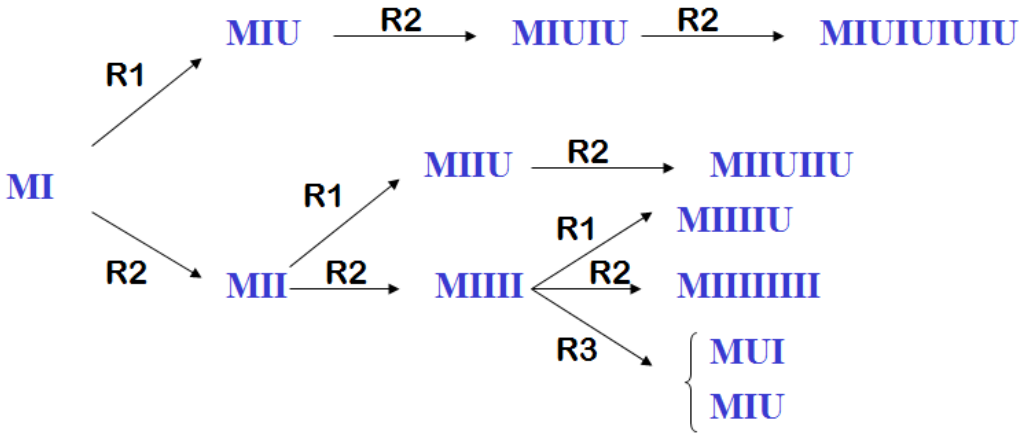
system of wrod generation

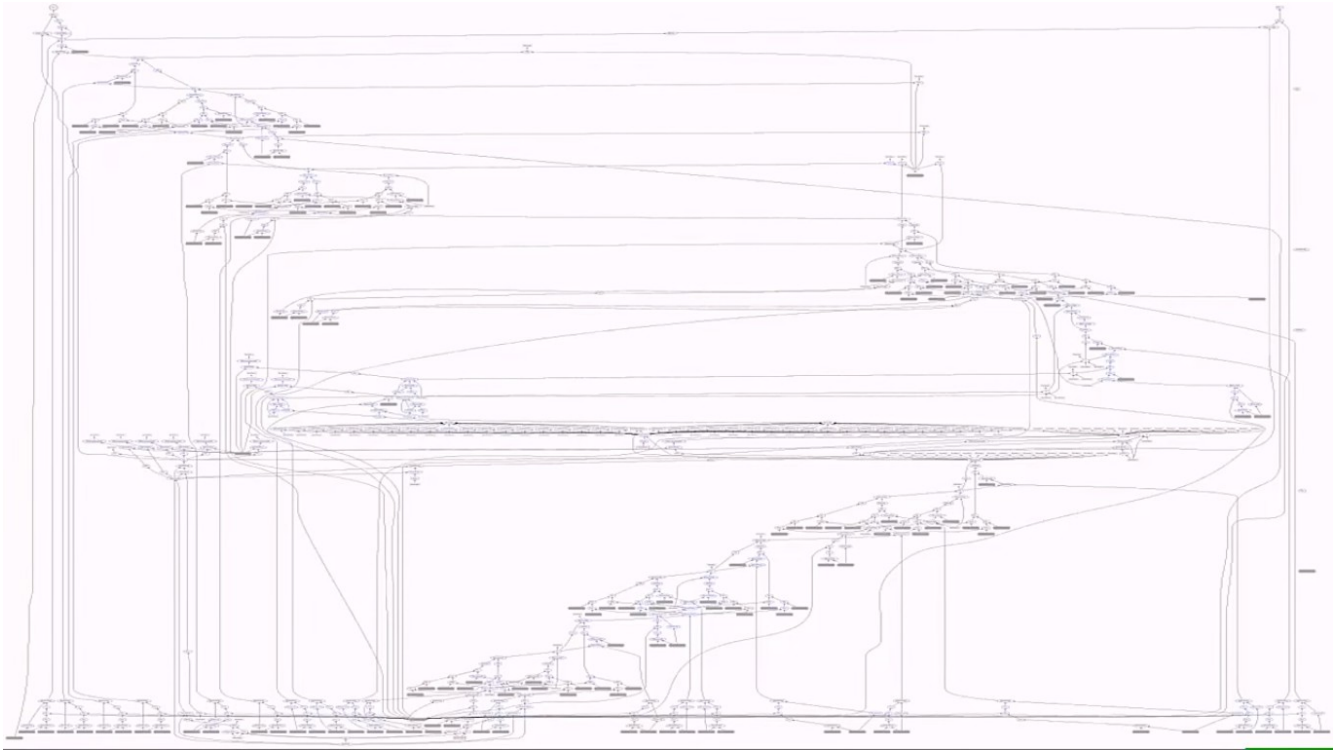
- **Axiome MI**
- **Régles : pour tout $\alpha, \beta \in V^*$**
  - **[R1]** $\alpha I \rightarrow \alpha IU$
  - **[R2]** $M\alpha \rightarrow M\alpha\alpha$
  - **[R3]** $\alpha III\beta \rightarrow \alpha U\beta$
  - **[R4]** $\alpha UU\beta \rightarrow \alpha\beta$

Système de production

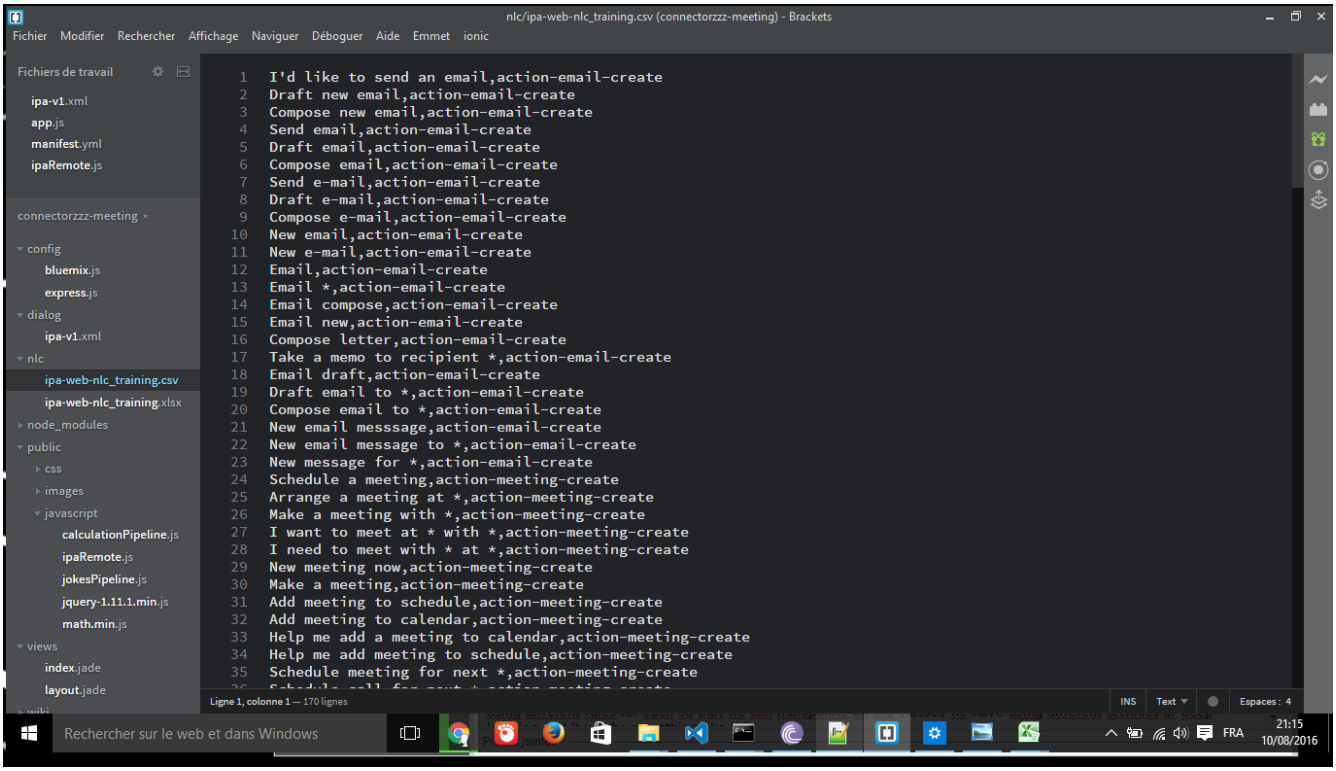
- **Peut on obtenir le mot MU ?**



And here we see that not all problems have solutions with calculations and mathematical models so our main goal is to create some system that answers simple queries and problems but then using deep neural learning techniques to generate some swarm intelligence using the power of the collective system that we will implement using one of different deep learning frameworks and toolkits and even libraries like caffee , torch , tensorflow or even microsoft CNTK like this example below



So to start understanding our system and use cases we need to classify our data like for example in watson NLC service here



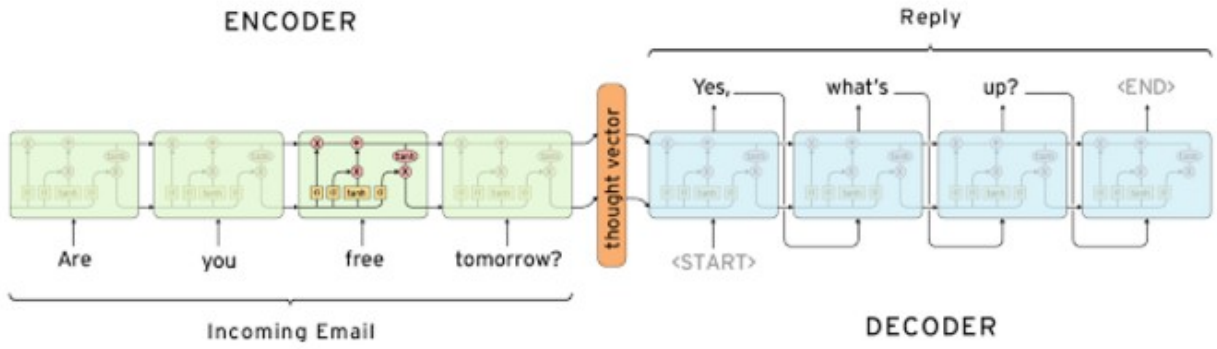
but then with big data the problem comes with which approche we zill use for defining our models because they can be from retrieval or even generated in complex systems

A TAXONOMY OF MODELS

RETRIEVAL-BASED VS. GENERATIVE MODELS

Retrieval-based models (easier) use a repository of predefined responses and some kind of heuristic to pick an appropriate response based on the input and context. The heuristic could be as simple as a rule-based expression match, or as complex as an ensemble of Machine Learning classifiers. These systems don't generate any new text, they just pick a response from a fixed set.

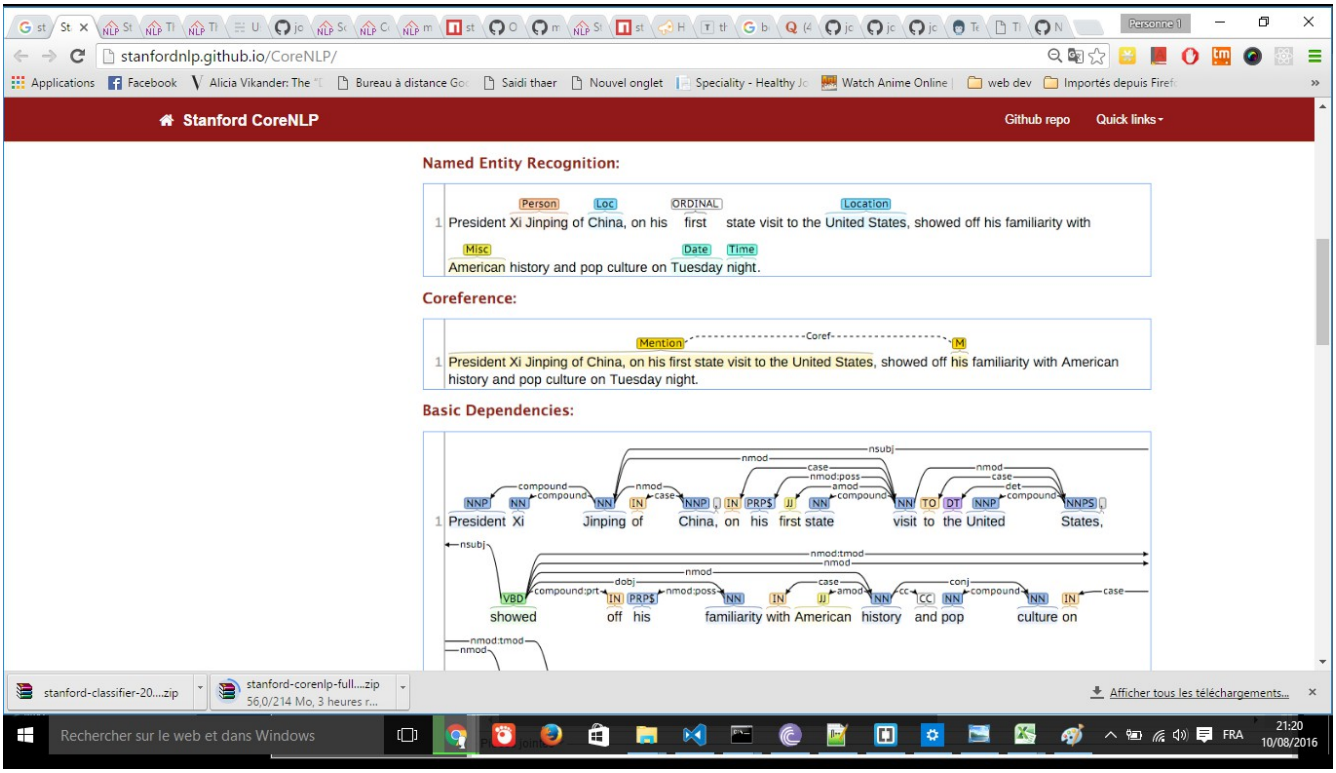
Generative models (harder) don't rely on pre-defined responses. They generate new responses from scratch. Generative models are typically based on Machine Translation techniques, but instead of translating from one language to another, we "translate" from an input to an output (response).



but then to try to encode queries and then decode a response will be using a third party system so we will will perform data cleaning, pre-processing, classification. While doing this one requires some NLP system to do grammatical and syntactical classifications

•Data tokenization
•Stop word removal
•Part of speech tagging
•Noun entity extraction
•Sentence parsing etc

NLP will help you here. When you do numeric classification, text classification or perform some recommendation; NLP helps to do it better



but then the need of deep learning methods and techniques is very important so that we can create our

models that can one of 2

OPEN DOMAIN VS. CLOSED DOMAIN

In an **open domain (harder)** setting the user can take the conversation anywhere. There isn't necessarily have a well-defined goal or intention. Conversations on social media sites like Twitter and Reddit are typically open domain – they can go into all kinds of directions. The infinite number of topics and the fact that a certain amount of world knowledge is required to create reasonable responses makes this a hard problem.

In a **closed domain (easier)** setting the space of possible inputs and outputs is somewhat limited because the system is trying to achieve a very specific goal. Technical Customer Support or Shopping Assistants are examples of closed domain problems. These systems don't need to be able to talk about politics, they just need to fulfill their specific task as efficiently as possible. Sure, users can still take the conversation anywhere they want, but the system isn't required to handle all these cases – and the users don't expect it to.

and then when we have different models for different classes we can coordinate between them using a dialog system like for this entity and intent used by api.ai services

formation.json


```json
{
  "id": "18cb5238-833d-4f18-81f6-5f4bbde3866b",
  "name": "Information",
  "isOverridable": true,
  "entries": [
    {
      "value": "Information",
      "synonyms": [
        "Information",
        "info",
        "news",
        "bin",
        "recycling"
      ]
    },
    {
      "value": "detail",
      "synonyms": [
        "detail",
        "detailed",
        "precise",
        "advance"
      ]
    }
  ],
  "isEnum": false,
  "automatedExpansion": true
}
```

tell me about @Information_Information.json


```
{
  "id": "85c4d5b5-0828-4aea-a73b-82238524f75b",
  "name": "tell me about @Information:Information",
  "auto": true,
  "contexts": [
    "tell_me_about_recycling"
  ],
  "userSays": [
    {
      "id": "ece54b8b-0959-45ae-b7b2-b24cce24f8d4",
      "data": [
        {
          "text": "give me "
        },
        {
          "text": "infos",
          "alias": "Information",
          "meta": "@Information",
          "userDefined": true
        },
        {
          "text": " about "
        },
        {
          "text": "recycling",
          "alias": "Recycle",
          "meta": "@Recycle",
          "userDefined": true
        }
      ],
      "isTemplate": false,
      "count": 0
    },
    { ... }
  ],
  "responses": [
    {
      "resetContexts": false,
      "action": "informaton_action",
      "affectedContexts": [
        {
          "name": "tell_me_about_information",
          "lifespan": 5
        },
        { ... }
      ],
      "parameters": [
        {
          "required": true,
          "dataType": "@Information",
          "name": "Information",
          "value": "$Information"
        },
        { ... },
        { ... },
        { ... }
      ],
      "speech": [
        "Recycling is something that we can all do. Children that are school age and older can help out too.",
        "The goal is to stop tossing out items that we can use again and again."
      ]
    }
  ],
  "state": "LEARNED",
  "priority": 500000,
  "webhookUsed": true,
  "fallbackIntent": false
}
```

but then

INTENTION AND DIVERSITY

A common problem with generative systems is that they tend to produce generic responses like "That's great!" or "I don't know" that work for a lot of input cases. Early versions of Google's Smart Reply tended to respond with "I love you" to almost anything. That's partly a result of how these systems are trained, both in terms of data and in terms of actual training objective/algorithm. Some researchers have tried to artificially promote diversity through various objective functions. However, humans typically produce responses that are specific to the input and carry an intention. Because generative systems (and particularly open-domain systems) aren't trained to have specific intentions they lack this kind of diversity.

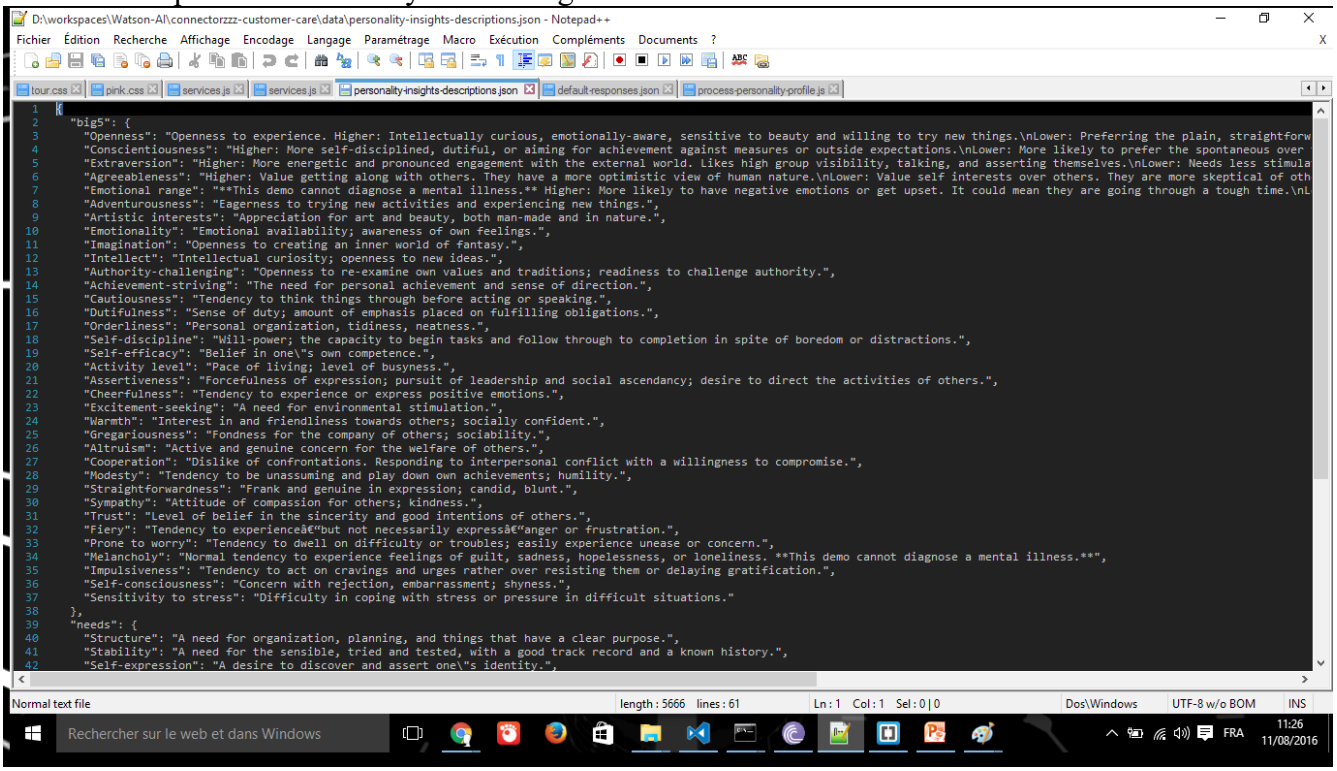
and then different studies were made for personality insights so that the AI system will formulate proper responses to all customer queries

COHERENT PERSONALITY

When generating responses the agent should ideally produce consistent answers to semantically identical inputs. For example, you want to get the same reply to “How old are you?” and “What is your age?”. This may sound simple, but incorporating such fixed knowledge or “personality” into models is very much a research problem. Many systems learn to generate linguistic plausible responses, but they are not trained to generate semantically consistent ones. Usually that’s because they are trained on a lot of data from multiple different users. Models like that in A Persona-Based Neural Conversation Model are making first steps into the direction of explicitly modeling a personality.

message	Where do you live now?
response	I live in Los Angeles.
message	In which city do you live now?
response	I live in Madrid.
message	In which country do you live now?
response	England, you?

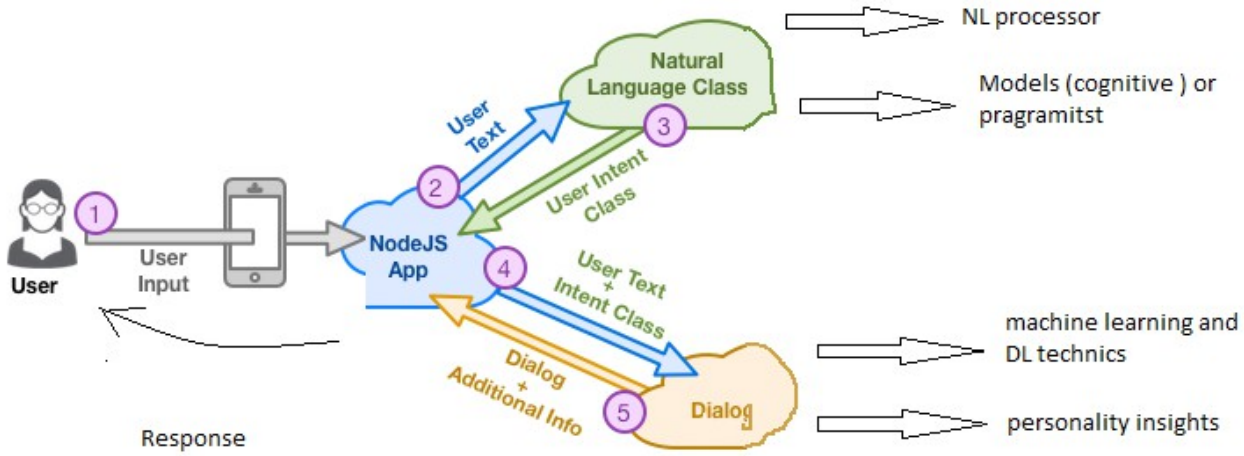
like for example the one used by watson cognitive services



and finally make a bot system that gives a response like the one used in api.ai for example


```
{
  "id": "69e47f32-39fe-4c50-b9c0-3d2
  "timestamp": "2016-08-11T06:10:30.
  "result": {
    "source": "domains",
    "resolvedQuery": "hey",
    "action": "smalltalk.greetings",
    "parameters": {
      "simplified": "hello"
    },
    "metadata": {},
    "fulfillment": {
      "speech": "Good day!"
    },
    "score": 0
  },
  "status": {
    "code": 200,
    "errorType": "success"
  },
  "sessionId": "9e7f96f9-2948·
}
```


and create a BOT system that contains all those different services and coordinate with them to make one simple use case



you know computers are not smart but the state of the art is to make them look like they are
but then this really can be made when meeting with big data to understand different patterns and contexts and create proper models for our different use cases