Quality of Information in the context of Ambient Assisted Living

Luís Lima[1], Ricardo Costa[1], Paulo Novais[2], Cesar Analide[2], José Bulas Cruz[3] and José Neves[2]

[1]College of Management and Technology - Polytechnic of Porto,Rua do Curral – Casa do Curral,4610-156 Felgueiras, Portugal

[2]Departamento de Informática/CCTC, Universidade do Minho,Campus de Gualtar,4710-553 Braga, Portugal

[3]University of Trás-os-Montes e Alto Douro ,Vila Real, Portugal

lcl@estgf.ipp.pt; rfc@estgf.ipp.pt; pjon@di.uminho.pt; analide@di.uminho,pt jneves@di.uminho.pt; jcruz@utad.pt

Abstract. With the use of new computational technologies and novel methodologies for problem solving, recurring to the use of Group Decision Support Systems, normally the problem of incomplete information is marginalized as if we were living in an ideal world. Common sense tells us that in the precise time a decision is make it is impossible to know all the information regarding to it, however decisions must be made. What we propose, in the ambit of the VirtualECare project, is a possible solution to decision making, through the use of Group Decision Support Systems, aware of incomplete information but, even so, able to make decisions based in the quality of the information and its source.

Keywords: Incomplete information, knowledge representation, group decision support system, idea generation and argumentation.

1 Introduction

Imperfect information is ubiquitous; we take most of our decisions, if not all, of our day to day life based on incomplete, not precise and even uncertain information. Most information systems just ignore this characteristic of the information about the real world and build upon models were some idealisation expunges the inherent uncertainty [1]. The result is model which never gives correct answers, because of its inability to model exactly what is going on. Instead, one should deal with the uncertainty in the model itself, even at the cost of less simplicity. To implement

useful information systems, namely knowledge based ones, it is necessary to represent and reason with imperfect information.

Examples of such systems are Group Decision Support Systems (GDSS) based on agent perception that we try to associate with the healthcare practice and respective information systems (e-Health systems), in which lack of verification of the quality of information is a key omission [2].

This work focus on a new class of systems from which VirtualECare [3], briefly described below, is an example. It represents a new and cost effective way for health care delivery in the intersection of telemedicine, virtual healthcare teams and electronic medical records. One of the components of VirtualECare is a knowledge-based GDSS. In this paper we define a method to evaluate the quality of knowledge involved in a GDSS, using VirtualECare as example, and present the foundations of a theory that permits to represent and reason with uncertain knowledge.

2 The VirtualECare Project

VirtualECare is an intelligent multi-agent system that will be able to monitor and interact with its customers, being those elderly people and/or their relatives. This system will be connected, not only to healthcare institutions, but also with leisure centers, training facilities, shops and relatives, just to name a few. The VirtualECare Architecture is a distributed one with different components interconnected through a network (e.g. LAN, MAN, WAN), each one with a different role (**Fig. 1**). A top-level description of the roles of the architecture components is presented below:

Supported User – Elderly with special health care needs, which are in constant supervision, thus allowing collecting of vital data information sent to the *CallCareCenter* and forwarded to the *Group Decision Supported System*;

Home – *SupportedUser* natural ambient, under constant supervision, being the collected information sent to the *Group Decision Supported System* through the *CallCareCenter* in case of clinical one, or sent to the *CallServiceCenter* for all the remaining, allowing a better, more comfortable (Intelligent) Ambient;

Group Decision – There can be more than one, being responsible for all the decisions taken in the VirtualECare. Our work is centered in this key component;

Call Service Center – Entity with all the necessary computational and qualified personal resources, capable of receiving, analyze the diverse data, and take the necessary steps according to it;

Call Care Center – Entity with all the necessary computational and qualified personal resources (i.e., healthcare professionals and auxiliary), capable of receiving and analyze the diverse data and take the necessary steps according to it:

Relatives – *SupportedUser* relatives with active supervising, giving complementary information and intervene in some more specific crises (e.g. loneliness).

In order to the *Group Decision Support System* produce correct decisions, a digital profile of the *SupportedUser* is needed allowing a better understand of

his/her special needs. So, we can have different kinds of relevant information, from the patient Electronic Clinic Process to their own personal preferences.
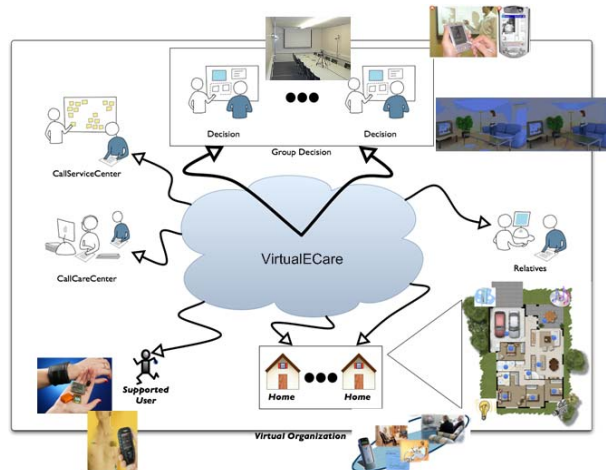

Fig. 1. The System Architecture, VirtualECare.

3 Group Decision Support Systems

In the last years, we have assisted to a growing interest in combining the advances in information society - computing, telecommunications and presentation – in order to create Group Decision Support Systems (GDSS).

Decision Support Systems (DSS) are interactive computer-based systems aimed to help decision makers use communication technologies, information, knowledge and/or models to solve problems. DSS and particularly (GDSS) will benefit from progress in more basic research on behavioral topics in the areas of organizational decision making, behavioral decision theory and organizational behavior [4].

We believe the use of GDSS in the Healthcare sector will allow professionals to achieve better results in the analysis of one's Electronical Clinical Profile (ECP). This achievement is vital, regarding the explosion of knowledge and skills, together with the growing need to use limited resources more efficiently.

3.1 Idea Generation and Argumentation in the Group Decision Module

The *Group Decision* module, as stated above, is a major module of our system. This fact, associated with the importance of decision-making in today business activity

and with the needed celerity in obtaining a decision in the majority of the cases that this key module will be defied to resolve, requires a real effectiveness of the decision making process. Thus, an Idea Generation tool is needed supporting the meetings, either face-to-face, asynchronous or distributed. After establishing individual ideas, the participants "defend" those ideas in order to reach consensus or majority. Each participant will argue for the most interesting alternatives or against the worst alternatives, according to his/her preferences and/or skills [5].

3.2 Meeting phases

In this work we will call *meeting* to all the phases necessary to the completion of a specific task, i.e., a meeting happens as an effect of the interaction between two or more individuals [6]. A meeting can be realized in one of the four scenarios: i) same time / same place; ii) same time / different places; iii) different times / same place; iv) different times / different places. Each one of these scenarios will require from the GDSS a different kind of action.
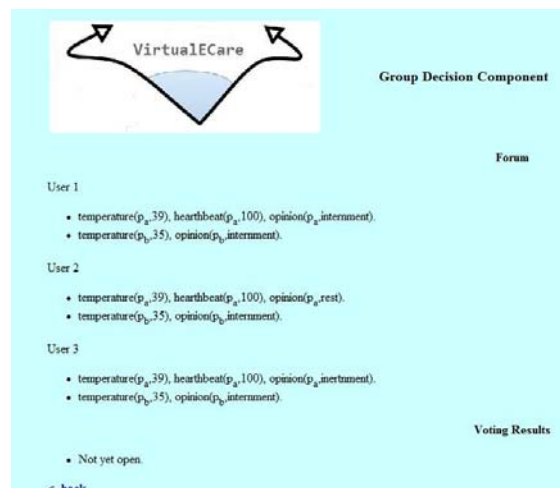

Fig. 2. Forum (Argumentation and Voting)

Besides the group members involved in the collaborative work process, it is very common to see a third element taking part in the course of action: the facilitator. The meeting facilitator is a person welcomed in the group, nonaligned, which arbitrate all the meeting phases [7]. The facilitator prepares the meeting, namely the group formation. The choice of the participants is a critical factor of success. In our model of the VirtualECare system, the GDSS assists the facilitator in this task, by providing quality metrics on the profile of each possible participant. In the In-Meeting phase, the participants will be working in order to accomplish the meeting goals and take de best decisions. In this, the participants use a knowledge database

and exchange information. Again, the system must provide a measure of the quality of this knowledge and information. In the Post-Meeting phase it is important to evaluate the results achieved by the group, as well as how much each group member is acquit with the achieved results (satisfied/unsatisfied).

The VirtualECare GDSS is composed of several modules being the most important for this work the Argumentation and the Voting which are used to carry out the discussion, through systematic registering and voting of the statements supporting each participant position, reaching, in the end, the final decision (**Fig. 2**).

4 Knowledge Representation

A suitable representation of incomplete information is needed, one that supports non-monotonic reasoning. In a classical logical theory, the proof of a question results in a true or false value, or is made in terms of representing something about one could not be conclusive. In opposition, in a logic program, the answer to a question is only of two types: true or false. It is a limitation of the knowledge representation, because no explicit representation of negative information exists. Additionally, the operational semantics applies the Closed-World Assumption (CWA) [8] to all the predicates. Most logic programs represent implicitly negative information, assuming the application of reasoning according to the CWA.

A logic program is a finite set of clauses in the form:

$$A_0 \leftarrow A_1 \wedge ... \wedge A_m \wedge not\ A_{m+1} \wedge ... \wedge not\ A_n \qquad (1)$$

such as $\forall i \in N_0$, A is an atom and the terms A_i and $not\ A_i$ are literals.

Weak negation - operator *not* in conventional LP - is the **negation-by-failure**: **not A** is true if it is not possible to prove **A**, and **not A** is false when is possible to prove **A**. This kind of reasoning would be enough in a CWA system, but is insufficient when there is incomplete information.

Extended Logic Programming (ELP) was first introduced by Gelfond & Lifschitz [9]. ELP introduces another kind of negation: strong negation, represented by the classical negation sign ¬. In most situations, it is useful to represent ¬*A* as a literal, if it is possible to prove ¬*A*. In EPL, the expressions *A* and *not A*, being *A* a literal, are extended literals, while *A* or ¬*A* are simple literals. Intuitively, *not p* is true whenever there is no reason to believe *p*, whereas ¬*p* requires a proof of the negated literal. An extended logic program is a finite collection of rules *r* of the form:

$$q \leftarrow p_1 \wedge ... \wedge p_m \wedge not\ p_{m+1} \wedge ... \wedge not\ p_{m+n} \qquad (2)$$

where q and every p_i are literals, i.e. formulas like a or $\neg a$, being a an atom, for $m, n \in N_0$.

The objective is to represent explicitly negative information, as well as directly describe the CWA for some predicates (*predicate circumscription* [1]). Three types of conclusions for a question are then possible: *true*, *false* or *unknown*, if there is no information to infer one or the other. The representation of null values will be

scoped by the ELP. In this work, we will consider two types of null values: one allowing the representation of unknown values not necessarily from a given set of values, and the second will represent unknown values from a given set of possible values. We will show now some examples using null values to represent unknown information. Consider the extensions of the predicates that represent some properties of the participants, as a measure of their "quality" for the decision process:

```
area_of_expertise: Entities x StrValue
role: Entities x StrValue
credible: Entities x Value
reputed: Entities x Value
```

The first argument denotes the participant and the second represents the value of the property (e.g. `credible(luis, 100)` means that the credibility of the participant `luis` has the value `100`).

```
credible(luis,100)
¬credible(E,V)← not credible(E,V)
```

Program 1. Extension of the predicate that describes the credibility of a participant

In **Program 1**, the symbol ¬ represents the strong negation, denoting what should be interpreted as false, and the term *not* designates negation-by-failure.

Now admit that the credibility of another participant *ricardo* has not, yet, been established. This will be denoted by a null value of the type unknown, and represents the situation in **Program 2**: the participant is credible but it is not possible to be certain (affirmative) about its value. In the second clause the symbol ⊥ represents a null value of an undefined type. It is a representation that assumes any value as a viable solution, but without being given a clue to conclude about which value one is speaking about. It is not possible to compute, from the positive information, the value of the credibility of the participant *ricardo*. The fourth clause (the closure of predicate credibility) discards the possibility of being assumed as false any question on the specific value of credibility for participant *ricardo*.

```
credible(luis,100)
credible(ricardo,⊥)
¬credible(E,V)← not credible(E,V),
               not exception(credible(E,V))
exception(credible(E,V))← credible(E,⊥)
```

Program 2. Credibility about participant *ricardo*, with an unknown value

In (Program 3) the value of the credibility of a participant is foreseen to be 60, with a margin of mistake of 15. It is not possible to be positive, concerning the credibility value. However, it is false that the participant has a credibility value of 80 or 100. This example suggests that the lack of knowledge may only be associated to a enumerated set of possible known values. As a different case, the credibility of the participant *paulo* is unknown, but one knows that it is specifically 30 or 50.

```
credible(luis,100)
credible(ricardo,⊥)
¬credible(E,V)← not credible(E,V), not exception(credible(E,V))
exception(credible(E,V))← credible(E,⊥)
exception(credible(carlos,V))← V ≥ 45 ∧ V ≤ 75
exception(credible(paulo,30))
exception(credible(paulo,50))
```

Program 3. Representation of the credibility of the participants *carlos* and *paulo*

Using ELP, a procedure given in terms of the extension of a predicate called `demo` is presented. This predicate allows one to reason about the body of knowledge presented in a particular domain, set on the formalism previously referred to. Given a question, it returns a solution based on a set of assumptions.

This meta predicate is defined as: *Demo: Question x Answer*, Where Question indicates a theorem to be proved and Answer denotes a truth value (see **Program 4**): true (T), false (F) or unknown (U).

```
demo(Q,T)← Q
demo(Q,F)← ¬Q
demo(Q,U)← not Q ∧ not ¬Q
```

Program 4. Extension of meta-predicate `demo`

5 Quality of Knowledge

In any decision making process, arguably, the decision is made without having all the information. How does a decision maker is confident about the reliability of the information? In group decisions the situation is more complex - each participant must be confident on: The reliability of the computer support system; The other decision makers; The information exchanged between participants. The Group Decision of the VirtualECare system above operates in one such environment. Let i ($i \in 1,…, m$) represent the predicates whose extensions make an extended logic program that models the universe of discourse and j ($j \in 1,…, n$) the attributes or those predicates. Let $x_j \in [min_j, max_j]$ be a value for attribute j. To each predicate is also associated a scoring function $V_{ij}[min_j, max_j] \to 0 … 1$, that gives the score predicate i assigns to a value of attribute j in the range of its acceptable values, i.e., its domain (for simplicity, scores are kept in the interval [0 … 1]), here given in the form: *all(attribute_exception_list, sub_expression, invariants)*

This denotes that *sub_expression* should hold for each combination of the exceptions of the extensions of the predicates that represent the attributes in the *attribute_exception_list* and the *invariants*. The invariants are integrity constraints in the form:

$$\leftarrow p_1 \wedge ... \wedge p_m \wedge not\ p_{m+1} \wedge ... \wedge not\ p_{m+n} \qquad (3)$$

where all p_i are literals, i.e. formulas like a or $\neg a$, being a an atom, for $m, n \in N_0$.

This is further translated by introducing three new predicates. The first predicate creates a list of all possible exception combinations (pairs, triples, ..., n-tuples) as a list of sets determined by the domain size (and the invariants). The second predicate recurses through this list and makes a call to the third predicate for each exception combination. The third predicate denotes sub_expression, giving for each predicate, as a result, the respective score function. The Quality of Knowledge (QK) with respect to a generic predicate P is therefore given by $QK_P = 1/Card$, where $Card$ denotes the cardinality of the exception set for P, if the exception set is not disjoint. If the exception set is disjoint, the quality of information is given by:

$$QK_P = \frac{1}{C_1^{Card} + \cdots + C_{Card}^{Card}} \qquad (4)$$

where C_{Card}^{Card} is a card-combination subset, with $Card$ elements.

The next element of the model is the relative importance that a predicate assigns to each of its attributes under observation: w_{ij} stands for the relevance of attribute j for predicate i (it is also assumed that the weights of all predicates are normalized:

$$\forall i \sum_{j=1}^{n} w_{ij} = 1 \qquad (5)$$

It is now possible to define a predicate's scoring function, i.e., for a value $x = (x_1, ..., n)$ in the multi dimensional space defined by the attributes domains, which is given in the form:

$$V_i(x) = \sum_{j=1}^{n} w_{ij} * V_{ij}(x_j) \qquad (6)$$

It is now possible to measure the QK by posting the $V_i(x)$ values into a multi-dimensional space and projecting it onto a two dimensional one. Using this procedure, it is defined a circle, as the one given in **Fig. 3**.
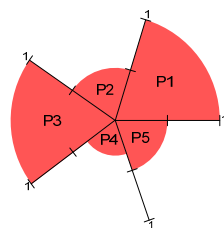

Fig. 3. A measure of the quality of knowledge for a logic program or theory P

The dashed n-slices of the circle (here built on the extensions of five predicates, named as $p_1 ... p_5$) denote de QK associated with each of the predicate extensions. Let's return to our case above and evaluate the global credibility of the system (**Program 5**). As an example we represent the QK for participant *luis*, depicted in

Fig. 4. In order to find the relationships among the extensions of these predicates, we evaluate the relevance of the QK, given in the form $V_{credible}(luis) = 1$; $V_{reputed}(luis) = 0.785$; $V_{role}(luis) = 0$. It is now possible to measure the QK associated to a logic program referred to above: the shaded n-slices (here n = 3) of the circle denote the QK for predicates *credible*, *reputed* and *role*.

```
¬credible(E,V)← not credible(E,V), not exception(credible(E,V))
exception(credible(E,V))← credible(E,⊥)
credible(luis,100)
credible(ricardo,⊥)
exception(credible(carlos,V))← V ≥ 45 ∧ V ≤ 75
exception(credible(paulo,30))
exception(credible(paulo,50))
role(luis,⊥)
role(ricardo,doctor)
exception(role(carlos,doctor))
exception(reputed(luis,80))
exception(reputed(luis,50))
exception(reputed(ricardo,40))
exception(reputed(ricardo,60))
reputed(carlos,100)
```

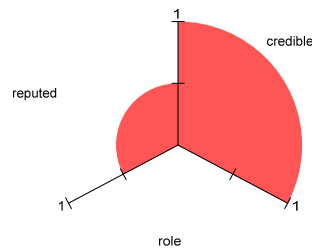**Program 5**. Example of universe of discourse

Fig. 4. A measure of quality of knowledge about participant luis

However, in order to accomplish the main goal of this work, we need to further extend the purpose of **Fig. 4**, i.e., we may define a new predicate, *reliance*; whose extension may be given in the form of the example below:

```
¬reliance(X,Y)← not reliance(X,Y), not exception(reliance(X,Y))
reliance(luis,((credible,1),(reputed,0.785)(role,0)))
reliance(ricardo,((credible,0),(reputed,0.785),(role,1)))
```

Program 6. Measuring the global quality

Besides being able to evaluate the quality of individual actors and individual pieces of information that flows in a group decision system, we aim to have an overall mechanism that allows one to measure the global quality of the system itself.

The same mechanism used to evaluate individual parts of the system is consistently used to evaluate all the system, through an extension process.

6 Conclusion

Our agenda is to apply the above Knowledge Representation with the respective Quality of its Information to the VirtualEcare GDSS module. Thus, the suggestions/decisions presented by this module will consider the existence of incomplete information, and, even so, will present a possible way to try and, if possible, resolve the actual problem. Incomplete information may arise from several sources (e.g. unreachable sensors, incomplete Patient Electronic Clinical Profile) but what is important is to be able to measure the quality of the information we have access to and the quality of the ideas presented by the participants, based in factors like reputation, credibility, namely, in the discussion. However, we are certain, that some vital information, if incomplete, may even so, compromise any suggestion / decision but, in the majority of situations, we believe this will not be the case.

References

1. Parsons, S., *Current approaches to handling imperfect information in data and knowledge bases.* IEEE Trans. on Knowledge and Data Eng., 1996. 8(3).
2. Cruz-Correia, R.J., et al., *Reviewing the integration of patient data: how systems are evolving in practice to meet patient needs.* BMC Medical Informatics and Decision Making, 2007. **7**(14).
3. Costa, R., et al., eds. *Intelligent Mixed Reality for the Creation of Ambient Assisted Living*. Progress in Artificial Intelligence, ed. J. Neves, M. Santos, and J. Machado. Vol. 4874. 2007, Lecture Notes in Artificial Intelligence - Spinger.
4. Conklin, J., *Dialogue Mapping: Building Shared Understanding of Wicked Problems*. 2006: Wiley.
5. Brito, L., P. Novais, and J. Neves, *The logic behind negotiation: from pre-argument reasoning to argument-based negotiaion*, in *Intelligent Agent Software Engineering*, V. Plekhanova, Editor. 2003, Idea Group Piblishing. p. 137-159.
6. Bostrom, R., et al., *Group facilitation and group support systems*, in *Group Support Systems: New Perspectives, Jessup and Valachic, Eds.* 2003, Macmillan.
7. Schwarz, R.M., *The Skilled Facilitator: Practical Wisdom for Developing Effective Groups*. 1994: Jossey Bass.
8. Hustadt, U. *Do we need the closed-world assumption in knowledge representation?* in *KI'94 Workshop*. 1994. Saarbrüken, Germany: Baader, Buchheit, Nutt (eds.).
9. Gelfond, M. and V. Lifschitz. *Logic Programs with Classical Negation*. in *Proceedings of the International Conference on Logic Programming*. 1990.