctions. Given

[40] https://www.cs.waikato.ac.nz/ml/weka/

that triples are the core tool of LOD, we model a feature based on ongoing and outgoing properties of a DBpedia entity. An outgoing property of a DBpedia entity is a property of a triple having the entity as subject. On the contrary, an incoming property is a property of a triple having the entity as object. For example, considering the triple `dbr:Rome :locatedIn dbr:Italy`, the property `:locatedIn` is an outgoing property for `dbr:Rome` and an incoming property for `dbr:Italy`. For each DBpedia entity, we count its incoming and outgoing properties, per type. For example, properties such as `dbo:birthPlace` or `dbo:birthDate` are common outgoing properties of an individual person, hence their presence suggests that the entity is an individual.

Outcome of SENECA. Following an exploratory approach, we decided to use the output of SENECA as a binomial feature (taking value "yes" or "no") for the classifiers (excluding Multinomial Naive Bayes classifier).

6.3.3 Computing the similarity between vectors representing objects and locations entities

Given an object o and a location l and let v_o and v_l be the vectors associated to o and l respectively, the cosine similarity between v_o and v_l provides a measure of the similarity of o and l. This score is an indicator of how typical is the location l for the object o. Given an object, we can create a ranking of locations with the most likely location candidates at the top of the list. Additionally, an empirical measure commonness of entities could be used to re-rank or filter the result to improve its generality. In [38], Basile et al. used a URI counts extracted from the parsing of Wikipedia with the DBpedia Spotlight tool [41] for entity linking. The similarity ranges from -1 (unrelated) to 1 (related). An example of relation is provided by table 5

Physical Object	Location	Similarity
Dishwasher	Kitchen	.803
Dishwasher	Laundry room	.788
Dishwasher	Utility room	.763

Table 5: An example of object-location relation.

6.4 Populating the MARIO Knowledge Base with Generic Procedural Knowledge

In order to exploit the Human Activities Dataset for improving and enhancing the MARIO abilities this dataset must be integrated in the MARIO Knowledge Base. Aligning this dataset with Framester also improves the robot understanding capabilities and connect

[41] DBpedia Spotlight, http://www.dbpedia-spotlight.org

the generic procedural knowledge with the lexical and factual knowledge provided by Framester. Since both the Human Activities Dataset and Framester are aligned with the DBpedia Knowledge Base, the two dataset are naturally aligned. However, only inputs and outputs of procedures contained in the Human Activity dataset are aligned to DBpedia. We disambiguated the description of the steps with respect to BabelNet [25] and we also extracted the occurrences of the Framester frames to create a machine-readable representation of the steps. For example, the machine-readable representation of a step of the procedure ”How to make homemade sweet bread” is ”Combine whisked eggs with milk and mix”. In this step there is an occurrence of the Framester's frame Cause_to_amalgamate: Cause_to_amalgamate(Part_1: dbpedia:Egg, Part_2: dbpedia:Milk). The machine readable representations of the steps within a procedure are stored in the MARIO knowledge base and the MARIO's abilities are able to access this kind of knowledge.

7 Discussion and Lessons Learned

A fundamental requirement for social robots like MARIO is the ability to capture knowledge from multiple domains and manage it in a form that supports different tasks and facilitates sharing, reuse and integration. In MARIO the potential of ontology-based knowledge representation approaches and Semantic Web technologies has been considered as part of the development of a robotic system and its applications that deal with knowledge representation, acquisition and processing.

At the heart of MARIO's knowledge management framework, we designed the MARIO Ontology Network (MON), a set of interconnected and modularized ontologies covering different knowledge areas (ranging from user profiles to life events, multimedia content, etc.) and defining reference models for representing and structuring the knowledge processed by the robot. The experience gained in the design of the MON confirms the importance of adopting and following a well-established methodology. The design methodology we followed is based on an extension of eXtreme Design, an agile design methodology for ontology engineering. An important aspect to be highlighted, and that contributed to successfully define the MON, is the direct and continuous involvement of domain experts, including professional caregivers from the different pilot sites. Their contribution was fundamental for identifying the reference uses cases and describing the nature of the knowledge that the robot has to deal with. In particular, the work in MARIO confirms our previous experience on the key role of *competency questions* in the process of identifying the ontology requirements and iteratively refining the knowledge domains.

From a technical perspective, the adoption of consolidated best practices for ontology engineering was effective to support the evolution and maintenance of the ontology network. We refer in particular to the adoption of Ontology Design Patterns (ODPs) as reference templates and the indirect re-use of external ontology modules with the definition of alignment axioms. Following these approaches, we have that the ontology guarantees interoperability with respect to external modules and, at the same time, allows defining domain-specific extensions that satisfy MARIO's requirements. Also, the adoption of a modular design approach (with a network of ontology modules, as opposed to the definition of a single, monolithic ontology) has proven effective to deal with the heterogeneity of the knowledge areas. This allowed us to start with the design of core modules and then iteratively update, refine and extend the network to address emerging requirements.

Two additional elements were fundamental to ensure the adoption and integration of the MON and knowledge base in the platform.

1. The provision of a dedicated *Caregiver Interface*, as a Web-based Graphical User Interface that supports caregivers and family members in the process of building a user-specific knowledge graph, centered around user's profile, family/social relationships and life events. This was important to allow non-expert users to populate and update the local knowledge base by abstracting the complexities of the underlying models and technologies.

2. The provision of a set of software interfaces for programmatic, language-independent access to the knowledge base. With respect to this, the Lizard tool was designed and developed for enabling transparent access to the ontology network and knowledge base by generating a middleware API for client applications. Developers in charge of building client applications and not familiar with languages for knowledge representations and querying (including OWL/RDF and SPARQL) were provided with an abstraction layer for creating/ storing knowledge and for querying the shared knowledge base.

Both elements positively contributed to the goal of providing PWD with personalised interactions and user experience, designed and implemented in MARIO's applications on top of user-specific knowledge graphs. This is confirmed by the popularity among PWD of applications such as *My Memories* (as reported in Deliverable 8.3 [46]), which largely exploits the PWD-specific knowledge graph for supporting the reminiscence process and build personalised human-robot interactions.

With respect to the role of semantics and semantic technologies for supporting language processing and understanding, the experience gained in the context of MARIO has been useful to identify their applicability scope and open challenges that required, and still require, research activities.

As discussed in this document, language processing and understanding involves different techniques, and there is no "one-size-fits-all" approach able to support and address the heterogeneity of tasks and requirements. For example, a deep semantic-based parsing approach might be an overkill and potentially useless when trying to interpret user answers to yes-no questions; similarly, the usage of basic pattern matching is impractical when the PWD is prompted to talk about important events in her life history. It has thus been important to identify the different scenarios (again with the valuable input and support of caregivers and pilot sites professionals), provide a range of NLP/NLU capabilities, and identify the best service (or combination of services) for each scenario.

The introduction of semantic features with different degrees of complexity has contributed to support the specific needs of MARIO's application. The ability of recognising entity mentions and their semantic category (persons, places, etc.) has been the basis for supporting applications whose logic depends on this capability (e.g., the My Chat, My Memories and CGA applications). This has been further expended to take advantage of MARIO's user-specific knowledge base, with the ability of establishing a link between mentioned entities and their representation in the local KB. Entity recognition is (together with intent classification and recognition) at the heart of existing frameworks for building goal-oriented chatbots and dialogue-based applications.

When introducing approaches and techniques based on frame semantics, different challenges come into play. From a methodological perspective, the selection, detection and typing of domain-specific frames has to be performed. This represents a non-trivial effort, as it requires to *(i)* consider user requirements and frame datasets; *(ii)* match requirements to frames; and *(iii)* identify relevant domain-specific frames to be considered. While an ex-

isting resource such as FrameNet represents a valuable starting point, the actual coverage and specificity of the available frames is still limited, in particular with respect to our target domain. It is often the case that available frames are either too generic or there is no frame description to address a specific requirement. This has an impact on frame recognition capabilities: different sentences, although expressing different situations, might be associated with the same frame that can thus not be considered as a discriminative feature. This in turn has an impact on the possibility of clearly defining a mapping between the occurrences of frames, user's intents and robot's capabilities or applications. These aspects have inspired and motivated our work that has led to Framester [15], an open large-scale linked data knowledge graph, covering and interlinking linguistic, ontological and factual knowledge. As part of its goals, Framester aims at extending and increasing the coverage and availability of frames.

Deep frame-based semantic parsing and machine reading, as supported by FRED[9], is a powerful and promising approach in the field of language understanding. However, some steps in the processing pipeline correspond per se to open research problems, in particular concerning frame detection and semantic role labelling. The complexity further increases when dealing with languages other than English. While the coverage of linguistic resources and models supporting the pipeline is high for English, the availability of models and resources for other languages is limited. With respect to the problem of frame recognition, a relevant contribution comes from the possibility of combining Babelfy's word sense disambiguation results with Framester's resources. Specifically, given a sentence in Italian, Babelfy is able to recognise the Babelnet's synsets associated with the words in the sentence. The association between synsets and frames available in Framester can then be used to detect the evoked frames. However, to provide full frame-based semantic parsing, an initial machine translation step is still required.

Another aspect to be considered is that existing NLP models and resources are often built starting from corpora of written text of encyclopaedic nature. However, spontaneous speech can be characterised by linguistic features that differ from those of written text. In the case of PWD, features such as long pauses, repetitions, unfinished sentences and re-starts have clearly an impact on deep semantic parsing. While frame detection might still be possible by considering words evoking frames, deep semantic parsing is negatively affected. This also motivates our choice to consider different language processing strategies, as in some scenarios even a few words processed with shallow techniques can be the trigger to understand user's intent.

From a general perspective, research activities and efforts are required to address the lack of resources able to provide robots with so-called common-sense background knowledge. Common Sense Knowledge (CSK) is knowledge about the world shared by all people but difficult to teach to artificial intelligence (AI) systems. However, common sense reasoning is at the core of many unresolved AI tasks such as natural language understanding, object and action recognition, etc. While semantic technologies provide access to Web-scale resources and knowledge, these knowledge graphs are mostly of encyclopaedic nature (e.g., DBpedia). To fill this gap, different frameworks are emerging to make available het-

erogeneous knowledge for robotic systems and applications. Research projects and initiatives like KnowRob[42], RoboEarth[43] and RoboBrain[44] go beyond local knowledge bases and, also with the emergence of cloud-based robotics, propose Web-scale approaches. As outlined in Sections 6.2-6.4, we are working and contributing in different ways along multiple research paths, ranging from prototypical object-location relation extraction, to empirical analyses of foundational distinctions in the Web of Data.

[42] http://knowrob.org/
[43] http://roboearth.ethz.ch/
[44] http://robobrain.me/

Relevant Publications

The work and activities carried out in the context of Task 5.2 and presented in this deliverable have contributed to and are further detailed in the following publications.

[M1] Aldo Gangemi, Valentina Presutti, Diego Reforgiato Recupero, Andrea Giovanni Nuzzolese, Francesco Draicchio, Misael Mongiovì: "Semantic Web Machine Reading with FRED". Semantic Web Journal, 8(6), 2017 http://www.semantic-web-journal.net/content/semantic-web-machine-reading-fred-1

[M2] Luigi Asprino, Aldo Gangemi, Andrea Giovanni Nuzzolese, Valentina Presutti, Diego Reforgiato Recupero, Alessanro Russo: "Autonomous Comprehensive Geriatric Assessment". In: Proceedings of 1st International Workshop on Application of Semantic Web technologies in Robotics (AnSWeR 2017), Portoroz, Slovenia, 2017 http://ceur-ws.org/Vol-1935/paper-05.pdf

[M3] Luigi Asprino, Aldo Gangemi, Andrea Giovanni Nuzzolese, Valentina Presutti, Alessandro Russo: "Knowledge-driven Support for Reminiscence on Companion Robots". In: Proceedings of 1st International Workshop on Application of Semantic Web technologies in Robotics (AnSWeR 2017), Portoroz, Slovenia, 2017 http://ceur-ws.org/Vol-1935/paper-07.pdf

[M4] Aldo Gangemi, Mehwish Alam, Luigi Asprino, Valentina Presutti, and Diego Reforgiato Recupero. "Framester: A Wide Coverage Linguistic Linked Data Hub". In: Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW). (Bologna, Italy). Ed. by E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali. Springer International Publishing, 2016, pp. 239–254. https://doi.org/10.1007/978-3-319-49004-5_16

[M5] Luigi Asprino, Valentina Presutti, and Aldo Gangemi. "Matching Ontologies Using a Frame-Driven Approach". In: Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management and Satellite Events, EKM and Drift-an-LOD (EKAW 2016). (Bologna, Italy). Ed. by P. Ciancarini, F. Poggi, M. Horridge, J. Zhao, T. Groza, M. C. Suarez-Figueroa, M. d'Aquin, and V. Presutti. Springer International Publishing, 2016, pp. 101–104. https://doi.org/10.1007/978-3-319-58694-6_9

[M6] Aldo Gangemi, Diego Reforgiato Recupero, Misael Mongiovì, Andrea Giovanni Nuzzolese, and Valentina Presutti. "Identifying motifs for evaluating open knowledge extraction on the Web". In: Knowledge-Based Systems (2016) https://www.sciencedirect.com/science/article/pii/S0950705116301125

[M7] Luigi Asprino, Valerio Basile, Paolo Ciancarini, and Valentina Presutti. "Empirical Analysis of Foundational Distinctions in Linked Open Data". In: Proceedings of the 27th International Joint Conference on Artificial Intelligence and the 23rd European

Conference on Artificial Intelligence (IJCAI-ECAI 2018). 2018, (to appear) https://arxiv.org/abs/1803.09840

[M8] Luigi Asprino, Aldo Gangemi, Andrea Giovanni Nuzzolese, Valentina Presutti, Diego Reforgiato Recupero, Alessandro Russo. "Ontology-based Knowledge Management for Comprehensive Geriatric Assessment and Reminiscence Therapy on Social Robots". In: Data Science for Healthcare: Methodologies and Applications. Ed. by Sergio Consoli, Diego Reforgiato Recupero, Milan Petkovic (to appear).

References

[1] E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali, eds. *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*. (Bologna, Italy). Springer International Publishing, 2016.

[2] MARIO Consortium. *D5.7 – Robot Semantic Sentiment Analysis*. 2017.

[3] T. Fong, I. Nourbakhsh, and K. Dautenhahn. "A survey of socially interactive robots". In: *Rob Auton Syst* 42.3-4 (2003), pp. 143–166.

[4] MARIO Consortium. *D1.1 – MARIO System Specification*. 2015.

[5] MARIO Consortium. *D3.1 – 4-Connect Community Module*. 2017.

[6] MARIO Consortium. *D3.3 – 4-Connect My Social Network Module*. 2017.

[7] MARIO Consortium. *D4.3 – MARIO Robot CGA Module*. 2016.

[8] MARIO Consortium. *D5.1 – MARIO Ontology Network*. 2016.

[9] A. Gangemi, V. Presutti, D. Reforgiato Recupero, A. G. Nuzzolese, F. Draicchio, and M. Mongiovì. "Semantic Web Machine Reading with FRED". In: *Semantic Web* 8.6 (2017), pp. 873–893.

[10] G. Hohpe and B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[11] MARIO Consortium. *D4.1 – Service Robot Enabled CGA Approach*. 2016.

[12] E. Pavlick, P. Rastogi, J. Ganitkevich, B. V. Durme, and C. Callison-Burch. "PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Beijing, China: Association for Computational Linguistics, 2015.

[13] A. Moro, A. Raganato, and R. Navigli. "Entity Linking meets Word Sense Disambiguation: a Unified Approach". In: *Transactions of the Association for Computational Linguistics (TACL)* 2 (2014), pp. 231–244.

[14] P. Ferragina and U. Scaiella. "TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities)". In: *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM)*. (Toronto, Canada). Ed. by J. Huang, N. Koudas, G. J. F. Jones, X. Wu, K. Collins-Thompson, and A. An. DOI:10.1145/1871437.1871689. ACM, 2010, pp. 1625–1628.

[15] A. Gangemi, M. Alam, L. Asprino, V. Presutti, and D. Reforgiato Recupero. "Framester: A Wide Coverage Linguistic Linked Data Hub". In: *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*. (Bologna, Italy). Ed. by E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali. Springer International Publishing, 2016, pp. 239–254.

[16] E. Agirre and A. Soroa. “Personalizing PageRank for Word Sense Disambiguation”. In: *Proceedings of the 12th conference of the European chapter of the Association for Computational Linguistics (EACL)*. (Athens, Greece). Ed. by A. Lascarides, C. Gardent, and J. Nivre. The Association for Computer Linguistics, 2009, pp. 33–41.

[17] V. Presutti, F. Draicchio, and A. Gangemi. “Knowledge extraction based on Discourse Representation Theory and Linguistic Frames”. In: *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*. (Galway City, Ireland). Ed. by A. ten Teije, J. Völker, S. Handschuh, H. Stuckenschmidt, M. d’Aquin, A. Nikolov, N. Aussenac-Gilles, and N. Hernandez. Vol. 7603. Lecture Notes in Computer Science. DOI:10.1007/978-3-642-33876-2_12. Springer, 2012, pp. 114–129.

[18] O. Etzioni, M. Banko, and M. J. Cafarella. “Machine Reading”. In: *Proceedings of the Twenty-first Conference on Artificial Intelligence (AAAI)*. (Boston, Massachusetts). Ed. by Y. Gil and R. J. Mooney. AAAI Press, 2006, pp. 1517–1519.

[19] H. Kamp. “A Theory of Truth and Semantic Representation”. In: *Formal Methods in the Study of Language*. Ed. by J. A. G. Groenendijk, T. M. V. Janssen, and M. B. J. Stokhof. Formal Methods in the Study of Language pt. 1. Mathematisch Centrum, 1981, pp. 277–322.

[20] J. Bos. “Wide-Coverage Semantic Analysis with Boxer”. In: *Proceedings of the Conference on Semantics in Text Processing (STEP)*. (Venice, Italy). Ed. by R. Basili, J. Bos, and A. Copestake. DOI:10.3115/1626481.1626503. The Association for Computational Linguistics, 2008, pp. 277–286.

[21] M. Steedman. *The Syntactic Process*. Cambridge, MA, USA: MIT Press, 2000.

[22] K. Kipper Schuler. “VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon”. PhD thesis. University of Pennsylvania, 2006.

[23] C. F. Baker, C. J. Fillmore, and J. B. Lowe. “The Berkeley FrameNet Project”. In: *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL)*. (Montréal, Quebec, Canada). Ed. by C. Boitet and P. Whitelock. Morgan Kaufmann Publishers / ACL, 1998, pp. 86–90.

[24] C. Fellbaum, ed. *WordNet: an electronic lexical database*. MIT Press, 1998.

[25] R. Navigli and S. P. Ponzetto. “BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network”. In: *Artificial Intelligence* 193 (2012), pp. 217–250.

[26] S. Baccianella, A. Esuli, and F. Sebastiani. “SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining”. In: *LREC*. Ed. by N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias. European Language Resources Association, 2010.

[27] L. Asprino, V. Presutti, and A. Gangemi. “Matching Ontologies Using a Frame-Driven Approach”. In: *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management and Satellite Events, EKM and Drift-an-LOD (EKAW 2016)*. (Bologna, Italy). Ed. by P. Ciancarini, F. Poggi, M. Horridge, J. Zhao, T. Groza, M. C. Suarez-Figueroa, M. d’Aquin, and V. Presutti. Springer International Publishing, 2016, pp. 101–104.

[28] L. Asprino, V. Presutti, A. Gangemi, and P. Ciancarini. “Frame-Based Ontology Alignment”. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence and the Twenty-Ninth Innovative Applications of Artificial Intelligence Conference (AAAI 2017)*. (San Francisco, California USA). Ed. by S. P. Singh and S. Markovitch. AAAI Press, Palo Alto, California, 2017, pp. 4095–4096.

[29] A. Gangemi and V. Presutti. “Towards a Pattern Science for the Semantic Web”. In: *Semantic Web* 1.1-2 (2010), pp. 61–68.

[30] J. Rouces, G. de Melo, and K. Hose. “FrameBase: Representing N-ary Relations using Semantic Frames”. In: *Proceedings of the 12th European Semantic Web Conference (ESWC)*. (Portoroz, Slovenia). Ed. by F. Gandon, M. Sabou, H. Sack, C. d’Amato, P. Cudré-Mauroux, and A. Zimmermann. Springer, 2015, pp. 505–521.

[31] M. T. Pilehvar, D. Jurgens, and R. Navigli. “Align, disambiguate and walk: A unified approach for measuring semantic similarity”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. (Sofia, Bulgaria). Ed. by M. P. Hinrich Schuetze Pascale Fun. Association for Computational Linguistics, 2013, pp. 1341–1351.

[32] D. Ritze, C. Meilicke, O. Šváb-Zamazal, and H. Stuckenschmidt. “A pattern-based ontology matching approach for detecting complex correspondences”. In: *Proceedings of the 4th International Workshop on Ontology Matching (OM) collocated with the 8th International Semantic Web Conference (ISWC)*. (Chantilly, USA). Ed. by P. Shvaiko, J. Euzenat, F. Giunchiglia, H. Stuckenschmidt, N. Noy, and A. Rosenthal. CEUR-WS.org, 2009, pp. 25–36.

[33] P. Hayes. “The Second Naive Physics Manifesto”. In: *Readings in qualitative reasoning about physical systems*. Ed. by D. S. Weld and J. de Kleer. San Francisco, CA, USA: Morgan Kaufmann, 1989.

[34] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. “Toward an Architecture for Never-Ending Language Learning”. In: *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI)*. (Atlanta, Georgia, USA). AAAI Press, 2010, pp. 1306–1313.

[35] R. Speer and C. Havasi. “Representing General Relational Knowledge in ConceptNet 5”. In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC*. (Istanbul, Turkey). Ed. by N. Calzolari, K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis. European Language Resources Association (ELRA), 2012, pp. 3679–3686.

[36] D. B. Lenat. “CYC: A Large-Scale Investment in Knowledge Infrastructure”. In: *Communications of the ACM* 38.11 (1995), pp. 32–38.

[37] P. Pareti, E. Klein, and A. Barker. “Linking Data, Services and Human Know-How”. In: *Proceedings of 13th International Conference ESWC*. Ed. by H. Sack, E. Blomqvist, M. d’Aquin, C. Ghidini, S. P. Ponzetto, and C. Lange. Vol. 9678. Lecture Notes in Computer Science. Springer, 2016, pp. 505–520.

[38] V. Basile, S. Jebbara, E. Cabrio, and P. Cimiano. “Populating a Knowledge Base with Object-Location Relations Using Distributional Semantics”. In: *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*. (Bologna, Italy). Ed. by E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali. Springer International Publishing, 2016, pp. 34–50.

[39] L. Asprino, V. Basile, P. Ciancarini, and V. Presutti. “Empirical Analysis of Foundational Distinctions in Linked Open Data”. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence (IJCAI-ECAI)*. 2018, (to appear).

[40] J. Camacho-Collados, M. T. Pilehvar, and R. Navigli. “Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities”. In: *Artificial Intelligence* 240 (2016), pp. 36–64.

[41] H. Paulheim and A. Gangemi. “Serving DBpedia with DOLCE – More than Just Adding a Cherry on Top”. In: *Proceedings of the 14th International Semantic Web Conference (ISWC), Part I*. (Bethlehem, PA, USA). Ed. by M. Arenas, O. Corcho, E. Simperl, M. Strohmaier, M. d’Aquin, K. Srinivas, P. Groth, M. Dumontier, J. Heflin, K. Thirunarayan, and S. Staab. Springer International Publishing, 2015, pp. 180–196.

[42] A. Gangemi, A. G. Nuzzolese, V. Presutti, F. Draicchio, A. Musetti, and P. Ciancarini. “Automatic Typing of DBpedia Entities”. In: *Proceedings of the 11th International Semantic Web Conference (ISWC), Part I*. (Boston, MA, USA). Ed. by P. Cudré-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, and E. Blomqvist. Vol. 7649. Lecture Notes in Computer Science. Springer, 2012, pp. 65–81.

[43] G. A. Miller and F. Hristea. “WordNet Nouns: Classes and Instances”. In: *Computational Linguistics* 32.1 (2006), pp. 1–3.

[44] F. M. Suchanek, G. Kasneci, and G. Weikum. “Yago: A Core of Semantic Knowledge”. In: *Proceedings of the 16th International Conference on World Wide Web (WWW)*. (Banff, Alberta, Canada). Ed. by C. Williamson, M. E. Zurko, P. Patel-Schneider, and P. Shenoy. DOI:10.1145/1242572.1242667. ACM, 2007, pp. 697–706.

[45] A. Gangemi and P. Mika. “Understanding the Semantic Web through Descriptions and Situations”. In: *Proc. of the International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE 2003)*. Catania, Italy: Springer, 2003, pp. 689–706.

[46] MARIO Consortium. *D8.3 – Evidence of Service Robots Benefits*. 2018.