Monitoring Gestational Diabetes Mellitus with Cognitive Agents and Agent Environments

Stefano Bromuri*, Michael Ignaz Schumacher *, Kostas Stathis†, Juan Ruiz‡

*Department of Business Information Systems
University of Applied Sciences Western Switerland Switzerland, 3 Technopole, 3960, Sierre,
Email: {stefano.bromuri,michael.schumacher}@hevs.ch

†Department of Computer Science,
Royal Holloway, University of London, Egham Hill, EGHAM, TW20 0EX
Email: kostas@cs.rhul.ac.uk

‡Department of Endocrinology
Lausanne University Hospital, 1011 Lausanne
Email: juan.ruiz@chuv.ch

***Abstract*—This paper presents an agent-based pervasive healthcare system (PHS) to support pregnant women with Gestational Diabetes Mellitus (GDM). Such an infrastructure uses a mobile application to monitor patients affected by GDM, whose parameters are then sent to and analysed by cognitive agents. We utilise a symbolic reasoning approach to formalise the events happening in the system, the entities participating in the interaction and the agent cognitive model for continuous monitoring of GDM. Such a cognitive model is based on deductive treatment adjustment rules to provide doctors with indications about the patient's treatment and on abductive rules to provide a diagnosis of the illness' current state. We evaluate our system by means of medical data to assess the precision of our infrastructure.**

***Keywords*-Agent Environments; Pervasive Health; Abduction Logic, Multi-agent Systems.**

I. Introduction

Gestational diabetes mellitus (GDM) [1] is a type of diabetes which temporarily affects 4% of otherwise healthy pregnant women, and typically improves or disappears after delivery. The factors associated with GDM are age, pregnancy weight, family history of diabetes, and ethnicity. GDM can increase the risk of health problems developing in an unborn baby, so it is important that the glucose levels in the pregnant woman blood are under control. If untreated, GDM can cause the baby to have macrosomia (excessive weight at birth) [2] or the woman to develop a hypertensive state called preeclampsia, which could lead to eclampsia, a serious condition that can lead to epileptic seizures and coma. One interesting fact about GDM is that patients are motivated to have a pregnancy that is as smooth as possible and they rarely lie about their health in the current practice, that is based on recording glucose levels and blood pressure on a notebook that is checked weekly by the doctors. There have been attempts to define a statistical model to deal with preeclampsia, such as the fullPiers model presented in [3], but between checks the woman may still develop poor glycemic control, leading to the aforementioned adverse effects. In this paper we focus on improving GDM care, by utilising a pervasive healthcare system (PHS) [4]. The primary goal of a PHS is to break the boundaries of hospital care, allowing patients to be monitored while living their day-to-day life and to keep in touch with healthcare professionals. Thus, we document the collaboration between computer scientists and doctors to give to GDM a formal representation, extrapolated from medical guidelines [5], in order to define a PHS for continuous monitoring, treatment adjustment and diagnosis of GDM, that can provide useful feedback to doctors and save their time when dealing with the physiological values of the patient, as usually multiple patients are assigned to one doctor.

We embedded such a GDM model in intelligent agents providing decision support to GDM caregivers. Agents are the most appropriate modelling abstraction for our PHS as they are understood to be autonomous software entities, that pursue a set of goals [6] in an intelligent way, by applying AI reasoning techniques such as abduction and deduction, and act proactively, without necessarily receiving a stimulus from a user. This set of properties can benefit the current definition of PHS, by having monitoring tools that are capable of complex and proactive reasoning on the current patient's physiological parameters.

Moreover, the agent environment concept [7], is becoming increasingly more important to simplify the definition and deployment of multi-agent applications, by mediating the interaction between the agents and resources deployed in the system, by hiding to the agents the complexity of dealing with the state of resources external to the agent, and by providing standard interfaces and standard descriptions to resources so that the agents can utilise them for their own goals.

The contribution of this paper is two-fold: from the medical perspective, we provide a formal model of GDM based on symbolic reasoning, which, to the best of our knowledge, is a new approach to GDM care; from the technological perspective, we present an heterogeneous and multi-disciplinary system that makes use of symbolic reasoning, agents and agent environments to define a PHS for decision support in GDM care. In particular, the main contribution of the paper consist in the definition of the abductive/deductive cognitive model of the agents to deal with GDM. In [8] we presented an earlier version of the PHS presented in this paper, where the agent environment was mapped to a set of locations of a real city. On one hand, with respect to [8] we extended our PHS in the following ways: we introduced a more complete interface for the mobile platform, so that the patients can visualise and introduce data about their blood glucose, their blood pressure, their weight and the current symptoms experienced; we extended the agent's cognitive model with a realistic deductive/abductive model for GDM; finally we provided a first laboratory evaluation of the agent's cognitive model, with realistic physiological parameters provided by GDM caregivers. On the other hand, this paper does not yet describe the full implementation of our system with the monitoring Web interface for the doctors, which will be subject of future publications.

The remainder of this paper is structured as follows: Section II discusses about the background technologies and concepts utilised in this paper; Section III discusses the components of our implemented Pervasive Healthcare System; Section IV discusses a laboratory evaluation of our system; finally, Section V concludes this paper and summarises our plans.

II. Background on the GOLEM Agent Platform

Our Pervasive Healthcare System (PHS) is based on the GOLEM agent platform [9]. GOLEM implements the agent environment concept, and it is built around four main entities: agents, avatars, objects and containers. In GOLEM agents are active cognitive entities with a body to perceive and produce events and a declarative mind capable of symbolic reasoning and decision making. Avatars are agents where the decision making is done by a human user via an interface.

Objects are reactive entities representing services for the agents. Containers represent a portion of the distributed agent environment, which is organised as a hierarchical structure in terms of super, sub and neighbour containers. GOLEM entities and events are specified in terms of C-logic

Predicate	Meaning
happens(E,T)	event E occurs at time T.
holds_at(Id,Cl,At,Val,T)	object Id of class Cl has an attribute At of value Val at time T.
instance_of(Id,Cl,T)	object Id is of class Cls at time T.
initiates(E,Id,Cl,At,Val)	event E initiates a value Val of an attribute At, in an object Id of class Cl.
terminates(E,Id,Cl,At,Val)	event E terminates a value Val of an attribute At, in an object Id of class Cl.
subscribes(E,S,T)	a sensor S subscribes to an event E at time T.
detectable(E,S,T)	the detectable/3 Given an environment state this predicate describes if an event E is detectable by a sensor S subscribed to the class of the event E at time T.
notify(E, S, T)	an event E is notified to a sensor S at time T.
possible(E,T)	the possible/2 defines if an event E can happen at a certain time T.
perceive(E,S,T)	a perception event E is notified to a sensor S at time T when the agent calls this predicate.
locally_at(C,P,P*,Id, Cl, At, Val, T)	Given a starting container C, this predicate allows querying the attribute of an object in C and in its sub containers with the same effects of an holds_at/5, where P* describes the current path and P the final path where the value of the object attribute holds.
neighbouring_at(C,P,P*, Max, Id, Cl, At, Val, T)	neighbouring_at/9 extends locally_at/8 by querying neighbour containers and their sub-containers, where Max describes the maximum length of the path.
regionally_at(C,P,P*,Max, Id, Cl,At,Val,T)	regionally_at/9 extends neighbouring_at/9, by querying super containers, their neighbours and their sub containers.

Table I
AEC Predicates

structures [10], a formalism that allows the description of complex objects. C-logic is a compact specification language that is particularly convenient to describe complex entities and that has a direct translation to the Ambient Event Calculus (AEC) [9], an Event Calculus [11] dialect, that handles the evolution of objects, agents and containers by means of production of events. In what follows, we will produce C-logic descriptions of the entities used in our PHS, to exemplify how GOLEM handles them. For example the following C-logic structure:

```
container:c1[address ⇒ ”container://one@134.219.7.1:13000”,
    location⇒coordinates:co1[longitude⇒0,latitude⇒0],
    entities ⇒ {caretaker:ag1,avatar:av1, caretaker:ag2, avatar:av2}]
```

describes a PHS' container that includes an address attribute whose value is container://one@134.219.7.1:13000, the location of the container pointing another object co1 with the longitude and latitude coordinates of the container, and the entities contained in the container is a set (multiple values attribute) of two caretaker agents ag1 and ag2 and two avatars av1 and av2. Entities within a container can have their own description, as shown for the caretaker agent ag1, below:

```
caretaker:a1[monitors ⇒ avatar:av1, sensors⇒{sight:s1,hearing:s2},
    effectors⇒{speak:ef1,arm:ef2},location ⇒ container:c1, activity ⇒ idle]
```

GOLEM containers behaviour is defined in terms of a declarative Prolog-like theory on top of the AEC formalism. In Table I we report the definition of the main predicates of the AEC and then we show how C-logic descriptions can be handled with the AEC, whose implementation details are discussed in [9]. The C-logic description reported above has a direct translation to a set of elementary Prolog-like assertions, bound to Java objects, that can be queried using the AEC, as also reported in [9].

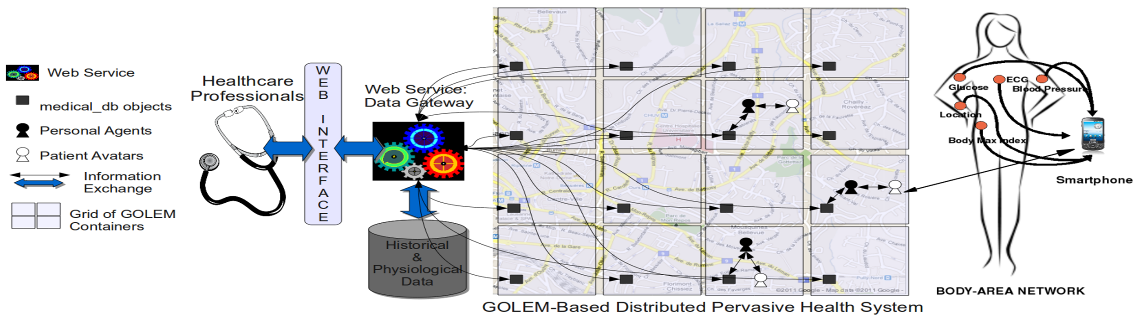

Figure 1. The Pervasive Health System Logic Architecture. A grid of GOLEM containers is deployed and mapped on a real environment representing a city. The disposition in a grid-like structure has been chosen to resemble the topology of a cellular network for mobile phones, as we imagine that a system like the one described here could easily run on such an infrastructure.

III. Pervasive Health Agent Environment Model

Fig. 1 presents the logic architecture of our GOLEM-based PHS. So far, we implemented the agent environment, the cognitive agents, the objects and the patient's mobile interface, while the Web interface and database on the hospital side are in development. Such a Web interface will allow doctors to insert historical patient's data that will be used in the agents' reasoning process. A personal caretaker agent is associated to every avatar connected to the grid using the mobile interface. Whenever a new set of physiological data is produced by the avatar, the caretaker agent records them in a database containing the patients' physiological data. Finally caretakers and patients are allowed to access their data by means of a Web interface. The reason to keep the computation within the infrastructure and not in the mobile interface is that the data related to the patient's history may vary in time, requiring the mobile interface to download the information multiple times. It is not optimal to have patient's data moving outside the hospital due to privacy and legal reasons. At the same time our approach only requires the patient to connect to the infrastructure when there is a change in the patient's physiological parameters, keeping the power consumption minimal.

A. The Mobile Interface

As shown in Fig. 1 every patient in our PHS is equipped with a mobile interface through which they can introduce their physiological parameters. The interface allows introducing blood pressure, glucose, weight and symptoms experienced by the patient. The interface works as a client towards the patient avatar in the agent environment. Thanks to the fact that the users are embodied in the agent environment as avatars, they can produce events in the agent environment as the following one:

```
pressure_reading:e1[avatar ⇒ avid1,caretaker_agent ⇒ ag1,
            systolic_pressure ⇒ 120, diastolic_pressure ⇒ 80].
glucose:e2[ avatar ⇒ avid1, caretaker_agent ⇒ ag1, glucose_level ⇒ 7].
```

Event e1 specified above is a pressure_reading event, produced by the avatar avid1 for the caretaker agent ag1, and it contains a systolic pressure value of 120 and a diastolic pressure value of 80. Similarly, event e2 is produced by the avatar to publish the blood glucose levels in the container for the personal caretaker agent.

B. The Distributed Agent Environment

The events produced by the avatars and directed to the caretaker agents are handled using a publisher/subscriber pattern [12]. Whenever an agent is deployed in the agent environment, it subscribes its sensors to perceive the events of the assigned avatar. For example, the following C-logic term:

```
listener:s1[agent ⇒ ag1, senses ⇒ event:A[actor ⇒ avatar:av1],status ⇒ open]
```

specifies that the sensor s1, owned by agent ag1 subscribes to the events produced by avatar av1, that will be notified to the agent by the environment, using the subscribe/3 predicate previously presented in Section II.

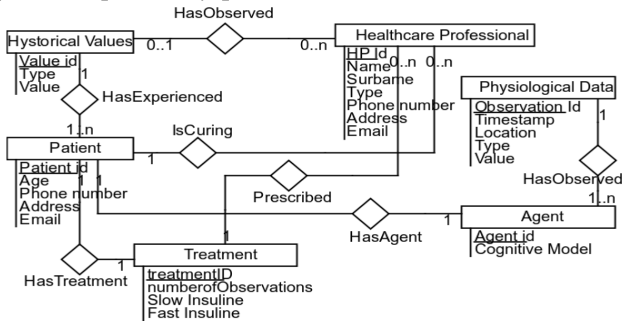

Figure 2. PHS Data Model.

In particular, for events notification, we define the following detectable/3 predicate:

```
detectable(event:E[actor⇒ avatar:AvID], Sensor, T)←
    holds_at(Sensor, owner, Agent, T), holds_at(AvID, caretaker, Agent, T).
```

this states that an event produced by a certain avatar in the agent environment is detectable by a sensor subscribing to that class of events, only if the sensor owner is the caretaker of the avatar that produced the event. The agents of our GOLEM-based PHS, must be able to obtain historical information about the patients, that are stored within the hospital, or caretaker, in charge of the patient. To achieve this, every container of the distributed agent environment is equipped with an object that connects to the database of the hospital, represented in C-logic as follows:

```
medical_db:db1[
    triggers ⇒ { store_event_trigger:tr1, historical_data_query_trigger:tr2 }
    emitter ⇒ { physilogical_data_emitter:em1, historical_data_emitter:em2 }]
```

The description above specifies a GOLEM object db1 of type medical_db, capable of storing information about physiological parameters through object trigger tr1, subscribing to store_event events, and capable of receiving queries about patient's historical parameters, through the object trigger tr2, that subscribes to historical_data_query events. Such medical_db objects contain a link to a Web service external to the agent environment, that allows storing and retrieving patient data. The data model of our system is presented in Fig. 2. Part of the data in the database of our PHS is added off-line by healthcare professionals (historical data, patient data and healthcare professional data), while, as shown in Fig. 1 the physiological data of the patient will be pushed in the database by the personal caretaker agents through GOLEM objects of type medical_db. Such objects are connected to a Web service that in turns is connected to the PHS to publish and retrieve data.

C. Cognitive Personal Caretaker Agents For GDM

The GOLEM caretaker agent mind is based on the Event Calculus like the agent environment, and its architecture is shown in Fig. 3.

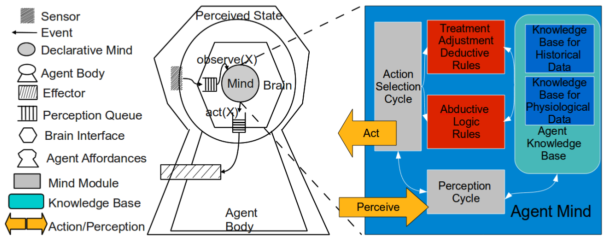

Figure 3. Caretaker Agent Mind Architecture. GOLEM agents are composed by an agent mind attached to an agent body that is equipped with a set of sensors and effectors to perceive and act in the agent environment.

The agent mind is composed by two control threads, one to perceive the events received by the agent body in the agent environment (called perception_cycle/1) and one to produce actions in the agent environment (called action_cycle/1). In the PHS based agent environment agents provide suggestions to professional caregivers, without any planning involved, meaning that the agents do not take any direct decision on how to proceed about a patient.

```
action_cycle(T)← choose(Act, T), execute(Act, T),
                 revise(Act,T), now(Tn), action_cycle(Tn).
perception_cycle(T)←see(Perc,T),revise(Perc,T),now(Tn),perception_cycle(Tn).
```

The two cycles described above, utilise then the Treatment Adjustment Deductive Rules and the Abductive Logic Rules to produce indicators for the medical doctors in charge of the patients. The choose/2 predicate is specified below:

```
choose(Action, T)← instance_of(PID, patient, T),
    findall(S, holds_at(PID,symptom,S,T), Symptoms),
    findall(A, select(Symptoms,A,T), Acts), higher_priority(Acts, Action, T).
higher_priority(ActList, Act, T)← member(Act, ActList), priority(Act, P, T),
    not (member(ActX, ActList), not ActX = Act, priority(ActX, PX,T), PX > P).
```

Such a predicate finds all the symptoms that the patient is experiencing and selects, amongst a set of actions, the best one to perform according to its priority.

Treatment Adjustment Deductive Rules

When a patient is screened with GDM, she discusses with a dietitian to adjust her diet in order to avoid the consumption of insulin. The problem is that patients develop a stronger insulin resistance with the progression of the pregnancy, consequently the introduction of slow and fast insulin may still become necessary. At the beginning of the treatment it is usual to perform four glucose checks per day: one before breakfast (at fasting) and one after two hours of having breakfast, one post-prandial (after eating) at lunch and one post-prandial at dinner. If poor glycemic control occurs, then two further pre-prandial (before eating) observations are introduced (one before lunch and one before dinner). Given these assumptions we derived the following set of rules to adjust the treatment. In the following rules the glycemia values are assumed to be in mmol/l (millimoles per liter), slow insulin is an insulin in tablets that has a long lasting effect during the day and fast insulin is an insulin that acts almost immediately. To express our rules, we used the AEC initiates/4 and terminates/4 predicates to deal with the evolution of the state of the patient in time.

```
R1) terminates(glucose_observation:Ev, ID, observations, _).
R2) initiates(glucose_observation:Ev,ID,observations,[Ev|Tail])←
      patient_of(Ev,ID), time(Ev,T), holds_at(ID,observations, Tail, T).
R3) initiates(glucose_observation:Ev,ID,hypoglycemia_risk, true)←
      patient_of(Ev, ID), time(Ev,T), holds_at(ID,observations, List, T),
      size(List, 20),not (member(M, List), glycemia(M,Gl), Gl >4).
R4) initiates(glucose_observation:Ev, ID, hypoglycemia_risk, true)←
      patient_of(Ev, ID),glycemia(Ev, Gl),time(Ev,T),
      holds_at(ID,observations, [Head|Tail], T), time(Head,T*),
      T - T* < 1, glycemia(Head, Gl2), Gl < 3, Gl2 < 3.
R5) initiates(glucose_observation:Ev,ID,postprandial_observation,[Ev|Tail])←
      is_postprandial(Ev),time(Ev,T),patient_of(Ev, ID),
      holds_at(ID,postprandial_observation, Tail, T).
R6) initiates(glucose_observation:Ev,ID,introduce_preprandial,true)←
      is_postprandial(Ev),time(Ev,T),patient_of(Ev,ID),
      holds_at(ID,postprandial_observation, List, T),
      sublist(Sub,List), size(Sub,6),foreach(Obs,Sub,(glycemia(Obs,G),G>8)).
select(_,A,T)←instance_of(P, patient,T),holds_at(P,introduce_slow_insuline,true,T),
  myID(ID),A=store_event:a1[actor⇒ID,recommendation:p1[
                              timestamp ⇒ T, value ⇒ introduce_slow_insulin]].
```

Rules R1)-R4) handle the case when the glucose readings of the patient indicate the possibility of hypoglycaemia. R1) terminates the old observation list value from the moment the event has happened, while R2) creates assigns to the observations list a new observation. R3) and R) deal with the case when the patient could have an hypoglycemic attack: R3) considers the last 20 observations and if they are all below 4 mmol/l, the patient is considered to be at a risk of hypoglycemia; R4) considers the last two observations, if the patient has two consecutive observations below 3 mmol/l, the patient is also considered at risk of hypoglycaemia. R5) and R6) specify how the additional pre-prandial observations are introduced: if there are a total of 6 values that are above 8 mmol/l in the post-prandial observations of the last five days, then two additional observations are introduced. Further rules are defined for the introduction of slow and fast insulin, but we leave these as they are very similar to the ones already illustrated. The flags that are initiated to be true by the rules specified above, are constantly checked by the agent action cycle, through the choose/2 predicate. If one of the flags is triggered then the agents produced an action to store the change of state of the patient and provide a recommendation to the doctors in charge of the patient. For example the select/3 rule above specifies a treatment adjustment recommendation about slow insulin.

Gestational Diabetes Abductive Model

Abductive logic programming (ALP) [13] is a knowledge-representation framework that can be used to solve problems declaratively based on the idea that a set of seemingly unrelated observed facts, are connected according to well known laws, offering an explanation of what might be true. Given a background theory T, and an observation G, the task of ALP is to compute a set of ground atoms Δ called explanation, and a ground substitution θ such that $\Delta \cup T \models G\theta$, where the set of atoms in Δ can belong to a set of predicates A, called abducibles that are predicates for which there is not complete information. Using ALP we formalised the medical knowledge associated to conditions related to GDM to a set of rules that given the symptoms provide a possible explanation of the current condition of the patient:

$DomainKnowledge$:
$previous\_macrosomia$ ← macrosomia.
$week\_of\_gestation(W)$← W >= 34, macrosomia; W >= 20, preeclampsia.
$glucose\_observation$(A = [Gl_1,Gl_2, . . . Gl_{20}]) ←
    $\exists$ Gl_k,Gl_j $\in$ A, j $\neq$ i, Gl_k >5, Gl_j >5, macrosomia.
$previous\_preeclampsia$; $blurred\_vision$; $severe\_headache$;
    $lower\_leg\_oedema$← preeclampsia; severe_preeclampsia.
$kidney\_disease$; $oliguria$; $protenuria$; $nausea$; $epigastric\_pain$ ←
    preeclampsia; severe_preeclampsia.
$blood\_pressure(Systolic, Diastolic)$←
    preeclampsia,protenuria, Systolic >= 140, Diastolic >= 90;
    not protenuria,severe_hypertension,Systolic>=160,Diastolic>=110;
    Systolic>=140,Diastolic>=90,protenuria,week_of_gestation(W),W < 35,
    (severe_headache;blurred_vision;nausea;vomiting),severe_preeclampsia.
IC :
    ← preeclampsia, (not protenuria;week_of_gestation(W), W < 20).
    ← protenuria, severe_hypertension.
    ← preeclampsia, blood_pressure(S,D),S < 140, D < 90.

select(Symptoms, A, T)←
  demo(Symptoms, Explanation),instance_of(ID,patient,T),myID(AID),
  member([preeclampsia], Explanation),full_piers(ID,Probability,T),
  A= store_event:act2[actor⇒ AID,physiological_data:ph2[type⇒observation,
            timestamp ⇒ T, value ⇒ Probability:preeclampsia]].

full_piers(ID,Probability,T)←holds_at(ID,platelet,PLA,T),holds_at(ID,spo2,SPO2,T),
  holds_at(ID,creatinine, CRE, T),holds_at(ID,aspartate, AST,T),
  holds_at(ID,gestational_age, Age,T),holds_at(ID,chestpain, Pain,T),
  Coeff = 2.68 + (-5.41*10^{-2}*Age) + 1.23*Pain + (-2.71*10^{-2}*CRE) +
  (2.07*10^{-1}*Plat) + (4*10^{-5}*$Plat^2$) + (1.01*10^{-2}*AST) +
  (-3.05*10^{-6}*AST^2) + (2.50*10^{-4}*CRE*PLA)+ (-6.99*10^{-5}*PLA*AST) +
  (-2.56*10^{-3}*PLA*SPO2), Probability is exp(Coeff)/1+exp(Coeff).

The rules above use a Prolog-like syntax where the head of the rules in the domain knowledge represents the observations on the current patient's state, while the body represents the abducible predicates that are part of the explanation associated to the observations or constraints on the observation itself. We considered the patient physiological signals (blood pressure, glucose levels, body mass index), her symptoms, her current and previous history (presence of protenuria, previous preeclampsia, previous macrosomia) and her current week of gestation. In our model we consider the existing guidelines defining preeclampsia [3] when systolic blood pressure is 140 and diastolic blood pressure is 90. The integrity constraints specify explanations that are not possible. For example, in the above definition it is impossible to give as an explanation preeclampsia if there is not protenuria as an observation. Then, we define select/3 rules to choose the best action to take given a diagnosis produced by a demo/2 predicate, that queries the abductive module. The implementation of demo/2 predicate is based on an adbuctive reasoner adapted from [14], which takes the symptoms, corresponding to the observations set G in ALP, to find an explanation, that corresponds to Δ in ALP. The knowledge base T of ALP is implicitly represented by the domain knowledge and the integrity constraints queried by the demo/2 predicate. As shown above we can define a select/3 rule for the case when preeclampsia is diagnosed with very high blood pressure. Such a rule specifies that, given a diagnosis of preeclampsia, the action performed by the agent is to store a physiological observation in the PHS database, holding the probability, according to the fullPiers model [3], that the woman has preeclampsia, so that the caregivers in charge of the patient can take an informed decision. The fullPiers model, whose equation is reported above in the full_piers/3 predicate, utilises a set of physiological parameters (collected offline by caregivers and available to the agents through the medical_db objects) to calculate the probability of adverse outcome of the pregnancy in the case that the woman presents preeclampsia symptoms. For macrosomia on the other hand, no model like the fullPiers has been developed yet. As a consequence, our system only provides an indication that macrosomia may be occurring, that alone is not conclusive to identify the condition, but that allows the caregivers to perform further checks.

IV. Laboratory Evaluation

We evaluated our system with 10 laboratory test scenarios taking 10 distinct patients' physiological data sets: 1) the physiological values are in range; 2) the patient experiences repetitive hypoglycemia; 3) the patient experiences hyperglycemia in the morning every two days in one week; 4) the patient takes too much weight with poor glycemic control; 5) the patient experiences hypoglicemia during the day and hyperglycemia at fasting; 6) the patient experiences blood pressure outside the threshold values for preeclampsia, and her blood test exams also are out of the range, triggering a preeclampsia alert with related probability of adverse outcome above 75%; 7) the patient experiences oedema, epigastric pain, and had previous preeclampsia, but the blood tests values are within range, triggering a preeclampsia alert with related probability of adverse outcome around 60%; 8) the patient has a set of post-prandial observations in the last week that are above 8mmol/l; 9) the patient experiences two consecutive values of glycemia below the 3 mmol/l; 10) the patient had policistic ovaric syndrome and poor glicemic control at the 30th week of pregnancy. The preeclampsia predictions (scenarios 6-7) respected the scores obtained applying the fullPiers model [3]. The problem at this stage is that the probability threshold to perform a decision about whether to hospitalise or not a patient varies with the population of the patients and with the place where the study is conducted as reported in [3]. The definition of such a threshold requires a study with a large amount of patients and such a study has not been performed in Switzerland at the current time. For this reason, we assumed that the decision on whether or not to hospitalise the patient is responsibility of the caregivers of the patient, using the probability calculated by the agents as a further information on which to take a decision. The alerts of hypoglicemia and treatment adjustment (scenarios 2-3-5-8-9), respond and identify correctly the situation, triggering the right alerts. Finally, the agents identified macrosomia correctly (scenarios 4-10), producing an alert for the doctors. One limitation of the system is that it is based on guidelines used by doctors, but the effectiveness of the deductive cognitive model needs to be tested with real patients as these may experience borderline values (i.e. rather than having above 8 mmol/l, the patient experiences 7.9 mmol/l). Since the guidelines are very generic, these situations are handled case by case by the caregivers, and at the current stage there is the necessity to refine the current deductive rules to a more realistic set of values.

V. Conclusion and Future Works

In this paper we presented a pervasive healthcare infrastructure based on abductive and deductive reasoning agents to monitor patients affected by GDM in their day-to-day activities. The prototype is defined in terms of a mobile application that allows the patients to insert their physiological values. Such values are then sent for treatment adjustment evaluation and for abductive diagnosis to the personal agents of the patients. Such personal agents reside in an agent environment that deals with the notification of the events to the agents and with the storage of the information produced by the mobile platform and by the agents recommendations for the doctors. To define the rules for the cognitive model of the patients, we relied on existing medical knowledge about gestational diabetes mellitus. We tested our infrastructure with models of patients proposed by medical doctors of the Lausanne University Hospital and we assessed that the system is capable to recognise these models and propose the correct indications for treatment adjustment and diagnosis purposes. Future work implies testing our system with real patients within a prolonged period of time, to assess the effectiveness of the platform in real settings. A future direction in this sense is to conduct a trial like the one proposed in [3] and define, by using our PHS, an equation for macrosomia to predict its occurrence probability. Finally, security concerns are also matter of future publications.

Acknowledgement

This work has been partially funded by the Hasler Stiftung and by the Nano-Tera consortium.

References

[1] D. C. Serlin and R. W. Lash, “Diagnosis and management of gestational diabetes mellitus,” *Am Fam Physician*, vol. 80, pp. 57–62, Jul 2009.

[2] J. M. Warren, “Pregnancy outcomes in women with gestational diabetes compared with the general obstetric population,” *Obstet Gynecol*, vol. 91, pp. 638–639, Apr 1998.

[3] P. von Dadelszen *et al.*, “Prediction of adverse maternal outcomes in pre-eclampsia: development and validation of the fullPIERS model,” *Lancet*, vol. 377, pp. 219–227, Jan 2011.

[4] U. Varshney, *Pervasive Healthcare Computing: EMR/EHR, Wireless and Health Monitoring*. Springer Publishing Company, Incorporated, 2009.

[5] N. Vogel, B. Burnand, Y. Vial, J. Ruiz, F. Paccaud, and P. Hohlfeld, “Screening for gestational diabetes: variation in guidelines,” *Eur. J. Obstet. Gynecol. Reprod. Biol.*, vol. 91, pp. 29–36, Jul 2000.

[6] M. Wooldridge, *MultiAgent Systems*. John Wiley and Sons, 2002.

[7] D. Weyns, A. Omicini, and J. Odell, “Environment as a first class abstraction in multiagent systems.” *Autonomous Agents and Multi-Agent Systems*, vol. 14, no. 1, pp. 5–30, 2007.

[8] S. Bromuri, M. I. Schumacher, and K. Stathis, “Towards distributed agent environments for pervasive healthcare,” in *MATES*, ser. Lecture Notes in Computer Science, J. Dix and C. Witteveen, Eds., vol. 6251. Springer, 2010, pp. 125–137.

[9] S. Bromuri and K. Stathis, “Distributed Agent Environments in the Ambient Event Calculus,” in *DEBS ’09: Proceedings of the third international conference on Distributed event-based systems*. New York, NY, USA: ACM, 2009.

[10] W. Chen and D. S. Warren, “C-logic of Complex Objects,” in *PODS ’89: Proceedings of the eighth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*. New York, NY, USA: ACM Press, 1989, pp. 369–378.

[11] R. Kowalski and M. Sergot, “A logic-based calculus of events,” *New Gen. Comput.*, vol. 4, no. 1, pp. 67–95, 1986.

[12] R. Blanco, J. Wang, and P. Alencar, “A Metamodel for Distributed Event-based Systems,” in *DEBS ’08: Proceedings of the second international conference on Distributed event-based systems*. New York, NY, USA: ACM, 2008, pp. 221–232.

[13] A. C. Kakas, R. A. Kowalski, and F. Toni, “Abductive logic programming,” *J. Log. Comput.*, vol. 2, no. 6, pp. 719–770, 1992.

[14] P. Flach, *Simply logical: intelligent reasoning by example*. New York, NY, USA: John Wiley & Sons, Inc., 1994.