­­­­­

M913 Dialogue Systems

A blue and white logo

Description automatically generated with low confidenceGraphical user interface, text

Description automatically generated

A picture containing hanger

Description automatically generated

Kylafi Christina-Theano (Theatina) – LT1200012

Final Project: UniPal

**Uni**versity **Pal** – Academic Assistant

Table of Contents

[Introduction 2](#_Toc121588282)

[Project Description 2](#_Toc121588283)

[File Structure 2](#_Toc121588284)

[System Configuration 3](#_Toc121588285)

[Requirements 3](#_Toc121588286)

[System Specifications 3](#_Toc121588287)

[Features 3](#_Toc121588288)

[Supported Scenarios (stories) 3](#_Toc121588289)

[Custom Actions 3](#_Toc121588290)

[NLU Pipeline 3](#_Toc121588291)

[Intent Recognition 3](#_Toc121588292)

[Feature Extraction 3](#_Toc121588293)

[Dialogue Policy 3](#_Toc121588294)

[Evaluation 4](#_Toc121588295)

[Test Stories 4](#_Toc121588296)

[Human Evaluation 6](#_Toc121588297)

[Questions 6](#_Toc121588298)

[Results 6](#_Toc121588299)

[Test Stories 6](#_Toc121588300)

[Configuration Comparison 6](#_Toc121588301)

[Human Evaluation 6](#_Toc121588302)

[Conclusions 7](#_Toc121588303)

[Future Work 8](#_Toc121588304)

[References 9](#_Toc121588305)

# Introduction

## Project Description

The task of the project at hand was to build a dialogue system (chatbot) focused on a specific domain, able to support functions (scenarios) with the use of domain-specific intents, slots and custom actions (web crawling, information extraction, text processing, etc. ).

In our case, **UniPal** [[1]](#footnote-1)– the chatbot that was built is an **academic assistant**, able to fetch various types of information from the University (National and Kapodistrian University of Athens) Department’s (Department of Informatics & Telecommunications) [website](http://www.di.uoa.gr/) (web crawling using custom actions), perform processing functions on the acquired data and then present them in a user-friendly, simple and comprehensible way. Analysis on the system’s specifications is performed in detail in the sections that follow.

## File Structure

Core system files:

* config.yml :
* domain.yml :
* stories.yml :
* test\_UniPal\_stories.yml :
* nlu.yml :
* rules.yml :
* actions.py :

Other files:

* scripts/ :
* slack/ :
* output files/ :
* notes/ :
* results/ :

# System Configuration

## Requirements

## System Specifications

### Features

### Supported Scenarios (stories)

### Custom Actions

## NLU Pipeline

### Tokenization

### Feature Extraction

### Intent Recognition

### Entity Extraction

## Dialogue Policy

### DIET

# Evaluation

The final system versions (parameter/configuration experiments) were evaluated based on both system tests (using test stories) and human evaluation (questionnaire after user-bot interaction/scenarios). There is no standardized way of measuring the quality of such virtual agents, therefore we have established our own standards specifically for this project, focusing on the system’s task completion efficiency domain-specifically.

**Keywords**: Effectiveness, user satisfaction, functional simplicity, ambiguity, flow,

(https://www.researchgate.net/publication/348014085\_Trends\_Methods\_in\_Chatbot\_Evaluation)

## Test Stories

(create test stories and compare results for different dialogue policies/configurations)

Test story **types** (in file test\_UniPal\_stories.yml) :

1. Latest University Announcements
2. Classes Timetable
3. Exams Timetable
4. Contact / Location / Access information
5. University’s Staff information
6. Psychological Support information

**Configurations** :

1. **NLU** Pipeline comparison

After the aforementioned comparison, it was decided to proceed using the 2nd (best f1-score on test set as shown in the results presented in section [**Results**](#_Results)) to perform the policy comparison.

1. **Policy** comparison

After the test stories evaluation process, the final 3 models (3 different policies) will be given to human testers for further evaluation, the results of which are presented below.

**Notes:**

* Intentional errors in messages/entities/values to check the system’s robustness
* Confusion matrices, plots, tables with training/test with cross val output

## Human Evaluation

### Questions

## Results

### Test Stories

### Configuration Comparison

##### NLU Pipeline Testing

Notes:

* Ascending -> more nlu examples would optimize performance

##### Policy Testing

Notes:

* Ascending -> more stories would optimize training / performance

### Human Evaluation

The final models that were selected after the previous evaluation processes, were tested through human evaluation.

##### 1st Phase

##### 2nd Phase

# Conclusions

# Future Work

**Notes**

1. Action uni class schedule: working well (due to different .xls format): 19-20, 20-21, 21-22

Fix: add more if’s to include them

1. Google calendar add reminders for exam/class
2. More stories – interactively to boost confidence and remove ambiguity
3. Intents: request exam/class timetable + inform programme/period/year/semester (the system is confused between request\_exam\_schedule and inform\_programme – normal outcome of the stories I feed the system! – future work to correct it)
4. Multi-intent classification (e.g. request\_exam\_schedule+inform\_programme, etc. ) to create better rules & stories (convo flows)

# References

(Articles & Papers the project at hand was based on)

1. [Trends & Methods in Chatbot Evaluation](https://dl.acm.org/doi/10.1145/3395035.3425319)
2. [Slack API: Applications | UniPal Slack](https://api.slack.com/apps/A02LRHXS7U2/event-subscriptions)
3. [How to call custom action from another custom action - Rasa Open Source - Rasa Community Forum](https://forum.rasa.com/t/how-to-call-custom-action-from-another-custom-action/11649/8)
4. [Medicare Locator](https://github.com/RasaHQ/medicare_locator/blob/master/actions.py#L228)
5. [Connect your chatbot with google calendar | The Rasa Blog | Rasa](https://rasa.com/blog/connect-your-chatbot-with-google-calendar/)
6. [Forms](https://rasa.com/docs/rasa/forms/#activating-a-form)
7. [NLU Training Data](https://rasa.com/docs/rasa/nlu-training-data/#visualizing-the-training-data)
8. [SimGus/Chatette: A powerful dataset generator for Rasa NLU, inspired by Chatito](https://github.com/SimGus/Chatette)
9. [Chatbots Using Python and Rasa - GeeksforGeeks](https://www.geeksforgeeks.org/chatbots-using-python-and-rasa/)
10. [How to create a FAQ Chatbot with Rasa?](https://www.steadforce.com/how-tos/how-to-create-a-faq-chatbot-with-rasa)
11. [Introducing DIET: state-of-the-art architecture that outperforms fine-tuning BERT and is 6X faster to train | The Rasa Blog | Rasa](https://rasa.com/blog/introducing-dual-intent-and-entity-transformer-diet-state-of-the-art-performance-on-a-lightweight-architecture/)
12. [Model Configuration](https://rasa.com/docs/rasa/model-configuration)
13. [Testing Your Assistant](https://rasa.com/docs/rasa/testing-your-assistant/#evaluating-an-nlu-model)

1. [GitHub Repository](https://github.com/theatina/UniPal) [↑](#footnote-ref-1)