Conventional Description of a Human-Machine Interaction on a Collaborative Information Retrieval Task

Jean-Baptiste Louvet

Conventional Description of a Human-Machine Interaction on a Collaborative Information Retrieval Task

Jean-Baptiste Louvet

April 2, 2020

Abstract

This document is a translation (from french) the 5^{th} chpter of the PhD thesis of the author [Lou19][1]. It illustrates the use of the collaborative interaction model (introduced in chapter 3 of the thesis, also introduced in a conference paper [LDC+17]) created by the author on a collaborative medical information retrieval task. The goal is to show how this model can be applied to a real world problem and how the different tools avaliable adapt to observed needs in concrete examples.

We study a human-human interaction corpus to model the problem solving process and then extract the main properties of this process to transpose it as possible behaviors[2] gathered in state tables.

Section 1 describes the particularities of collaborative medical information retrieval (CMIR). It also introduces the Cogni-CISMeF corpus, on which our human-human CMIR scenario rely. Following sections develop the stages of this scenario re-written with the state table formalism: section 2 gives the opening, verbalization and verbalization/terminology alignment stages, section 4 gives the results evaluation stages and section 5 gives the query repair stages. Section 6 describes the modularity levels of our model. Section 7 highlights the propreties of CMIR kept by our model and its limits. Section 8 gives the formal definition of the predicates used in the tables in previous sections.

1 Human-human collaborative medical information retrieval example

This section gives the characteristics of the collaborative medical information retrieval (CMIR). Our model of the human-human CMIR relies on work carried out during the Cogni-CISMeF project (PI CNRS TCAN 2004-2006).

[1] It can be found at the following URL: `https://phd.louvet.io/documents/manuscrit/Manuscrit_doctorat_Jean-Baptiste_Louvet.pdf`

[2] In this article, a possible behavior is described as a sequence of dialogue acts played with a given goal (inform, request, offer...) during a dialogue. A possible behavior can be either expected or performable. An *expected* behavior is initiated by the human and a *performable* behavior is initiated by the agent. See Dubuisson Duplessis' work for detailed explanations [DPCK17, DD15, Dub14].

First (section 1.1), we explain the collection of a human-human CMIR corpus. Then, we give (section 1.2) the CMIR scenario originating from the study of this corpus. Eventually (section 1.3), we discuss the modification made to this scenario while switching to a human-machine interaction.

1.1 The Cogni-CISMeF corpus

1.1.1 The CISMeF project

The CISMeF (Catalogue et Index des Sites Médicaux de langue Francaise[3] [CIS]) project was launched by the university-affiliated hospital of Rouen in 1995. It indexes all the main websites and documents on health information in french language, with more than 123 000 sites and documents at 2018/09/13. The CISMeF terminology that indexes the ressources is structured in four hierarchy levels: *meta-terms*, corresponding to biological or medical specialties, the *keywords*, based on a terminology derived from MeSH("Medical Subject Headings", a medical metadata system), the *quantifiers*, a hierarchy associated to each keyword and the *ressource types*, describing the nature of the conveyed information. This project lead to the release in 2000 of Doc'CISMeF, a search engine associated to this catalogue [DLD+01].

1.1.2 The Cogni-CISMeF project

The use of CISMeF by an ordinary user is impeded by the fact that the query language is complex and that the mastery of the CISMeF terminology is necessary to perform efficient queries [Loi08]. To alleviate this hassle, the Cogni-CISMeF project emerged in order to design an assistant agent to ease the access to this service [CDH+10]. A human-human interaction corpus was collected to be used as a base to the development of this agent. It consists of assistance dialogues on a CMIR task using the search engine of CISMeF. Relatively the Golovchinsky's taxonomy [GPB09, GQPG09], the kind of CIR of the Cogni-CISMeF corpus is *of explicit intention*, *synchrone* and *co-localized*.

During these dialogues, each participant has his own role: a *user* having a medical information need and a CISMeF *expert* helping him finding relevant resources. The expert has an access to CISMeF and leads the search by collaborating with the user. In order to have more information on the progression of the interaction and the use of CISMeF, the expert was instructed to verbalize his actions as mutch as possible. Two project members played the expert role and 21 members of the laboratory played the user role. These latter had no expertise in the medical domain. 21 dialogues (9 for an expert and 12 for the other) were recorded and transcribed in this corpus for a total of 1 800 exchanges. An example of dialogue extracted from the corpus is given in table 1.

Loisel's work [Loi08, LKC08] from this corpus lead to a first model of assistant agent for an IR task on CISMeF, based on GoDiS [LLC+00]. This prototype inherited of the limitations of GoDis on the extension to a complex task and the addition of dialogue moves and was not further expanded [Dub14]. This corpus was later annotated with the multifunctional scheme DIT++ and used by Dubuisson Duplessis for the dialogue game model of Dogma [Dub14].

[3]Catalogue and Index of French Language Medical Sites

A1	[...] Perhaps we can try to widen the search in our case if we consider the words we used
B2	We didn't use that much already
A3	Ummm, forget it then
B4	Why remove / we can remove "analysis"
A5	So let's remove "analysis"
B6	And "diagnostic"
A7	Yes
[...]	
A8	[...] I am almost tempted to put diagnostic anyway because / because we will see what it yields
B9	Yes normally it's a diagnostic / ok / let's try like this
A10	We will try like this otherwise we will remove extra things to have some / so I launch the search again with the "cancerology" thematic access, the CISMeF keyword "colon" and the qualifier "diagnostic" without specifying the type of searched resources

Table 1: Translated extract of a dialogue from the corpus (VD06). A is the expert and B the searcher.

1.2 The CMIR scenario

The annotation made with DIT++ on the Cogni-CISMeF corpus shows that the main dimension of the dialogue is *Task* (dialogue acts used to carry out the dialogue's underlying activity), which represents 68,30% of the anotated utterances. The four following dimensions are *Time Management* (9.93%), *Auto-Feedback* (9.44%), *Own Communication Management* (5.31%) and *Turn Management* (2.76%). These four dimension respectively correspond to the dialogue acts signaling that the locutor needs time to formulate his contribution, to the dialogue acts bringing information of the processing by the locutor of the previous utterance, to the dialogue acts indicating that the locutor is modifying his current contribution and to the dialogue acts used to agree on the locutor's role.

We focus on the dimension *Task* and leave the others aside. This decision is motivated by the fact that our goal is to structure the sequencing of the dialogue patterns contributing directly to the achievement of the interaction's underlying task. The other dimensions concern phenomena specific to human-human oral interaction and their integration to our model is not critical.

The analysis of the corpus made it possible to identify and to describe the different stages of the CMIR task, each playing a specific role to carry out the task [LDC+16, LDC+17]. We identified five stages:

Verbalization – the setting of the search topic between the two interlocutors (the user and the expert). The user starts by expressing his information need to the expert. After a first formulation, he can potentially add spontaneous precisions to it. As a reaction, the expert can ask precision if he juges the initial formulation not accurate enough or if he didn't understand it. He also can try to reformulate it in order to have the user validating it. If he considers the formulation as sufficient, or if the user has nothing to add, the expert starts to construct the first query;

Query construction – search of terms in the CISMeF terminology corre-

sponding to the user's verbalization. The aim for the interlocutors is to collaboratively find an alignment between the terminology of the search engine and the usual vocabulary of the user to fill the query form;

Query launch – execution of the current query by the expert. This stage is often implicitly carried out, i.e. without being verbalized;

Results evaluation – this stage starts by the explicit evaluation of the query results by the expert. If they do not seem satisfying, he decides to repair the query without expecting feedback from the user. If they seem satisfying, he shows them to the user. If he juges them as satisfying, the goal is reached and the search ends, the information need being met. If the user juges them as partially satisfying, i.e. not adapted to his profile or not addressing all the aspects of ris information need, or not satisfying, the query must be repaired. The user can also give up the search after a negative evaluation;

Query repair – the expert and the user use tactics to modify the query while respecting the information need of the user. Three tactics where observed: the *precision*, refining the query by adding keyword to it or by specifying them, the *generalization*, simplifying the query by removing keywords or by simplifying them, and the *reformulation*, if the query is not too specific nor too general but give documents that do not satisfy the user. However, these tactics are not mutually exclusive: it is possible to combine a precision or a generalization with a reformulation. The dialogue sample in table 1 est un exemple de réparation de requête.

On top of these five stages, an openinng stage and a closing stage were observed. The opening stage is optional and consists in greetings. The closing stage can bring propositions of a new search.

The scenario depicted in Figure 1 synthesizes the stages and their possible sequence. The states in dotted lines correspond to actions that can be implicitly carried out, i.e. without verbalization form the interaction participants. We highlight the presence of a launch/evaluation/repair loop.

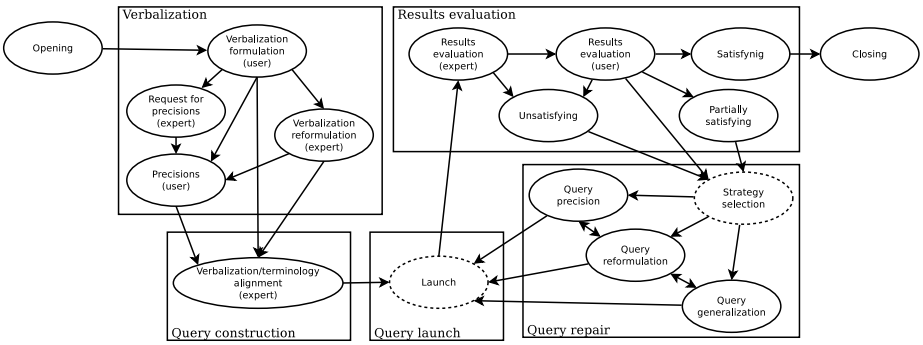

Figure 1: Scenario from the analysis of the corpus representing the possible sequencing of the stages of the CMIR task.

Different properties of information retrieval and collaborative information retrieval were highlighted in this corpus. In particular, the problem solving process used by the participant is an iterative, opportunistic, strategic and interactive process [Dub14]:

Iterative – requires to repeat the same process to meet the information need of the user by successively refining the query formulation [SE98, MW07]. The user's information need is not fixed but evolves when he discovers new resources. This aspect is illustrated by the repetition of the query launch/evaluation/repair pattern, depicted by a loop in the secnario directed graph of figure 1;

Opportunistic – the discovery of some resources can shift the user's perception of his information need and lead the search in an *unexpected* direction. It corresponds to Bates' *berrypicking* model, that describes information retrieval as a dynamic process evolving depending on the found ressources during its execution [Bat89]. This opportunistic aspect makes the interaction hard to predict and so hard to plan from the very start.

Strategic – according to the observations of Bates [Bat79, Bat90], different levels of strategies were observed in the corpus, corresponding to the *moved*, *tactics*, *stratagems* and *startegies* moving the IR task forward;

Iteractive – the search of an answer to the information need of the user requires numerous exchanges between him and the helping expert. During these exchanges, the participants to the search mutually influence each other to further the task, corresponding to the definition of an interaction [Wag94].

1.3 Human-human and human-machine CMIR comparison

We now discuss the modifications brought by the shift from a human-human interaction to a human-machine. To design the states corresponding to the CMIR scenario, we do not have an approach *by emulation*, but by compcomplementarity, which acknowledges the assimetry between human and machines and makes use of it to develop interaction and collaboration means [Ter95, Fis01, Suc07]. The structure of the human-machine task turns out to be modified when compared with the human-human introduced in section 1.2.

We now discuss the structural divergences of the human-machine model vis-à-vis the human-human model.

Our goal while shifting to a human-machine representation is not to copy the human-human interaction but to keep its main properties to help the user to perform his task. The interaction is *de facto* modified on three aspects:

- the transition from a human to a collaborative agent bring a change in the skills shared by the speakers. for example, the agent is able to launch queries "in private", that is to say without informing the user, in order to evaluate them and take decisions before interacting with the user. Hence, the results evaluation by the expert, which was explicitly performed during the human-human interaction, will be implicitly performed during the human-machine interaction, and will therefore not be turned into a state (cf. section 4.1);

- the different repair strategies are merged into a single one. Indeed, if it relevant to distinguish between the three strategies that may be used to repair the query during the study of the information retrieval task, this distinction is not necessary anymore while modeling the intraction and a single state will group these three strategies;
- the collaborative situation is not the same. In the COGNI-CISMEF corpus, the CISMEF expert have the upper hand on the interaction and leads the search, the user observing his actions. In the human-machine situation, we prefer letting the priority to the user, considering that his decisions prevail over the machine's ones. We apply here Lieberman's rule, that suggests, to desing an autonomous interface agent, to *suggest rather than act.*

In order to let the upper hand to the user over the interaction and the task, the system will not take decisions instead of him and will only be able to make offers to him. It's according to this point of view that the rules of the states are built, to take into account the implicit part of the user's behavior an to ensure that the system is following the user's actions and not the other way around.

2 Opening, verbalization and construction

This section and the three following (3, 4 and 5) introduce the application of our collaboration formal model (introduced in previous work [LDC+17]) to the CMIR introduced in section 1.2. The translation of each state of the human-human into a state table is discussed. Each table bulit from the scenario comes with explanations necessary to its understanding ant of the desciption of the prediates it contains. As discussed in section 1.3, a state of the human-human scenario is not necessarily turned into a state table and therefore can not appear in the human-machine scenario.

We first describe the opening, verbalization and query construction phases using states table of opening (table 2.1), of verbalization formulation (table 2.2), of verbalization reformulation (table 2.3), of request for precision (table 2.4), of precision of the verbalization (table 2.5) and of verbalization/terminology alignment (table 2.6).

Writing conventions We use x to represent the user and y to represent the system. We consider i as the current time (T_i represents the current version of te dialogue gameboard) and j the itme at which the behavior in question is closed.

The preconditions and prerequisite are defined with predicates, seen as deductions made by the agent over the interaction state. This way, we are not concerned by their definition while writing states, but simply by the semantics. There si a clear split done between their use in the states and the way they are computed. They link up the current state of the interaction and the activation of a dialogical behavior. Their formal definition is given in section 8

We make a difference between the predicates conveyed by propositional commitments, and the predicates computed by the system. The computed predicates start by an uppercase letter and the ones conveyed by propositional commitments start by a lowercase letter. For example, the predicate verbComplete

is used for the user's commitment on the fact that he ended his verbalization, thus we find it in the commitment C(x, verbComplete, **Crt**). The system will use this commitment to deduce the fact the user's verbalization is over, respresented by the predicate VerbalisationComplete which definition is given in predicate 4. The predicates deduced on the dialogue gameboard must be taken from the point of view of the system, as they will only intervene in its deductions.

2.1 Opening

The opening state is very simple, it is the one in which the interaction starts. Its table is given in table 2.1. It contains only one performable behavior, which function is to inform the user (using the predicate function(*helpForIR*)) that the system's function is to help him performing his information retrieval task.

Opening	
Prerequisite	InteractionNotStarted
Rules	$\varnothing$
Performable behavior	
Dialogue pattern	Inform(y, function(*helpForIR*))
Precondition	$\neg$ SystemFunctionShared
Rules	$\varnothing$
Effects	$\mathrm{T}_j \ni \mathrm{C}(y, \text{function}(\textit{helpForIR}), \textbf{Crt})$

State table 2.1: Opening state

With, as deduced predicates:

InteractionNotStarted true if the interaction didn't start yet;

SystemFunctionShared true if the system shared with the user the fact that its function is to help him on an IR task.

2.2 Verbalization formulation

The scenario given in section 1.2 indicates that the verbalization formulation state follows the opening of the interaction. The table of this state is given in table 2.2. Its prerequisites correspond to the fact that the system shared its function with the user (SystemFunctionShared defined in predicate 2) and that the latter didn't yet formulate any verbalization (VerbalisationMissing defined in predicate 3). It corresponds to the moment when the user describes his information need in natural language and contains only expected behaviors. The first behavior corresponds to the fact that the user expresses a part of his information need, using the information transfer communication game Inform on the predicate verbExpression(*expr*). The second behavior corresponds to the fact that the user states that he is done with expressing his information need, by using the information transfer communication game Inform on the predicate verbComplete.

This state uses a new deduced predicate

VerbalisationMissing true if the user didn't yet formulate his verbalization.

Verbalization formulation		
Prerequisite	$\wedge$	SystemFunctionShared VerbalisationMissing
Rules	$\varnothing$	
Expected behavior		
Dialogue pattern	Inform(x, verbExpression(*expr*))	
Rules	$\varnothing$	
Effects	$T_j \ni C(x, \text{verbExpression}(expr), \textbf{Crt})$	
Expected behavior		
Dialogue pattern	Inform(x, verbComplete)	
Rules	$\varnothing$	
Effects	$T_j \ni C(x, \text{verbComplete}, \textbf{Crt})$	

State table 2.2: Verbalization state, $i < j$

2.3 Verbalization reformulation

In the reformulation sate, the system tries to interpret the user's verbalization and to express it in another way in order to check if its interpretation is correct according to the user. This state is described in state table 2.3. Its prerequisites describe the fact that the user's verbalization must be over (VerbalisationComplete defined in predicate 4) and that it must be complete enough (VerbalisationSufficient defined in predicate 6).

The state contains only one behavior, performed by the system, where it uses a verification game to check if its reformulation is correct. This verification game concerns the predicate isReformulation(*expr*, *ref*), which is true if *ref* is a reformulation of *expr*. The precondition of this behavior is that the system found a reformulation of the user's verbalization, corresponding to the predicate Reformulation(*expr*, *ref*), defined in predicate 7.

Verbalization reformulation		
Prerequisite	$\wedge$	VerbalisationComplete VerbalisationSufficient
Rules	$\varnothing$	
Performable behavior		
Dialogue pattern	CheckQuestion(y, ?isReformulation(*expr*, *ref*))	
Precondition	Reformulation(*expr*, *ref*) $\wedge$ VerbExpression(*expr*)	
Rules	$\varnothing$	
Effects	$\vee$	$T_j \ni C(x, \text{isReformulation}(expr, ref), \textbf{Crt})$ $T_j \ni C(x, \neg\ \text{isReformulation}(expr, ref), \textbf{Crt})$

State table 2.3: Verbalization reformulation state, $i < j$

With, as deduced predicates:

VerbalisationComplete true if the user stated that his verbalization is complete;

VerbalisationSufficient true if the user's verbalization is rich enough;

Reformulation(*expr*, *ref*) true if the system considers that *ref* is a reformulation of *expr*;

VerbExpression(*expr*) true if *expr* is an expression of the verbalization.

2.4 Request for precision of the verbalization

The state of request of precision occurs when the user ended his verbalization but system considers that it is not complete enough. So the state has for prerequisite the predicate VerbalisationComplete. The predicate PrecisionNotAsked is added to prevent the system to ask repeatedly the precision of his verbalization to the user. The predicate QueryEmpty prevents asking to the user to precise his verbalization if he's already formulating a qury. The only possible behavior of the state is emited by the system, which asks to the user to perform the action to precise his verbalization (represented by preciseVer) with the Request dialogue game. The preconditions of this behavior correspond to the fact that the user's verbalization is not sufficient. The rules describe that if the user accepts the system's request, then his commitment on the fact that the verbalization is complete is *canceled*, that is to say switched from the state **Crt** to **Ina**.

Request for precision of the verbalization		
Prerequisite	$\wedge$	VerbalisationComplete PrecisionNotAsked QueryEmpty
Rules	$\varnothing$	
Performable behavior		
Dialogue pattern	Request(y, preciseVerb)	
Precondition	$\neg$ VerbalisationSufficient	
Rules	Succes $\Rightarrow$ C(x, verbComplete, **Ina**)	
Effects	$\vee$	$\wedge$ $\mathrm{T}_j \ni$ C(x, preciseVerb, **Crt**) $\mathrm{T}_j \ni$ C(x, verbComplete, **Ina**) $\mathrm{T}_j \ni$ C(x, preciseVerb, **Fal**)

STATE TABLE 2.4: Request for precision of the verbalization, $i < j$

This state uses the following new predicates:

PrecisionNotAsked true if the system didn't yet ask the user to precise his verbalization;

QueryEmpty true if the query does not contain any keyword.

2.5 Verbalization precision

The verbalization precision state occurs when the user wants to bring a complement to his verbalization. Its state table is shown in table 2.5. In the human-machine interaction preformed ont the CMIR task, we consider that the complement to the user's verbalization will be performed only on the system's demand, done in the request for precision state (as opposed to the human-human interaction where the user can give spontaneus precisions). The prerequisites of the precision state are only the fact that the user is commited to precise his

verbalization, with the predicate UserWillPreciseVerbalisation. In this state, two behaviors are expected. The first corresponds to the fact that the user adds informations to his verbalization, using the Inform communication game. The rules of this behavior specify that the opening of an Inform dialogue pattern on an expression of the verbalization lead to the satisfaction of the extra-dialogical action commitment made in the request for precision state. This latter then switches into the **Ful** state. The second expected behavior corresponds to the fact that the user indicates that he ended his verbalization, using an Inform communication game. Its rules state that if the user opens an Inform dialogue pattern on the fact that his verbalization is over, but is still commited on the action of completing it (so he didn't add anything to the verbalization), then he *violates* this action commitment, which state is switched to **Vio**.

Verbalization precision		
Prerequisite	UserWillPreciseVerbalisation	
Rules	$\varnothing$	
Expected behavior		
Dialogue pattern	Inform(x, verbExpression($expr$))	
Rules	Entry $\Rightarrow$ C(x, preciseVerb, **Ful**) $\lfloor$ $T_i \ni$ C(x, preciseVerb, **Crt**)	
Effects	$\vee$	$T_j \ni$ C(x, verbExpression($expr$), **Crt**) $T_j \ni$ C(x, preciseVerb, **Ful**)
Expected behavior		
Dialogue pattern	Inform(x, verbComplete)	
Rules	Entry $\Rightarrow$ C(x, preciseVerb, **Vio**) $\lfloor$ $T_i \ni$ C(x, preciseVerb, **Crt**)	
Effects	$\vee$	$T_j \ni$ C(x, verbComplete, **Crt**) $T_j \ni$ C(x, preciseVerb, **Vio**)

State table 2.5: Verbalization precision state, $i < j$

This state uses the following deduced predicate:

UserWillPreciseVerbalisation true if the user commited himself to precise his verbalization.

2.6 Verbalization/terminology alignment

The query construction phase contains only one state, the one of verbalization/terminology alignment, introduced in table 2.6. Its prerequisites cover three situations: the user verbalized his information need (VerbalisationComplete, predicate 4) and this one is sufficient (VerbalisationSufficient, predicate 6), or he gave a verbalization and the agent didn't ask for precisions (VerbalisationComplete and PrecisionNotAsked, predicate 8) or the request for precisions from the system failed (VerbalisationPrecisionFailed, predicate 12). In any case, to have the prerequisites activated, it is necessary that the query was not launched. This state contains an expected behavior, corresponding to the addition of a keyword to the query, performed using the Request dialogue game on the action of adding the desired keyword (addKeyWord(kw)). The rules of this behavior indicate that the system is not allowed to turn down this request and that he

instantly performs the action, leading to the satisfaction of the corresponding action commitment and to the creation of the commitment on the fact that the keyword is part of the query.

The other possible behavoir of the state corresponds to the same action but initated by the system. The system can offer to the user to add a keyword to the query, using de Offer dialogue game. The preconditions indicate that the keyword kw must not be part of the current query ($\neg$QueryKeyWord(kw), predicate 13) and that the addition of the keyword to the query is relevant (RelevantForAddition(kw), predicate 14). The rules of this behavior correspond to the direct execution of the action suggested by the system.

Verbalization/terminology alignement		
Prerequisite	$\wedge$	QueryNeverLaunched $\vee$ $\wedge$ VerbalisationComplete VerbalisationSufficient $\vee$ PrecisionNotAsked $\vee$ VerbalisationPrecisionFailed
Rules	$\varnothing$	
Expected behavior		
Dialogue pattern	Request(x, addKeyWord(kw))	
Rules	$\neg$ Failure Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succes $\Rightarrow$ C(y, addKeyWord(kw), **Ful**)	
Effects	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**)
Performable behavior		
Dialogue pattern	Offer(y, addKeyWord(kw))	
Precondition	$\neg$ QueryKeyWord(kw) $\wedge$ RelevantForAddition(kw)	
Rules	Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succes $\Rightarrow$ C(y, addKeyWord(kw), **Ful**)	
Effects	$\vee$	$\wedge$ $T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**) $\vee$ $T_j \ni$ C(y, addKeyWord(kw), **Fal**)

STATE TABLE 2.6: Verbalization/terminology alignment state, $i < j$

With these new deduced predicates:

QueryNeverLaunched true if no query launch has already been performed;

VerbalisationPrecisionFailed true if the verbalization precision failed, that is to say that the user declined to precise his verbalization or commited to do it but did not;

QueryKeyWord(kw) true if the keword kw is a keyword of the current query;

RelevantForAddition(kw) true if the addition of the keyword kw to the query is relevant to precise the query.

3 Query launch

The query launch phase contains only one state, that corresponds to the action of launching the current query by one interlocutor or the other. This state is presented in table 3.1. The preconditions of this state simply indicate that the query must be launchable (QueryLaunchable, predicate 15). The expected behavior makes it possible for the user to ask the system to launch the current query using the Request dialogue game. As for an addition of a keyword to the query, the system can not turn down this request and executes it instantly, leading to the satisfaction of the corresponding action commitment and to the creation of the commitment on the fact that the query has been launched.

The system can also offer to the user to launch the current query using the Offer dialogue game. The preconditions indicate that the current query launch must be relevant (QueryLaunchRelevant, predicate 16). As for the previous behavior, the rules correspond to the direct execution of the action offered by the system.

Query launch		
Prerequisite	QueryLaunchable	
Rules	$\varnothing$	
Expected behavior		
Dialogue pattern	Request(x, launchCurrentQuery)	
Rules	$\neg$ Failure Succes $\Rightarrow$ C(y, currentQueryLaunched, **Crt**) Succes $\Rightarrow$ C(y, launchCurrentQuery, **Ful**)	
Effects	$\wedge$	$T_j \ni$ C(y, currentQueryLaunched, **Crt**) $T_j \ni$ C(y, launchCurrentQuery, **Ful**)
Performable behavior		
Dialogue pattern	Offer(y, launchCurrentQuery)	
Precondition	QueryLaunchRelevant	
Rules	Succes $\Rightarrow$ C(y, currentQueryLaunched, **Crt**) Succes $\Rightarrow$ C(y, launchCurrentQuery, **Ful**)	
Effects	$\vee$	$\wedge$ $T_j \ni$ C(y, currentQueryLaunched, **Crt**) $T_j \ni$ C(y, launchCurrentQuery, **Ful**) $T_j \ni$ C(y, launchCurrentQuery, **Fal**)

State table 3.1: Query launch state

With the new deduced predicates:

QueryLaunchable true if the current query respects the minimum criteria to be launched;

QueryLaunchRelevant true if the system judges that it is relevant to offer to the user to launch the current query.

4 Results evaluation

The results evaluation phase is the one which structure will be the most impacted by the switch from the human-human scenario to the human-machine

one. As a matter of fact, some parts of the scenario are performed explicitly during the human-human interaction but will be performed implicitly during the human-machine interaction and will therefore not be represented as states. It is in particular the case of the results evaluation by the expert, discussed in section 4.1. It is also the case where states are judged as partially or not satisfying, discussed in section 4.4. Only two states are ultimately kept in this phase. We introduce then in the following of this section: the evaluation of the results by the user, in section 4.2 (table 4.1) and the closing in case of positive evaluation, in section 4.3 (table 4.2).

4.1 Results evaluation by the expert

In the human-human scenario, the evaluation of the results returned by the search engine is explicitly performed by the expert, who comments them to share his judgment to his interlocutor. This situation will be completely modified during the shift to a human-machine interaction, given that the system will have access to the query results before they are presented to the user. It is thus possible to evaluate them in an "hidden" way, that is to say during the computation of the predicates, in the states before the query launch. This situation makes it possibe to get ahead on the consequences that the offer to adde a keyword to the query may have or to launch it. It is for example what is done with the predicates QueryLaunchable (predicate 15) and QueryLaunchRelevant (predicate 16), that take into account the results returned by the current query in their evaluation.

The ability to anticipate the query results evaluation gives an edge to the system which then have the ability to know the consequences of an action that he considers to offer to the user. The predicate QueryLaunchable (predicate 15) illustrates this interest as it prevents to offer to the user to launch a query that may return no result (this is what justifies its use as precondition of the query launch table 3.1).

This implicit evaluation performed *a priori* by the sistem is part of its private mecanics and is not visible for the user. Therefore, it doesn't exists a state of query results evaluation by the system strictly speaking. The only informations on a query evaluation by the system that are made public are the ones that the system judges as relevant during the query construction. Instead of having access to the integrality to the evaluation performed by the system, the user will only have access to the part that the system considers as useful. It is then to the user to decide if what the system offers is interesting and to take it into account or not.

4.2 Results evaluation by the user

The results evaluation phase is greatly simplified in our vision of the human-machine interaction and is mainly focused on the results evaluation by the user, described in table 4.1. The prerequisite of this state is the fact that the current querry has been launched (CurrentQueryLaunched, predicate 17). The three expected behaviors in this state correspond to the fact the user sees ressources (seenResource(r)), evaluate them (relevantResource(r) or $\neg$ relevantResource(r)) and gives feedback on the satisfaction of his information need

by the results of the current query(currentQueryResultsSatisfying or ¬ currentQueryResultsSatisfying). These behaviors are performed using the Inform communication game. The performable behavior of this state corresponds to the fact that the system asks to the user if his research is osver, using the ChecrQuestion dialogue game, on the predicate researchOver. The preconditions of this behavior specify that the user must have evaluated as satisfying the results of the current query (CurrentQueryResultsSatisfying, predicate 18), that he didn't express the fact that his research is over (ResearchNotOver, predicate 19) and that the system didn't already ask him if his research is over (ResearchOverNotAsked, predicate 20) since the last time he expressed that the query results are satisfying. In the case where the user negate the fact that his research is over (he plays a *disconfirm* act), leading to a failure of the dialogue pattern, a rule is applied and disables the commitment of the user on the fact that the current query results are satisfying, if this commitment is activated.

Results evaluation (user)		
Prerequisite	CurrentQueryLaunched	
Rules	∅	
Expected behavior		
Dialogue pattern	Inform(x, seenResource(r))	
Rules	∅	
Effects	$T_j \ni C(x, \text{seenResource}(r), \textbf{Crt})$	
Expected behavior		
Dialogue pattern	Inform(x, relevantResource(r) ⌊ ¬ relevantResource(r))	
Rules	∅	
Effects	$T_j \ni C(x, \text{relevantResource}(r) \lceil \neg \text{relevantResource}(r), \textbf{Crt})$	
Expected behavior		
Dialogue pattern	Inform(x, currentQueryResultsSatisfying ⌊ ¬ currentQueryResultsSatisfying)	
Rules	∅	
Effects	$T_j \ni C(x, \text{currentQueryResultsSatisfying} \lceil \neg \text{currentQueryResultsSatisfying}, \textbf{Crt})$	
Performable behavior		
Dialogue pattern	CheckQuestion(y, ?researchOver)	
Precondition	CurrentQueryResultsSatisfying ∧ ResearchNotOver ∧ ResearchOverNotAsked	
Rules	Failure ⇒ C(x, currentQueryResultsSatisfying, **Ina**) ⌊ $T_i \ni C(x, \text{currentQueryResultsSatisfying}, \textbf{Crt})$	
Effects	∨ $T_j \ni C(x, \text{researchOver}, \textbf{Crt})$	
	∧ $T_j \ni C(x, \neg \text{researchOver}, \textbf{Crt})$	
	∧ $T_j \ni C(x, \text{currentQueryResultsSatisfying}, \textbf{Ina})$	

State table 4.1: Results evaluation by the user state

With the deduced predicates:

CurrentQueryLaunched true if the current query has been launched;

CurrentQueryResultsSatisfying true if the results of the current query were judged satisfying by the user;

ResearchNotOver true if the user didn't express the fact that his research is over;

ResearchOverNotAsked true if the system didn't ask to the user if his research is over since the last time he expressed the fact that the query results are satisfying.

4.3 Closing

The closing of the interaction corresponds to the moment when the task is completed and where the research session is over. The closing state is then corresponding to the moment when the system informs the user that their interaction is over. This action is acptured in the possible behavior of the state, which dialogue pattern is the Inform communication game. The prerequisite of the state, along with the preconditions of the behavior, are the fact that the research is over (ResearchOver, predicate 21).

Closing	
Prerequisite	ResearchOver
Rules	$\varnothing$
Performable behavior	
Dialogue pattern	Inform(y, interactionOver)
Precondition	ResearchOver
Rules	$\varnothing$
Effects	$\mathrm{T}_j \ni \mathrm{C}(y, \text{interactionOver}, \mathbf{Crt})$

State table 4.2: Closing state

ResearchOver true if the user states that his research is over.

4.4 Partially- and non-satisfaying results

As for the results evaluation by the system, this state is not transcribed in the human-machine version of the interaction on a CMIR task. Indeed, as the results evaluation by the system is implicitly performed (cf. section 4.1), the rejection of the query results will not be to its initiative in the results evaluation phase. On top of that, the satisfaction level of the user according to the results returned by the query has already been expressed in his results evaluation state. The fact that the query results are satisfying or partially satisfying will be determined only from the user's actions during the results evaluation.

5 Query repair

In the query repair phase, we find the three strategies used in the human-human scenario: query precision, generalization and reformulation. As discussed in section 1.3, we do not define a state by strategy. As a matter of act, each stratey requires simple operations on the query (addition, suppresion or substitution of keyword) and it's the preconditions of each associated behavior that will make it possible for the system to use some of these operations to apply these strategies. The distinction between the different strategies in thus done during the computation of the predicates rather than during the definition of the query repair state itself.

The query repair state containing numerous behaviors, we split them into two tables: one for the behaviors expected from the user (table 5.1) and one for the behaviors performable by the system (table 5.2).

Comportements de réparation (attendus)		
Prerequisite	$\wedge$	CurrentQueryLaunched $\neg$ CurrentQueryResultsSatisfying
Rules		$\_ \Rightarrow$ C(x, $\neg$ currentQueryResultsSatisfying, **Crt**) $\lfloor$ $T_i \ni$ C(x, $\neg$ currentQueryResultsSatisfying, **Ina**)
Expected behavior		
Dialogue pattern		Request(x, addKeyWord(kw))
Rules		$\neg$ Failure Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succes $\Rightarrow$ C(y, addKeyWord(kw), **Ful**) Succes $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effects	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**)
Expected behavior		
Dialogue pattern		Request(x, removeKeyWord(kw))
Rules		$\neg$ Failure Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succes $\Rightarrow$ C(y, removeKeyWord(kw), **Ful**) Succes $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effects	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Ina**) $T_j \ni$ C(y, removeKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**)

State table 5.1: Comportements attendus de l'état de réparation de la requête.

The main rule states that if we enter in a state of the repair phase, it means that the current query results are not satisfying. The first expected behavior corresponds to the request from the user to add a keyword to the query, performed with the Request dialogue game. This expected behavior is similar to the one from the verbalization/terminology alignment state depicted in table 2.6. Its rules specify that the system is not allowed to turn down the request of the user and that this one is executed right away. The second expected behavior is the request is his request to delete a keyword from the query, using the Request dialogue game. The rules of this behavior indicate that the system cannot turn down this request and executes its without delay. For the two expected behaviors, in regard to the verbalization/terminology alignment state, a rule is added indicating that the acceptance by the system of this request deactivates the commitments on the fact that the current query has been launched.

On the system's side, three behaviors are performable. The two first are similar to the user's ones: the first consists in offering to the user to add a keyword to the query, as in the verbalization/terminology alignment state. The seconds consists in offering to the user to remove a keyword from the query. The preconditions to make it possible to offer this behavior ensure that the offered keyword (kw) belongs to the current query and that its removal is relevant. The

Comportements de réparation (réalisables)		
Prerequisite	$\wedge$	CurrentQueryLaunched $\neg$ CurrentQueryResultsSatisfying
Rules		_ $\Rightarrow$ C(x, $\neg$ currentQueryResultsSatisfying, **Crt**) $\lfloor$ $T_i \ni$ C(x, $\neg$ currentQueryResultsSatisfying, **Ina**)
Performable behavior		
Dialogue pattern		Offer(y, addKeyWord(kw))
Precondition		$\neg$ QueryKeyWord(kw) $\wedge$ RelevantForAddition(kw)
Rules		Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succes $\Rightarrow$ C(y, addKeyWord(kw), **Ful**) Succes $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effects	$\vee$	$\wedge$ $T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**) $T_j \ni$ C(y, addKeyWord(kw), **Fal**)
Performable behavior		
Dialogue pattern		Offer(y, removeKeyWord(kw))
Precondition		QueryKeyWord(kw) $\wedge$ RelevantForRemoval(kw)
Rules		Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succes $\Rightarrow$ C(y, removeKeyWord(kw), **Ful**) Succes $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effects	$\vee$	$\wedge$ $T_j \ni$ C(x, queryKeyWord(kw), **Ina**) $T_j \ni$ C(y, removeKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**) $T_j \ni$ C(y, removeKeyWord(kw), **Fal**)
Performable behavior		
Dialogue pattern		Offer(y, substituteKeyWord(kw, skw))
Precondition		QueryKeyWord(kw) $\wedge$ RelevantForSubstitution(kw, skw)
Rules		Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succes $\Rightarrow$ C(x, queryKeyWord(skw), **Crt**) Succes $\Rightarrow$ C(y, substituteKeyWord(kw, skw), **Ful**) Succes $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effects	$\vee$	$\wedge$ $T_j \ni$ C(x, queryKeyWord(kw), **Ina**) $T_j \ni$ C(x, queryKeyWord(skw), **Crt**) $T_j \ni$ C(y, substituteKeyWord(kw, skw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**) $T_j \ni$ C(y, substituteKeyWord(kw, skw), **Fal**)

STATE TABLE 5.2: Comportements réalisables de l'état de réparation de la requête.

rules indicate that if the user accepts this offer, is its put into action immediately by the system. The third behavior makes it possible for the system to offer to the user to substitute a keyword by another. This behavior is useful especially for the implementation of query repair strategies that imply the simplification

or the specification of keywords. The preconditions check that the keyword to substitute (*kw*) belongs to the current query and that the substituting keyword (*skw*) is relevant.

With the deduced predicates:

RelevantForRemoval(*kw*) true if the deletion of the keyword *kw* from the query is relevant;

RelevantForSubstitution(*kw*, *skw*) vrai if the substitution of the keyword *kw* by *skw* is relevant to modify the query, without necessarily precising or simplifying it.

6 Modularity of the model

Our model has three main modularity levels. The first level (section 6.1) makes it possible to generalize the writing of the possible behaviors by focusing only on the semantic content of the associated dialogue pattern and its impact on the task. The second level (section 6.2) shows the interest of the rules to specify the consequences of a dialogue act in a given context. The third level (section 6.3) shows a clear distinction between the conventional of the interaction and the underlying intentions which make it possible for the agent to decide which behavor to play.

6.1 Behaviors as atomic operations

The main element of a behavior is the associated dialogue pattern. This pattern conveys an information transfer or an action discussion. It corresponds to a *atomic operation* that can be performed to push the task forward. The organization of these operations with resect to each other explain the way the task is performed. A possible behavior is a brick to put at the appropriate places of the task execution. It's in this way that the state tables are defined, grouping these operations into coherent units. Indeed, some behaviors ar not unique to a single state and can be resued somewhere else. One operation can fit at different times of the task completion, that is to say in different states. It entails that one possible behavior can be present in several states. The writing of a behavior can thus be seen not as a definition specific to a limited part of the interaction, but as the design of a brick that one will put in use at the appropriate moments of the task performance.

It is for example the case of the overlap of the possible behaviors in the states of verbalization/terminology alignment and query repair.

This point of view makes it possible to consider designing an interaction by separating the expected actions of the interlocutors to perform the task and the interactions necessary to perform these actions. To model the task execution, it is only necessary to organize the information transfert and the action discutions contributing to pushing it forward. It seems possible, as soon as one has defined a set of behaviors encapsulation these phenomena of information transfer and actions discussion, to do not care of the interactions that will convey them and to focus on their organization among states.

It also greatly simplifies the reading and the writing of the tables, these ones boiling down for each line to an information transfer or to an action discussion,

this representation synthesizing the important informations and hiding the ones specific to the interaction.

6.2 The rules as contextual specifications

This "behavior-wise" modularity must be put into perspective depending on the context of each state. As explained in the description of a state (see the author's word for the definition of a state [LDC$^+$17, Lou19]), the possible behaviors do not only correspond to a given dialogical patter, and also include rules. However, for a given dialogue pattern, the rules of the behavior to which it belongs can vary depending on the state in which it is. For example, when the user formulates a verbalization,t he rules change depending on the context. In the case of the verbalization formulation case (section 2.2), no rule is applied, wheras if the case of the state of verbalization precision (section 2.5), a rule exists specifying that the opening of the pattern by the user leads to the completion of the associated extra-dialogical commitment (creation of C(x, preciseVerb, **Ful**)). Both these behaviors are reported in the tables 6.1 for the verbalization formulation state and 6.2 for the verbalization precision state.

Verbalization formulation behavior	
Expected behavior	
Dialogue pattern	Inform(x, verbExpression($expr$))
Rules	$\varnothing$
Effects	$\mathrm{T}_j \ni \mathrm{C}(x,$ verbExpression($expr$), **Crt**)

STATE TABLE 6.1: Verbalization expression formulation behvior taken from the verbalization formulation state (table 2.2).

Verbalization precision behavior		
Expected behavior		
Dialogue pattern		Inform(x, verbExpression($expr$))
Rules		Entry $\Rightarrow \mathrm{C}(x,$ preciseVerb, **Ful**) $\lfloor\ \mathrm{T}_i \ni \mathrm{C}(x,$ preciseVerb, **Crt**)
Effects	$\vee$	$\mathrm{T}_j \ni \mathrm{C}(x,$ verbExpression($expr$), **Crt**) $\mathrm{T}_j \ni \mathrm{C}(x,$ preciseVerb, **Ful**)

STATE TABLE 6.2: Verbalization expression formulation behavior taken form the verbalization precision state (table 2.5).

However, we stand for the idea the it is possible to give a "minimalistic" definition of a possible behavior and its rules, and to specify them depending on the situation we want to apply it to. Numerous rules do not depend on the current context of the interaction but on the general context governing the interaction.

Our agent being collaborative, we decided that it must accept all the requests of the user. It implies that when the user opens a Request dialogue game, the system is forced to ensure that it yields to a success, no matter the action convey by the dialogue game and no matter the current context of the interaction. The same way, when the system is commited to perform an action, it performs

it immediately. This phenomenon is captured by the writing of rules in the expected behavior associated to the dialogue pattern commiting the system on the action (often an Offer or Request dialogue game). The table 6.3 illustrates this aspect on the addition of a keyword to the query: the rules describe the fact that the system can not decline the user's request ($\neg$ Failure) and that it directly adds the keyword (creation of the commitments C(x, queryKeyWord(kw), **Crt**) and C(y, addKeyWord(kw), **Ful**)).

Keyword addition request behavior		
Expected behavior		
Dialogue pattern	Request(x, addKeyWord(kw))	
Rules	$\neg$ Failure	
	Succes $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**)	
	Succes $\Rightarrow$ C(y, addKeyWord(kw), **Ful**)	
Effects	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**)
		$T_j \ni$ C(y, addKeyWord(kw), **Ful**)

State table 6.3: Example of behavior with rules corresponding to constrains corresponding to the general context of the interaction.

Thus, to a certain extent, the rules of the possible behaviors correspond to general principles related to the interaction context. In the case where rules are applied to a specific state, it is possible to add them to the behavior in this state without loosing its genericity. To this is added the mechanism of rules of the whole state, applied during the entry in this state and that make it possible to modify the dialogue state in order to make it correspond to the context of the state. This use of specific rules is a second level of modularity.

6.3 Dissociation of interactions and decisions

A third level of modularity is the independence between the definition of predicates and their use. The way they are computed is not taken into account during the desing of the states. The definition of the predicates given in section 8 for the human-machine interaction on a CMIR task is only a suggestion. We made the choice to define them with logical formulas but it is entirely feasible to do it otherwise without having to modify the conventional structure of the task ginven in sections 2, 4 and 5. The way the predicates are defined is totally independent of the structure of the interaction and it is the responsability of the designer to choose their definition. It makes it possible to modifiy their definition without modifiying the states' definition, dissociating clearly the datastructure used by the agent's reasoning and the task structure.

To sum up, we can say that our possible behavior model dissociates the *why*, the *how* and the *what*:

The why is the precondition of a performable behavior. It explains why the system can perfom this behavior and is activated at the conclusion of deductions performed by the system;

The how it the dialogue pattern of the performable behavior. It descibes how the behavior must be conventionally performed by the participants to the dialogue;

The what is the semantic content of the dialogic pattern of the performable behavior. It corresponds to what is conveyed by the initiator of the pattern as an action or a proposition.

7 Synthesis

We reconsider here the CMIR task and describe how the states given in sections 2, 3, 4 and 5 correspond to this model. We also give the current limits of the model and how to overcome them.

7.1 Preserved properties

We saw in section 1.2 that the CMIR model is interactive, opportunistic, iterative and strategic. These properties are kept during the shift to the human-machine model.

Interactive – By construction of the model, the CMIR task can only be performed by interacting. Any action performed by the human or the machine is integrated to dialogue games, that are interactive patterns themselves organized in coherent structures described by the states;

Opportunistic – Not planning the task makes it possible to change its running in unpredictable ways. The decision-making of the machine is based on the current state of the interaction an can, according to it, offer spontaneously behaviors impossible to anticipate. The same way, the contextualization mechanisms of a user's move make it possible to adapt to numerous actions from him, giving him a wide range of actions and not restricting him to a small set of dialogue moves.

Iterative – The loop launch/evalution/repair is visible in the state tables of the CMIR scenario. The launch is performed thank to behaviors of Request or Offer patterns on the action launchCurrentQuery. After this launch, the predicate CurrentQueryLaunched is true, which makes it possible for the user to enter the results evaluation state (table 4.1). If the evaluation is negative, then the commitment C(x, ¬currentQueryResultsSatisfying, **Crt**) is created. The creation of this commitment y ensure that the prerequisite of the query repair state are satisfied. This states sequencing can be repeated as many times as necessary until the user finds a satisfying answer to his information need;

Strategic – The fact that the task is not explicitly planned does not prevent the use of strategies. It is for example the case for the query repair, where the system decides the most relevant operations on the query depending on the current situation, corresponding to the level of *tactics* of Bates [Bat90]. On top of that, the behaviors expected from the user in the query repair state make it possible for him to use all the necessary moves to develop tactics, stratagems or strategies (according to the meaning of Bates).

In the Cogni-CISMeF corpus, when the query results are not satisfying for the user, he enters directly in the query repair phase without even verbalizing it. The only explicit actions transcribed in the corpus concern the fact that the

query is launched, that the expert submits the results to the user and that the latter directly offers modifications to the query. It means that the user shortcuts the results evaluation state and does not emits any explicit evaluation on the query results. The transition to the repair is thus implicitly done and is deduced by the expert from the user's behavior. However, this deduction is not obvious for the system as it corresponds to a situation where the user does not respect the order in which he is supposed to perform his actions.

It is nevertheless important to make this possible and give to the system the capacity to integrate these shortcuts in its interpretation of the user's behavior. It is with this in mind that the rule of the query repair state has been written (cf. tables 5.1 and 5.2). This rule indicates that, at the entry in this state, the user rejects the results of the last query launched. The user is free to make feedbacks or not on his evaluation of the results of the query, and the system is able to interpret his actions, event implicit ones.

7.2 Limits

It is not possible, in the current state of our model, to perform some dialogue moves, often implicit if the human-human interaction. It is for example the case of the implicit approval from the user of an offer or request from the system. For example, in the request for precisions state, (table 2.4), if the system asked the user to precise his verbalization with the Request(y, preciseVerb) dialogue game and that the user, instead of explicitly accepting with an acceptRequest(x, preciseVerb), directly starts to complete his verbalization with an inform(x, verbExpression(*expr*)), the acceptance will not be implicitly played. The request game will not be answered, wheras it is implicitly over.

We hit here a limit of the model, which as such dose not make it possible do describe all the possible implicit behaviors of the user. However, an extension is possible while preserving the current structure of the formalism. Taking into account these kind of implicit behavior is possible thanks to the rules of the behavior with which the user implicitly accepts the system request. The writing of these rules implies to directly manipulate the dialogical actions of the dialogical pattern of the implicitly manipulated behavior, which is not possible for now. It requires to take into account dialogical action commitments, that we didn't discuss in this work as they are closer to the manipulation of structures specific to dialogue management than to the management of the task itsefl by the interaction. Hence, we let the consideration of the implicit acceptation of action discussion phenomena to future developments.

8 Predicates definition

This section gives a definition, formal if possible, of the predicates used in the previous sections of this document. The value of the predicates depends on the context (mainly the dialogue gameboard). In some definitions, we consider that we have access to external ressources enriching the knowledge and the reasoning capacities of the system, particularly on the domain of natural lanugage processing and information retrieval.

We use the notation $\mathrm{T}_i \ni \mathrm{C}(p, pred|act, s)$ to describe the fact that the dialogue gameboard T_i contains a commitment of the participant p on the

predicate *pred* or the action *act* and that this commitment is on the state *s*.

8.1 Opening, verbalization and query construction

8.1.1 Opening

The predicate InteractionNotStarted used as prerequisite means that the interaction did not start yet (we are at the very beginning of the interaction). It is true if no interaction took place already. It simply checks that the dialogue gameboard is empty.

Prédicat 1 – InteractionNotStarted **:**

$\mathrm{T}_i = \varnothing$

The precondition of the performable behavior is the negation of the predicate SystemFunctionShared. This predicate is true if the system shared with the user the fact that its function is to help him on an IR task. Thus, it checks if the commitment of the system on its functions has been created.

Prédicat 2 – SystemFunctionShared **:**

$\mathrm{T}_i \ni \mathrm{C}(y, \text{function}(\textit{helpForIR}), \textbf{Crt})$

8.1.2 Verbalization formulation

The predicate VerbalisationMissing used as precondition indicates that the user did not yet formulate a verbalization. It checks that the dialogue gameboard does not contain any commitment of the user of a verbalization expression.

Prédicat 3 – VerbalisationMissing **:**

$\nexists e, \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(e), \textbf{Crt})$

8.1.3 Reformulation

The predicate VerbalisationComplete is true if the user asserted that his verbalization is complete. It checks if the user is commited on this proposition.

Prédicat 4 – VerbalisationComplete **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{verbComplete}, \textbf{Crt})$

The predicate VerbExpression(*expr*) is true if the expression *expr* is part of the verbalization of the user's information need.

Prédicat 5 – VerbExpression(*expr*) **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(\textit{expr}), \textbf{Crt})$

The predicate VerbalisationSufficient is true if the user's verbalization is rich enough. We consider that it is rich enough if it contains three different expressions.

Prédicat 6 – VerbalisationSufficient **:**

$$\wedge \left| \begin{array}{l} \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(a), \textbf{Crt}) \\ \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(b), \textbf{Crt}) \\ \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(c), \textbf{Crt}) \\ \mathrm{a} \neq \mathrm{b} \neq \mathrm{c} \neq \mathrm{a} \end{array} \right.$$

The predicate Reformulation(*verb*, *ref*) is true if the linguistic ressources of the system give *ref* as reformulation of *verb*.

Prédicat 7 – Reformulation(*verb*, *ref*) **:**

Use of a natural language processing and/or domain ressource to compute this predicate.

Discussion on these predicates The correct computation of the predicates VerbalisationSufficient and Reformulation(*expr*, *ref*) requires to make use to linguistic or domain ressources that we do not have for our work. Thus, we let aside the computationt of these predicates. Even if we do not address this aspect, we are aware of its importance in the conduct of the task underlying to the interaction. As a matter of fact, the construction of a structured representation of the information need of the user can bring a huge profit for the rest of the interaction, especially for the verbalization/terminology alignment. The interpretation from a natural language formulation to a database query language was subject to an in-depth development in the PhD work of Pradel [Pra13]. This work relies in particular on the concept of *pivot query*, which is an intermediary representation between the natural language and the target query language.

8.1.4 Request for precision

The predicate PrecisionNotAsked is true if the system didn't yet ask to the user to precise his verbalization.

Prédicat 8 – PrecisionNotAsked **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{preciseVerb}, \textbf{Ina})$

The predicate QueryEmpty is true if the current query is empty.

Prédicat 9 – QueryEmpty **:**

$\nexists i, \mathrm{T}_i \ni \mathrm{C}(\_, \text{queryKeyWord}(kw), \textbf{Crt})$

8.1.5 Verbalization precision

The predicate UserWillPreciseVerbalisation is true if the user is commited to precise his verbalization. It checks that the latter commited on the action to precise the verbalization.

Prédicat 10 – UserWillPreciseVerbalisation **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{preciseVerb}, \textbf{Crt})$

8.1.6 Verbalization/terminology alignment

The predicate QueryNeverLaunched is true if no query launche has already been performed.

Prédicat 11 – QueryNeverLaunched **:**

$\nexists i$, $T_i \ni C(\_, \text{currentQueryLaunched}, \textbf{Crt})$

The predicate VerbalisationPrecisionFailed is true if the verbalization precision failed, that is to say that the user declined to precise his verbalization or was commited to do it but did not.

Prédicat 12 – VerbalisationPrecisionFailed **:**

$$\vee \left| \begin{array}{l} T_i \ni C(x, \text{preciseVerb}, \textbf{Fal}) \\ T_i \ni C(x, \text{preciseVerb}, \textbf{Vio}) \end{array} \right.$$

The predicate QueryKeyWord(kw) is true if the keyword kw is a keword of the query q.

Prédicat 13 – QueryKeyWord(kw) **:**

$T_i \ni C(\_, \text{queryKeyWord}(kw), \textbf{Crt})$

The predicate RelevantForAddition(kw) is true if the addition of the keyword kw to the query is relevant to precise the query.

Prédicat 14 – RelevantForAddition(kw) **:**

Makes use of the domain tools to determine the most relevant keyword to add to the query.

8.2 Query launch

The predicate QueryLaunchable is true if the current query respects the minimal criterion to be launched.

Prédicat 15 – QueryLaunchable **:**

Depends on the information retrieval tool used. Typicalli, the current query must not be empty and return at least a result.

The predicate QueryLaunchRelevant is true if the system judges that it is relevant to offer to the user to launch the current query.

Prédicat 16 – QueryLaunchRelevant **:**

Depend on the domain and information retrieval tool used and on how much the system judges the results of this query as responding to the user's information need.

8.3 Results evaluation

8.3.1 Results evaluation by the user

The predicate CurrentQueryLaunched is true if the current query has been launched. It checks if the system is commited on the launch of the current query.

Prédicat 17 – CurrentQueryLaunched **:**

$T_i \ni C(y, \text{currentQueryLaunched}, \textbf{Crt})$

The predicate CurrentQueryResultsSatisfying is true if the results of the query q were judged satisfying by the user. It checks that the commitment by the user of the fact that q is satisfing is active.

Prédicat 18 – CurrentQueryResultsSatisfying **:**

$T_i \ni C(x, \text{currentQueryResultsSatisfying}, \textbf{Crt})$

The predicate ResearchNotOver is true when the user did not commit himself on the fact that his research is over.

Prédicat 19 – ResearchNotOver **:**

$T_i \not\ni C(x, \text{researchOver}, \textbf{Crt})$

The predicate ResearchOverNotAsked is true when the system did not ask to the user if his search is over since the last time the user signaled the fact that the current query's results are satisfying.

Prédicat 20 – ResearchOverNotAsked **:**

The definition of this predicate requires to manipulate dialogical action commitments to know if the system asked the user if his search is over and when. The dialogical actions commitments not being discussed in this document, we will not give a formal definition of this predicate here.

8.3.2 Closing

The predicate ResearchOver is true if the user states that his search is over. It checks that the latter is commited of the proposition of end of search.

Prédicat 21 – ResearchOver **:**

$T_i \ni C(x, \text{researchOver}, \textbf{Crt})$

8.4 Query repair

The predicate RelevantForRemoval(kw) is true if the removal of the keyword kw of the query is relevant.

Prédicat 22 – RelevantForRemoval(kw) **:**

This predicate makes use of information retrieval and domain resources to be

computed.

The predicate RelevantForSubstitution(*kw*, *skw*) is true if the substitution of the keyword *kw* by *skw* is relevant to modify the query.

Prédicat 23 – RelevantForSubstitution(*kw*, *skw*) **:**

This predicate makes use of information retrieval and domain resources to be computed.

References

[Bat79] Marcia J. Bates. Information search tactics. *JASIS*, 30(4):205–214, 1979.

[Bat89] Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5):407–424, 1989.

[Bat90] Marcia J. Bates. Where should the person stop and the information search interface start? *Inf. Process. Manage.*, 26(5):575–591, 1990.

[CDH+10] Nathalie Chaignaud, Valérie Delavigne, Maryvonne Holzem, Jean-Philippe Kotowicz, and Alain Loisel. Étude cognitive des processus de construction d'une requête dans un système de gestion de connaissances médicales. *Revue des Sciences et Technologies de l'Information - Série TSI : Technique et Science Informatiques*, 29:991–1021, 2010. 29 pages.

[CIS] Cismef – catalogue et index des sites médicaux de langue française. `http://www.chu-rouen.fr/cismef/`. Consulté le 13/09/2018.

[DD15] Guillaume Dubuisson Duplessis and Laurence Devillers. Towards the consideration of dialogue activities in engagement measures for human-robot social interaction. In *International Conference on Intelligent Robots and Systems*, Designing & Evaluating Social Robots for Public Settings Workshop, pages 19–24, Hambourg, Germany, September 2015.

[DLD+01] Stefan Jacques Darmoni, Jean-Philippe Leroy, Magali Douyère, Josette Piot, Saida Ouazir, Benoit Lacoste, Christophe Godard, Isabelle Rigolle, Martial Brisou, Stphane Videau, Myriam Quéré, Eric Goupy, Habib Abdulrab, and Benoit Thirion. Doc'CISMeF : un outil de recherche internet orienté vers l'enseignement et la formation à distance en médecine. *Pédagogie Médicale*, 2(3):170–178, aug 2001.

[DPCK17] Guillaume Dubuisson Duplessis, Alexandre Pauchet, Nathalie Chaignaud, and Jean-Philippe Kotowicz. A conventional dialogue model based on dialogue patterns. *International Journal on Artificial Intelligence Tools*, 26(1):1–23, 2017.

[Dub14] Guillaume Dubuisson Duplessis. *Modèle de comportement communicatif conventionnel pour un agent en interaction avec des humains: Approche par jeux de dialogue.* PhD thesis, INSA de Rouen, 2014.

[Fis01] Gerhard Fischer. User modeling in human–computer interaction. *User Modeling and User-Adapted Interaction*, 11(1):65–86, Mar 2001.

[GPB09] Gene Golovchinsky, Jeremy Pickens, and Maribeth Back. A taxonomy of collaboration in online information seeking. *CoRR*, abs/0908.0704, 2009.

[GQPG09] G Golovchinsky, P Qvarfordt, J Pickens, and B Gray. Collaborative information seeking. *Collaborating Finding common ground for multiparty problems San Francisco JosseyBass*, 42(3), 2009. [Original String]:Golovchinsky, G., Qvarfordt, P., & Pickens, J. (2009). Collaborative information seeking. IEEE Computer, 42(3) Gray, B. (1989). Collaborating: Finding common ground for multiparty problems. San Francisco: Jossey-Bass.

[LDC+16] Jean-Baptiste Louvet, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, and Laurent Vercouter. Recherche collaborative de documents : comparaison assistance humaine/automatique. pages 161–166. Journées Francophones d'Ingénierie des Connaissances, 2016.

[LDC+17] Jean-Baptiste Louvet, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Laurent Vercouter, and Jean-Philippe Kotowicz. Modeling a collaborative task with social commitments. volume 112, pages 377–386, Amsterdam, The Netherlands, The Netherlands, September 2017. Elsevier Science Publishers B. V.

[LKC08] Alain Loisel, Jean-Philippe Kotowicz, and Nathalie Chaignaud. An issue-based approach to information search modelling: Analysis of a human dialog corpus. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, Speech and Dialogue*, pages 609–616, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[LLC+00] Staffan Larsson, Peter Ljunglöf, Robin Cooper, Elisabet Engdahl, and Stina Ericsson. Godis: an accommodating dialogue system. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational systems-Volume 3*, pages 7–10. Association for Computational Linguistics, 2000.

[Loi08] Alain Loisel. *Modélisation du dialogue Homme-Machine pour la recherche d'informations: approche questions-réponses.* PhD thesis, INSA de Rouen, 2008.

[Lou19] Jean-Baptiste Louvet. *Collaboration humain-machine à l'aide de motifs dialogiques pour la réalisation d'une tâche complexe. Application à la recherche d'information.* PhD thesis, INSA Rouen Normandie, July 2019.

[MW07] Gary Marchionini and Ryen White. Find what you need, understand what you find. *Int. J. Hum. Comput. Interaction*, 23(3):205–237, 2007.

[Pra13] Camille Pradel. *D'un langage de haut niveau à des requêtes graphes permettant d'interroger le web sémantique*. PhD thesis, 2013. Thèse de doctorat dirigée par Haemmerlé, Ollivier et Hernandez, Nathalie Intelligence artificielle Toulouse 3 2013.

[SE98] Alistair Sutcliffe and Mark Ennis. Towards a cognitive theory of information retrieval. *Interacting with computers*, 10:321–351, 1998.

[Suc07] Lucy A. Suchman. *Human-Machine Reconfigurations: Plans and Situated Actions*. Cambridge University Press, Cambridge, 2 edition, 2007.

[Ter95] Loren G. Terveen. Overview of human-computer collaboration. *Knowledge-Based Systems*, 8(2):67–81, 1995. Human-computer collaboration.

[Wag94] Ellen D. Wagner. In support of a functional definition of interaction. *American Journal of Distance Education*, 8(2):6–29, 1994.