2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)

An Agent Program in an IoT System to Recommend Plans of Activities to Minimize Childhood Obesity

Lucas V. Alves
Atlantic Institute
Graduate Program in Computer Science
State University of Ceara
Fortaleza, Brazil
e-mail: lucas_alves@atlantico.com.br

Rodrigo T. de Melo
Atlantic Institute
Graduate Program in Computer Science
State University of Ceara
Fortaleza, Brazil
e-mail: rodrigo_melo@atlantico.com.br

Leonardo F. da Costa
Atlantic Institute
Graduate Program in Computer Science
State University of Ceara
Fortaleza, Brazil
e-mail: leonardo_costa@atlantico.com.br

Cleilton L. Rocha
Atlantic Institute
Fortaleza, Brazil
e-mail: cleilton_rocha@atlantico.com.br

Eriko W. de O. Araujo
Atlantic Institute
Fortaleza, Brazil
e-mail: eriko@atlantico.com.br

Gustavo A. L. de Campos
Graduate Program in Computer Science
State University of Ceara
Fortaleza, Brazil
e-mail: gustavo.campos@uece.br

Jerffeson T. de Souza
Graduate Program in Computer Science
State University of Ceara
Fortaleza, Brazil
e-mail: jerffeson.souza@uece.br

***Abstract*— Overweight and obesity in children is a recognized worldwide epidemic. They are associated with several current and future chronic diseases. OCARIoT is a joint EU-Brazil joint that aims to develop a sophisticated, noninvasive, unobtrusive, personalized IoT system to detect and normalize the behaviors that put a child at risk of developing obesity or eating disorders. In a recent written work, we proposed the design of an agent-based approach to recommend individual physical and food-related activities, based on data collected from wearable devices. In this paper, we present the design of an expanded approach that, in addition to recommendations for individual activities, should recommend activity plans, i.e., sequences of activities organized to minimize childhood obesity. The first results with the extended version were very promising. During the experiments, the selected individual activities and sequences of activities organized by the approach proved to be effective in conducting children, with different profiles and initial states, to the desired states of various attributes associated with childhood obesity.**

Keywords-Childhood obesity; IoT systems; Recommendation systems; Intelligent Agents

I. INTRODUCTION

Childhood obesity has been considered a global epidemic by all renowned health organizations, being regarded as one of the biggest medical problems of the 21st century. The number of school-age children and adolescents with obesity has risen more than 10-fold in the last four decades, from 11 million to 124 million in 2016, and the number of overweight children was estimated to be around 216 million in that year [1]. The number of overweight or obese children in the world under the age of five was estimated to be over 38 million in 2017, increasing to over 70 million by 2025, if the current trend continues [2, 3]. From 1975 to 2016, the prevalence of obesity in children increased from 0.7% in girls to 5.6%, and from 0.9% to 7.8% in boys [1].

There is a lack of knowledge about the main biological factors, causes and risk factors of obesity. The causes include unhealthy diet and eating habits, sedentariness, sleep deprivation, including genetic and medical problems. Behavioral and eating habits are considered as determinants of obesity [4]. Research confirms the strong connection between a sedentary lifestyle and obesity, which we should focus on finding strategies to incorporate nutrition-related habits and physical activities in children [5].

OCARIoT is a joint EU-Brazil project that aims to develop a sophisticated IoT system to detect and prevent the behavior of children who develop obesity or eating disorders [6]. The objective of OCARIoT is to promote the improvement of eating and physical habits and the prevention of the appearance of obesity in children between 9-12 years of age, enabling them to take control of their health. In a recent article we described the design and testing processes of an agent system specially designed to recommend customized preventive and corrective actions for each individual child of the IoT system [7].

This new work proposes another important component in the IoT system, to provide a personalized coaching plan to prevent obesity. We present the design of an expanded approach to recommending activity plans to children, i.e.

978-1-7281-7303-0/20/$31.00 ©2020 IEEE
DOI 10.1109/COMPSAC48688.2020.0-181

sequences of activities over time, to minimize childhood obesity, while allowing children to remain active and engaged in their well-being and healthy habit management. Our approach is based on notions of Artificial Intelligence (AI), especially the notion of intelligent artificial agents [8]. AI techniques can help predict health problems, identify risk factors for different diseases, and recommend health interventions.

The paper is organized in more six sections. Section II presents the main related works. Section III informally defines the subproblems components of the recommendation problem. Section IV presents the theoretical framework supporting the approach. Section V describes the recommender system. Section VI, the main results and the analyses of the system performance. Finally, Section VI presents the conclusion and the future works.

II. Related works

In [8], we propose an agent-based approach to recommend eating and physical activities to children, based on data collected from wearable devices. It was the first computational approach in the context of OCARIoT system. It is described in Section IV. In this section, we report on the scientific literature related to the application of information technology, specially related to artificial intelligence mechanisms, addressing different aspects of childhood obesity, including simulation, prediction, and intervention.

In [9], authors proposed a multi-agent-based simulation system for determining the complex relationships between physical activity and childhood obesity, by attempting to simulate the daily life of 2 to 18 years old children, while considering the child's social and physical environments, his/her psychological state and physical health. The system employs a two-level architecture. The Environmental Level is composed of a Physical Environment, containing physical components (home, school, etc.), which can offer activities (play, sleep, etc.) and traveling activities (take the bus, walk, etc.), and a Social Network, containing individuals simulated by artificial agents. The Intra-Agent Level targets the internal components of each simulated individual. It models both the physical environment and social networks as perceived by the individual. The proposed system was not implemented and therefore evaluated.

In [10], authors propose an agent-based simulation framework to study the impact of physical activity on the prevalence of childhood obesity. Child agents are represented by eight attributes: age, gender, weight, height, body mass index, weight status category, daily caloric intake, and energy expenditure. The three first are independent, and body mass index and weight status category are based on their values. Caloric intake and energy expenditure are random variables. To generate the caloric intake and the energy expenditure, the system considers a probability distribution, child profile, and physical activity policies at school. Simulation results suggested that even moderate-intensity physical activities can help reduce the fraction of overweight and obese children.

In [11], authors evaluated six machine learning techniques (RandomTree, RandomForest, ID3, J48, NaiveBayes, and BayesNet) to predict childhood obesity based on clinical data collected from age 2 to 10. The experiments considered 167 attributes collected before the patient's second birthday. After the second birthday, based on the BMI percentile, a binary data label of OBESE was created. The Random tree and ID3 methods had the highest sensitivity in the analyses, with ID3 obtaining a positive predictive value of 84% and a negative predictive value of 88%.

In [12], authors applied a variety of classifiers (Bayes Net, J48, Naive Bayes, MLP and SMO) to predict childhood obesity. The study considered 12 years old students from 153 schools in Malasia. Besides BMI and the result of physical tests (step-up, push-up test, partial curl-up and sit and reach), a questionnaire was applied to collect socio-demographic, physical activity, and dietary information. The classifiers J48 and SMO achieved the best results, an overall accuracy of 82.56% and 82.08%, respectively, when CfsSubsetEvaluator with genetic search was used to select features.

In [13], authors bring a systematic literature review examining published research on the effect of health video games on childhood obesity. Authors considered papers published between 2010 and 2013. At the end of data collection, 14 articles representing 14 different interventions were selected. The authors found statistically significant differences in obesity related outcomes (specially BMI). In one study they found a significant reduction only when considering the first half of the intervention and, in another, authors found this outcome among girls only.

III. Recommendations of Plans of Activities

The recommendation problem previously addressed was defined as a specific monitoring problem [8]. Given the continuous information perceived about the states of a child, the task in our problem is to select messages of effective actions related to the prevention of the child's obesity states. The process of solving the problem should interpret the child's input information and recommend appropriate intervention actions.

To base the actions to be issued, we first decide to solve the problem of classifying the child's perceived states of obesity. Given a set of n known classes of child health patterns and the perception of a new pattern of a specific state of the child, the task in our classification problem is to associate the new pattern with the known class whose patterns are closest to the new perception.

In the OCARIoT classification specific subproblem, classes were defined by a set of rules, which defines, based on some features, whether a given child should be classified into a given class. The general form of a heuristic rule adopted for deciding about classes looks like: if (a feature value is one of the predefined specific and possibly dangerous situations), then (report on the situation and consequences).

Given the solution to the classification proble, the current and desired state descriptions for the child, the task in our recommendation problem is to decide, from a set of possible preventive or corrective action messages, which should be sent to the child. The general form of a rule for making these choices is: if (possible dangerous health situation) then (send the actions to be taken), that is, common associations that are made available by an expert in the subject.

The planning problem addressed in this work considers that the current and desired state descriptions for the child and the effect of each possible action on the change of each possible current state are given. The task in the planning problem is to organize a sequence of corrective and preventive actions, to be sent to the child, that is, to change the current state to the desired state of health of the child.

IV. THEORETICAL FRAMEWORK

In most theoretical discussions, solving the planning problem on a computer is characterized as a search process through a tree, more precisely a directed graph, whose 'nodes' contains information about states, or situations, and whose 'branches' contains information about operations, or actions, that transform situations into one another. In general, these graphs contain a starting node and one or more nodes containing descriptions of desired states. In this scenario, a problem-solving system must be able to find a sequence of actions that transform an initial situation into a desired situation, that is, a path from the starting to some goal node

The design of our recommender system is inspired in the Anytime Rational Agent with Limited Performance Hardware (AT-RALPH) architecture [14]. The design problem concerns both the recommender system's *architecture* and its *program*. Its architecture extracts stimuli information from the environment, defined in a set possible states S, and runs an agent program to select/plan rational actions, defined in a set of possible actions A. The program is a representation of an agent function in a programming language, i.e., the state-action mapping agent: $S^* \rightarrow A$.

In the AT-RALPH architecture, *Behaviour is* generated by the execution of the agent program on the architecture, modelled as a sequence of internal states, through which the system passes as percepts arrive and the program executes. Given the environment E and a program expressed in the language £, the recommender system's architecture M generates a behaviour: $M\colon E \times £ \rightarrow S^*$. The notion of rationality is captured by a performance measure, performance: $S \times A \rightarrow \mathbb{R}$, that maps a real number to every episode in the history of the agent in the environment, history: $(s^0, a^0) \rightarrow (s^1, a^1) \rightarrow \ldots \rightarrow (s^k, a^k)$, such that s^0 is the agent's initial health state, s^k is the state in the beginning of the k-th interaction, obtained with the execution of the action a^{k-1}, and a^k is the action by the agent in s^k.

The AT-RALPH architecture ties the problem of limited rationality: "how a system with finite computational resources can generate a utility maximizing behavior in a non-static environment?" It ties together: (a) the design of *metareasoning* decision procedures to select which actions to direct the course of another decision procedure; (b) the *compilation* of decision processes that maximize utility into efficiently-executable policies and goals, and their integration into the decision procedure; (c) the generation of *planning* behavior directed to a represented goal [15, 16].

The decision procedures in AT-RALPH are implemented by four distinct *execution architectures* (EA), running in parallel, each one characterized by one or more types of six types of knowledge employed, where P and P′ represent arbitrary predications, $\underline{s}$ represents a state, $\underline{a}$ an action, U represents a utility function such that U(s) belongs to the set of real numbers: (a) synchronic environment model, $P(s) \Rightarrow P'(s)$, (b) diachronic action model, $P(s) \Rightarrow P'(result(a,s))$; (c) absolute or relative utility model, $P(s) \Rightarrow U(s)$; (d) condition-action rules, $P(s) \Rightarrow best(a,s)$; (e) action-utility rules, $P(s) \Rightarrow U(result(a,s))$; (f) condition-goal-action, $goal(result(a,s)) \Rightarrow best(a,s)$.

AT-RALPH architecture builds the four EA from these six categories: (1) Production systems, (2) Systems based on goals, and (3) Theoretical decision systems, and (4) Action-utility systems. Theoretical Decision Systems combine the knowledge of types (a), (b) and (c) to find the best action. The Goal-based system combines the knowledge of types (a), (b) and (f) to suggest actions that achieve the desired objective condition. The Action-utility system uses knowledge of type (e) to evaluate various actions to select the most valuable one. The Production system uses knowledge of type (d) to provide action choices directly.

Different EAs use equal categories of knowledge. Decision procedures share a single knowledge base. They can operate in parallel under a metalevel control, which decides the priority of each AE in operation based on a cost function. EAs generate the possibility to produce a knowledge compilation process, which must operate on the instances of actions selected by each EA and must generate the compiled knowledge such as condition-action/plans rules for the EAs Production system, from the actions selected by the EAs based on goals, employing uncompiled knowledge about the states of the world, the results of the actions, and the current values of the states of the world [15].

More recently, these EAs were incorporated in the four basic kinds of agent programs, suggested by Russell and Norvig [8], as embodying the principles underlying almost all intelligent systems: (1) Simple reflex agents, (2) Model-based reflex agents, (3) Goal-based agents, and (4) Utility-based agents. In this paper, the formalism for describing the decision-making procedure in each kind of agent program was synthesized from the works about intelligent agents in [8] and [17]. Each program is described as an information processing system decomposed in three subsystems, represented by three functions: the see perception subsystem, the next internal state subsystem, and the action decision-making subsystem.

Simple reflective agents select actions based on their current perception, mapped by the see function, and a set of condition-action rules. The Model-based reflex agent is another type of Production system, that maintains a internal state of its environment in memory and uses the function to update this description. The action function of the Goal-based agent programs selects its actions using the information processed by the next function and the information about desired situations (goal states) in the environment. The utility-based agent action function uses a utility function to map descriptions of internal states in real numbers.

Related to this work, it is worth mentioning the problem-solving agent, a kind of goal-based agent that solves systematic search problems in a fully observable and deterministic environment [8]. This agent characterizes the solution of the problem as a search process through a directed graph that contains a starting node and one or more objective nodes. The problem-solver program developed and currently propagated in the classrooms of at least 750 universities in 85 countries, was designed to find solutions that are sequences of actions, generated from a process of systematic search in the state space of the problem.

Two types of searches can be defined for the agent to perform this process: searches without information, exhaustive in the space of states in length or width, and searches with exhaustive information, but with the aid of heuristics that aim to select portions of space that are in the direction of the desired states.

V. AN AGENT-BASED APPROACH TO RECOMMEND PLANS

In this section we describe the skeleton of the projected agent-based approach. It is composed of three agents of two types: (1) Health_Advisor, a model-based reflex agent, designed to recommend customized preventive and corrective actions for an individual child; (2) Child, a model-based reflex agent, designed to represent each child in the IoT system; and (3) Coach_Planning, a goal-based reflex agent, designed to interact with the Health_Advisor agent and recommend customized plans to achieve the child's health goal states, and send it knowledge in the form of condition-plan rules, compiled from the uncompiled knowledge about the child's perceived states, the results of Health_Advisor's action on the child's current states, after the action has been recommended and executed, and the knowledge of the child's desired states. Figure 1 illustrates the abstract architecture related to the interactions between the three agents.

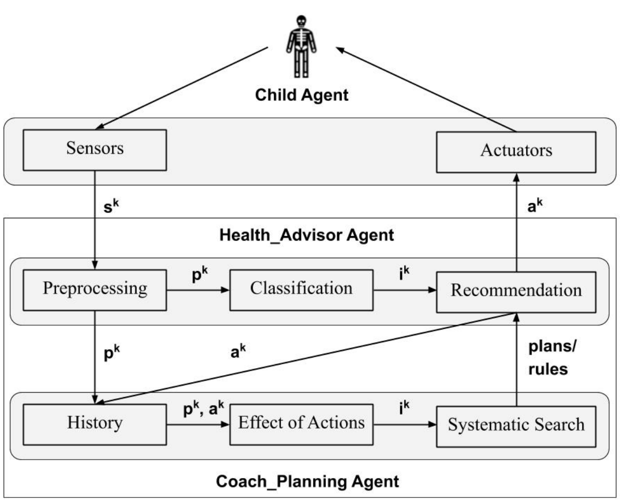

Figure 1. Scheme of Interactions between the three agents.

A. Abstract Architecture

Each agent is an information processing system decomposed in three subsystems, represented by three functions: see, next and action. The Child agent interacts with the Health_Advisor agent that interacts with the Coach_Plannig agent. The scheme assumes that at any moment k in a total time K, through sensors, the Health_Advisor agent receives stimuli information about the Child, $s^k \in S$, i.e., a current state defined on the set of n possible health states for a child, and:

(1) its perception subsystem, named Preprocessing subsystem in Figure 1, see: $S \rightarrow P$, processes the health state information $s^k \in S = \{s_1, s_2, \ldots\}$, and maps it to one possible perception $p^k \in P = \{p_1, p_2, \ldots\}$, that are computational representations of aspects in the health state s^k;

(2) its internal state update subsystem, named Classification in Figure1, next: $P \times I \rightarrow I$, maps the current perception in $p^k \in P$ and the current internal state, $i^{k-1} \in I = \{i_1, i_2, \ldots\}$, to a new internal state $i^k \in I$, considering a model of the Child agent;

(3) its decision-making subsystem, named Recommendation subsystem in Figure 1, action: $I \rightarrow A$, processes the internal states $i^k \in I$ and selects one action in the set of possible actions for the agent, $a^k \in A$, according to a set of condition-action rules;

(4) its actuators recommend the selected action $a^k \in A$ to the Child agent and sends the episode (p^k, a^k) to the Coach_Planning agent.

In parallel, at any moment k, the Coach_Planning agent receives the stimuli information about the Child, $(p^k, a^k) \in SxA$ sent by the Health_Advisor agent, i.e., an episode in history of the interaction between the Child and the Health_Advisor agents, and:

(5) its perception subsystem, named History, assembles the perceived items in the episodes the previous history $(p^0, a^0) \rightarrow (p^1, a^1) \rightarrow \ldots \rightarrow (p^k, a^k)$;

(6) its internal state update subsystem, named Effect of Actions in Figure 1, learns the effect of the Health_Advisor's recommended actions $a^k \in A$, in the Child' health states $p^k \in P$, result:$PxA \rightarrow P$;

(7) its decision-making subsystem, named Systematic Search subsystem in Figure 1, designs plans, i.e., sequences of actions that transform the current state $p^k \in P$ in the health desired states of the Child;

(8) its actuators compiles condition-plans rules and sends them to the Health_Advisor agent, that can now recommend plans formed by at least one corrective or preventive action.

In interaction k+1, the Child agent changes its current state $s^k \in S$ to a new state $s^{k+1} \in S$, according to the executed action $a^k \in A$, and the Health_Advisor agent initiates another cycle involving the perception of the environment through the see function, the updating of its internal state through the next function, the selection of a new action by the action function, the recommendation through its actuators to the Child agent, and sending an episode to the Coach_Planning.

B. The Health_Advisor Agent

The Health_Advisor in Figure 1 was designed in our first approach to the recommendation problem [7]. The agent adopts a factored representation of the health states measured by the sensors monitoring the Child agent, splitting up each observed perception $p^k \in P$, the computational representation of health state $s^k \in S$, into a fixed set of N attributes, $X = \{x_1, x_2, ..., x_N\}$, each of which has a value defined in a given domain, e. g., $p^k = (valx_1^k, valx_2^k, ...)$. So, considering the perception $p^k \in P$ in the output of the see function and a set of rules, suitable designed to compute the class of each current attribute value, its next function updates its internal state with the information about the class of each attribute value, indicating its adequacy and the level of this adequacy. The rules were elaborated considering the existence of a fictional ruler, that must be defined for each attribute in the set X.

The model of the Child agent, considered by the Health_Advisor agent to update its internal state in (3), is a kind of knowledge in the class of synchronic environment model, in the context of the execution architectures (EA) in the AT-RALPH architecture previously outlined in Section III. As proposed in the first version of this agent program, this knowledge was available in the problem domain as an expertise in the form of a set of heuristic rules (theories). After the Health_Advisor's next function has updated its internal state i^k, inserting the information about the perception p^k and about the classes of all current attribute values in p^k, indicating the adequacy and the severity level of the current values, in (4) its action function selects a recommendation action $a^k \in A$, based on a set of condition-action rules, stated in the form of a single input and a single output rule, that must be applied to each attribute $x_i \in X$.

The consequent in the rules are defined employing three functions with specific arguments. The preventive_rec function was defined to return a suitable preventive recommendation action, considering the adequate health classes information in the Health_Advisor's internal state. The corrective_rec_above function was defined to return a suitable corrective recommendation action, considering the above inadequate health classes in its internal state. The corrective_rec_bellow function was defined to return a suitable corrective recommendation action, considering the bellow inadequate classes. Each recommendation action in the rulers employed to solve the classification problems, was represented in the form of an atomic message or of a sequence of messages considering two fundamental messages: inform and request [15].

C. The Coach_Planning Agent

According to Figure 1 and the (1)-(8) information processing phases in the two artificial agents, described in the first subsection, the learned information in (6), about the effect of the Health_Advisor's recommended actions in the Child's health states, result:PxA $\rightarrow$ P, is employed by the Coach_Planning agent in the design of rational plans. In this paper we considered that the value of each health attribute $x_i \in X$, perceived in $p^k = (valx_1^k, valx_2^k, ...)$, is defined in the set of real numbers. So, to each attribute, the learned result function computes a consequent range of values in a discrete subset of real numbers $Cons_i = [Inf_2, . . ., Sup_2]$, as the effect of each action $a^k \in A$ over an antecedent range of values in a discrete subset of real numbers $Ant_i = [Inf_{1i}, . . ., Sup_{1i}]$, such that $|Ant_i| = |Cons_i| = n$; Inf_{1i}, Sup_{1i}, Inf_{2i} and Sup_{2i} are real numbers; and $Inf_{1i} \leq Sup_{1i}$ and $Inf_{2i} \leq Sup_{2i}$.

To compute the approximate consequent value p'^{k+1} when the antecedent value p'^k are not yet known, that is, $result(p'^k, a^k) = p'^{k+1}$ and p'^k was not perceived in any history generated by the interactions of the Health_Advisor and Child agents, we developed an interpolative linear associator [18], that must be specialized to compute the approximate value of each attribute $x_i \in X$. The associator considers the values in the learned ranges as exemplars patterns and maps antecedent values in consequent values, representing the effect of actions in the values of health attributes, that is, Ω_i: $Ant_i \times A \rightarrow Cons_i$, such that: (1) $\Omega_i([Inf_{1i}, . . ., valx_i^k, . . ., Sup_{1i}], a^k) = [Inf_2, . . ., valx_i^{k+1}, . . ., Sup_2]$ when $valx_i^k \in Ant_i$; and (2) $\Omega_i([Inf_{1i}, . . ., valx_i^k + d, . . ., Sup_{1i}], a^k) = [Inf_2, . . ., valx_i^{k+1} + e, . . ., Sup_2]$ when $valx_i^k \in Ant_i$ but $valx_i^k + d \notin Ant_i$, d and e are real numbers, $Inf_{1i} \leq valx_i^k + d \leq Sup_{1i}$ and $Inf_{2i} \leq valx_i^{k+1} + e \leq Sup_{2i}$.

The Coach_Planning agent's action function in (7) employs the result function to solve the problem of designing rational plans to be sent to the Child agent. More concretely, its action function was designed as a Problem-solving agent [8]. This agent characterizes the problem solving as a search process through a directed graph, whose 'nodes' contains information about states $p \in P$ and whose 'branches' contains information about actions $a \in A$, that transform states into each other, according to the learned result function. The graph contains a starting node, that contains information about $p^0 = (valx_1^0, valx_2^0, ...)$, and one or more goal nodes, that contain information about desired health states for the Child agent.

The agent's action function considers information about these desired states, described in terms of another function Goal_Test: $P \rightarrow \{V, F\}$, which maps states $p \in P$ to a true or false value. When true, the Coach_Planning agent knows that it can end the search process, as it has generated a desired state $p' \in P$. The graph_search function in the agent was programmed using a data structure called 'frontier', more specifically a list that serves to store the nodes that have not yet been expanded. This list is initialized with the node that contains the description of the initial state of the problem. The program repeatedly performs a cycle of three main actions: (1) removes the first node from the frontier, (2) tests to see if the state described in the removed node is a meta state and, if not, (3) branches this node, applying all possible actions in the state represented in the node, generates new nodes and inserted them in the frontier list according to some order, at the beginning or at the end.

The cycle of actions continues until the removed node contains a desired state or the entire state space of the problem has been covered. The order in which the graph_search function inserts the generated nodes in the frontier list is determined by the search strategy under consideration. For example, employing uninformed search strategies to decide which will be the next node to be branched in the search tree, we have that the Breadth-first search simply inserts the nodes that were branched at the end of the frontier, that is, the list functions as a queue, and the Depth-first search inserts the new nodes at the beginning of the frontier, that is, the list acts as a stack. Employing informed search strategies, such as Greedy best-first search and A* search, that associates a heuristic assessment function value with each node in the search tree, the frontier list acts as a priority queue.

As in the AT-RALPH architecture, in the design of our recommender system, the two kinds of agents are proposed to run in parallel, overlapping in their knowledge needs and using a single knowledge base. The metalevel control system must connect to its object level in the appropriate way. The decision procedure must correctly integrate the conclusions of each of the agents under metalevel control, and the agents are connected to each other by *compilation* processes, that convert in (8) instances of execution of the goal-based agent Coach_Planning into knowledge appropriate to the model-based reflex agent Health_Advisor. This knowledge, in the form of condition-plans rules, is compiled from the uncompiled knowledge about the states of the Child, the results of the Health_Advisor's action in the Child's states, after the actions have been recommended and performed, and the knowledge about the Child's desired states.

VI. Example of Application and Validation

This section presents some initial results of the interactions between the Health_Advisor with the Child agent and with the Coach_Planning agent in various scenarios. In the previous paper [7] we focused on the evaluation of the health state classification and on the actions recommended by the Health_Advisor agent to the Child agent, the effect of these actions in its health states, and the performance measured in the whole history of the interactions between them. This paper focus on the evaluation of the plans generated and the rules compiled by the Coach_Planning agent. The first subsection presents the validation design and the second presents the results and analyses of the designed agent-based approach.

A. *Validation Design*

As in the previous paper, we chose two important attributes to characterize the child health state, that is, the Body Mass Index (BMI) and the Sleep Duration (SD). Tables I-III specify some of the different scenario settings to evaluate the designed agents. Table I presents the standard weight status categories associated with BMI ranges (Kg/m^2) standard sleep status categories associated with SD ranges (h) for children 6-13 years. But also, the agent can make diet, exercise, and healthy sleep tips recommendations in other to increase the child's SD quality. Table II presents nine examples of these recommendation actions. Table III presents the expected impact in the value of each standard category of the attributes BMI and SD.

TABLE I. Standard Categories and Associated Ranges

BMI Category	Range	SD Category	Range
Obesity	(30.0, 40.0]	Not recommended_1	(11, 15]
Overweight	(25, 30.0]	Recommended	[9, 11]
Health weight	[18.5, 25.0]	May be appropriate	[7, 9)
Underweight	[10.0, 18.5)	Not recommended_2	(7, 4]

TABLE II. Examples of Recommendation Actions

Action	Description
a_1	More than 300 min. of moderate-intensity activity a week
a_2	Eat 2-4 servings of low-fat or no-fat dairy products/day
a_3	> 150 min. moderate/vigorous-intensity aerobic activity
a_4	> 150 min. moderate-intensity muscle-strengthening activ.
a_5	Ask your doctor or a dietitian for advice
a_6	Exercise daily
a_7	1 hour moderate/vigorous exercise 2-3 hs before bedtime
a_8	Practice a relaxing bedtime ritual
a_9	Go to sleep and wake up at the same time every day

TABLE III. Impact of Action in the Attribute Classes

	Action	Class	Impact	
BMI	a_1	'Obesity'	very_very_high_upper	-0.9
	a_2 a_3 a_4	'Health weigth'	zero	0.0
	a_5	'Underweight'	medium_below	0.5
	a_6	'Recommended'	zero	0.0
SD	a_7	'Notrecommended1'	medium_upper	-0.5
		May be appropriate	very_very_low_below	0.9
	a_8	'Notrecommended2'	very_high_below	0.7
	a_9	'Recommended'	zero	0.0

The information in Table I is employed by the Health_Advisor agent to classify its percepts about the health state of the Child agent and to update its internal state by its next function. The information in Tables II and III are employed by the agent, in addition to the information in the internal state, to select and to recommend actions to the Child. In this paper the human agent Child was simulated by another model-base reflex agent. So, in interaction k+1, by means of its see function, the Child agent perceives the recommendation action $a^k \in A$, sent by the Health_Advisor agent, and generates its current perception p^{k+1}. Considering p^{k+1} and the information about the previous internal state i^k, its next function generates a new internal state i^{k+1}. In the sequence, its action function considers i^{k+1} and selects an adequate action, i.e., it computes the Child agent's new state $s^{k+1} \in S$, to be sent to the Health_Advisor agent.

To compute s^{k+1}, domain-specific information, named ModelChild [8], was considered by the Child agent's next and action function. We encapsulate in the simulated Child agent one kind of knowledge related to its physical twin Child agent, that is, about the effect of the execution of possible recommendation actions $a^k \in A$ in the last Child's health state, $(i^k, a^k) \Rightarrow i^{k+1}$. To implement this kind of internal state updating, besides the information about the value of each attribute $x_i \in X = \{BMI, SD\}$, defining a health state, the Child agent also employs a fictional ruler associated to each x_i, that can be equal or not the respective fictional ruler employed by the Health_Advisor agent to classify and recommend actions.

Besides the fictional rulers the effect of recommendation actions is dependent of possible parameters to set up. For example, there are parameters to control the convergence velocity of the effects of executed actions to states in the health margin of each attribute. Depending on the value of these parameters, it may happen that, instead of converging to the health margin, the effects of the actions performed cause fluctuations in the values of the resulting states. The initial impact of actions in some states depends on initialization of some parameters, which can affect the convergence velocity too. So, the performance of the agent approach's recommendations will be affected by the properties of the rules encapsulated in the Child agent and by the values of all these parameters. The next subsection presents two experiments: the first one, E1, where the agents have the same fictional rulers, and second one, E2, where they have different rulers.

B. Results and Analyses – Health_Advisor Agent

Figure 2 shows two sequences of actions recommended by the Health_Advisor agent to the Child agent, and the effect of each action in terms of changing the values of BMI and SD attributes, during a thirty parts period (K=30) and seven interactions (N=7) between the agents. Table IV shows the agent's log. Table V, the first four episodes of its history the values of attributes BMI and SD in each interaction k. The last two columns in Table IV are related to the classification mechanism (Classification 1) described in the Health_Advisor, such that the letters 'i' and 'a' represent respectively the classes 'inadequate' and 'adequate', the letters 'a' and 'b' in the first term of each class represent respectively the positions in the rulers employed for classification, named 'above' and 'below' the health margins, the second term represents the difference from the current value BMI^k and SD^k and the ideal values BMI^* and SD^*, and the third term represents the impact that an adequate action must perform, that is, a value in the interval [-1, 1].

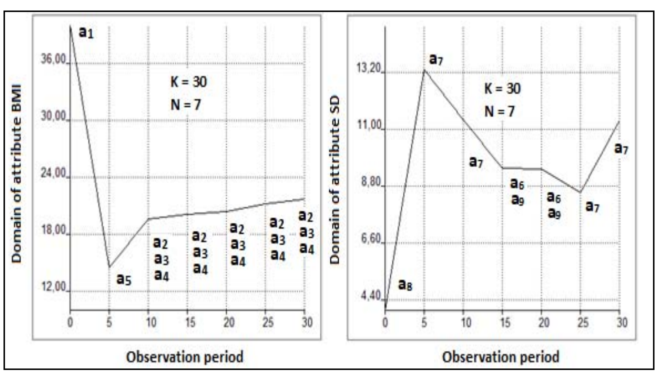

Figure 2. Effect of Actions in the the Values of the Attributes.

TABLE IV. HEALTH_ADVISOR AGENT'S LOG – PART I

k	BMI^k	SD^k	$Class^k_{BMI}$	$Class^k_{SD}$
1	40.0	4.0	i(a,15,-1.0)	i(b,5,1.0)
2	14.5	13.3	i(b,4,0.5)	i(a,2,-0.6)
3	19.6	11.4	a(1,0)	i(a,0,-0.1)
4	20.1	9.5	a(1,0)	a(1,0)

TABLE V. HEALTH_ADVISOR AGENT'S LOG – PART II

	Classification 2		Decision-making	
k	$Class^k_{BMI}$	$Class^k_{SD}$	a^k_{BMI}	a^k_{SD}
1	Obesity	Notrecommended_2	a_1	a_8
2	Underweight	Notrecommended_1	a_5	a_7
3	Health weight	Notrecommended_1	a_2 a_3 a_4	a_7
4	Health weight	Recommended	a_2 a_3 a_4	a_6 a_9

The columns in Table V are related to the second part of the classification mechanism (Classification 2) described by the attribute category values in Table II, and to the recommendation actions selected by the Health_Advisor's decision-making subsystem. The classification mechanism and the recommended actions were satisfactorily selected in each interaction, as the performance pointed maximally in almost all seven episodes, and the differences measured have fallen from one interaction k to another interaction k+1, when the selected recommended actions by the decision-making subsystem were performed in each interaction k.

The Child agent was set in two configurations. The first and previous configuration illustrates the cases where the Health_Advisor agent and the Child agent have the same fictional rulers (E1). The second, the cases where the agents have different parameters set up from the first configuration (E21) and the case where the agents have different rulers (E22). Figure 3 shows the actions effect in terms of changing the values of SD attributes during a thirty parts period (K = 30) and eleven interactions (N = 11) between the agents. The graph on the left, illustrates the case of experiment E21. The graph on the right, illustrates the case of E22.

The changes in the parameters did not cause any substantial change in values obtained in E21. However, in the first interactions, the change accelerated the convergence of SD values to the health margins, such that in the interaction N=3 the current value was classified as adequate, and it stayed in the adequate class till the interaction N=8. Second, the change in the changes in the parameters caused an oscillation in the SD value, as it escaped the health margin values and was classified as inadequate in the remaining interactions. The changes in the fictional rulers caused a fluctuation in the SD value in E21.

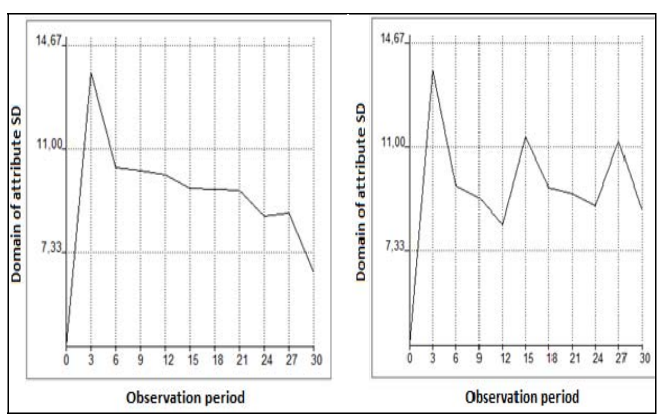

Figure 3. Effect of Actions in the Values of SD in E21 and E22.

TABLE VI. HEALTH_ADVISOR'S PERFORMANCE IN E1 AND E2

	BMI^k		SD^k	
	E1	E2	E1	E2
Ideal	21.0	26.0	10.0	7.0
Σ Diff	44.8	**71.9**	17.6	**33.0**

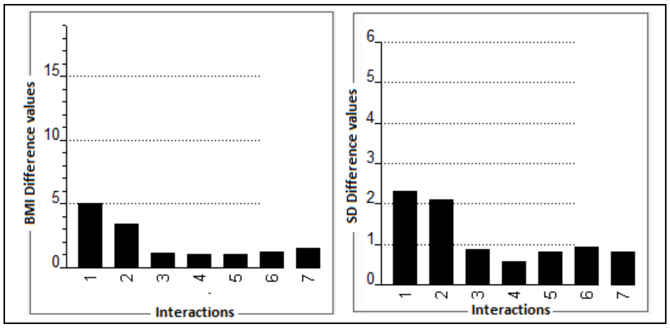

Figure 4. Means of difference values in E1.

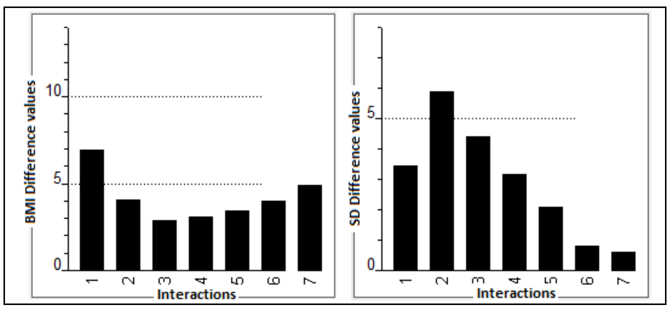

Figure 5. Means of difference values in E2.

To better verify the effect of the recommended action, in terms of the reduction of the differences in each interaction, Table VI shows the Health_Advisor's performance measured in E1 and E2, considering a suite of test with 240 test cases. Figure 4 and 5 shows graphically the means of difference values associated to BMI and SD attributes, in each of seven interactions (N=7) in a twenty parts period (K=20). Table VII shows the means of difference values, the standard deviation and variance values in each interaction.

TABLE VII. MEANS OF DIFFERENCE VALUES IN E1.

	BMI			SD		
k	$Mean^k$	$St Dev^k$	Var^k	$Mean^k$	$St Dev^k$	Var^k
1	5.2	5.4	29.3	2.3	1.7	2.8
2	3.6	2.5	6.0	2.1	1.2	1.4
3	1.2.	1.1	1.1	0.9	0.4	0.1
4	1.1	1.1	1.3	0.6	0.3	0.1
5	1.2	1.2	1.4	0.8	0.5	0.2
6	1.3	1.1	1.3	0.9	0.4	0.1
7	1.6	1.1	1.2	0.8	0.4	0.1

Figures 3 and 4 are in accordance with the expected. The parameters controlling the effect of actions in the Child side produced similar results considering the suite of test evaluated. The sharper oscillation in Figure 4 than in Figure 3 equals a larger reduction in the differences observed int Table VI and VII. Regarding the performance measured in this table, these figures show that the recommendations made were oriented to bring the current values of the two attributes into the health margin, mainly from the first interactions in which there is a greater reduction in the means of differences and their values of variance and standard deviation.

C. Results and Analyses – Coach_Plannig Agent

Tables IV and V show an example of the main components of the history of the interactions between the Health_Advisor and the Child agents. More specifically, the columns 2 and 3 in Table IV show respectively the perceived values of BMI and SD attributes, $valx_{BMI}{}^k$ and $valx_{SD}{}^k$ $(k = 1, 2, 3, 4)$. The columns 4 and 5 in Table V show the selected actions, $a^k{}_{BMI}$ and $a^k{}_{SD}$, by the Health_Advisor agent. For example, Table VIII shows the partial history registered with four episodes in the two tables. Based on this last table, Table IX illustrates the respective components of the result function learned by the Coach_Planning agent considering this partial history.

The assembly scheme in Table IX is employed by the Coach_Planning agent, by its next function, to learn the result function every time the Health_Advisor agent interacts with the Child agent. In order to evaluate the plans generated by the Coach_Planning we begin with an empty database, that is, without the knowledge about the result obtained by the execution of actions and, to populate the database, we promote the interaction of the Health_Advisor with the Child in several initial health states, considering the ranges of the attributes described in Table I. Table X shows these initial states for the two attributes ($valx_{BMI}{}^0$ and $valx_{SD}{}^0$) and their corresponding range of desired health states ($valx_{BMI}{}^*$ and $valx_{SD}{}^*$).

TABLE VIII. HISTORY OF THE INTERACTION BETWEEN AGENTS

k	$valx^k{}_{BMI}$	$a^k{}_{BMI}$	$valx^k{}_{SD}$	$a^k{}_{SD}$
1	40.0	a_1	4.0	a_8
2	14.5	a_5	13.3	a_7
3	19.6	a_2 a_3 a_4	11.4	a_7
4	20.1	a_2 a_3 a_4	9.5	a_6 a_9

TABLE IX. ACTIONS EFFECTS LEARNED BY COACH_PLANNING

$valx^k{}_{BMI}$	$a^k{}_{BMI}$	$valx^{k+1}{}_{BMI}$	$valx^k{}_{SD}$	$a^k{}_{SD}$	$valx^{k+1}{}_{SD}$
40.0	a_1	14.5	4.0	a_8	13.3
14.5	a_5	19.6	13.3	a_7	11.4
19.6	a_2 a_3 a_4	20.1	11.4	a_7	9.5
20.1	a_2 a_3 a_4	-	9.5	a_6 a_9	-

TABLE X. INITIAL AND DESIRED STATES OF BMI AND SD

$valx^0{}_{BMI}$	$valx^*{}_{BMI}$	$valx^0{}_{SD}$	$valx^*{}_{SD}$
40.0	[18.5, 25.0]	15.0	[9.0, 11.0]
10.0	[18.5, 25.0]	4.0	[9.0, 11.0]
14.0	[18.5, 25.0]	6.5	[9.0, 11.0]
32.5	[18.5, 25.0]	12.5	[9.0, 11.0]
39.5	[18.5, 25.0]	14.5	[9.0, 11.0]

The order in the table is important to consider since, in the beginning of the tests, the Coach_Planning's database is empty and, therefore, the agent cannot design plans from any initial states. As it goes through new initial states the database is being populated, increasing the chances of the agent perceiving initial states previously perceived and generating sequences of actions that take him to some desired state.

Table XI shows two plans designed by the agent, for the case of BMI attribute, when it starts with the last initial states in Table X, that is, after it has experienced the others first four initial states. The table also presents the states the Coach_Planning explored till it could conclude the two plans designed. It is worth mentioning that, although the agent had designed thirteen plans for the BMI attribute, it did not generate plans for the SD attribute, since the first four experiences in Table X passed by the agent was not sufficient to produce the necessary knowledge to do it. In order to minimize this problem, that is, the necessity to have a lot of experiences till the agent can be able to design, we proposed: (1) to learn the result function as to compute a consequent range of values in the set of real numbers, as the effect of an antecedent range of values in the same set, for each possible action; (2) to approximate the consequent values when the antecedent values are not yet known employing an interpolative linear associator.

TABLE XI. PLANS AND STATES FOR BMI

	BMI
Plan 1	$a_5 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6 \rightarrow a_4 \rightarrow a_4 \rightarrow a_1$
	39.7 → 38.9 → 37.4 → 35.4 → 32.8→ 29.9 → 26.7 → 22.9
Plan 2	$a_5 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6 \rightarrow a_4 \rightarrow a_4 \rightarrow a_1 \rightarrow a_8 \rightarrow a_8$
	39.7 → 38.9 → 37.4 → 35.4 →32.8 → 29.9 → … → 22.8

Table XII shows examples of some components of this new result function after the agent has experienced the fifth case in Table X. The first and third components in the new result function are respectively the antecedent values and the consequent values attained with the execution of the action, in the second component in the function, collected during the Coach_Planning experience. Table XII shows the plans designed by the agent when its initial state is equal to the fifth case in Table X. It shows the states the agent passed and the antecedent and consequent values that the agent perceived previously, respectively in the second and third lines associated to each plan.

As expected, the agent designed the same plan designed for the case of BMI attribute, showed in Table XI. Therefore, different from the SD case, where there is not any plan in the table, the Coach_Planning agent designed a plan composed of only one action almost at the end of the history of the interaction between the Health_Advisor and the Child agent. To make this earlier, the agent must experience other cases, extending the numbers of states experienced. Finally, considering the number of plans designed during its last experience, the Coach_Planning agent compiled some knowledge in the form of condition-plans rules. Table XIV shows the rules compiled after the agent has experienced the fifth case in Table X.

TABLE XII. THE NEW RESULT FUNCTION

BMI
result([40, 39.74, 38.9], a_5, [39.74, 38.9, 37.42])
result([37.42, 35.36], a_6, [35.36, 32.76])
result([32.76], a_4, [29.9])
SD
result([15, 14.9, 14.61, 14.1], a_{19}, [14.9, 14.61, 14.1, 13.38])
result([13.38, 12.54], 12.54, 13.38), a_{11}, [12.54, 11.55])

TABLE XIII. PLANS AND STATES FOR BMI AND SD

BMI
$a_5 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6 \rightarrow a_4 \rightarrow a_4 \rightarrow a_1$
39.7 → 38.9 → 37.4 → 35.4 → 32.8→ 29.9 → 26.7 → 22.9
[40,39.74,38.9,39.5] → [39.74,38.9,37.42] → [39.74,38.9,37.42] → [35.36,32.76] → [35.36,32.76] → [29.9] → [26.7] → [22.99]
SD
a_{10}
10.44 → 10.3
[10.41,10.31,10.21,10.08,9.98,10.27,10.17,10.07,9.97,10.51, 10.11,10.01,9.91] → [10.31,10.21,10.11,9.98,10.08,10.17, 10.07,9.97,10.41,10.01,9.91]

TABLE XIV. COMPILED CONDITION-PLAN RULES FOR BMI AND SD

Rules for BMI
do(((a_4 or a_7 or a_8) and (a_4 or a_7 or a_8))) if BMI >= 20.21 and BMI =< 23.02
do((a_5 and a_5 and a_6 and a_6 and a_4 and a_4 and a_1 and (a_4 or a_7 or a_8) and (a_4 or a_7 or a_8) and (a_4 or a_7 or a_8)) if BMI >= 38.9 and BMI =< 40
Rule for SD
do(((a_{10} or a_{18}) and (a_{10} or a_{18}))) if SD >= 9.91 and SD =< 10.51

The compiled rules were generated in the form 'plans if condition'. The plans expression in rules consequent have been mined from the lots of plans generated for each initial state and from the number of histories of interactions that the Coach_Planning agent perceived. As the rules can indicate more than one plan to be executed, the consequent sequence of actions to be sent to the Child must be selected considering the semantics of 'and' and 'or' logical connectives. New rules and new conditions for the antecedents of old rules must be compiled as the Health_Advisor and Child agents interact, and the Coaching_Plan learn new component of result function.

Although an effective compilation process depends on the experiences the agent has lived, the compiled rules are very important to maximize the performance measure of the multiagent system, mainly in those cases where the Health_Advisor agent doesn´t know well the Child agent as, for example, in the case that they have different fictional rulers, as in the case we postulate in the E2 experiment. Different from the condition-action rules employed by the Health_Advisor agent, that are made available by a person specialized in the subject, the mined condition-plans rules are learned by the Coach_Planning agent considering the perception of the real impact of the recommended actions on the health states of each specific child.

VII. CONSCLUSIONS AND FUTURE WORKS

Childhood obesity has been considered a global epidemic by all renowned health organizations, being regarded as one of the biggest medical problems of our century. The OCARIoT (EU-Brazil joint project) aims to develop a personalized IoT system that can detect and normalize the behaviors that put a child at risk of developing obesity. The environment task is composed by the children's home and school, the children and their parents, doctors, and teachers. In this paper, we described an agent-based approach to recommend eating and physical activities to children. The validation process showed that the approach is promising, although it will be continued to better validate the proposed agents Health_Advisor and Coach_Planning.

We still intend to elaborate new experiments trying to perceive the influence of the parameters associated to the simulated agent called Child, and the real consequences for the performance of agent approach in the case that the Health_Advisor agent has a different ruler from the Child. We postulate that this will be the real situation in which the agent recommending action. Therefore, the Health_Advisor agent will be adapted to encapsulate the knowledge about the Child, learned and compiled by the Coach_Planning agent, and to correct its parameters and fictional rulers.

In the next stages of IoT system development, we plan to elaborate two other agent programs to be executed in parallel with the Health_Advisor and Coach_Planning agents. The Health_Group agent will be able to group children based on their profiles. The Utility_Planning agent will be able to design plans of corrective and preventive actions for the children based on an utility function and, as consequence, to compile knowledge for the Health_Advisor in the form of condition-plans rules, but whose consequent plans will have more quality than the ones generated by the Coach_Planning agent, as they will be designed oriented by the utility function.

REFERENCES

[1] NCD Risk Factor Collaboration. 2017. Worldwide trends in body-mass index, underweight, overweight, and obesity from 1975 to 2016 a pooled analysis of 2416 population-based measurement studies in 128Âů9 million children, adolescents, and adults. The Lancet 390, 10113 (2017), 2627–2642.

[2] World Health Organization. 2017. Facts and figures on childhood obesity. https: //www.who.int/end-childhood-obesity/facts/en/

[3] World Health Organization. 2018. Joint child malnutrition estimates - Levels and trends (2018 edition). https://www.who.int/nutgrowthdb/estimates2017/en/

[4] K. Kuzbicka and D. Rachon. 2013. Bad eating habits as the main cause of obesity among children. Pediatric endocrinology, diabetes, and metabolismo 19 (01 2013), 106–10.

[5] C. Boreham and C. Riddoch. 2002. The physical activity, fitness and health of children. Journal of sports sciences 19 (01 2002), 915–29. https://doi.org/10. 1080/026404101317108426

[6] OCARIoT Project. Smart childhood obesity caring solution using iot potential. [Online]. Available: https://ocariot.eu/, last accessed on December 2019.

[7] J. T. Souza, G. A. L. Campos, C. Rocha, E. Werbet, L. F. Costa, R. T. Melo, L. V. Alves, "An agent program in an IoT system to recommend activities to minimize childhood obesity problems," SAC '20: The 35th ACM/SIGAPP Symposium on Applied Computing, online event, [Brno, Czech Republic], March 30 - April 3, pp. 654-661, 2020. http://doi.org/10.1145/3341105.3373927.

[8] S. Russel and P. Norvig. 2010. *Artificial Intelligence: A Modern Approach*. Pearson Education, 3rd. Edition (2010).

[9] R. Aziza, A. Borgi, H. Zgaya, and B. Guinhouya. 2014. Physical Activity and Childhood Obesity: A Multi-Agent Simulation. In DCAI 2014 - Distributed Computing and Artificial Intelligence, 11th International Conference. Switzerland , Switzerland. https://hal.archives-ouvertes.fr/hal-01718872

[10] A. Ramirez-Nafarrate and J. O. Gutierrez-Garcia. 2013. An agent-based simulation framework to analyze the prevalence of child obesity. In 2013 Winter Simulations Conference (WSC). 2330–2339. https://doi.org/10.1109/WSC.2013.6721608

[11] T. Dugan, S. Mukhopadhyay, A. Carroll, and S. Downs. 2015. Machine Learning Techniques for Prediction of Early Childhood Obesity. Applied Clinical Informatics 6 (10 2015), 506–520. https://doi.org/10.4338/ ACI-2015-03-RA-0036

[12] Fadzli Syed Abdullah, Nor Saidah Abd Manan, Aryati Ahmad, Sharifah Wajihah Wafa, Mohd Razif Shahril, Nurzaime Zulaily, Rahmah Mohd Amin, and Amran Ahmed. 2017. Data Mining Techniques for Classification of Childhood Obesity Among Year 6 School Children. In Recent Advances on Soft Computing and Data Mining, Tutut Herawan, Rozaida Ghazali, Nazri Mohd Nawi, and Mustafa Mat Deris (Eds.). Springer International Publishing, Cham, 465–474.

[13] A. Lu, H. Kharrazi, F. Gharghabi, and D. Thompson. 2013. A Systematic Review of Health Videogames on Childhood Obesity Prevention and Intervention. Games for health journal 2 (06 2013), 131–141. https://doi.org/10. 1089/g4h.2013.0025

[14] S. Zilberstein and S. J. Russel. Efficient Resource-Bounded Reasoning in AT-RALPH. Artificial Intelligence Planning Systems, Proceedings of the First Conference (AIPS 92) 1992, Pages 260-266.

[15] S. J. Russell. Execution Architectures and Compilation. In*Proceedings of the International Joint Conference on Artificial Intelligence*, 1989.

[16] S. J. Russell. An architecture for bounded rationality. *ACM SIGART Bulletin*, 2(4):146–150, 1991. (doi:10.1145/122344.122374).

[17] Michael Wooldridge (2009). An introduction to multiagent systems. Wiley, 2009.

[18] M. Hazewinkel ed. (2001) [1994], Linear interpolation", *Encyclopedia of Mathematics*, Springer Science+Business Media B.V. / Kluwer Academic Publishers, ISBN 978-1-55608-010-4