Introduction to IBM’s Watson and its services

Autor:

Luis Fernández Ortega

Tutor:

Francisco Serradilla García
Resumen:

En este proyecto fin de grado se presenta la plataforma Watson de IBM, un sistema cognitivo nacido para dar respuesta a la necesidad de procesar la información escrita en lenguaje natural.

En este proyecto se plantea una introducción a los conceptos fundamentales de Watson, una descripción de los servicios que lo componen y algunos de los casos de uso en los que pueden aplicarse; haciendo especial hincapié en el desarrollo de asistentes virtuales.

En concreto, este proyecto tratará de proporcionar una solución al problema de la saturación de los buzones de dudas; centrándose para ello en el del Departamento de Prácticas de esta escuela (la Etsisi).

Dicha solución ha sido desarrollada para desplegarse en la nube IBM Cloud (anteriormente conocida como Bluemix); haciendo uso de los distintos entornos de desarrollo necesarios, e integrada mediante Node-RED.

Este documento se ha estructurado de la siguiente forma: En primer lugar, se presenta una breve historia de Watson. A continuación, se detalla cada uno de sus servicios junto con una breve descripción de los mismos. Después se estudia con mayor profundidad el servicio Watson Assistant (sobre el que se basa la solución propuesta). Finalmente, se plantean una serie de conclusiones, así como los posibles caminos que se pueden seguir en desarrollo de la solución.
Abstract:

This Grade Final Project presents the IBM Watson Platform, a cognitive system born to fulfill the need of processing information written in natural language.

This project offers an introduction to the basic concepts of Watson, a brief description of its services and some use cases which can be solved using one or more of them; making special remark on the development of virtual assistants.

Specifically, this project will try to purpose a solution to the saturation problem present in many virtual mailboxes, focusing on the mailbox of the Department for Academic Practices of the Etsisi.

The mentioned solution has been developed to be deployed on the IBM Cloud (formerly known as Bluemix); making use of several of the development environments offered by IBM, and using Node Red to integrate the whole solution.

This document has been structured according to the following guidelines: First comes a brief history of Watson; followed by a list of its services. After that, comes a detailed study of the service Watson Assistant (upon which is based the practical case). Lastly, a series of conclusions accompany the options for continuing the project in the future.
# Table of contents

1. Introduction .................................................................................................................. 1
2. Goals ............................................................................................................................. 3
   2.1 Theoretical goals: ...................................................................................................... 3
   2.2 Practical goals: ......................................................................................................... 3
3. Motivation ....................................................................................................................... 4
4. Background .................................................................................................................... 5
   4.1 Cognitive computing: ............................................................................................... 5
   4.2 What is Watson: ........................................................................................................ 6
   4.3 Deep Language processing: .................................................................................... 6
   4.4 Watson’s decision making: ..................................................................................... 7
5. State of the art: .............................................................................................................. 9
   5.1 Watson nowadays: .................................................................................................... 9
   5.2 Introduction to the services of Watson: ................................................................. 10
      5.2.1 Watson Assistant (Former Watson Conversations): .............................................. 10
      5.2.2 Watson Discovery: ............................................................................................ 10
      5.2.3 Watson Knowledge Catalog: ............................................................................ 12
      5.2.4 Watson Knowledge Studio: ............................................................................. 12
      5.2.5 Watson Language Translator: .......................................................................... 13
      5.2.6 Watson Machine Learning: .............................................................................. 14
      5.2.7 Watson Natural Language Classifier: ............................................................. 15
      5.2.8 Watson Natural Language Understanding: .................................................... 16
      5.2.9 Watson Personality Insights: ........................................................................... 16
      5.2.10 Watson Speech to Text: .................................................................................. 17
      5.2.11 Watson Text to Speech: .................................................................................. 17
      5.2.12 Watson Tone Analyzer: .................................................................................. 18
      5.2.13 Watson Visual Recognition: .......................................................................... 18
      5.2.14 Watson Studio: ............................................................................................... 19
      5.2.15 Watson Explorer: ............................................................................................ 20
   5.3 Other IBM Cloud services: ....................................................................................... 20
      5.3.1 Node-RED: ...................................................................................................... 20
   5.4 Success Stories: ....................................................................................................... 21
      5.4.1 Watson and the Spanish Tax Agency ................................................................. 21
      5.4.2 Woodside’s knowledge pool ............................................................................. 23
Regarding the front end: ............................................................................. 49
Results of the process: .................................................................................. 49
Steps prior to the development: .................................................................... 51
Formalization and classification of questions: ................................................. 51
First elicitation of requisites: ......................................................................... 51
Development of the MVP: .............................................................................. 51
Enrichment and first deploy: .......................................................................... 52
Addition of a new environment: ..................................................................... 53
Second round of questions: ............................................................................ 53
Designing for economy: .................................................................................. 53
Adding new capabilities: ................................................................................ 54
Solving the problem with number one: ........................................................... 55
Conclusions and future work ......................................................................... 56
Conclusions: ..................................................................................................... 56
Conclusions derived from the theoretical study. .............................................. 56
Conclusions of the practical experiment: ......................................................... 57
Future Work: .................................................................................................. 58
Continue training the assistant: ...................................................................... 58
Updating the knowledge of the assistant: ......................................................... 58
Expanding the dominion of the assistant: ......................................................... 59
Integrating with Watson Discovery: ................................................................. 59
Embedding the assistant into the Faculty Web: ............................................... 60
Adding a voice interface: ............................................................................... 60
Social, environmental and ethical implications: .............................................. 61
Bibliography: ................................................................................................. 62
Annex 1: Deploying the assistant through Telegram. .................................... 63
Recovering the credentials from the assistant: .............................................. 63
Configuring Telegram: ................................................................................... 64
Creating the Node-RED instance: ................................................................. 66
Annex 2: Starting data set .............................................................................. 76
List of FAQ: .................................................................................................... 76
# Table of Figures:

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Default workspaces view</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Intents overview</td>
<td>30</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Entities overview</td>
<td>33</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Dialog node conditions</td>
<td>34</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Logical operators inside a condition</td>
<td>35</td>
</tr>
<tr>
<td>Figure 6</td>
<td>accessing testing tool</td>
<td>39</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Manage context</td>
<td>40</td>
</tr>
<tr>
<td>Figure 8</td>
<td>&quot;Try it out&quot;</td>
<td>40</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Accessing Improve</td>
<td>41</td>
</tr>
<tr>
<td>Figure 10</td>
<td>– Improve - Overview</td>
<td>42</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Improve - User conversations</td>
<td>43</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Sample architecture</td>
<td>44</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Architecture of the solution</td>
<td>50</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Annex 1 - Accessing deploy</td>
<td>63</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Annex 1 - Retrieving credentials</td>
<td>64</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Annex 1 - Adding BotFather</td>
<td>64</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Annex 1 - Creating a new bot</td>
<td>65</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Annex 1 - Retrieving bot token</td>
<td>66</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Annex 1 - Choosing the service</td>
<td>67</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Annex 1 - Creating the instance</td>
<td>67</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Annex 1 - Configuring the service</td>
<td>68</td>
</tr>
<tr>
<td>Figure 22</td>
<td>Annex 1 - Accessing the toolkit</td>
<td>68</td>
</tr>
<tr>
<td>Figure 23</td>
<td>Annex 1 - Accessing manage content</td>
<td>69</td>
</tr>
<tr>
<td>Figure 24</td>
<td>Annex 1 - List of bundles</td>
<td>70</td>
</tr>
<tr>
<td>Figure 25</td>
<td>Annex 1 - Installing the bundle</td>
<td>70</td>
</tr>
<tr>
<td>Figure 26</td>
<td>Annex 1 - Design of the flow</td>
<td>71</td>
</tr>
<tr>
<td>Figure 27</td>
<td>Annex 1 - Connecting the nodes</td>
<td>71</td>
</tr>
<tr>
<td>Figure 28</td>
<td>Annex 1 - Adding the bot to Node-RED</td>
<td>72</td>
</tr>
<tr>
<td>Figure 29</td>
<td>Annex 1 - Adding the assistant to Node-RED</td>
<td>73</td>
</tr>
<tr>
<td>Figure 30</td>
<td>– Annex 1 - Coding the first function node</td>
<td>73</td>
</tr>
<tr>
<td>Figure 31</td>
<td>Annex 1 - Coding the second function node</td>
<td>74</td>
</tr>
<tr>
<td>Figure 32</td>
<td>Annex 1 - Deploying the solution</td>
<td>75</td>
</tr>
</tbody>
</table>
1 Introduction

Nowadays, information has become one of the most valuable actives of the companies. Thus, the ability of extracting value from various forms of data is one of the most compelling needs in the current business world.

To cope with the rising demand for tools capable of transforming, analysing and presenting data, IBM S.A. has developed a portfolio of services (fundamentally based on cloud) grouped under the umbrella of Watson technology.

Seven years have passed since Watson competed in the television show Jeopardy. In this time, Watson has evolved from a single supercomputer tuned for one specific purpose (being able to accumulate and process information to answer the show’s particular type of questions) to a highly configurable, cloud based platform that allows everyone to tailor their solutions to their needs and easily tune them whenever needed.

This document intends to present Watson core services, along with some real-world problems that benefit from its use.

The document follows the following structure:

First comes an introduction to all the services conforming the Watson core offering, focusing mainly on their features and use cases.

---

1 In the show Jeopardy, instead of providing an answer to a question, the contestants are given a statement and are asked to provide a suitable question that might have originated that answer.
After that, follows a series of use cases developed by IBM and its partners to solve various problems of the latter ones.

Next to the success stories, the document delves into a detailed review of one of the services offered by Watson (Watson assistant) selected due to the role it plays on the practical exercise developed for this project.

Finally, this paper presents a use case relevant to the faculty (the Etsisi\(^2\)) and details the creation of a possible solution to the problem presented using Watson technologies.

\(^2\) Etsisi: Escuela Técnica Superior de Ingeniería de Sistemas Informáticos – Universidad Politécnica de Madrid
2 Goals

This PFG\textsuperscript{3} has two distinct parts; each with its own objectives. The first part, and its goals are focused on theoretical explanations, whereas the second part is centered on practical goals.

2.1 Theoretical goals:

Theoretically speaking, this project aims to introduce the reader to the basics of IBM Cloud\textsuperscript{4} and Watson, reviewing important concepts ranging from the history of Watson to the definition of cognitive computing.

This project also expects to provide the reader with basic knowledge about all of Watson core services, and even some ideas regarding possible use scenarios.

Lastly, the final goal of the PFG is to make the reader acquainted with the service Watson assistant, and with the help of the practical part, guide even the most profane of developers through the process of designing a virtual assistant.

2.2 Practical goals:

On a practical level, the goal of this PFG is to generate a solution in the form of a virtual assistant. The purpose of the assistant will be answering questions posed by students of the Etsisi regarding the subject of external practices in the school.

Knowing that the casuistry of the questions is too huge to define, the purpose of the assistant will be to answer the frequently asked questions, leaving the most intricated ones to the personnel responsible for the practices at the Etsisi.

Starting from only a document containing a list of FAQs (Frequently Asked Questions) and the requirements elicited from the personnel, the assistant that results of the project should be able to solve a significant amount of the student’s doubts about the matter.

\textsuperscript{3} Proyecto de Fin de Grado – End of grade Project

\textsuperscript{4} Formerly known as IBM Bluemix
3 Motivation

In today’s world, the dynamic nature of the knowledge paired with the ever-growing amount of people who are required to access it, makes the task of informing increasingly more difficult.

Relying on people to explain sets of knowledge means accepting certain restraints, such as: limited span of time, limited reach, inconsistency (due to a myriad of reasons), etc.

On the other hand, entrusting the transfer of knowledge to machines has many advantages such as: scalability, nigh ubiquity and most important of all complete consistency. However, despite all this upsides, and maybe because of them, machines lack the ability to improvise, to go out of their script; in other words, they lack the “human factor”.

Fortunately, not all fields of knowledge require that level of adaptability, and here is where virtual assistants (i.e. machine teachers) come at hand.

A great deal of documents, especially those containing sets of rules (or laws) can benefit of having a directed guide to the specific matter of interest for each particular user.

Likewise, many of the doubts that may emerge of the study of any document, are shared by most users and can, therefore be defined to be answered by a virtual assistant.

Knowing all this, is easy to conclude that despite being unable to substitute humans in the transfer of knowledge, machines can help reduce the volume of human interaction necessary, doing the heavy-duty work and lowering the impacts associated with any possible increase of demand of said knowledge.
4 Background

To fully understand Watson Services once they are presented, it is vital to comprehend some of the key concepts related to them. Therefore, the next pages introduce some of those concepts for the readers consideration:

4.1 Cognitive computing:

Though there is no formal definition of cognitive computing, it is widely accepted as cognitive any software or hardware component that imitates the behaviour of the human brain.

Despite being originally considered a new paradigm of programming; cognitive computing seems to be evolving into a full new era of computing (coming after the tabulating machines era and the current programming era).

Many experts in the field\(^5\) believe that the era of the cognitive computing is deeply intertwined with the growing level of interconnection between humans and machines. Thus, seeing how the latter has grown in the past few years and continues growing today, it is only logical that the former keeps gaining presence proportionally.

This new era of computing might change the way people interact with machines, relying more and more on them to handle the immense amount of information that everyone must deal with on a daily basis.

The more akin a computer can get to the human way of processing information the easier it will be to make the transition between the two, increasing the amount of tasks humans can delegate to computers.

\(^{5}\) See Dharmendra S. Modha – Cognitive Computing (item 1 of the bibliography)
4.2 What is Watson:

Watson is the answer given by IBM to the challenge of making a computer capable of “thinking” like a human. Its name is an homage to the late Thomas J. Watson, the first CEO of IBM.

Watson was originally designed as a super-computer capable of processing human language in a form never seen before. With its incredible computational capabilities, it was able to gather and process a truly enormous amount of knowledge, gaining at a time some context on practically any conversation in which it could take part.

On the year 2011, Watson’s capabilities for analysing language were put to the test against human competitors in the American quiz show Jeopardy.

With 100 practice games against past Jeopardy winners, on February 14, 2011 Watson won an exhibition match against the all-time champions of the game Ken Jennings and Brad Rutter.

On November 2013, following the Jeopardy success, IBM made public its intention to make Watson API available for software developers.

After that, a Business unit was created to continue working on the Watson initiative, making progressively more of Watson functionalities available to the general public via Bluemix, improving those functionalities and even developing new ones such as Visual Recognition capabilities.

4.3 Deep Language processing:

Whilst other language recognition systems focus on the more traditional approach to the task (referred to as “Shallow Language Processing” or SLP in this document), Watson is based on a newer paradigm called Deep Language Processing.

---

6 See index 4.3 - Deep Language Processing
7 Former name of IBM Cloud
Shallow language processing relies on strict sets of rules and pre-defined patterns in order to extract the meaning of the input text. This approach, though very accurate in terms of the literal meaning of the phrases, loses all its prowess when dealing with complex phrases with more than one correlation between subject and verb (and/or qualificatives).

To illustrate this behaviour, when presented with the sentence: “I tried out the new Cafeteria, it is great to be able to have lunch at the campus again”, a SLP based system would pick up the Cafeteria (generally its brand name) and the qualificative great, and assume that great refers to the quality of the food and / or service of the cafeteria, thereby losing the real meaning of the sentence.

To counter this effect, Watson aims to incorporate an abstract concept into the equation: context.

Gathering information beyond the “building blocks” that make the sentence is the only way of identifying any meaning not literally conveyed in the expression; it is the only way to understand “the context”.

But, how does Watson gather context about the phrase? Firstly, it tries to recover as much information as possible about the text (finding meaning for the abbreviations…). After that, it searches on its knowledge base, targeting the specific subjects relevant to the text at hand, hoping to find matches and information not directly conveyed on the phrasing. Then, it gathers all those pieces of information, and makes a decision. This process is called “searching a corpus of information”.

4.4 Watson’s decision making:

Stemming from the use of Deep Language Processing comes a problem: Once the system has started retrieving the corpus of information, it is not uncommon that it finds different possible interpretations for the phrase it is analysing. Faced with this dilemma, Watson must make a decision, opting for one of the meanings and processing the information accordingly.
Having in mind the fickle nature of language, and even though the decision must be taken among a defined set of options, seems like the resolution can never be categorical. As such, instead of giving a yes or no answer, Watson relies on Confidence percentages (that can in most cases be tweaked by the developer) to choose the path to follow.

The next scenario could be an example of Watson’s decision-making process:

Say that using Watson technology, a company has made a system that, being presented with a photo of either a bike or a bicycle, is able to difference whether the item shown in the photo belongs to one group or the other. Now, bear in mind that bikes and bicycles are not so far apart (both have two wheels, a handle, etc). If the Application is correctly trained, upon “seeing” a photo of a chopper\(^8\), it should settle around 60% confidence on bike and 40% bicycle, concluding at last that it is a bike.

\(^8\) A type of bike whose frame resembles that of a bicycle from some angles
Second chapter:
Theoretical Work

5 State of the art:

5.1 Watson nowadays:

Seven years after its participation in Jeopardy, Watson’s functionalities have expanded and diversified, giving birth to several services\(^9\) that improve upon different individual aspects of the original system.

Exceeding its intended reach, Watson has evolved from a single super-computer to a set of cloud based APIs and applications that can be easily accessed by anyone and, in many cases, can be used free of cost.

The use cases of Watson have also expanded, ranging nowadays from information classifiers to emotional analysers; adding even possibilities such as the development of visual recognition solutions, all based on the same cognitive principle.

It is important to remark that, even though most of the services originated from Watson are native to the IBM Cloud, there are a few such as Watson Explorer that must be deployed directly on premises (on a Power Based system for example).

\(^9\) Understood in the sense of Systems architecture, that is: sets of functionalities with common purposes that can be used by different clients and repurposed by them.
5.2 Introduction to the services of Watson:

5.2.1 Watson Assistant (Former Watson Conversations):

Watson Assistant is a service that allows developers to design virtual assistants (commonly known as chatbots) as a standalone feature or as part of a greater system (usually solutions designed using other Watson services).

The assistants designed with this service have the advantage of using Deep Natural Language Processing to analyse the input seamlessly; incorporating the state of the art technology of Watson to the process.

For a detailed explanation of this service, see Chapter 2, section 5.

5.2.2 Watson Discovery:

Watson discovery is a service conceived as a cognitive information engine. In other words, its purpose is to give structure (classifying, finding patterns, etc.) to the information it receives. This structure is built with the idea of making the knowledge easier to understand, or to garner value from previously opaque content (be it due to its size or complexity).

Furthermore, Watson discovery allows the developer\textsuperscript{10} to improve the final result by training the service on the specific Corpus of knowledge relevant for the target business (see section 4.2.4 – Watson Knowledge studio).

Regarding its complexity, even though it is highly tuneable (and when done correctly can multiply the value obtained from the solution) Watson Discovery has been designed to yield high value results with little effort and a bare minimum of knowledge about Watson technologies and any of the AI practices it uses.

\textsuperscript{10} The client company creating a solution using the service
Being able to handle (and successfully combine) both structured and unstructured data, one of its main uses is to process natural language queries to return any information relevant to the subject of the query. This functionality can be put to great use when Watson Discovery is paired with a way to interact with the user (such as an instance of Watson Assistant).

An example of its potential, would be as a question answerer for a given subject:

Imagine that the subject of Roman Laws, due to the vast amount of knowledge it requires, ends each academic year with complaints from the students related to the obscurity of the texts.

To solve this problem, a development team decides to create a virtual assistant to retrieve given passages of any of the different laws the students may ask about.

Traditionally, the development team should start by gaining insight about roman law, and then proceed to dissect the laws and program them into the assistant\(^\text{11}\). After that would come the testing phases and so on.

This process can radically be shortened using Watson Discovery, resulting in the following steps:

First, the development team should create a simple assistant (with basic functionalities, but without “teaching” it any specific knowledge about roman law).

After that, the team would need to set up a Watson Discovery instance and raw feed all the texts in accepted formats (XML as an example of structured one and PDF as unstructured for example).

Lastly, they would connect them in such a way that each time the Assistant would be asked, it would call Watson Discovery, which at a time would query the requested passage among its corpus, returning the results back to the assistant and on due time to the user.

Note that the advantage comes in the time saved by avoiding having to learn and understand the specific corpus, and also skipping the process of program it into the assistant.

\(^{11}\) A chatbot, not necessarily Watson Assistant.
5.2.3 **Watson Knowledge Catalog:**

Watson Knowledge Catalog’s name is quite self-explanatory, although with a tweak. Chiefly, as its name promises, it can serve as a container for many kinds of data; guaranteeing of course data compliance with any laws regarding the management of information. Its mission in this regard would be to shape any data “owned” by a company into an asset\(^\text{12}\) for that company, making it easy to access (via categorization), and allowing the definition of custom policies regarding particular sets of data.

But, the tweak is that its potential does not end with written data. Knowing the increasing relevance of AI models, and the volubility of the models in production, IBM designed Watson Knowledge Catalog in such way that it is able to store and categorize a vast array of machine learning (& deep learning) models for safekeeping, tracking or sharing purposes.

Both the “written data” and the AI models hosted by this service can be recovered and put to use with ease by other Watson Services; making Watson Knowledge Catalog the most convenient storage for solutions mainly composed of Watson Services.

5.2.4 **Watson Knowledge Studio:**

Watson Knowledge studio is a service designed to teach Watson the language domain of any given business. On a practical level, this means defining custom entities and relations (called artifacts) unique to the context on which the service will be deployed.

To illustrate the functionality of this service, let’s suppose that a particular instance of Watson Knowledge Studio must analyse feedback from students of each of the UPM’s campuses and determine the correct campus to which each comment should be sent.

\(^{12}\) In the sense of valuable property
To be able to distinguish between each of the campuses, we should teach Watson (via Knowledge studio) to recognize “Campus Sur” (for example) as a valid campus name, but given the wide array of names that students might use to refer to this particular campus, it would probably be productive to include some other synonyms such as “Etsisi” (for considering it a part of the southern campus).

With all the new entities pinpointed, the comments might then be sent to another service for further processing. The next service, would then be able to carry out its normal duties with the added functionality of being able of distinguishing between the different campuses of the UPM.

Given the limited value this service can produce on its own, Watson Knowledge studio is best used in combination with other services, such as enriching the domain of a Watson Discovery instance, increasing the accuracy of Watson Language Understanding (through custom annotation models), or widening the range of Watson Explorer.

5.2.5 Watson Language Translator:

Designed as an easy way to traduce a great volume of documents as quickly and reliably as possible. Watson Language translator was meant to eliminate language barriers from data: unifying all the data relevant to a business in the same language, or making it accessible (through translation) to any eligible user.

Apart from translating texts, Watson Language translator is also capable of identifying the language in which a given text was written. It was designed to identify even some of the languages that it is not able to translate to/from.

Similar to Watson Knowledge studio, one of the best ways to “squeeze” the most out of this service is to use it as a link inside a chain of other services.
5.2.6 Watson Machine Learning:

Watson Machine learning is a set of APIs which can be called from almost any application to apply machine learning techniques without straining the resources available on premises.

Similar to other Watson services, Watson Machine Learning was built around a visual toolkit to allow almost any user (even those without much machine learning knowledge) to take advantage of its functionalities. In any case, it also supports many of the most common machine learning frameworks (TensorFlow, Keras, Caffe, Pytorch, Spark MLlib, Scikit learn, Xgboost and SPSS) for users already familiarized with their use. The idea behind going from purely visual editing to the most intricate of designs seems to be to satisfy the demands of the widest possible crowd.

One of the main advantages of using Watson Machine Learning is the readiness to comply with Data policies, being able to connect with several types of data stores (30 at the moment\textsuperscript{13}) as a part of IBM Cloud.

The other main advantage, being able to divert the computational peaks, though inherent to all cloud based services, becomes especially relevant in scenarios such as the use cases of Watson Machine Learning. This is due to the high computational strain to which most models are subjected during training or peak moments of use.

Being free of the necessity to purchase expensive hardware equipment to cope with the peaks on demand (High availability systems, load balancers…) companies can save enormous amounts of money; paying accordingly to the use they make of the service instead of having to pay to even be able to set up the solution.

There is yet another great feature unique of Watson Machine Learning, though it requires a certain level of technical expertise to really exploit it to its full potential: integrating Watson Machine Learning with other Watson Services.

\textsuperscript{13} May 2018
When connected with Watson Knowledge studio, Watson Machine Learning becomes able to classify and safely store lots of different models, allowing for precise control over the AI models; and when paired with Watson Studio, it becomes a part of a unified environment, simplifying the steps needed to take advantage of the results the machine learning (or deep learning) models produce.

5.2.7 Watson Natural Language Classifier:

Watson Natural Language Classifier is a service which, given a phrase, is able to dissect and classify its content; and then, return the results. These results come mainly in the form of the different classes identified in the text, along with the most plausible intentions ranked by percentage of confidence.

To exemplify, suppose that through an application (whose role is that of a secretary), Watson Natural Language Classifier receives the following sentence: “What time can Mr…. meet me?”. In order to reply, the application needs to receive feedback from the Natural Language Classifier service, which would (if trained correctly) infer the intentions of the sentence: Firstly, by deducing that is a question; then by identifying what is being requested (in this case information about gaps on a certain schedule); after that, whose schedule it is, Mr….’s.; finally communicating its conclusions to the calling application.

The main feature of this service is its ability to be trained and classify information in multiple languages, being able to keep up on a variable environment.

This service is commonly used in the generation of chatbots (not based on Watson Assistant), for being able to assume the all the computation related to the understanding of the inputs; leaving only the responses and the communications to the chatbot itself.
5.2.8 **Watson Natural Language Understanding:**

Watson Natural Language understanding exists to extract insight from both structured and unstructured text. One of its main goals is to deduce the intention of the text at an emotional level.

Basically, it distinguishes among different concepts, highlighting keywords and semantic roles; identifying entities and finding relations between them; and presenting conclusions about the sentiment and the emotions intended on its writing. Furthermore, this service is able to associate various emotions to specific keywords on the text, increasing the depth of the analysis.

The full capabilities of Watson Natural Language Understanding can be unlocked though integration with Watson Knowledge studio, which allows for custom entities and keywords to be recognized and to extend the dominion of the solution to thirteen different languages.

5.2.9 **Watson Personality Insights:**

Watson personality insights has two main uses, which will be addressed separately:

On the one hand, this service can, given enough text written by the subject of the study, formulate hypothesis about the personal traits of the author (based on the patterns Watson has found studying thousands of individuals and their productions\(^1\)). The main objective being getting to know a specific person better prior to meeting them (very useful in business negotiations, for example). The text feed to the service may come from a formal written piece (such as an essay) or from casual text (for example social media content), with varying levels of accuracy.

On the other hand, instead of focusing on a single individual for a detailed study, Watson Personality insights can be used to analyse text from a wide array of subjects (mainly from social media) to extract common interests, buying patterns, etc.

\(^1\) All this work was done by IBM and does not require any user action.
The main market application of this service nowadays is either the development of highly targeted content (product recommendations, custom offers…) or the leverage of the conversion rates of an isolated segment of the market\textsuperscript{15}.

5.2.10 Watson Speech to Text:

Watson speech to text is a service whose main purpose is, as its name suggests, take an audio recording of a speech or a conversation and generate an accurate transcription.

Some the most notable features of this services are:

- The use of machine intelligence to combine grammar with language structure and context of the conversation to derive a fitting transcription
- The accuracy of the analysis of audio signals, which allows for a sharper transcription of the conversations.
- The possibility to recognize and operate on several languages such as (English, Spanish, French, Portuguese, Korean, etc…)

5.2.11 Watson Text to Speech:

The twin of Watson Speech to Text, this service is the polar opposite of the previous one. Instead of taking a conversation and generate a transcription, it makes audio files from the texts it receives. It is widely used on systems without a visual interface or designed for visual impaired people.

As will be shown on the chapter dedicated to success stories, both Watson Text to Speech and Watson Speech to text tend to be used together as the nexus between the user and the main system, giving the users the chance to interact with Watson as a verbal interface.

\textsuperscript{15} In this scenario acting more as a thermometer rather than as the lever itself.
5.2.12 Watson Tone Analyzer:

Watson Tone Analyzer was meant as a way to process the emotions imbedded in day to day conversations, distinguishing between the raw emotions (anger, disgust, fear, joy and sadness) but also indicating socially meaningful qualities (openness, conscientiousness, extroversion, agreeableness…) and language styles (confident, analytical…).

In a similar way to Watson Natural Language Understanding, Watson Tone Analyzer is able to work with any size of document, from small sentences posted on social media to Full Scale Documents and even books.

This service can be used to listen to social tendencies about certain topics or improve the customer service experience for example.

To illustrate the latter use, suppose the Etsisi wants to implement a feedback system to each class. It is important though, to make sure no destructive content makes its way into the system. An instance of Watson Tone Analyzer could be set to filter the inputs, signaling any comment whose main emotions are anger or disgust or whose style is akin to a set of predefined patterns.

5.2.13 Watson Visual Recognition:

Watson Visual Recognition is a service which uses Deep Learning mechanism to analyse images searching for visual clues. In can be trained to work in a wide variety of scenarios, but the service comes with several predefined models for specific purposes. Each of these models has been configured to better suit the requirements of each of their use cases. As of May 2018, the models available are:

- General model: Used to identify certain traces on images associated with predefined classes of keywords\textsuperscript{16}. Its main purpose is to describe the contents of the image for classification.

\textsuperscript{16} i.e. identify certain pre-established objects on images.
- Custom model: The custom model allows for the generation of unique visual recognition tools relevant to the specific context of the visual classifier, or to identify concepts that are not supported by the general model. In other words, it is used to design classifiers that do not fit into the other models.

- Face model: The face model has been designed to identify all the human faces contained on the images it is shown, plus determining the gender of each person and approximate the range of age.

- Food model: This model was trained specifically for the food domain. It can recognize food items and dishes with great accuracy. It is important to note that this model is still on beta state.

- Explicit model: The explicit model was created to detect sensible / adult content on the images, acting as a filter wherever one is needed.

- Text model: The purpose of this model is to process text found on natural scene images (an angled photo of a credit card for example). Sadly, this model is still on closed beta.

### 5.2.14 Watson Studio:

Watson studio is a service whose purpose is to accelerate the process of development and deployment of artificial intelligence solutions and the learning curve of the developers.

To achieve this, Watson Studio offers a wide variety of tutorial samples and repositories of previously designed models. The models designed inside the confines of an organization can also be shared among the organization.

As an additional feature, it also includes visual tools that enable the design of AI models without the need of coding a single line.

Watson Studio is based on Open Source applicative such as Jupyter Notebooks and supports many of the popular IA related libraries.
The main advantage of this service is its ability to connect and upgrade the functionality of other Watson services; adding at a time an elegant way of presenting the results of the processing (via graphical display).

5.2.15 **Watson Explorer:**

Watson explorer is a service designed to explore and analyze the content of natural language data. It can process both structured and unstructured data to get as much side information as possible about said information.

The goal with this service is to recover and group all the information generated, presenting it in such way that helps with the process of decision making, hopefully saving costs related to analyze data by hand or using a shallow language analyzer.

It is important to know that, unlike most Watson services, Watson Explorer must be deployed on client premises (using a power based system, for example) locally or as part of a private cloud (mainly IBM Cloud Private or ICP).

5.3 **Other IBM Cloud services:**

5.3.1 **Node-RED:**

Node-RED is a programming tool designed to integrate hardware and software. Node-RED is based on a visual toolkit and a palette. The palette is composed by a myriad of nodes that either add functionality or represent one of the components that needs to be integrated. It is important to mention though, that to be able to add a component to the solution, its node must exist either on palette or as part of a plugin. Anything from API’s to online services, cloud… can be connected in a few steps as long as there is a node for it.
It is important to know that Node-RED is independent from IBM, and it is mainly used due to the simplicity of its basic functionalities, being able to integrate a solution involving different systems with little to know effort. For the integration of complex solutions, or performance heavy ones, it would be advisable to change to another integration platform, such as IBM API Connect or IBM App Connect.

5.4 Success Stories:

To understand the use that can be made of the Watson Services, IBM offers a list of “success stories” or, in other words, a list of real use cases in which Watson provided great value to the customer. Following this paragraph comes a selection of those cases, chosen either by impact, lateral thinking of relevance.

5.4.1 Watson and the Spanish Tax Agency

On the year 2017, IBM announced a partnership with the Agencia Tributaria\textsuperscript{17} in order to develop an assistant tasked with helping the Large Companies in dealing with the IVA (a tax applied to every product by the Spanish Government).

To be specific, there is a particular ruling regarding the IVA that forces the company to provide information to the Tax Agency within 4 days of the emission of each ticket.

It seems that this ruling, known as the SII\textsuperscript{18}, has generated a wide array of individual situations that require further instructions.

To cope with this problematic, a Virtual Assistant was made to cover as many of the doubts as possible, giving a breath of air to the team responsible for answering questions regarding the SII.

\textsuperscript{17} The Tax Agency of the Spanish government
\textsuperscript{18} Sistema de información Inmediata in Spanish
One of its most interesting features, is the ability to validate the authenticity of any conversation held with the assistant. This means that any answer given by the assistant to a company exempts said company of any further responsibility if correctly followed; such is the level of reliance of the assistant. To accomplish this, each conversation generates a special code that is stored by the Tax Agency and in the conversation file given to the company. If on a later date, the company is required to show the conversation (due to an infraction) the codes are compared to see if they match, thus verifying the authenticity of the conversation.

The assistant was trained both by members of IBM and the Spanish Tax Agency, and was released into production satisfying only a few requirements that have increased over time until its span covered the whole SII dominion.

Now, the assistant gives service 24 hours a day, 7 days a week and 365 day a year, offering answers an any point of the year.

When it first entered production in July 2017, its reception was quite timid with a little over 200 questions a week, but its popularity has risen to a peak of more than 2000 on November of the same year.

Since it entered production, the assistant has managed to reduce the doubts reaching the team by an 80% (from 900 messages a week to 165).

Part of the success of the assistant comes from the finesse put into the design of the Watson Assistant service that forms its core, but it was not the only factor.

It is nigh impossible to code an entire system of dates (specially one as complex as the one needed for this use case), knowing this, the development team from IBM designed a date calculator to support the assistant when dealing with time, letting the users know when do they have to take certain actions for example.

Another turn point for the performance of the assistant was the decision to introduce a set of possible inputs as a way of answering the questions of the assistant, instead of letting the users type whatever they wanted. This decision undoubtedly helped to reduce the learning curve of the users, which mainly want to know as fast as possible what action they need to take and when must they take it.
5.4.2 Woodside’s knowledge pool

Woodside is one of the world’s largest oil and gas companies. As such, one of the day to day problems they have to deal with, is the acquisition of new talent and the passing of the accumulated knowledge of the senior members of their teams.

One of their biggest concerns in this regard was the solution of the hazards natural to their trade. Dealing with quite dangerous materials, and needing as much precision as possible, the company delved into the problem-solving process.

They found that up to 80% of the lapse between the occurrence of the problem and the solution was dedicated to search for the information needed, and only 20% was actually spent on solving the problem.

Therefore, they decided to create a knowledge pool condensing the information gathered during their decades of experience and to make it available to their employees. To carry out this task, they enlisted the help of Watson:

First, Woodside gathered input from all across their staff, going as far as asking retirees for their opinion on certain matters.

Then, they collected the information (more than 600,000 pages of it) and feed them to a Watson fueled machine learning solution.

After that, they trained and tested the machine learning models in order to be able to identify how past experience can be useful to present deeds.

And finally, as it happens with most AI solutions, they released it and have been improving upon it ever since, tuning it from feedback given by their employees.

Nowadays, Woodside claims that the time employed of gathering meaningful information has been reduced by 75% thanks to the help of Watson.
5.4.3 Guiding eyes solution:

Guiding eyes is a company dedicated to the raise and training of dogs for visually impaired people. The problem they faced is the prohibitive cost of this process (up to 50,000 dollars for each dog) and the relatively low success on doing it, successful only in one out of three cases.

The objective here then, is to improve the success rate as much as possible. Starting from the company’s archive, lots of information about dog genetics, medical records, trainers, etc. were pulled out and fed into a Watson solution.

Having more than half a million inputs between medical records and temperamental records, Watson tries to pair each dog with the right owner.

With the help of Personality Insights and Natural Language Understanding, Watson tries to gain insight about the future owners to better find the perfect match.

Reports from the company affirm that the solution has helped rising the success rate of dog upbringing by up to 20%, just by properly managing and analyzing the vast amount of data they already had.
Chapter 2: Watson Assistant:

6 Detailed explanation: Assistant

6.1 Introduction to Watson Assistant:

Watson assistant (formerly known as Watson Conversations) is one of the services offered by IBM S.A. through IBM Cloud. Its purpose is the design of virtual assistants (referred commonly as chatbots) centered on a specific domain, and with a predefined reach.

The assistants designed using Watson Assistant (without integrating it with other Watson services) can conversate and answer questions within their domain with predefined answers (outputs) written by the development team.

These assistants have a wide variety of uses, some of the most prominent are:

- Allowing users to easily access information that was either contained in obscure documents or too tedious to find.

- Help the users grasp a set of concepts explaining the hardest bits (via question answering) without having a human expert interacting with each user separately.

- Guarantee the same correct answers for the same questions (especially relevant to domains related to laws).

- Improving the user experience when doing tasks such as filling out forms, by making the experience interactive and providing feedback.
- Serve as the interface for a larger solution (an assistant integrated with discovery, for example).

Even though this service can be deployed as a standalone solution, it is important to note that it will need at least to be integrated with a front end in order to communicate with the user.

6.2 Regarding conversations:

Starting from this point, any interaction between the user and the assistant will be referred to as conversation. These conversations will likely take the form of a series of questions and answers sharing a single common thread.

If, by requirements of the user, the interaction must encompass more than one particular matter, and in order to avoid confusion, they will be considered different conversations to all effects.

Determining the outcome of a conversation is crucial to the design process of virtual assistants, as it pinpoints the weak points of the conversation tree that need to be improved. Any conversation (even if part of a larger interaction) will have one of three possible outcomes:

- Successful: A conversation will be considered successful if the assistant is able to answer with accuracy, giving the user the information he was looking for.

- Unsuccessful: If the assistant is able to respond to the user demands, but the information given is either irrelevant to the matter or wrong. A conversation will also be considered unsuccessful if the assistant is unable to provide an answer to a question or gets stuck and is unable to carry on with the conversation.

- Undefined: If the conversation ends but it is unclear whether or not the conversation was successful (if, for example, the user does not

Note that Undefined should be (at least during the training process) a temporal state. Each undefined conversation should be reviewed by a human supervisor to determine if it was successful or unsuccessful.
6.3 About workspaces:

Depending on the pricing plan, the development team may have up to a certain amount of workspaces to work in (the basic plan includes up to five of them).

Each workspace is independent from the others and can contain different solutions. However, regarding the development of complex solutions, it is not unusual to see different workspaces containing different versions of the same assistant19. For example, the development team may set up the following environments (all dedicated to the same assistant):

- an environment for development with continuous deploys.
- A pre-production environment, with periodical deploys from the changes made in the development workspace.
- And finally the production workspace, with controlled deployments coming from successful testing on pre-production.

![Figure 1 - Default workspaces view](image)

6.4 About Intents:

An intent is the ultimate user’s goal behind each of his messages. In other words, it would be “the action that the user wants to communicate”. Examples of intents would be:

19 See section 6.10 - Creating Several Environments
- Saying or negating a statement
- Asking for the hour
- Ordering the assistant to open something

Any intent defined by the development team must contain a name (which will always start with #) that cannot be changed once the intent is created; and a set of user examples that will serve to train the assistant to recognize the intent. Intents may also contain a description to elaborate the use of the intent if the development team considers that the name falls short on this matter.

In order for the assistant to properly recognize the user's intentions, every intent must be defined in such a way that there are no overlaps and with as little ambiguity as possible. Even though this is a task for the development team, Watson Assistant comes with an integrated tool that detects overlaps to some degree, preventing the user from creating meaningless intents.

### 6.4.1 User examples:

User examples are sentences introduced by the development team as examples of possible user inputs that should be recognized as the intent at hand by the assistant.

Once introduced, the assistant trains itself for a few moments. After that, it should be able, if the examples introduced were good enough, to recognize any sentence introduced by the user referring to that intent as such.

The number of user examples that Watson Assistant needs to learn to recognize an intent is surprisingly low, but it is important to remember that the user examples must be as varied as possible.

First, the development team must think about all the possible ways in which the user might refer to the intent; and then input them as user examples.
Since everyone speaks differently, it is more than likely that some of those ways manage to elude the development team; but not to worry, the assistant can be improved afterwards as it interacts with the users, needing only a few seconds to retrain itself.

To minimize the improvement effort of the assistant, it is commendable to add as much variety as possible to the user examples. To illustrate, imagine that the assistant must be trained to recognize the intent of “turning off something”. The following are a set of some good user examples:

- Watson, please, can you turn off …...?
- Turn off ....
- Switch of .... 
- Shut down ....
- Halt ....

The key factor here is to express the command in as many different registers as possible, including different verbal voices. Sometimes it is very hard to know how wide a range of users the assistant will face.

---

20 For more information about this see section 6.9 - Improving the Assistant
6.4.2 Visual Guide for intents

The image above shows a view of the default intents perspective. The most prominent areas are highlighted in different colors and will be reviewed one by one:

Inside the red square is the main panel of the perspective, which is common to all the views of the assistant. From this panel, the developer can change the workspace and the view of the assistant.

Squared in blue is the button to add new intents. Its functionality is pretty straightforward and does not require further explanation.

Highlighted in green are the options to import, export and delete intents. The first two serve to bring previously exported intents (usually from a different solution), and to export said intents respectively. The delete option permanently erases (after asking for confirmation) all selected intents.

Finally, inside the orange line, every intent inside the assistant is listed by name, along with the latest modification date and the number of user examples contained inside each intent.
6.5 About Entities:

Entities are specific words (or sets of words) that give a certain sentence a distinctive or special meaning. Frequently, they are subject or the sentence or the objective of the action, and the tend to be substantives.

Each entity must have a name, which starts automatically with the symbol “@” (for organizational purposes); one or more values\(^{21}\), and can include a description to help clarify the content of the entity if necessary.

Contrary to intents, which are recognized based on patterns (learnt by the assistant through training), entities can only be recognized if they are introduced by the user literally. This behavior was programmed to avoid confusion between entities and more specifically between different values of the same entity (which can be very similar).

Identically to intents, entities can also be both exported to other solutions and imported from them.

6.5.1 Values inside an entity:

In the context of entities, a value is a defined and distinctive representation of the concept contained within that entity. For example: “Juan Lopez” would be an example of value for an entity called “People”; and “1”, “2” or “3” would be values for an entity called “natural numbers”.

When working with entities, the developers can set the level of granularity required for each case, being able to set the assistant to recognize just the entity, or make a difference based on the specific value to look for in the user’s message.

One of the most useful features of the values is the ability to apply logical operations and comparisons between them (always between the core names of the values).

\(^{21}\) See section 6.5.1 – Values inside an entity
6.5.2 Synonyms of values:

Knowing the limitations of having to recognize each entity (or more accurately each entity’s values) literally in the input given by the users; the developers of Watson Assistant developed a feature called synonyms.

For each value of each entity, Watson assistant allows for synonyms to be introduced. if one of these synonyms is found in the text, will equally be recognized as the entity value they strain from.

This case contains a good example of the use of synonyms:

Knowing that not every user writes the same way, the synonym “The home of the braves” was added as an equivalent to “United States” which itself is a value of the entity Countries. If this process were to be repeated for each country that had to be recognized, the success rate\(^{22}\) of this fictional assistant would greatly increase.

6.5.3 System entities:

Apart from the entities defined by the development team, Watson Assistant comes with a list of “common use entities” that may be toggled (and thus added to the entities panel) with the push of a button. These entities cover things such as numbers (both in character and written), percentages, currency….

By design, none of the system entities can be altered by the development team of a solution, and its values cannot be toggled separately. Although this feature was added to guarantee the integrity of said entities, there are cases in which this rigidity creates enough problems to render them useless.

It is also important to mention that the system entities have been designed for the English language, and if a team wants to add an entity containing the Spanish terms for the natural numbers, for example, they would need to create a custom entity and introduce the numbers one by one\(^{23}\).

---

\(^{22}\) Meaning the amount of successful conversations compared with the total number of conversations

\(^{23}\) More about this issue on section 7.11 – Solving the problem with number one
6.5.4 Visual guide for entities:

The entities perspective is quite similar to the intents one in many ways. Apart from all the default panel common to every view of the assistant, the main panel (squared in red), for example, has the same options as the previous one. The area emphasized in green is also identical to the one on the intent’s perspective only this time it refers to entities.

The biggest difference with the other perspective is the area highlighted in blue. This menu allows the developer to exchange between viewing custom created entities and pre-defined entities (system entities).

Once inside the system entities tool, the developer can activate and deactivate said entities clicking on the corresponding switch.

Finally, if the perspective is centered on custom entities, a list of every entity currently available (save for the system ones) is shown. This list contains the names, descriptions (if inserted), the date of modification, and most important, a list with the values of each entity.
6.6 About the Dialog tree:

The dialog tree is a set of nodes (grouped as a multiple branched tree) that represent all the possible stages of a conversation between the user and the assistant.

Any dialog node must have certain features (though as will be shown later, some of them can be meddled with to better accommodate the solution’s requirements) which are:

- A unique name
- A condition in order to enter the node, denoted by “if bot recognizes”
- An answer to the user’s input, in the form of “Then respond with”
- Indications about what to do next, called “And finally”

6.6.1 Setting up the conditions:

Each component of the dialog tree must have a set of condition that have to be met before the assistant can access the node. This conditions usually consist on the assistant recognizing a pair formed by an intent and an entity within the user’s input.

Forming a condition with Watson Assistant is quite simple. First, the developer must select the first component of the condition (introducing an # will prompt the list of intents, and writing @ will prompt the list of entities).

Figure 4 - Dialog node conditions
If the component is an entity, and a should be a certain value, clicking after the name written will show a list of operators, which can be followed by the name of any value.

![Logical operators inside a condition](image)

After the first component is finished, the developer can add components and logical operators until the conditions required are properly defined.

### 6.6.2 Defining the answer:

Once the conditions are set, it is time to define the answer given to the user when the node is accessed.

Watson Assistant includes the possibility of adding variations to the response, should the node be accessed several times during a single conversation. This variation will always be used in order and will only be reused if the list of unused ones is empty. It is also possible to set the order of the variants to random, in which case, they will be shown randomly in cycles.
6.6.3 Behavior Between nodes:

When the node has given an answer to the user’s input, the assistant must carry on with the conversation. By default, the assistant will wait for the user to introduce another input (this option is called “Wait for users input”). If, by requirements of design the assistant must jump to a specific node (to end the conversation immediately, for example), the option “jump to” will prompt a menu to decide where the assistant should jump to.

6.6.4 Customizing the node:

Some special situations require functionalities that default nodes cannot process; to solve this, Watson Assistant includes a customization node that allows the developer to alter the normal structure and behavior of the node. There are four options for node customization, which are:

- **Slots:** If enabled, the developer can configure different slots of information. Each slot represents a piece of information that must be acquired by the node from the user. When entered, the node will ask the user for information to fill the slot, and will continue to do so until it is filled.

- **Multiple responses:** Multiple responses is a feature designed to simplify the number of nodes involved on a dialog tree. If enabled, allows for the setting of multiple answers based on a single condition. For example, with this option, a node could be entered based on a pair intent – entity; but the answer could be given according to the specific value of the entity.

- **The two options of digressions:** The remaining two options deal with subject changes in the middle of the conversation. In other words, if enabled, these options allow to alter the normal flow of the conversation, changing the topic as required.
6.6.5  Context variables:

Context variables are the answer to the need of storing certain values relative to the flow of the conversation (or fed by the user) for later use.

Context variables are independent from nodes but can be accessed and updated by them. They also can be used as part of conditions.

These variables have a name (Which starts with $ and can be used to access the context variable list on the conditions tab), a type of value and the value itself.

These fields can be accessed by name in the answer field (to provide a tailored answer).

6.6.6  Building the dialog tree:

With the concepts regarding nodes clear, it is time to build the conversation tree. In order to ease the task of building the tree, Watson Assistant offers the development team the chance to incorporate other elements besides dialog nodes.

- **Child nodes:** Child nodes are similar in every regard to normal dialog nodes, with the exception that they must depend of a “parent” node, meaning that they can only be accessed in the conversation flow has entered previously on the parent node and if its own conditions are satisfied once the parent node has ended its functions.

- **Folders:** Folders serve as a way of nesting the tree’s structure to make it easy to the eye. They also help delimitate the different aspects of the assistant’s domain, or even to distinguish between expansions of said domain. Unlike nodes and child nodes, folders only have a name and might have a set of conditions, if the development team want to extract a common condition (like making a common factor in math) from all of them.
All nodes and folders (if not included inside a folder) are disposed in a vertical array hanging from the node that kickstarts the conversation. This information will be relevant in order to understand the navigation of the dialog tree.

### 6.6.7 Navigating the dialog tree:

Navigation of the dialog tree represents is of the nodes/folders that are accessed during a certain conversation. The navigation path would be the order (and the number of times) each node or folder is accessed between the start of the conversation and the closing of the assistant.

Thus, it is very important to understand the rules of navigation in order to design a proper assistant. Fortunately, they are very simple:

- The assistant will always look to satisfy the conditions of the uppermost node, ignoring the presentation node (the one that starts the conversation). If the conditions of said node are not met, it will continue descending until it finds a node whose conditions can be satisfied or the end of the conversation (if there are no suitable nodes).

- Unless the current node has an instruction “Jump to” (which would directly jump to a specific node) the assistant will first try to satisfy the conditions of the child nodes pending form the current node (in descending order), and if it is unable to do so, it will restart the process described in the previous paragraph.

### 6.7 Content Catalog:

The content catalog is a repository of commonly used topics. It contains intentions regarding several dominions of the assistant. The content of each of those dominions can be added to the assistant with a single click. Once added to the workspace, the intents contained within the dominion are incorporated to the already existing ones.

Once added to the rest, the new intents can be accessed, customized and even deleted as required, same as if they were user defined intents.
As a final note regarding the Content Catalog, it is important to mention a few things:

Firstly, the content catalog is intended to serve only as a guide, and not as the sum total of possibilities regarding the dominions included. As such, it should be view more as a template to develop solutions and less as a frame.

And secondly, it is important to mention that sadly, the Content Catalog only exists in English, and if the assistant in question in written to operate on a different language, it is probably better to start the assistant from scratch.

6.8 Testing the assistants:

The Watson Assistant service comes with a built-in tool to live test the performance of the assistant without the need to deploy it. The tool, called “Try it!” allows the developer to interact with the assistant via a graphical interface, and apart from showing Watson’s answer to the input, it shows the different intents and entities detected on the user’s input.

To use the tool, one must only press the button named Try it on the upper right corner of the screen:

![Figure 6 - accessing testing tool](image)
Once opened, it shows a window similar to those of instant messaging, where each of the user inputs is accompanied by the assistant answer and the list of intents and entities detected (see figure 8).

Apart from the normal view of the chat, there is also a page dedicated to monitor the state of all context variables defined in the assistant. To access it, the developer must press “Manage context” on the default “Try it out” perspective (see figure 9).

6.9 Improving the assistants:

During the development phase of any virtual assistant, an even if the development team have a set of final users available for testing, they will only be able to register a certain amount of language permutations.
To be able to achieve the most success in conversations, the assistant needs to be readjusted to fit new-found expressions and patterns employed by the users. In order to achieve this goal, the training process of the assistant is usually extended for a prolonged period of time after the initial release.

In many regards, the improvement process is similar to that of regular software maintenance. The amount of work and effort necessary depends greatly on the resources available, the level of accuracy required, the size and variety of the user base and the reach of the assistant.

To help with this task, the Watson Assistant toolkit comes with a view entitled “improve” that contains lots of information regarding the performance of the assistant. Is it a truly powerful tool, offering detailed statistics about frequency and reliability in the detection of each intent and entity, and a visually displayed ranking of the most common interactions.

This window can be accessed through the panel on the left side of the workspace, marked with a statistic line:

![Figure 9 - Accessing Improve](image)

Once inside the tool itself, one can distinguish to separate views:
The first one is called Overview, and it contains all the information kept by the assistant regarding general metrics of the interactions, and can be grouped in many ways.

The figure above shows the Overview window with its main parts differentiated.

The area marked in blue contains search-boxes to change the information show on the other two areas according to specific searches, along with general metrics about the assistant.

Inside the red rectangle is the main feature of this view, a set of graphics showing statistical information about the assistant or some of its parts. Being a graphical view, this feature helps detecting the worst pain-points very easily.

Contained in the green square are the rankings, showing the top entities and intents ranked by the number of times they show up on the user’s inputs.
The second view is called User Conversations, and it contains a detailed recollection of each conversation held by the assistant while outside of “try it” mode. It is the ultimate tool for determining what went wrong in any given conversation and whether or not the fault was in the assistant.

![Image of User Conversations]

*Figure 11 - Improve - User conversations*

*This view has only one main area, that show an entry for each user input, offering the possibility to open it to see also the answer given by the assistant.*

Combining both views, the development team can easily conclude which of the unsuccessful conversations ended that way because of the assistant and improve upon them consequently, making the solution more accurate each time.

Development teams must not fear this process, because should they include a feature with a negative impact in the assistant, shortly enough it would show up on the statistics and could be fixed without major consequences.
6.10 Creating several Environments:

Like most software solutions, Assistants can benefit from having several environments to safely test changes and new features. What is more, since the assistants cannot control different concurrent versions (each change is committed to the deployed solution as soon as Watson has finished training) it is probably even more necessary.

So, to generate a developer environment, the best option is to create another workspace inside of the same account. That workspace is completely independent, and once the version developed is stable, it can be transferred to the production environment quite easily.

To add a layer of security, both the developer environments and the integration one (should it be needed), can be separated into another IBM Cloud account under a lite license\(^24\). Once the solutions (tested through “try it” are stable) they can be moved by a member of team with sufficient clearance to the account containing the production (and preproduction in needed) environments.

The following diagram details the scenario described above. It shows three accounts: two for developers, free and with only an instance of Watson Assistant; and one containing all the services necessary to deploy an assistant twice (once for production and once for integration/pre-production).

\(^{24}\) Lite licenses are offered for free or at a discount price
6.11 Limitations of the service:

Having reviewed every aspect of the Watson Assistant service, and with the experience of developing several solutions with the service, there are some aspects of its design that must be considered before embarking on the design of an assistant.

6.11.1 Training of users

One of the most crucial points for the success of a Virtual Assistant based of Watson is the training of the users for the use of said assistant.

The problem with most human users is the tendency to interact with the assistant in the same manner as they would with a search engine. For example: many users, when asking for the retrieval of a certain law article would input only the name of the article. Introducing this information, the assistant will detect the name of the article, but most likely it will not know what to do with it. A correct phrasing alternative would be “show me the article….”

Ironically this issue affects mostly the younger generations, which are most familiarized with the keyword based recognition employed by most natural language understanding system (is one of the most popular forms of shallow recognition).

Though this problem can be mended with targeted design (i.e. training the assistant to recognize keyword based patterns), the best results come from a proper use of the assistant.

For example, in the case of the Spanish Tax Agency, an explanatory video demonstrating the proper use of the assistants was created to train the users. It yielded a considerable increase on the success rate of conversations, further proving this point.

---

25 Section 5.4.1 – Watson and the Spanish Tax Agency
6.11.2 Handling inputs referred to a set of entity values:

Being trained to recognize inputs based on contexts, assistants struggle to recognize isolated entity values. These inputs can be found in cases when the assistant must ask the user for clarification about a certain enunciate. In these cases, the users not always enter a value that can be recognized.

To illustrate, when asked whether a user comes from inside of Spain or outside, one of the natural answers would be to enter “outside” which will not, and should not be recognized as a value. Thus, the assistant will fail on its purpose.

A possible solution to this problem is the introduction of answers buttons, limiting the options the users have to answer. Sadly, this is not always possible (due to multiple factors).

Another solution, though less elegant, provides an equally great result. This solution consists on designing a loop that forces the user to keep entering phrases until one of them is recognized.

6.11.3 Developing for languages other than English:

Since all of the pre-defined work advanced in the form of system entities and content catalog was designed for use in English, it can result unreliable when trying to adapt it to other languages due to differences in grammatic or language structure.

Based on the experience of the author of this project, sometimes it is even productive to ignore certain good practices\(^\text{26}\) to better suit the specific structure of the language in which the assistant will be released.

\(^{26}\) In the sense of advice given by the software developer
Third chapter:
Design of the solution

7 Practical Problem

7.1 Presentation of the problem:

The team responsible for managing the external practices of the students at the Etsisi finds itself flooded with doubts regarding the procedure: both face to face, with students coming to see the responsible of the practices; and remotely, either via telephone or email.

The goal of the solution generated in this TFG\textsuperscript{27} is to reduce as much as possible the amount of time spent answering doubts; questions that, in many cases, are very similar to each other,

Nowadays, a document containing the Frequently Asked Questions (or FAQ) is available through the web of the faculty\textsuperscript{28}, but not many students take advantage of it. Upon interviewing with the personnel of the department, it was hinted that many students, either do not know about the existence of said document, or are unable to find it by searching the web of the faculty.

\textsuperscript{27} End of grade Project
\textsuperscript{28} See Annex 2 – Starting Data Set
7.2 First decisions:

It has been decided that, in order to attack this problem, a great solution would be the creation of a virtual assistant. This assistant should be based on the Frequently Asked Questions, and deployed in such a way that it could be easily accessed by the students in need of clarification.

Knowing the limited time and resources available, as well as the scarcity of testing subjects (apart from the developer and some fellow students); it has been agreed that the goal of the assistant will be to solve as many questions as possible, leaving the most complex (or specific) of questions to the personnel of the faculty.

It has been agreed, that due to the previous theoretical study having been dedicated to IBM Watson, the solution resulting from this practical part should also be based on IBM Watson as well.

7.3 Architectural design of the solution:

7.3.1 Regarding the core of the assistant:

Having a predefined set of questions defining the dominion and the reach of the assistant, and being this set of questions of manageable size; it has been decided that Watson Assistant should be the standalone core of the solution. It has also been decided that the knowledge should be fed to the assistant manually. Automating this process (via integration with Watson Discovery\(^{29}\), for example) would probably be counter-productive, skyrocketing the learning curve and the testing required to only save a minimal amount of time to the development team. At any rate, a door will be left open to possible integrations in the future; mainly to avoid hindering the assistant’s scalability.

\(^{29}\) See section 8.2.4 - Integrating with Watson Discovery
7.3.2 Regarding the integration platform:

Not having to perform a small-grain integration of services, Node-RED seems to be the obvious choice to build the integration solution: Not only is it provided in the catalog of IBM Cloud, but also has pre-developed connections to all the parts involved.

To connect with Watson Assistant (and any other service added further “down the road”), Node-RED has pre-built nodes available through the IBM Extension; and to integrate the Telegram front end, the connection nodes are provided with a free bundle30.

7.3.3 Regarding the front end:

Considering that the design of a custom front end is not contemplated on the scope of this project, but needing a way to expose the solution to the final user, it has been agreed that simple as it may be, the front end must be easily accessible by the final users.

To this end, the application Telegram has been chosen to act as a front end. The users will be able to communicate with the assistant through instant messaging, been able to access it from anywhere like any phone number.

In fact, many IBM Cloud based applications (and particularly so Virtual Assistants) are at least tested once deployed through Telegram.

7.3.4 Results of the process:

After choosing between the different options for the development of the solution, and due mostly to organizational reasons, it has been decided that the solution is going to be divided in three layers as can be seen in the figure on the next page:

30 The downloading process is detailed in the Annex 1
The development of these three layers will be sequential, starting from the innermost layer and ending with the Front end.

The third layer stands for the assistant itself. It has been named “Target Layer” due to the assistant being an exposed API (called from the integration solution). The layer consists of a Watson Assistant instance supported by a Cloudant Database, both based on IBM Cloud technology.

The second layer is the integration level. It will be developed after the minimum viable product (MVP) of the assistant is ready. The Node-Red instance along with the integration flow conform this second layer.
The first layer is going to be the front end, and does not require any specific design or development effort. A Telegram bot will serve as the whole of the Third layer.

7.4 Steps prior to the development:

Before starting to consider the development of the assistant, some questions needed to be addressed to avoid possible misconceptions or mistakes during development or testing.

7.4.1 Formalization and classification of questions:

To properly delimitate the dominion of the assistant, the first step was the formalization of the knowledge that was going to be fed to the assistant. Taking advantage of the need to change the knowledge base (i.e. the FAQ document), the chance was also used to change the writing of the answers: actualizing the questions and altering the tense to accomplish a more adequate voice.

7.4.2 First elicitation of requisites:

With an updated list of the questions, it was decided that it was the perfect moment to elicit the actual requisites and prevent any future deviations. After meeting with the responsible of the practices department, some questions (which didn’t originally feature in the list) were added to the dominion of the assistant.

7.5 Development of the MVP:

The first task in the development of the project was the creation of a minimum viable project of the assistant itself.
The MVP only contained a running instance of IBM Watson Assistant with rough recognition of the questions via a small set of intents and entities. The answers were fed to the assistant literally\(^31\), without any customization attending to the formulation of the question.

Once finished, the MVP was tested to look for mismatches or overlaps between the intents. Since there was no need to deploy the solution after each patch, the problems were solved immediately after they were found; until the MVP covered the basic requirements of the Project.

Note that, at this state, the assistant could only recognize doubts formulated with roughly the same words of the original questions.

### 7.6 Enrichment and first deploy:

Having polished the MVP, the next step in the development was the addition of user examples to all of the intents to allow the assistant to recognize a wider variety of enunciates.

For each question, both the writer of these lines and two faithful companions\(^32\) tried to come up with as many phrasings as possible, trying to imagine all the possible scenarios that would require the answer provided with the original question.

This last part of “reversing the logic” is quite important, because the final users will not know which questions can be solved by the assistant, and will try to formulate their own doubts in their own terms.

After the assistant was able to recognize different enunciates for each question, it was made available through Telegram\(^33\), mainly to take advantage of the improvement views (only available once the service is deployed)

---

\(^{31}\) Without considering the voice of the questions that may originate them  
\(^{32}\) Fellow students from the faculty  
\(^{33}\) The process of deployment is detailed on the Annex I
7.7 Addition of a new environment:

After the initial release into production, it was noticed that due to Watsons design (which applies changes to production as soon as they are introduced into the solution) a development environment was necessary in order to safely apply modifications.

Fortunately, the procedure to add a second environment to Watson Assistant is as simple as creating a second Workspace (which can be done in a few clicks) and copying the entirety of the assistant.

In this particular case, it was decided that the development environment was not going to be deployed. Once the changes introduced passed a batch of test through “try it out”, they were moved into the production environment and were further tested, this time through Telegram, along with the rest of the solution (as a whole).

7.8 Second round of questions:

With the assistant already available via Telegram, and the basic requisites satisfied, it was decided (during an interview with the personnel responsible for the base document) that the assistant was still lacking some basic concepts about the dominion.

Resulting from that conversation, several new questions were formulated along with their answers. Once the questions were clearly defined, they were put through the step 7.6 (Enrichment) and then added to the deployed assistant.

7.9 Designing for economy:

One of the main concerns for this project was the potential cost of the solution. To reduce as much as possible said cost, the assistant was designed to use as little assets as possible.

34 In the form of SAAS monthly payment
Since the size of the assistant determines the pricing plan, and being the Entities the limiting factor in this particular scenario, the number of entities had to be trimmed down.

Up until this point there were nine entities, one for each combination of values relevant in at least one scenario. After the cut, only four remained. This was achieved by creating more abstract entities capable of englobing a wider array of values (avoiding always conflicting descriptions).

Even though the goal of this stage was to reduce the cost, the main premise was that the functionality of the assistant should not be hindered in any way.

7.10 Adding new capabilities:

Up until this point, the solution was able to answer to any question given by the user (if they were included in the assistant’s domain), but the interactions that it was able to have were no true conversations.

Without trying to make the assistant a general purpose chatbot (which would have gone way out of the original reach) or making the assistant “self-aware”35, some nodes and branches were added in order to give a sense of depth to the conversation.

This was a clear opportunity to introduce a thermometer of the satisfaction of the users; in this case, through a final question from the assistant regarding its own performance. From this point onwards, in order to know whether the user was satisfied with the information provided or not, it was only required to look (on the improvement view) for the frequency of either answer.

Note that the exchange regarding the satisfaction of the user was designed to take place at the end of each conversation. This means that in the same interaction, the user may have to answer multiple times to this question (one for each conversation).

---

35 Meaning that the assistant is able answer questions about its own nature and its domain
7.11 Solving the problem with number one:

During the development of the practical part, a crucial aspect for the recognition of some questions was based on the accurate recognition of numbers. Luckily, Watson assistant comes with a pre-configured entity that recognises any of natural number, either if written with numbers or letters.

Nevertheless, during early testing, it was discovered that due to the particular nature of the use of the expression “un/una” meaning one, but also meaning “a”; as in: “Un estudiante de informática” meaning “A computer science student” heavily tampered with number recognition (recognizing almost always “one” as the number looked for) rendering some branches of the conversational tree inaccessible\(^{36}\).

Since the functionality lost was minimal, on early stages of development a workaround was implemented to reduce the impact of the issue. Having satisfied most of the requirements, the next item on the list was truly solving this issue:

In order to solve this problem, a new entity was added to the design. This entity, (called “@numeros”) contains the definition of the 40 first natural numbers (more would have exceeded the expected dominion) excluding the verbal definition of “one”. This means that despite having to sacrifice the possibility of recognising the number one on its written form (which given the context is highly unlikely) the assistant in now capable of recognizing numbers once again with relative easy.

\(^{36}\) This issue was discussed on the section 6.11.3 – Developing for languages other than English
8 Conclusions and future work

8.1 Conclusions:

8.1.1 Conclusions derived from the theoretical study.

After having the opportunity to talk with several experts in Watson technology and having tried out most of the services that conform Watson, the first idea that comes to mind is the gigantic, almost unmanageable, size of the platform. To fully understand the possibilities presented by each service on its own would require full time dedication for an extended period of time.

Even imagining all the use cases that can emerge from the use of the services as standalone solutions is staggering, but being designed as a "Brick Building" platform (where each service represents a brick), the applications become nearly endless.

Due to the lack of depth of most of the POTs\textsuperscript{37} (Proof of technology) undertaken by the author of these lines, it is very difficult to give an objective and accurate opinion about most of Watson’s services.

That being said, and based on present experience, all of the services have delivered what they promised.

One of the points that rose more skepticism was the true complexity of the toolkits (despite the claims of the company) and the level of technical expertise required to create a viable solution.

To be fair, it is true that to get the most out of a service, one must have a high level of expertise (some functionalities are not easy to deduce and/or find); but it is also true that to build a working solution only a few hours of learning (even self-taught) are in most cases more than enough to develop a satisfactory solution.

\textsuperscript{37} POC (proof of concept) or POT both refer to guided tutorials reviewing the basics of a technology or a tool.
Continuing in the same line, the videos provided by IBM through various social platforms helped to significantly shorten the learning period required to be able to work with each service. These videos, which range from visual introductions to full demos of the services (and even videos dedicated to showing solutions to common errors and misconceptions) truly helped when dealing with difficult scenarios.

### 8.1.2 Conclusions of the practical experiment:

After the development of the Virtual Assistant (that was the objective of the practical part) several inferences become obvious:

Firstly, that at first sight it is very difficult to decide which services should be involved in the development of a non-trivial solution. For instance, in the case of the problem with the practices of the Etsisi, the first choice was to create a Watson Discovery driven solution, integrated to Watson assistant (only in this case, the instance of the latter would not contain any knowledge per se, and would need to rely in the former to give any answers). Though from a theoretical standpoint this solution may seem more adequate (considering ease of scalability and retrain), if chosen, would have probably proven to be too costly, both in effort and in monetary cost (because of the added cost of the API calls to Watson Discovery). Ironically, due to the wider scope of Watson Discovery, it is more than likely that the accuracy of the solution would have also diminished in such a specific context.

And secondly, once working hands-on with the development toolkit, it became apparent that despite being a visual builder, there were still many possible solutions to each of the problems encountered, but not all of them were equally efficient. Unfortunately, the only way to determine the way to proceed in each situation was though trial and error. In any case, once encountered, if a similar dilemma appeared, the decision was almost immediate.

So, to summarize: even though developing a working Virtual Assistant with Watson is a relative easy task, adequately doing so is not so simple. Moreover, addressing some of the problems inherent to the development of any assistant beyond a simplistic chatbot requires a certain “know how” that can only be achieved though experience in the development of said Assistants.
8.2 Future Work:

Regarding the future of the solution proposed, there are several courses of action that could improve substantially the impact of the solution. Before deciding to undertake any of the courses, it is important to know that some of them are mutually exclusive, so the decision should not be taken lightly.

8.2.1 Continue training the assistant:

This is possibly the most obvious action to take, and it is also the one that could have a greater impact without having to renounce to any other course.

The idea, as described before\textsuperscript{38}, is to continue training the assistant beyond its production release, incorporating to the system any commonly occurred phrasing not already within the domain.

For example, nearing the end of each academic year, the person in charge of maintaining the assistant could make a quick review of the unsuccessful conversation and add to the user examples of the intents any phrasing that was previously missing.

8.2.2 Updating the knowledge of the assistant:

To be of use to the students, and to avoid giving wrong information, it is crucial to keep the assistant coherent with any changes that affect the rules of the external practices.

In fact, this measure goes beyond future work, because even when the assistant ceases to be improved, it would still need (for as long as it remains in production) to have updated and relevant information.

\textsuperscript{38} See section 6.9 – Improving the assistants
8.2.3 Expanding the dominion of the assistant:

If the assistant finally reaches a productive state, and it turns out to be a valuable tool, it would seem logical to expand its dominion: increasing the amount of questions it can answer, and covering even more topics regarding the practices.

To expand said dominion, more intents and entities should be added as required, and the dialog tree would subsequently be expanded to meet the new requirements.

Note though, that an expansion of this sort would not require extensive testing, since Watson assistant can be easily modified even while remaining in production.

8.2.4 Integrating with Watson Discovery:

This option is an alternative to the solution presented in the previous entry (8.2.3) as it serves the same purpose although with different methods.

Should the expansion cover many new topics, or information relative to strict sets of laws, the development team could explore the possibility of adding the analytic power of Watson Discovery to the solution.

With Watson Discovery, all the new knowledge would not need to be manually taught to the assistant, instead it would only need to be introduced into the Discovery instance.

Once correctly integrated and trained, the new solution would transform the user’s questions into Discovery queries, showing the user the information retrieved.

As a final note to this action, the increase in monetary cost associated with the use of a Discovery instance should also be considered before committing to the implementation of this option.
8.2.5 Embedding the assistant into the Faculty Web:

The first of the alternatives to add a proper user interface to the solution, embedding the assistant into the web would probably be the most logical option.

Sadly, if the webpage was created as a custom design or with a tool that does not support Watson technology it would not be a simple task.

To be successful, the assistant should be controlled by a custom designed orchestrator, in charge of regulating the chat window, the API calls, etc.

On the bright side, the presence of an orchestrator allows for the implementation of specific answer buttons, solving one of the limitation of the Watson Assistant service.\(^\text{39}\)

8.2.6 Adding a voice interface:

To make the assistant more accessible, one of the best options is to add a voice interface. This would allow visually impaired people (or anyone with problems to type) the chance to benefit from its use.

Probably the easiest way to do so would be adding instances of Watson Speech to Text and Text to Speech, and integrating them into the solution. Please, note that integrating the voice interface with a webpage can be trickier.

It is also important to mention that the theoretical cost of each conversation would rise, since each input would have to go through two more services before getting back to the user.

\(^\text{39}\) See section 6.11.2 – Handling inputs referred to a set of entity values
8.3 Social, environmental and ethical implications:

The solution created in the practical part of this project is not expected to have a huge impact neither socially nor environmentally. Nevertheless, the theoretical concepts behind it do indeed pose some interesting ethical questions.

Firstly, the premise of delegating on machines the transfer of knowledge is a the very least unnerving.

And secondly, though the virtual assistants imitate quite well the human interaction, they are not able implicate with the users, to listen them beyond their guidelines.

These two questions stand on marshy ground, since categoric answer would either severe the course of human progress or accept eliminating humans from the equation as a viable option.

In opinion of the author, the best course of action would be the middle ground between the extremes or, in other words, relying in the machines for easy or systematic tasks, but leaving the vicissitudes to humans.

Of course, the is only the fear of machines taking jobs from humans but, at least on this case, machines should operate beyond human competences, performing tasks that are either too big to handle by humans or so repetitive that it would be a waste to assign human workforce to them.

To summarize, it seems that the only way to avoid an ethical (and even moral) dilemma is to learn to limit the use we make of machines, and leaving space for human decision whenever It really matters.
9 Bibliography:


10 Annex 1: Deploying the assistant through Telegram.

Once the Watson Assistance instance is ready to be deployed, it is time to integrate it with an interface; which, in this case will be a Telegram bot. The next pages show how to do that relying only on the services provided by IBM Cloud (except for Telegram itself).

10.1 Recovering the credentials from the assistant

To be able to connect the assistant with the Node-RED flow, the credentials for the specific instance of Watson Assistant must first be recovered.

From the default view of the workspace, pressing the “deploy” button on the vertical menu (on the left of the screen) will open the “credentials view” as can be seen on the next figure:

Once inside the view, both the user name and the password for the service must be recovered from the Service credentials tab. The workspace ID found under Workspace details must also be recovered.
10.2 Configuring Telegram

To deploy a bot in Telegram, it is required to download either the mobile application or the computer version. Once inside the application, it is mandatory to start a conversation with a default bot named “bot-father”.

Figure 15 - Annex 1 - Retrieving credentials

Figure 16 - Annex 1 - Adding BotFather
After starting a new conversation with this bot, we can create a new bot by typing “/newbot”:

![Telegram screenshot]

Figure 17 - Annex 1 - Creating a new bot

After introducing the command, the BotFather will ask for a new name for the bot and a username that must end with the suffix “-bot” once both of them are introduced, it will create and show a Token to access the API. This will serve as an identifier and must be saved for later use.

After the creation of the bot, some of its settings can be altered at any time using the command “/editbot” though for the purpose of this tutorial none of them will be altered.
10.3 Creating the Node-RED instance:

The Third step of the deployment is the creation of a Node-Red Instance. As it was previously mentioned, this service will play the role of integration platform. As can be seen on the next figure, the service required is called Internet of Things Platform Starter, and can be found in the boilerplates section.
Once selected, IBM Cloud will ask for a unique name and will offer some settings, for this tutorial, nothing will be altered except the name. Once everything is correct, it is time to press create.

After the creation of the service, it will be accessible through the Dashboard panel view on the IBM Cloud account. The first time is accessed, it will ask to enter a username and password or allow anyone to access the editor. It is highly recommended to secure the solution by adding a username and a complex password.

Note that after the creation, the instance will require some time to build itself before it can be accessed. This process ends once the status shows a green dot and the word Running.
After the user and password are introduced, two more informative steps must be completed by pressing next and finish respectively. They basically cover the foremost features of the service and inform the developer about the existence of a library containing examples of flows ready to be deployed.

Having passed those screens, the next step is entering the flow editor, which can be done by pressing the button shown on the image bellow:
Once inside the flow editor, but before creating the solution, it is mandatory to import some of the nodes required (since they are not standard nodes).

To do this, starting from a brand-new flow, and pressing the “**hamburger**” on the top right side of the screen will open a set of options. To download new nodes, the option **Manage Palette** must be selected.

![Image of flow editor with Manage Palette option](image.png)

_Figure 23 - Annex 1 - Accessing manage content_

Once Manage Palette is selected, it will prompt a view of all the bundles (of nodes) available for download.

Up until recently, the bundle used was “**node-red-contrib-chatbot**”. Now, even though it is still available, with IAM authentication being added to IBM Cloud services, in opinion of the writer, the bundle “**node-red-contrib-telegrambot**” is a better option (since it allows for control of API keys).

---

40 Hamburger is a term used to denote the icon (commonly used for menus) conformed by three parallel and horizontal lines.
The bundle can be found under the install tab, writing its name on the search-bar.

Once the button install is pressed, a list of nodes will be added to the palette under the denomination of *telegram*.

With all the nodes ready to use, it is time to design and create the flow that will connect the assistant with telegram.

To add a node to the flow, one does only need to drag and drop it into the mat. To connect it, it is only required to click in one of the terminals of the starting node and drag the mouse to a terminal of the receiving node. To access the setting of a specific node, double clicking on it will open a window with all the options.

The first nodes added to the flow will be a telegram receiver, that will input into the flow the user’s messages and a telegram sender, that will send to the user the answers generated by the assistant.

After the telegram nodes, two function nodes, whose purpose is to tweak the parameters to allow communication between Telegram and the IBM Cloud must be added.

Finally, an instance of the node *Assistant* (in the folder IBM Watson) must be added to connect with the Assistant itself.
Additionally, a debug node can be added to diagnose issues during the coding of the Node-RED solution.

If done correctly, the final picture of the flow should look something like this:

![Figure 26 - Annex 1 - Design of the flow](image)

Now is time to connect the nodes. It is important to know, that not every exit terminal receives the same output, so the connections must be made exactly as shown in the next figure:

![Figure 27 - Annex 1 - Connecting the nodes](image)
With all the nodes connected, it is time to edit the settings of each node in order for the solution to work.

First, the Telegram Receiver; once the node is opened, a new bot must be added:

![Image: Adding the bot to Node-RED](image.png)

Figure 28 - Annex 1 - Adding the bot to Node-RED

To add the bot, both the name and the Token must be recovered from Telegram\textsuperscript{41} and be introduced on the new bot’s settings, apart from that, personal experience suggests that it is better to increase the polling time to 1000, to avoid connection failures due to internet connection.

No parsing needs to be set for the bot (since the key parameters will be manipulated in the function nodes).

A similar process must be repeated for the Telegram Sender node, only this time, the bot created for the sender already appears on the view, and can be directly selected.

With the Telegram nodes configured, it is time to prepare the Assistant node. This step requires the Service Credentials recovered earlier\textsuperscript{42}.

Apart from introducing the credentials, the option multiple users must be selected. This option will allow different users to have simultaneous conversations with the assistant.

\textsuperscript{41} See section 10.2 - Configuring Telegram
\textsuperscript{42} See section 10.1 - Recovering the credentials from the assistant
The end result of this stage can be seen on the following image:

Figure 29 - Annex 1 - Adding the assistant to Node-RED

After the Assistant node is appropriately configured, it is time to write the functions inside the homonymous nodes.

The first function name (following the path of the flow) will receive information from Telegram and pass it to the Watson Assistant.

Figure 30 – Annex 1 - Coding the first function node

The image above shows the “Edit” view of the node. Inside, a distinct name (to help identify which node is being altered) can be seen along with three lines of code:
msg.user = msg.payload.chatId;
msg.payload = msg.payload.content;
return msg;

The sole purpose of these lines is to identify the input context variables with the parameters required by the assistant node.

The “msg.user” parameter helps identify the author of the input message, allowing (only if the “multiple user” option was selected during the creation of the bot) for multiple users to have simultaneous conversations with the assistant.

The “msg.payload” simply points the actual message to the assistant node.

For the second function node, the code is quite more elaborated:

```javascript
msg.payload = {
  chatId: msg.user,
  type: "message",
  content: msg.payload.output.text[0]
};
return msg;
```

Figure 31 - Annex 1 - Coding the second function node
Basically, this segment of code takes the information given by the assistant node (the identity of the user and the content of the answer given by the assistant) and maps it in order for it to be legible by the Telegram sender node.

Once this step is finished, the solution is ready to be deployed by pressing the button deploy on the top right corner of the screen:

![Figure 32 - Annex 1 - Deploying the solution](image1)

Now the solution is ready to be tested through telegram. As a first indicator, if the telegrams nodes are correctly connected, they should read “connected” under the node box.

If everything went well, the user should be able to initiate now conversation with the assistant via Telegram Chat.
11 Annex 2: Starting data set

This annex contains the Frequently Asked Questions that served as a base for the design of the assistant. Each question has been numbered; these numbers will be relevant in the next annex for they stand as the question associated.

Note that, since the assistant was designed to operate in Spanish, both the questions and the answers are written on that language.

11.1 List of FAQ:

<table>
<thead>
<tr>
<th></th>
<th>¿Cuándo puedo empezar mis prácticas externas?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Debes tener aprobados 120 créditos antes de poder hacer las prácticas externas. Con carácter excepcional y con autorización del responsable de prácticas externas, se permite empezar con 110 créditos aprobados.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>¿Cómo empiezo mis prácticas externas?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Tienes que registrarte en la web del COIE (<a href="https://www.coie.upm.es/">https://www.coie.upm.es/</a>) y optar a algunas de las prácticas publicadas. Si ya has hablado con alguna empresa por tu cuenta, la empresa tiene que darse de alta en el COIE, firmar un acuerdo con la UPM y publicar la oferta a la que quiere que tú optes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>¿Cuántos créditos puedo cursar?</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Puedes cursar 3, 6, 9, 12, 15 o 18 créditos, dependerá del número total de horas de prácticas. Como norma general, el máximo son 25 horas a la semana y un crédito de prácticas equivale a 30 horas, así que en un mes se pueden obtener 25 x 4 / 30 = 3,33 créditos. Redondeando un poco se puede estimar que para 6, 12 y 18 créditos se necesitan aproximadamente 2, 4 y 6 meses de prácticas.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>¿Mis prácticas deben ser curriculares o extracurriculares?</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Para que se te puedan reconocer es imprescindible que sean curriculares.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>¿Puedo cobrar por ellas?</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>No se puede percibir un salario o beca por las prácticas curriculares, pero las empresas pueden pagar pequeñas cantidades en concepto de ayuda al estudio. Estimamos que entre 550 – 700 € al mes es una cantidad razonable en nuestro sector. Por tanto, no es imprescindible que las prácticas sean extracurriculares para poder cobrar una ayuda</td>
</tr>
<tr>
<td>Número</td>
<td>Pregunta/Instrucción</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------</td>
</tr>
<tr>
<td>6</td>
<td>¿Cómo me matriculo?</td>
</tr>
<tr>
<td></td>
<td>Una vez elegidas las prácticas y admitido por la empresa deberás ponerte en contacto con la Oficina de Prácticas Externas para que ésta autorice tu matriculación. Esto puede ser en cualquier momento del curso académico, aunque debes tener en cuenta el plazo necesario para realizar las horas correspondientes antes del momento de la evaluación. Una vez autorizada la matrícula podrás realizarla en Secretaría del Centro.</td>
</tr>
<tr>
<td>7</td>
<td>¿Qué pasos tengo que seguir?</td>
</tr>
<tr>
<td></td>
<td>Puedes consultar nuestro diagrama de secuencia para ver todo lo que tendrás que hacer o en qué fase de realización estás.</td>
</tr>
<tr>
<td>8</td>
<td>¿Necesito un supervisor académico?</td>
</tr>
<tr>
<td></td>
<td>Es imprescindible tener un supervisor académico para que las prácticas puedan ser curriculares. La oficina de prácticas te asignará uno.</td>
</tr>
<tr>
<td>9</td>
<td>¿Necesito un tutor profesional?</td>
</tr>
<tr>
<td></td>
<td>Es imprescindible que la empresa te asigne un tutor profesional y que exista un programa formativo asociado a tus prácticas. Recuerda que NO eres un trabajador, sino un estudiante, y que haces las prácticas para aprender.</td>
</tr>
<tr>
<td>10</td>
<td>¿Sirve para algo la valoración final que hago de la empresa?</td>
</tr>
<tr>
<td></td>
<td>Por supuesto, tanto a la empresa como a la Oficina de Prácticas Externas nos interesa mucho la opinión final de los estudiantes, tanto si es buena como mala, porque nos permitirá elaborar un ranking que ayude a futuros estudiantes a elegir las prácticas con mayor calidad formativa. Por ello es importante que des una opinión sincera al rellenar tu informe final.</td>
</tr>
<tr>
<td>11</td>
<td>¿Cuándo apruebo mis créditos?</td>
</tr>
<tr>
<td></td>
<td>Cuando haya transcurrido el tiempo mínimo para cubrir las horas a partir de la matriculación y el momento del inicio de prácticas deberás realizar una memoria de prácticas y entregarla en nuestra plataforma (Moodle). También es imprescindible un informe final tuyo, del tutor profesional y del supervisor académico. Toda esta documentación es necesaria para que tu práctica sea calificada en la siguiente reunión del Tribunal de Prácticas Externas.</td>
</tr>
<tr>
<td>Número</td>
<td>Pregunta</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>12</td>
<td>¿Cuándo se reúne el tribunal de prácticas?</td>
</tr>
<tr>
<td>13</td>
<td>¿En qué convocatoria contabiliza el mes de Agosto?</td>
</tr>
<tr>
<td>14</td>
<td>¿Puedo convalidar las prácticas si ya he trabajado en el sector?</td>
</tr>
<tr>
<td>15</td>
<td>¿Cuántas horas a la semana puedo hacer?</td>
</tr>
<tr>
<td>16</td>
<td>¿Soy alumno extracomunitario, ¿por qué no puedo hacer más de 20 horas?</td>
</tr>
<tr>
<td>17</td>
<td>Ya he hecho unas prácticas extracurriculares, ¿se me pueden reconocer?</td>
</tr>
</tbody>
</table>
Estoy trabajando en una empresa del sector, ¿Puedo hacer prácticas curriculares a la vez?

Si estás trabajando en el sector y desarrollando labores de Ingeniero Informático tienes dos opciones para aprobar las prácticas externas:

-Procedimiento A: si quieres que se te reconozca como prácticas externas el trabajo realizado con anterioridad a la matrícula en la asignatura de prácticas externas tendrás que hacer una solicitud a la Comisión de Reconocimiento de Créditos, que evaluará si el trabajo que has realizado cumple con los requisitos. Tendrás que presentar un informe de los trabajos realizados y una vida laboral. Si se te reconoce la asignatura tendrás un 5.0.

-Procedimiento B: si vas a continuar trabajando en la empresa lo mejor es que te pongas en contacto con la Oficina de Prácticas Externas, que te asignará un profesor supervisor. La empresa deberá asignarte un tutor profesional y elaborar un plan de prácticas que tendrá que ser aprobado por nuestro subdirector. En este caso no es necesario que la empresa firme ningún acuerdo con el COIE porque ya estás bajo contrato. Una vez realizadas las horas correspondientes y presentada la misma documentación que se requiere para las prácticas curriculares tus prácticas serán evaluadas y calificadas en la siguiente reunión del tribunal de la asignatura.

Tengo una beca de investigación oficial de la UPM, ¿Se me puede reconocer?

Sí, las becas oficiales con credencial de becario se reconocen como prácticas externas.