BotWheels: A Petri Net based Chatbot for Recommending Tires

Francesco Colace[1], Massimo De Santo[1], Francesco Pascale[1], Saverio Lemma[2] and Marco Lombardi[2]
[1]*DIIn, University of Salerno, Fisciano (SA), Italy*
[2]*SIMASLab, University of Salerno, Fisciano (SA), Italy*

Keywords: Chatbot, Petri Net, Recommender Systems.

Abstract: Technological progress seems unstoppable: large companies are ready to implement more and more sophisticate solution to improve their productivity. The near future may be represented by so-called Chatbot, already present in the instant messaging platforms and destined to become more and more popular. This paper presents the realization of a prototype of a conversational workflow for a Chatbot in tires domain. The initial purpose has focused on the design of the specific model to manage communication and propose the most suitable tires for users. For this aim, it has been used the Petri Net. Finally, after the implementation of the designed model, experimental campaign was conducted in order to demonstrate its enforceability and efficiency.

1 INTRODUCTION

Make a Chatbot that can be as close and similar to a human being, today is a very important challenge, both as regards its possible applications, especially web and social, both for its use in the research scope. One of the biggest problems is to try to make it look like the Bot as close as possible to a human being. There is the propensity of persons to give more listening to a person rather than to a robot: it is because in some way, since the people used to talk to each other, it is difficult to interact with an automaton, both because a robot does not always behave during a natural dialogue as we expect. In research fields, the issue of Chatbot and Bot in general has been discussed for many years, although it has seen an increasingly gradual slowdown in recent years. This is due in part to the inherent difficulty of the same, both since it is very difficult to appropriately describe the evolution of a discussion, without making the appropriate considerations and upstream reflections. In fact, the amount of investment by companies in trying to create a Bot as similar to an operator is growing (Casillo et al., 2016). Indeed all the major social platforms have now integrated this service and offer the ability to customize and then to map the conversations between a bot and a user. This happens because the company generally considers it. Currently the technology limits, however, prevent existing Chatbot to have good performance since the bottleneck is that generally a user during a conversation is unpredictable. As seen from the Turing test to be able to assume that a machine can be like a man, it is important that this more than five minutes of talk time. This means that the conversational flow plays an important role in to the project of a Chatbot. To create and manage the workflow of a Bot appears to be today one of the main objectives, because the existing systems are not always responsive given a user input, and often fail in the recognition and processing of user-written sentences. The main aim of this paper is to describe a software module prototype, called workflow manager, who is responsible for the management of the flow of conversation between a Chatbot and a user, applied at a real case. Our case study concerns a Chatbot, called BotWheels, which will serve for help the choice of tires to buy. After an analysis on the state of art, we have realized the general pattern of operation of the bot and then using the Petri Nets, we have realized the workflow model. Finally, it was conducted the experimental phase that has highlighted the strengths and weaknesses points of the Bot. In the next section, the related works are presented.

2 RELATED WORKS

One of the biggest problems on Bots always has been to be able to understand, in a sentence uttered by a

Colace, F., Santo, M., Pascale, F., Lemma, S. and Lombardi, M.
BotWheels: a Petri Net based Chatbot for Recommending Tires.
DOI: 10.5220/0006491903500358
In *Proceedings of the 6th International Conference on Data Science, Technology and Applications (DATA 2017)*, pages 350-358
ISBN: 978-989-758-255-4

person, his/her intent, that is what our user is asking the automaton. The usually used approach is to create an ontology of reference, in which inside there is mapped throughout the domain of knowledge. Already from this observation, you can understand how the definition of the correct reference ontology is fundamental to the proper functioning of Bot. From this, it is possible to extrapolate the object of which you are speaking with Latent Semantic Analysis (LSA) algorithm. To observe how very similar thing was done, it is possible using LSA, the conversational skills of an agent has been improved. It is therefore possible to identify in this way the topic of the conversation and structure a dialogue flow (Pilato et al., 2007). Another important work shown how is possible implement a dialogue management mechanism through formalisms and finite state machines. It shows how now one of the issues of greatest interest and critical problem has been to develop a model of the discussion flow between human and robot (Kronlid and Lager, 2007).

In the last times, there have been some interesting works that worth mentioning. An important work presents how the interaction between man and some Chatbot present on the internet in order to make a classification and to verify the actual efficiency (Gianvecchio et al., 2011). This was done analyzing the conversation log between Bot and users through log-based classification, that unlike Turing test, it differs for the duration and for the comparative metrics of the discussion (e.g. length of sentence and delay between question and answer). A case study, where the relations are between users and their messages, was shown in another work (Gligorijevic et al., 2012). One of the first prototypes of Chatbot for social networks was developed in last recent times. The critical point remains, as noted earlier, the workflow management. In this case, an algorithmic approach is used and it provides for a very flexible management of the conversation (Rodrigo et al., 2012). Last times was improved a tool to describe a Chatbot conversational model, which extrapolating certain keywords from texts on Wikipedia and it is able to take on a particular conversational lexicon based on training (Haller and Rebedea, 2013). In recent times, the greatest efforts have focused on trying to create a robot that could be as much as possible like a human: the dialog flow plays a fundamental role in the modeling of a Chatbot. For describe a model that helps the Bot to maintain conversations with people through the development of natural language, was improved a software tool (Ravichandran et al., 2015). All this was achieved through a conversational Flow Chart, which describes the operation that occurs through the activation of appropriate responses based on input. Another example is a self-learning Chatbot model, through user-entered phrases. This system carries out the matching of new constructs: a POS-tagged tokens system performs the matching of phrases, then makes an assessment of the correct answers and finally all mated sentences are been analyzed (Bang et al., 2015). An improved version of AluxBot software was developed, which is a Chatbot developed in Visual Basic with integration of speech recognition of OS Windows (Peniche-Avilés et al., 2016). AluxBot is focused on promoting the care of the environment and sustainable development for school children, in order to provide an alternative use of new technologies. It can be possible observe a particular architecture of a Chatbot prototype used without a server platform, in which all the necessary features are implemented on the inside, without the need to request external services (Yan et al., 2016). Another example is a Bot to Bot system that it can write controlled applications by voice using the natural language processing and robotic control in order to make the voice command of a user and translate it into a structured intent for the robot (Fischer et al., 2016). Furthermore, it is possible to instruct the robot via a Chatbot, which is responsible for providing input the desired command. Another usefully study is on the usefulness of ontology-based Chatbot technology to the design of intelligent agents for medical diagnosis (Edwards et al., 2016). It is apparent from this study the need for a technology to support the diagnosis, which can not be separated from the use of medical information of the patient, such as his/her clinical record and his/her medical history, which will be needed for a higher quality for interaction with the Chatbot. The system also should be regularly trained to stay updated. A very interesting case of study present a teaching Chatbot assistance, named CS 221, for students, where the system classifies any question under three aspects: Policy, Assignment and Conceptual (Chopram et al., 2016). The union of the classification of the three types provides the answer to make on output. Finally, another important work is a Chatbot model created by IBM and named Curious Cat: it is a crowdsourcing model that guides the user to provide information of interest, using a base of existing knowledge, and helps him/her in the process acquisition, requesting coherent questions and checking the answers (Bradeško et al., 2016).

3 MODELING AND FORMALIZATION

In order to model and formalize the flow of execution of a Chatbot on more possible contexts (Colace et al., 2016), evaluating what today is present in research, it was analyzed the point of view of the individual conversation atom between human and the automaton. In fact, a conversation can be seen as a succession of user's questions/answers and questions/answers of the automaton. It has been mapped the single interaction in order to determine what are the topic in question, at a particular time, and what are the attributes of the argument on which the automaton will have to ask a question or give an answer. It allows to be able to model in a general way a possible operation of Chatbot, regardless of its actual realization, and to formalize the problem analytically.
Through the analysis of the sentence obtained from a semantic engine, which extrapolates the intent (e.g. tire change) and its associated attributes (e.g. height, width, diameter), it is possible to think about modeling, starting from a predetermined context (e.g. tires), any sentence during a conversation with two fundamental information:

- TOPIC: what we are talking (1 word)
- TAG: the topic attributes (N words)

This pair of information then constitute the current ITEM of information concerning the sentence considered. Then it can be defined the two bounded sets:

- A: finite set of topics
- B: finite set of tag

T is defined as a Cartesian product of A and B: T = A x B. C is a subset of A x B, that is defined as all possible items of possible conversation, in a given context and intent.

$$C \subseteq A\,x\,B \quad \forall C \rightarrow I + Co \qquad (1)$$

where Co is the context and I is the intent.
It is possible to have multiple tags for topic but no more topic for tag. This means that the topic must always be one.
This helps to define a priori, in current context and with a specific intent, all the inputs of all interactions that Chatbot can capture and can give an answer or question depending on the requirement. Therefore is possible to use a logic model to be able to describe the dynamic operation for the single interaction.
It is possible to treat the finite state machine model, in particular the Mealy machine, that is a sextuple, {S, S0, Σ, Λ, T, G}:

- finite set of state (S)
- an initial state S0, that is an element of (S)
- a finite set called the input alphabet (Σ)
- a finite set called the output alphabet (Λ)
- a transition function (T : S × Σ → S)
- an output function (G : S × Σ → Λ)

In general, the Mealy machine is a finite state automaton whose output values are determined by the current state and input; unlike the Moore machine, that works only in the current state function. An example of Mealy Machine model is reported in the figure 1.

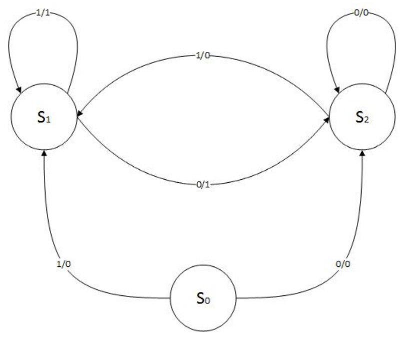


Figure 1: Example of Mealy Machine.

Through this model, it can be possible obtain the logical model of human-robot interaction. Considering the single interaction, there are two states: Si state, where the user provides in input a proper conversation item, and Si+1 state, where the available information will update and the automaton will provide a response or an answer. The outputs are calculated according to the inputs and the previous state. The inputs are the items of the current conversation (user's response or answer); the outputs will be the actions to be taken. Into the state the item of conversation will be preserved in the previous

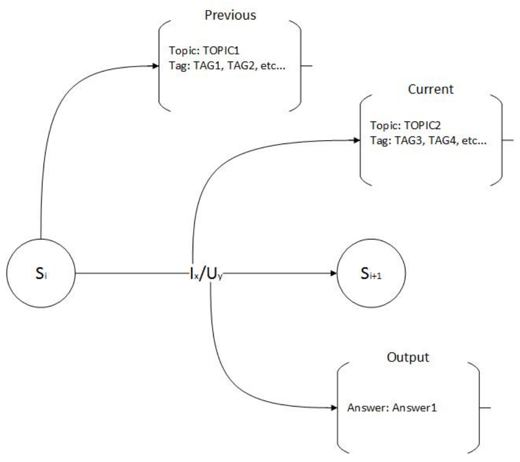
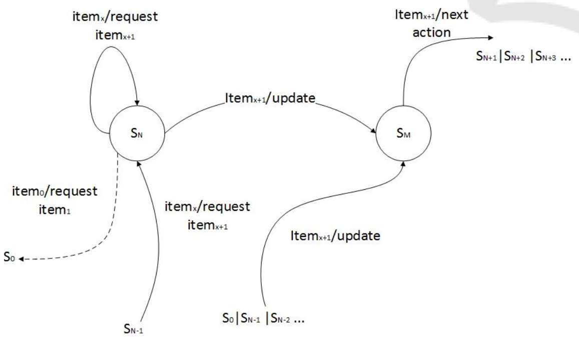


Figure 2: General Model of Finite State Machine for two states.

point so that they can process the correct next action. The outputs depend on the current input and the previous state, which means that for each item of the current conversation, and since the previous state, it must map the appropriate output (answer/question) to be provided.
Remembering that at every item of conversation can have only one topic, which must have a finished number N of tags, is important to consider two things: a topic change involves a change of information (tags) to be requested, and it can not have two items in the same topic. In fact, at every step of the conversation, the automaton must update the information on the current state (TOPIC + TAGS) and then, according to these, gives the appropriate output (REQUEST/RESPONSE).
Defined K as the number of finished levels of interaction for which, given topic of interest, it is possible to produce a final output of conversation and describe the operation of the machine in this way:

$$\sum_{K=0}^{N}(S \; x \; \Sigma) = \Lambda \tag{2}$$

where S is the current state, Σ is the set of the input alphabet $\Sigma = C \subseteq A$ x B, Λ is the set of output alphabet. This means that it can be defined the level of items according to their specific weight within our workflow. According to this concept, this definition can be extended by a finite state machine with Sn states that represent the current state with respect to the user and Sm state that represent the current state with respect to the automaton. A general model of Finite State Machine for N state is reported in the figure 3.



item$_x$/request item$_{x+1}$
Item$_{x+1}$/next action
$S_{N+1}|S_{N+2}|S_{N+3}$ ...
Item$_{x+1}$/update
S_N
S_M
item$_0$/request item$_1$
item$_x$/request item$_{x+1}$
Item$_{x+1}$/update
S_0
S_{N-1}
$S_0|S_{N-1}|S_{N-2}$ ...

Figure 3: General Model of Finite State Machine for N states.

4 BOTWHEELS: WORKFLOW MANAGER DESIGN

The design of BotWheels's Workflow Manager was divided in two phases: in the first place, it is provided to define a model able to explicitly describe and represent the considered knowledge domain (tires). In the second place, the purpose was to define a workflow module that was able to navigate the reference model realized in the first step, in order to maintain a logical discourse with the user and to generate a flow of operations that would allow to identify the most appropriate output. Therefore, the first aim has been to build an ontology, shown in figure 4, to describe the reference taxonomy. The advantage of this approach is to be able to represent concepts, properties, attributes and constraints of the one part and of the other part a reference model for the workflow. The general data structure of a common ontology is a directed graph whose nodes represent concepts and its links represent the relations between these. The root may have several descendants: in this work, it can be considered the child pneumatic concept. It has several children including the vehicle node, whose descendants are the cars and motorcycles node, and the node size. Each of these nodes presents, in turn, determined descendants. A choice of this type has been effected because, generally, to find a specific pneumatic, must be indicated in detail the characteristics of the vehicle or the size of the eraser.
Then, the ontology allows to achieve a knowledge and the links between concepts (for example, we can see that we have various types of vehicles or that the concept of car is made up of the attributes of the brand, model, version and year).

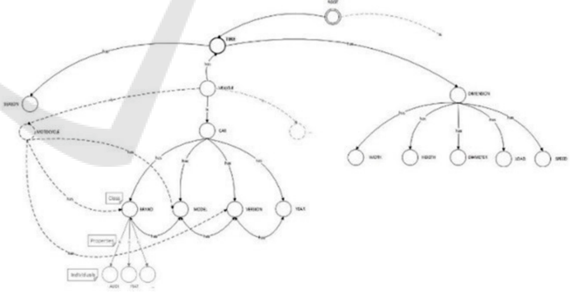


Figure 4: Reference Ontology.

The next goal was to provide a reasoner that can access in a reference ontology and that it was able to generate and follow a consistent and efficient workflow, defined by a sequence of steps. The conversational module then must be able to navigate autonomously the ontology, moving through the concepts relationships.
The selected model is the Petri Net (figure 5 and figure 6), because it lends well for this kind of applications. This model is composed of a set of basic

objects: places, transactions and arcs, graphically represented by circles, rectangles and oriented lines.
The first aspect relates to the identification of the user's intents by BotWheels: it is necessary in practice to understand what he/she is interested. Various options are considered, including changing tires, the workshop nearest search and request for information.

- In particular, the conversation begins through the activation of the transition "start conversation", then the system asks the intent ("ask intent" transition) as long as it is not understood; below, after obtaining the intent ("get intent" transition), through "ask vehicle" transition, system will be asked to user the vehicle (car, motorcycle, etc...).
- Assuming it has been identified the intent of wanting to change the car tires, it can begin the process of tires recommendation for car: the system will randomly ask general information on the car or the tire size, activating the "ask car" or "ask tire" transition. Obtained an answer, "get response for cars" or "get response for tire size" transition is activated.
- If it is aware of the brand and model, "get car" transition is activated, and then it can be possible to ask later the car version and year; as well as if it is aware of the width, height and diameter, the transition "get size" is activated, requiring later load and speed.
- Finally, "get tire from car data" or "get tire from tire data" transition is activated only when the system will have obtained all the information (brand, model, version and year) or the context of the tire (width, height, diameter, load and speed). Obtained result will be represented from one or more tires that correspond at the research done, based on the data provided.
- The result obtained can then be further filtered based on optional parameters, such as the season, in order to activate "recommend tire" final transition.
- The transition ends the conversation is activated and the conversation ends.

The most delicate initial aspects to consider are as follows:

1. Controls relating to the information that the user communicates;
2. The next question that the system will has to make;
3. Any flow of communication changes.

For this purpose, the monitor places were used, which are a control structure of the system evolution: recording some events, commanding the other eligible events and by disabling other.

1. In detail, monitor places related to car context (*brand mp*, *car_model mp*, *version mp*, *year mp*) and to the dimension context of tire (*width mp*, *height mp*, *diameter mp*, *load mp*, *speed mp*) have been defined: they tell what information of the context the system knows.
   For example, in the case where the system has requested information on the car and the user has provided the brand (brand monitor place fills with a token), "get car" transition could not enabled, having the need to know also model; information that the system will require. Obtained this information (model monitor place fills with a token), the transition is activated and the system will prompt version and year. Also received these information, "get tire from car data" transition is activated.
   The monitor places may be filled by direct method and indirect method. In the first case, a determined monitor place is filled directly following an explicit user response, for example relative to the car model. A monitor place can be filled indirectly also, if for example, the user provides the model and can be traced back to the brand in that, in the ontology of reference, there is a single association: in this case, also brand monitor place will be filled.
   For example, the user provides Panda model, can be traced back to FIAT brand.
2. To determine which question the bot will ask the user, we have considered the monitor places in a denied context, graphically defined by an underline name. For example, for car denied context, a token in <u>MODEL MP</u> indicates that the system does not know the car model.
   For example, when the user has already provided the brand of his/her car, <u>BRAND MP</u> will be filled and "ask brand" transition will not be active. Instead, BRAND MP of the car context will be filled and consequently "ask model" transition will be active. Obtained brand and model, "get car" transition is enabled; <u>VERSION MP</u> and <u>YEAR MP</u> are filled and subsequent transitions to enable will be "ask version" and "ask year".
3. When the weighted sum of the missing information on the total of a context is smaller than the current value, there is a *change context.* Suppose the user after providing the car brand, gives information about the height, width and diameter tire; there is a change of context from the current context C (drive) to another context N (size) because the weighted sum of the missing tokens of N on total - the product of the

weighted sum of monitor places null for the number of monitor places null divided by the total number of monitor places in N- is less than the weighted sum of the missing tokens of C. When this condition occurs, the monitor place of context switch is enabled, in the specific case, *Context_Dimension MP*: it allows to transfer the tokens by brand / model / version / year in height / width / diameter / load / speed allowing the change of the communication flow for the new context. In this case, the monitor places related to the height, width and diameter will be occupied by its token: "get size" transition can be turned on and the system will have to ask the load and speed. When user has provided this information, "get tire from tire data" transition is enabled.

Analytically, it must check the following expression:

$$(\sum_{\mathrm{i}} a_i) * \frac{MPN\ N}{MP\ N} < (\sum_{\mathrm{u}} \beta_u) * \frac{MPN\ C}{MP\ C} \qquad (3)$$

where MPN N and MPN C are the monitor places null of N and C; MP N and MP C are the monitor places total of N and C; α and β represent the weight associated to a specific question of monitor place null, of N and C.

A context switch can also occur due to a situation of *context stall*, when the tokens of a place in that context and the related monitor places are equal to zero.

- Following a context stall, there is a change of context or a reactivation of context.
- If all contexts are stalled, the system does not have minimum information to produce a result.

Following the resolution of the listed problems, the goal was to handle situations such as those in which a value is unknown or in any case is not provided, and the cases in which the user updates the values of the current context.

For this purpose, we inserted a maximum (equal to 1) to the number of tokens that may be present in brand, model, version and year places, as well as diameter, width, height, load and speed places. This allows the system, in normal situations, to request a question for the user only once.

In special situations, it can be possible a *context reactivation* current that allows to "recharge" the tokens present in the places indicated.

In particular, to have a *context reactivation (first level)* of current context if occurrence a deadlock, (1) the current context has more tokens of the context to the previous state, or (2) is identical to the previous context but we have a greater number of tokens in the *optional context*, containing the monitor places for the information considered optional, as the season or rubber speed.

- For example, initially the user provided information about the car brand. Later, when the system asks the model, he/she returns the car information. In this case, we will have a deadlock since model monitor place and model place are null. The current context (2) it has more tokens of the context to the previous state (1), we will have a context reactivation (first level) that will allow to recharge tokens of brand / model / version / year, allowing the system to request the model at the user.
- A similar situation is in the case of the question on the model, the user responds by providing optional information, additional to those already present.

The design approach enables to set a limit to the number of times in which the system will ask certain questions, thus avoiding loops in situations of information not provided and / or unknown.

It is possible to have a *context reactivation of second level*: it occurs when, starting from a change of value of one or more information, the number of tokens in that context is less or equal to the number of tokens in the same context to the previous state.

- Suppose that the user has provided brand and model of car, FIAT and Panda respectively. The system asks about the year of the car and user answers TOYOTA.

Every monitor place is correlate at data JSON object representing the context, we are aware that there has been a value change of two information. The number of tokens in the current context (equal to 1) is less than the number of tokens in the same context to the previous state (equal to 2): we have, therefore, a context reactivation of second level. It follows that, since the only occupied monitor place is the brand, the system will ask model.

Finally, it was analyzed the case in which information is not mandatory for the purposes of obtaining a result. An example is "select season" transaction. It is activated with enabling of "get tire from car data" or "get tire from tire data" transaction: a particular choice allows to enable "recommender tire" transaction and then further filter the results previously obtained, while a lack of response from the user does not affect what has already been obtained.

- Examples of optional information are the season or the tire data. The relevant questions are asked when it does not know the answers (and then observing the optional context denied).

Finally, some experimental are discussed in the next chapter.

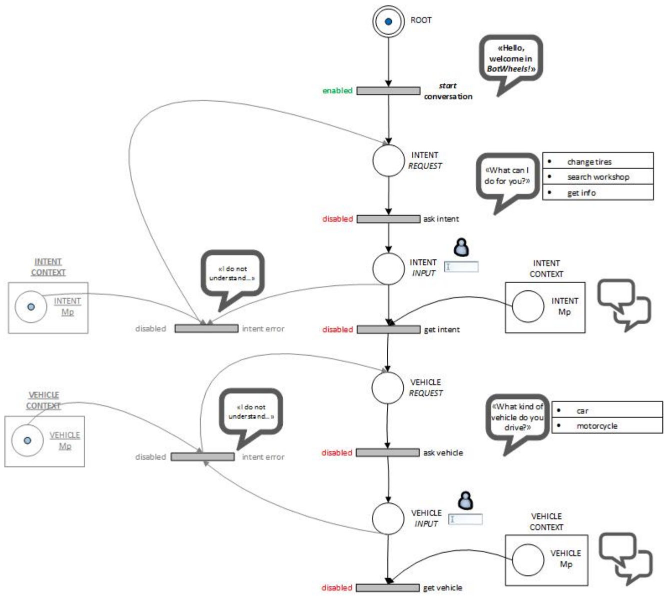


Figure 5: Petri Net of BotWheel's Workflow Manager (Part I).

5 EXPERIMENTAL RESULTS

For evaluating the performance of the proposed system an experimental campaign has been developed. In particular, the aim of the experimentation has been the evaluation of the system effectiveness in the recognition of the users' requests. In addition, the usability of the system has been evaluated. An implementation of the Chatbot has been developed and inserted in a Web Site of a tires seller. At the end of the chat session, an email with the suggested model tires has been sent to the potential customer. The email, showed in the store, guaranteeing a 5% discount on tires' price. In this way, it was possible to check whether the tires suggested by the system were right for the car and the needs of the customer. In two months about five hundreds potential customers (identified with the email address) used the Chatbot and 173 of them showed the email in the store and bought tires. The experimental analysis has been conducted about these 173 customers. First of all the performance of the Chatbot in providing the correct suggestions to the user has been evaluated. In particular, three different situations has been considered:

- Chatbot furnishes a correct suggestion
- Chatbot furnishes a correct suggestion, but it does not fit with the real needs of the customer
- Chatbot furnishes a wrong suggestion

The obtained results are the following Chatbot furnished the following results:

A) Correct Suggestion: 113 - 65,32%
B) Correct Suggestion, but not suitable for the needs of the customer: 24 - 13,87%
C) Wrong Suggestion: 36 - 20,81%

Analyzing the Wrong Suggestion case, we noticed that the system fails when customer talks about a model that have various versions because it proposes tires of different dimensions. Another critical aspect occurs when the system does not understand what kind of vehicle the customer is considering. It happens, for example, when it is not clear if we are considering a car or a motorcycle produced by an automotive group. In the case of Correct Suggestion, but not suitable for the needs of the customer the main problem is in the identification of the real user needs. For example, when the customer says that works in a snowing city but lives in city on the sea the system fails to identify the correct tires. From the point of view of the usability a questionnaire about his/her interaction with the Chatbot was submitted to each customer. In general, they find the Chatbot easy to use and user friendly. Comparing it with other Chatbot (for example Telegram Chatbot or similar) customers says that BotWheels is more simple and effective.

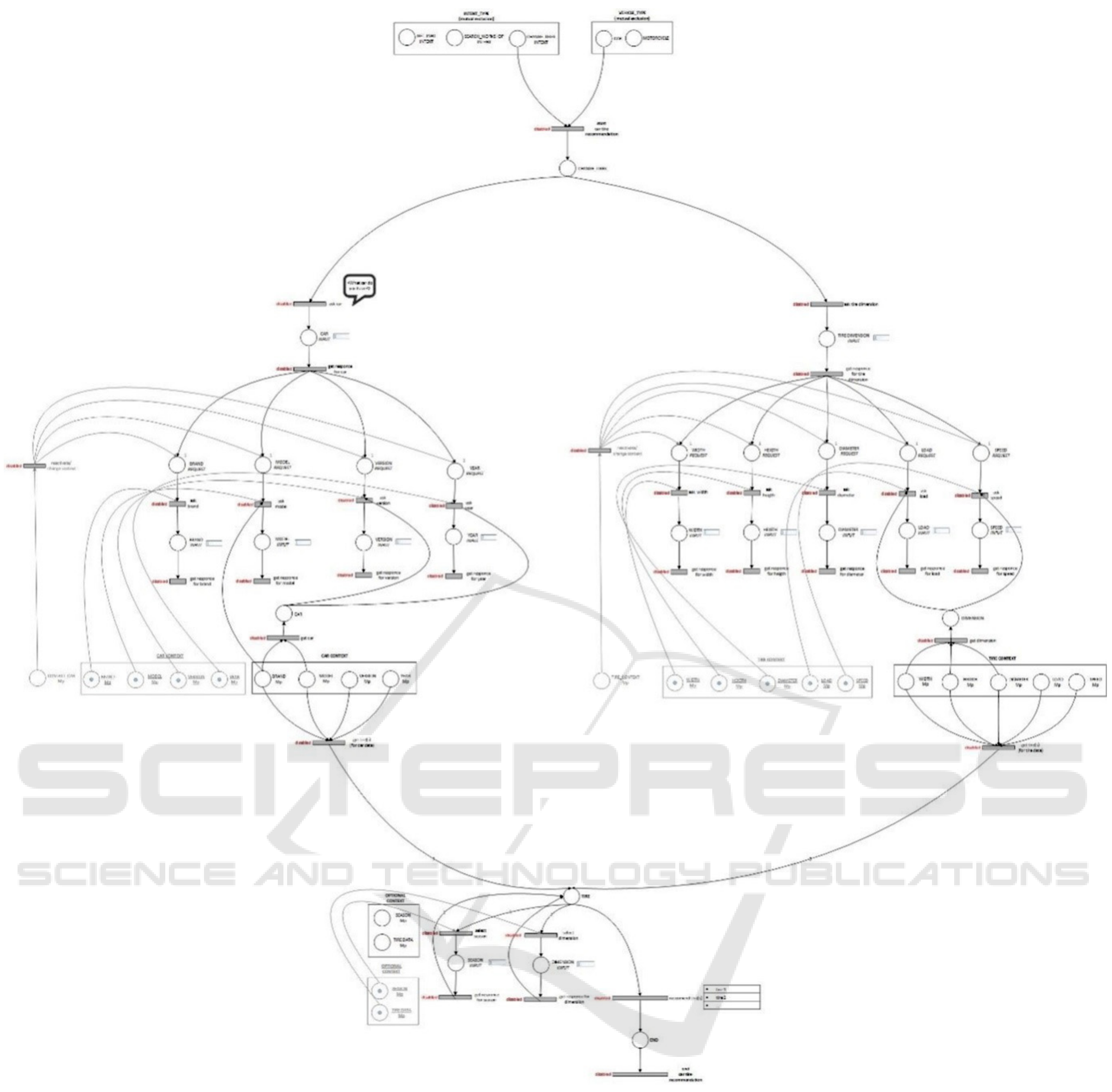


Figure 6: Petri Net of BotWheel's Workflow Manager (Part II).

6 CONCLUSIONS

In this paper, an original approach to a Chatbot has been introduced. In particular, the proposed system is based on the Petri Net formalism. A real case has been investigated developing a Chatbot, BotWheels, for a tires' seller. The results obtained by the experimental campaign are satisfying and show the good perspective of this kind of approach. Further developments involve the application of the proposed approach in various contexts and an improvement of the recommender approach.

REFERENCES

M. Casillo, F. Colace, S. Lemma, M. Lombardi, and A. Pietrosanto, "An Ontological Approach to Digital Storytelling," *In Proceedings of the The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016, Data Science 2016 (p. 27). ACM, 2016.*

G. Pilato, A. Augello, G. Vassallo and S. Gaglio, "Sub-Symbolic Semantic Layer in Cyc for Intuitive Chat-Bots," *International Conference on Semantic Computing, 2007.*

F. Kronlid and T. Lager, "Implementing the Information-State Update Approach to Dialogue Management in a

Slightly Extended SCXML," *Proceedings of the 11th International Workshop on the Semantics and Pragmatics of Dialogue (DECALOG), 2007.*

S. Gianvecchio, M. Xie, Z. Wu, and H. Wang, "Humans and Bots in Internet Chat: Measurement, Analysis, and Automated Classification*," IEEE/ACM Transactions On Networking, Vol. 19, No. 5, 2011.*

V. Gligorijevic, M. Skowron and B.Tadic, "Directed Networks of Online Chats: Content-Based Linking and Social Structure," *Eighth International Conference on Signal Image Technology and Internet Based Systems, 2012.*

S. M. Rodrigo, and J. G. F. Abraham, "Development and Implementation of a Chat Bot in a Social Network," *Ninth International Conference on Information Technology- New Generations, 2012.*

E. Haller and T. Rebedea, "Designing a Chat-bot that Simulates an Historical Figure*," 19th International Conference on Control Systems and Computer Science, 2013.*

G. Ravichandran, N.M. Reddy and G. S. Shriya, "dBot: AI Based Conversational Agent," International Journal of Science and Research (IJSR) *ISSN (Online): 2319-7064 Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391.*

J. Bang, H. Noh, Y. Kim and G.G. Lee, "Example-based Chat-oriented Dialogue System with Personalized Long-term Memory*," International Conference on Big Data and Smart Computing (BigComp), 2015.*

J. Peniche-Avilés, C. Miranda-Palma, L. Narváez-Díaz and E. Llanes-Castro, "AluxBot - A Chatbot that Encourages the Care for the Environment*," IJCSI International Journal of Computer Science Issues, Volume 13, Issue 6, November 2016.*

M. Yan, P. Castro, P. Cheng and V. Ishakian, "Building a Chatbot with Serverless Computing*", Proceedings of the 1st International Workshop on Mashups of Things and APIs, 2016.*

M. Fischer, S. Menon and O. Khatib, "From Bot to Bot: Using a Chat Bot to Synthesize Robot Motion," *The AAAI Fall Symposium Series: Artificial Intelligence for Human-Robot Interaction Technical Report FS-16-01, 2016.*

B. Edwards, I. Muniru and A. Cheok, "Robots to the Rescue: A Review of Studies on Differential Medical Diagnosis Employing Ontology- Based Chat Bot Technology," 6 December 2016.

S. Chopram, R. Gianforte and J. Sholar, "Meet Percy: The CS 221 Teaching Assistant Chatbot," *ACM Transactions on Graphics, Vol. 1, No. 1, Article 1, Publication date: December 2016.*

L, Bradeško, J, Starc, D, Mladenic, M, Grobelnik, and M. Witbrock, "Curious Cat: Conversational Crowd based and Context Aware Knowledge Acquisition Chat Bot," *IEEE 8th International Conference on Intelligent Systems, 2016.*

F. Colace, L. Greco, S. Lemma, M. Lombardi, D. Yung, and S.K. Chang, "An Adaptive Contextual Recommender System: a Slow Intelligence Perspective," *The Twenty-Seventh International Conference on Software Engineering and Knowledge Engineering (SEKE), pp. 64-71, 2015*