

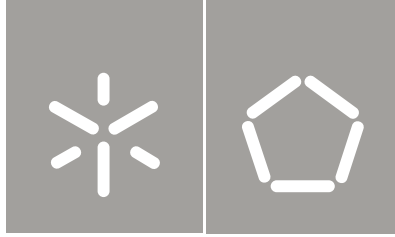


Universidade do Minho
Escola de Engenharia

Vitor Leandro Vieira Oliveira

Personalized-Based Dietary
Recommendation for Weightlifting

Vitor Leandro Vieira Oliveira
Personalized-Based Dietary
Recommendation for Weightlifting



Universidade do Minho
Escola de Engenharia

Vitor Leandro Vieira Oliveira

Personalized-Based Dietary Recommendation for Weightlifting

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia Eletrónica Industrial e Computadores

Trabalho efetuado sob a orientação do
Professor Doutor Paulo Cardoso

outubro de 2013

DECLARAÇÃO

Vitor Leandro Vieira Oliveira

Endereço eletrónico: leandrooliveira0387@gmail.com Telefone: 917911956

Número do Bilhete de Identidade: 13251601

Título da Dissertação:

Personalized-Based Dietary Recommendation for Weightlifting

Orientador:

Doutor Paulo Cardoso

Ano de conclusão: 2013

Dissertação submetida na Universidade do Minho para a obtenção do grau de Mestre em Engenharia Eletrónica Industrial e de Computadores

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA DISSERTAÇÃO/TRABALHO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE;

Universidade do Minho, ___/___/_____

Assinatura: _____

ACKNOWLEDGEMENTS

It was a pleasure for me to work with all the wonderful people in our lab here in Guimarães. First of all, I would like to thank Paulo Cardoso for being a great counselor.

Also a big thanks to Professor Adriano Tavares for his advices, his ideas and tremendous support had a major influence on this thesis. He spent a lot of time helping me as well as all the other people in our lab. I learned a lot during this time and I am convinced that this knowledge will help me in the future. I would like to thank Piyaporn Turnmark for co-operating in my thesis. I enjoyed her interest in my research as well as the fruitful discussions. Also I would like to thank all my friends and family for all the amazing support they gave during the development of this dissertation.

SUMMARY

This dissertation focuses on the development of an expert system for nutrition and menu recommendation, aiming to assist top level sports athletes with their nutrition guideline for better sports performance. This dissertation will describe the food and nutrition ontology using a knowledge-based framework. The developed ontology will work with a rule-based knowledge to provide specific menus for different times of the day and different train phases for the athlete's daily nutritional needs and the athlete's personal preferences for a better sport performance. The main components of this system are the food and nutrition ontology, the athlete's profiles and the nutritional rules for sports athletes.

This dissertation focus on Weightlifting sport, but can be adapted to different sports in the future, since for every different sport the nutritional needs will be different.

Since some needed input values are not constant (weight etc...), a sensor will be used to get all the non-constant values, so an interaction between the application and an external sensor will be made, making this a multi-disciplinary dissertation.

Being the competitive sport activity highly practiced by thousands of people around the world, this is a very attractive area, so this project is very catchy. On the other hand the technologies, both software and hardware, are an interesting way to put the engineering in the service of sport.

This is an innovative project, it could be a viable product for the market today, making this project even more interesting.

Keywords: Ontology, Weightlifting, Nutrition, Sports

RESUMO

Esta dissertação é focada no desenvolvimento de um sistema de recomendação de menus alimentares, com o objetivo de auxiliar de atletas profissionais com diretrizes nutricionais para ajudar a atingir o melhor desempenho desportivo. Esta dissertação irá descrever uma ontologia alimentar e nutricional usando uma estrutura baseada numa base de dados. A ontologia desenvolvida irá trabalhar com uma base de dados baseada em regras para fornecer menus específicos para diferentes momentos do dia e diferentes fases de treino, para ajudar os atletas a atingir a sua *performance* ideal. Os principais componentes deste sistema são as ontologias de comida e nutrição, os perfis do atleta e as normas nutricionais para atletas.

Esta dissertação foca-se em Halterofilismo, mas pode ser adaptado para diferentes modalidades desportivas no futuro, uma vez que para cada desporto existem diferenças a nível nutricional.

Uma vez que alguns valores de entrada necessários, não são constantes (peso, etc...), um sensor será usado para obter todos os valores não constantes, portanto, uma interação entre a aplicação e um sensor externo será realizada, tornando esta dissertação um projeto multidisciplinar

Sendo a actividade desportiva competitiva e profissional, praticada por milhares de pessoas a volta do globo, esta é uma área muito atraente, por isso este projeto é muito cativante. Por outro lado, as tecnologias, *software* e *hardware*, são uma forma interessante de colocar a engenharia a serviço do desporto.

Este é um projeto inovador, que poderia ser um produto viável para o mercado de hoje, fazendo com que este projeto ainda mais interessante.

Palavras-chave: Ontologia , halterofilismo, Nutrição , Esportes

INDEX

Acknowledgements	iii
Summary	v
Resumo	iv
Figure Index	viii
Table Index.....	x
Acronyms.....	xi
Chapter 1	1
Introduction	1
1.1. Motivation.....	1
1.2. Objectives	1
1.3. Structure.....	2
Chapter 2.....	5
Project Background.....	5
2.1. Relevant works on Ontology in sports and nutrition	5
2.2. Periodized Nutrition for the yearly training programme	6
2.2.1. Energy Requirements	7
2.2.2. Daily Menu	8
2.3. Nutrition Background for Weightlifting	9
2.4. Ontology Meaning	10
Chapter 3.....	18
System Analysis	18
3.1. System Overview	18
3.2. System Requirements	18
3.3. Menus calculation	19
3.4. Selection of Sensor and communication Protocol	23
3.5. ANT+ Protocol	24
3.6. Communication with the Scale	25

3.7. User Cases Diagram for the enter menu	33
1. Project Design	35
4.1. Ontology Design	35
4.2. Enter and report interface application analyses	42
4.3. Enter personal information and report interface application subsystems design 43	
2. Project implementation.....	49
5.1. Java Application	49
5.2. Sensor data acquiring Implementation	56
5.3. JNI.....	63
6. Results.....	65
6.1. Interface Results	65
7. Conclusions and future work	75
References.....	77

FIGURE INDEX

FIGURE 1: GENERAL NUTRITION RECOMMENDATIONS DURING DIFFERENT YEARLY TRAINING PHASES FOR STRENGTH AND POWER ATHLETES. NUTRITION RECOMMENDATIONS FOR A 70-KG STRENGTH AND POWER SPORT ATHLETE. (ADAPTED FROM [2, 11])	7
FIGURE 2: THE CUNNINGHAM AND THE HARRIS-BENEDICT EQUATION FOR ESTIMATING ENERGY REQUIREMENT FOR ADULTS (ADAPTED FROM [12])	8
FIGURE 3: THE METABOLIC EQUIVALENT FOR WEIGHTLIFTING. [13]	8
FIGURE 4: THE ARCHITECTURE OF WEB SEMANTICS	12
FIGURE 5: RDF TRIPLE	13
FIGURE 6: JAVA APPLICATION AND OWL INTERACTION	16
FIGURE 7: OVERVIEW OF THE SYSTEM	18
FIGURE 8: TANITA BC 1000	24
FIGURE 9: INSERT PERSONAL INFORMATION INTERFACE PAGE USE CASES	34
FIGURE 10: ONTOLOGY MAIN CONCEPTS	35
FIGURE 11: OBJECT PROPERTIES	39
FIGURE 12: DATATYPE PROPERTIES	40
FIGURE 13: PROTÉGÉ INDIVIDUAL FORM	42
FIGURE 14: ANT+ IN SPORTS DOMAIN	25
FIGURE 15: APPLICATION FLOW CHART	42
FIGURE 16: INSERT PERSONAL INFORMATION FLOWCHART	44
FIGURE 17: FOOD PREFERENCES SUBSYSTEM	45
FIGURE 18: PICK A MENU SUBSYSTEM FLOW CHART	46
FIGURE 19: QUERY'S SUBSYSTEM FLOW CHART	47
FIGURE 20: CHANNEL COMMUNICATION	26
FIGURE 21: ESTABLISH A CHANNEL BETWEEN MASTER AND SLAVE	27
FIGURE 22: SENSOR APPLICATION FLOWCHART	48
FIGURE 23: MAIN PAGE INTERFACE	49
FIGURE 24: FOOD PREFERENCES INTERFACE	50
FIGURE 25: CHOOSE YOUR MEALS FOR TODAY INTERFACE	51
FIGURE 26: REPORT PAGE INTERFACE	52
FIGURE 27: MALE A USER PROFILE	66
FIGURE 28: CHOOSE A MEAL FOR PROFILE MALE A EXAMPLE	67
FIGURE 29: REPORT FOR USER MALE A	68

FIGURE 30: MALE B PROFILE INFORMATION	68
FIGURE 31: ORIGINAL TANITA SCALE SOFTWARE	69
FIGURE 32: SYSTEM SENSOR MEASUREMENTS.....	70
FIGURE 33: FEMALE A PROFILE INFORMATION	70
FIGURE 34: CHOOSE A MEAL FOR PROFILE FEMALE A EXAMPLE.....	71
FIGURE 35: REPORT FOR USER FEMALE A	71
FIGURE 36: MALE B PROFILE INFORMATION	72
FIGURE 37: APPLICATION REPORT SENSOR DATA.....	72
FIGURE 38: ORIGINAL SOFTWARE FEMALE A PROFILE DATA.....	73
FIGURE 39: USER FAVOURITE INGREDIENT TOMATO	73
FIGURE 40: ITEMS WITH INGREDIENT TOMATO	74
FIGURE 41: USER NON FAVOURITE INGREDIENT TOMATO.....	74
FIGURE 42: ITEMS WITHOUT TOMATO INGREDIENT.....	74

TABLE INDEX

TABLE 1: AVAILABLE PROCESS TYPES	41
TABLE 2: LEVEL OF ACTIVITY	20
TABLE 3: TABLE OF TRAINING STAGE AND THE MET'S VALUES	21
TABLE 4: WEIGH GOAL.....	22
TABLE 5: TABLE OF TEE FOR GAIN, MAINTAIN AND LOSE WEIGHT	22
TABLE 6: MEALS TOTAL ENERGY.....	23
TABLE 7: INPUT VALUES NEEDED.....	23
TABLE 8: MAX AND MINIMUM VALUES OF THE INSERTED PARAMETERS.....	43
TABLE 9: COMBO BOXES	46
TABLE 10: CHANNEL TYPE CONFIGURATIONS.....	28
TABLE 11: TRANSMISSION TYPE CONFIGURATIONS	29
TABLE 12: CHANNEL CONFIGURATION FOR A WEIGHT SCALE	30
TABLE 13: TABLE OF USER PROFILE DATA PAGE.....	31
TABLE 14: DATE PAGE 1	31
TABLE 15: DATE PAGE 2	32
TABLE 16: DATE PAGE 3	33
TABLE 17: DATE PAGE 4	33
TABLE 18: SETTING NETWORK KEY TABLE.....	58
TABLE 19: ASSIGN A CHANNEL FUNCTION TABLE	58
TABLE 20: ASSIGN CHANNEL ID FUNCTION TABLE.....	59
TABLE 21: SETTING CHANNEL FREQUENCY FUNCTION TABLE.....	60
TABLE 22: SETTING CHANNEL PERIOD FUNCTION TABLE.....	60
TABLE 23: SEND BROADCAST MESSAGE TABLE	61
TABLE 24: USER PROFILES FOR TESTING THE SYSTEM	66

ACRONYMS

TEE - Total Energy Expenditure

OWL - Ontology Web Language

RMR - Resting Metabolic Rate

BMR - Basal Metabolic Rate

RDF - Resource Description Framework

METs – Metabolic Equivalent Task

SWRL – Semantic Web Rule Language

SQWRL – Semantic Query Web Rule Language

XML – Extended Markup Language

CHAPTER 1

INTRODUCTION

When athletes with enormous talent, highly motivated and with high-level training are presented in competition the difference between victory and defeat is small and the attention to details can make the difference. Nutrition and eating habits are a very important factor in the preparation of an athlete who wants to reach the top of his sport performance. Before, during and after sports event the fluids intake and nutrient intake influences the performance of the athlete.

The Main goal of this thesis is to improve performance in competition and also to improve the diet of normal athletes.

1.1. Motivation

The creation of a recommendation tool for dietary, can be very beneficial for athletes who want to reach their sports peak in time for the biggest competitions (Olympics, championships and international competitions).

Since sports it's an activity practiced globally and by millions every day, creating an engineer platform for sports it's always motivating. Associated with competitive and professional sports there are always involved big industries that moves millions of euros, and just recently some products were release to help high competition athletes with their training and nutrition so it's still a recent market and with a big receptivity for new products to enter that specific market.

1.2. Objectives

The objective of this dissertation, is the development of a nutrition recommendation tool for weightlifter athletes and it will be started by studying the nutritional domain and development of its ontology. The development of the knowledge base, basically, will rely on nutritional concepts, that consists of two kinds of knowledge: ontology and queries. The knowledge rule-based model will generate results for a recommendation menu for the user

depending on his personal informations. The personalized nutritional ontology will be developed using the editor Protégé ontology, this ontology will be developed as use class hierarchies, properties and property restrictions and will be modelled based on two main concepts: personal profile and nutritional menus.

It will also be necessary to study the needed personal athlete information such as weight, their age, gender, level of muscle mass etc..., it's also necessary to study the implementation of report and results engine for the ontology. That engine will process the data in OWL and the knowledge base will be rule based to generate the results. It will be used the OWL API for java for handling the ontology data and for the questioning of the knowledge base will be used SQWRL language.

A software for the insertion of all the athlete needed profile data will be developed in java, it will also generate the recommendations menu for different types of training and different athlete profiles. When all of those are implemented, the system will be tested, testing it for all the recommendations for different profiles and proper operation of sensors and ontology and then writing the dissertation, in which will be explained all the work done and all tests.

1.3. Structure

The first chapter is dedicated to the introduction of the system and what are the motivations behind this dissertation, it's explained all the objectives and the desired final accomplishments.

The second chapter is dedicated to the state of art, in this chapter it's studied the nutrition in professional athletes, with a focus on the background of nutrition in weightlifters. Also it's studied all the previews developments in the area of engineering in sports and the study of how previews works involving ontologies. It was studied the energy and nutrition needs of a processional weightlifter during all his training phases. In this chapter we also give an introduction to what is an ontology and all subsequent used languages in this project like SWRL and SQWRL.

The third and fourth chapter are responsible for all the system design and analysis, here we specify the used hardware for the needed measurements. We analyse how the user interface will work, and how everything will glue together at the end. We will analyse in

detail the sensor communication system and how to get the measurements. So in this chapter all subsystem and their needs will be analyse.

In chapter fifth all the system implementation will be detailed explained, how all query's where construct, how the sensor and interface were implemented and everything works and glues together.

In the sixth chapter we demonstrate and analyse the system results using user profiles for making comparisons between different users, and test our application in comparison with the sensor original one.

In the seventh chapter we have the work conclusion and the future work that it's possible to be implemented in this system.

The last chapter is the seventh and presents all the bibliography used in the development of this dissertation.

CHAPTER 2

PROJECT BACKGROUND

2.1. Relevant works on Ontology in sports and nutrition

There are some relevant research works about Ontologies in Sports and Nutrition, such as those described in [6-9]. The research work by Fodholi et al [6] designed and developed a daily menu assistance in the context of a health control system of a population. This project uses ontologies to model a nutrition needs domain, implementing a rule-based inference engine. The system is implemented as a semantic web application, where users enter abstinence foods and personal information so the system can calculate several parameters and provide an appropriate menu from database.

Cantais et al. [9] designed a food ontology for diabetes control from a nutrition viewpoint to support health care of diabetes patients. The ontology was developed based on some referenced nutrition guides for diabetes patients. The food ontology consists of 177 classes 53 properties and 632 instances. Thirteen major classes of food types were defined including unprocessed aliments, major miscellaneous categories and food types determined by the main ingredient. Some of the defined properties include nutrition elements such as fat, fibre, carbohydrate, etc.

Also for diabetes control, Hong and Kim [8] implemented web-based expert system for nutrition counselling and management, also based on ontologies. This system uses food, dish and menu database which are fundamental data to assess the nutrient analysis. Clients can search food composition and conditional food based on nutrient name and amount. The system is able to organize food according to Korean menus, and it is able to read nutrient composition of the each food, dish and menu.

The Food-Oriented Ontology-Driven System (FOODS) [7] is another ontology based expert system of a counselling system for food or menu planning. It uses a food-oriented ontology to implement a system which has two user interfaces, one for who cooks and another for costumers or users that want advice on meals.

Suksom et al. [10] implemented a rule-based system for a personalized food recommender system, aimed to assist users in daily diet selections based on some nutrition guidelines ontology and focused on personalization of recommendation results by adding user's health status information that may affect his nutrition need.

This work adopted some food ontology design schemes from [6-10]. However, comparing to [8-12], our work was focus only in weightlifting athletes. It was extended the personalized food ontology defined in [8, 10] by adding information related to athletes' training program that may affect their nutrition need and unifying the food and sport ontologies. Our recommendations are based on sport nutrition guidelines, which were transformed into rule-based knowledge.

2.2. Periodized Nutrition for the yearly training programme

In the course of a year, athletes training is divided in different cycles, manipulating variables such as volume, frequency and intensity in order to meet season demands and schedule. As these variables change, also the food of athletes should also change to meet different nutritional demands. Usually, a one year training cycle, a macro-cycle, is divided in micro-cycles, ranging from one day to a week, and meso-cycles, ranging from a week up to three months. Also, four phases can be identified in training: general preparation, specific preparation, competition, and transition. A training plan comprises phases and cycles to manipulate training variables, so food needs should be in synch with this plan in order to maximize the athlete results in each phase (Fig. 1)

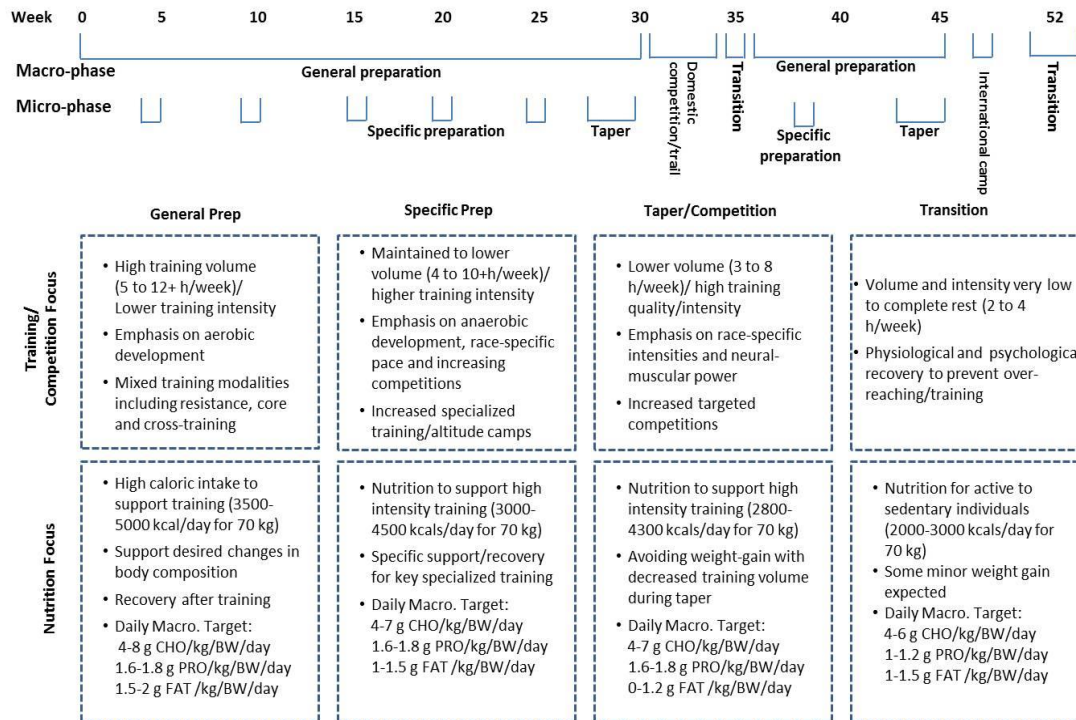


Figure 1: General nutrition recommendations during different yearly training phases for strength and power athletes. (Adapted from [2, 11])

2.2.1. Energy Requirements

Meeting energy needs is a nutrition priority, in order to maximize athletic performance. To this goal, the energy provided to the athlete should equal the energy he/she needs, being this estimation a critical factor for nutrition. Besides the training plan, energy needs of an athlete depends on his/her prior nutritional status and sex, being factors such as heredity, age, body size, influential in such calculation

One of the most common ways to estimate total energy expenditure (TEE) is first estimate resting metabolic rate (RMR) using a prediction equation and then multiply RMR by an appropriate activity factor (both general and specific activity) (1). This is called the factorial method.

$$TEE = (RMR \times \text{General Activity}) + (RMR_{\text{(per hour)}} \times \text{METs}) \quad (1)$$

In general it is best to use the RMR prediction equation most representative of the population with whom you are working. For both active men and women, The Cunningham equation best predicted RMR; the Harris-Benedict equation was the next best predictor [12]

$$RMR = 500 + 22(LMB) * LBM = \text{Lean Body Mass}$$

$$\text{Males: } RMR = 66.47 + 13.75(wt) + 5(ht) - 7.76(age)$$

$$\text{Female: } RMR = 655.1 + 9.56(wt) + 1.85(ht) - 4.68(age)$$

<i>Level of activity</i>	<i>Activity factor (xRMR)</i>	<i>Level ff activity</i>	<i>Activity factor (xRMR)</i>
<i>Sedentary</i>	<i>1.2</i>	<i>Moderate activity</i>	<i>Men 1.7 Women 1.6</i>
<i>Very Light activity</i>	<i>Men 1.3 Women 1.3</i>	<i>Heavy activity</i>	<i>Men 2.1 Women 1.9</i>
<i>Light activity</i>	<i>Men 1.6 Women 1.5</i>	<i>Exceptional activity</i>	<i>Men 2.4 Women 2.2</i>

Figure 2: The Cunningham and the Harris-Benedict equation for estimating energy requirement for adults (Adapted from [12])

Type of activity	METs
Weight Lifting, power lifting, or body building, vigorous	6.0
Weight Lifting, power lifting, or body building, vigorous light or moderate effort	3.0

Figure 3: The metabolic equivalent for weightlifting. [13]

2.2.2. Daily Menu

A menu defines the food composition that is consumed by an athlete for his/her meal in one meal time or in one day. A balanced menu is a menu which consists on varied foods in appropriate quantities and proportions, to fulfill the nutritional needs of the athlete [13].

To calculate the total calories contained in every menu, in this project was used the data from the dataset “Nutritive values Thai food” provided by Nutrition Division, Department of Health, Ministry of Publish Health (Thailand) and a software named INMUCAL provided by the Institute of Nutrition, at Mahidol University.

2.3. Nutrition Background for Weightlifting

Weightlifting demands extreme strength and power to lift very heavy weights in a controlled manner. The aim of these athletes is to build muscle bulk and target the main muscles that are used for the bar movement. A high level of muscularity is therefore required by both male and female competitors. Maintaining low body fat is also a physical requirement often demanded to optimize the power to weight ratio of lifters, helping to achieve best performance. Despite requiring a high muscle mass, control of athletes weight is very important given that competitions are divided into weight-class categories [1].

Besides providing the energy for training and for its recovery, in the case of weightlifting and other strength-power sports, nutrition also promotes training adaptations, including skeletal muscle hypertrophy [2]. A summary of the reported dietary intake of adult strength-power athletes in training, regarding to macronutrient consumption, reported that weightlifters consume a greater number of daily servings of protein-rich sources when compared with other athletes. As a result, the protein intake of male weightlifters has been reported to range between 1.6 g/kg/day and 3.2 g/kg/day [3-5], which is high when compared with the recommended 1.2–1.7 g/kg/day for resistance training athletes. Furthermore, weightlifters derive approximately 40–44% of their daily energy intake from dietary fat [1-5], which is also well above the acceptable range for health and athletic performance of 20–35%. This is probably a consequence of their greater intake of protein-rich animal products.

Conversely, the reported carbohydrate intakes in weightlifters of 2.9–6.1 g/kg/day [1, 3-5] are insufficient according to the current recommended levels of 7–8 g/kg/day for athletic individuals [6]. Combined, these reports suggest that the dietary habits of male weightlifters may not yield the desired training gains and/or health benefits due to the emphasis placed on protein consumption (with high fat) at the expense of adequate carbohydrate ingestion [1].

Ontology-based personalized dietary recommendation for weightlifting is one way to help athletes meet their requirements, in comparison with a database schema for example the ontology have some big advantages since the ontology is a way of sharing meaning shared understanding when the database focus is on data only with minimal focus on formal semantics when in a ontology the focus on formal semantics is very strong, another advantage is that an ontology can always be reuse when a DB schema is rarely reused since its locked to the same

set of queries when the ontology queries are usable on other systems. So the trade-off of using an ontology vs. a DB schema is that you gain function and flexibility with the ontology but you lose performance that is better in the DB schema. Since we need flexibility in this project and we want this data to be reused in future works the ontology is a better option in this case.

2.4. Ontology Meaning

From the philosophic point of view an ontology is the study of the being, becoming, existence or reality and their relations, the theory of objects and their ties. So an ontology deals with all the questions about entities that exist or can be said to exist and how are they grouped and related in a hierarchy and divided according to similarities.

The word Ontology applies into multiple fields being computers science one of them, the word ontology comes from Greek word (Ov) and means entity, in all the fields the basics about ontology is describing the world and representing all their entities. Historically the Greek Philosopher Parmenides as said to be the first one to propose the meaning of ontology, it comes from the branch of metaphysics which deal with nature of reality, during the 20th century some research was made about ontologies, that led to the recognize that ontologies can be very important for the developments in the fields of AI (artificial intelligence), since capturing knowledge is very important for the development of intelligent systems.

Ontology in the computer science field is a data model that describes concepts (classes) in a specific domain and also describes their relationships. Ontology was successfully used to share concepts across applications and exchange information; ontology exchange information based on semantics rather than using syntax.

During the 90s it was released a paper by Tom Gruber "*Toward Principles for the Design of Ontologies Used for Knowledge Sharing*" about ontologies, where it was introduced conceptualization and the ontology was deliberate a term in this field. Tom Gruber said in that paper that an ontology, was a description of all the concepts and the relationships between them, and that they can formally exist for an agent or a community of agents. So ontologies are taxonomic hierarchies of classes and classes definitions and their relationships.

Another way to see what is an ontology is comparing with the object oriented languages, the big different between the object-oriented languages and ontology type languages, is that

programming with object-oriented language we centre around methods on classes, and we make design decisions based on the operational proprieties of classes, while in ontology we make the decisions based on the structural proprieties of a class.

Depending on the expressivity of the ontology, it can be classified in different ways. It can be classified by their proprieties or description logical based (Food has propriety hasIngredient, hasCalories etc...) by text definitions (Food is defined by a phrase “a type of food”) or by logical definitions. For this project it will be used a description logical based ontology.

- relation between classes (food type class and ingredients class)
- relations between individuals (type of menu has a specific ingredient)
- relations between classes and individuals

Since it will be necessary to get back the some specific data from the knowledge base, we will need a infer engine, further in this paper the engine will be explained in more detail.

2.4.1. *Main elements on a ontology*

There are five main elements in an ontology, concepts or classes, individuals, properties and relationships, that four elements together make the ontology a knowledge base.

- Classes are collections of objects, sets or abstract groups, describing concepts in the ontology specific domain; they can contain both a subclass that describes more specific concepts. An example of a class would be a Food class that would contain various subclasses like food type and food group.
- Individuals are the basic components of ontology. The individuals may be concrete concepts like a specific menu or an ingredient or an abstract one like numbers as calories in a menu.
- Properties are related to individuals or class, as they are something that define or explain them. There are two types of properties: *datatype* used to assign a valor to a property or class, (e.g., a menu hasEnergie 150j) or *object* type one object can be attributed to other (e.g., menu A hasIngredient b).

- The last of the main elements of ontology are the relationships that consist in all relations between classes and individuals.

2.4.2. *Web Semantics*

According to widely known proposals for a Semantic Web architecture, ontologies will play a key role as they will be used as a source of shared and precisely defined terms that can be used in such metadata.

The Web Semantics is an extension of the web that will allow humans and computers to work together, by connecting the meaning of works and attribute a meaning to all internet contents so both humans and computers can understand it. It has the objective of creating languages that make the information readable by computers and machines and allow the global sharing the knowledge assisted by computers. The semantic web won't be seen but will work in the background allowing a better experience for users.

The architecture of Web Semantics are as illustrated in the figure 4. In the figure we can see its divided in 7 layers.

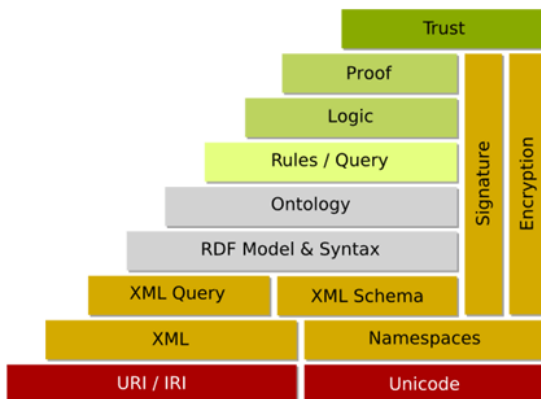


Figure 4: The architecture of Web Semantics [21]

The first layer is UNICODE and URI. Unicode is a computer standard that allows computers to manipulate all types of texts from all writing systems. The URI (Uniform Recourse Identifier) is string of characters that identify a web resources and enables interactions with those resources, typically www (world wide web) .

In the second layer is XML (Extensible Markup Language), is the common syntax used in semantic web, it's a general purpose mark-up language containing structured info,

The third layer consists of RDF (Resource Description Framework), it is used to represent information in the internet. Consists on a model of data know as metadata, that has the objective of creating a simple model to describe data. It's a technology recommended by W3C since 1999, and its written in XML. RDF decomposes any knowledge into triple pieces with some semantic rules in each of those pieces. Those triples are are Subject, Predicable and Object as exemplified in figure 5. The subjects, predicables and objects are names for simple name for things, like this subject is john smith and the predicate is that he plays the object cricket , deslikes the object insects and that cricket (and object here) is a sport. Predicates are always relations between a subject and a objects , and those 3 triples form a graph of data and the RDF serves as a description of those graph.

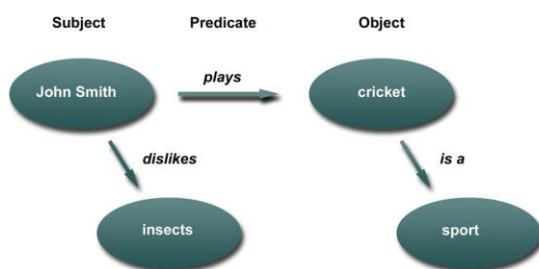


Figure 5: RDF Triple

The forth layer is the ontology vocabulary the one this project is focused on. And ontology vocabulary or ontology language is a language used to construct ontologies, so allowing the encoding of knowledge into specific domain's allowing reasoning. There are various types of ontology languages, in this case it will be used the W3C recommended OWL (ontology web language). The fifth layer is the query and rules layer that will be explained in the SWRL and SQWRL section in this paper. The rest of the layers are not used in this project.

2.4.3. *Ontology Web Language*

The importance of ontologies in semantic mark-up has prompted the development of several ontology languages specifically designed for this purpose, leading up to OWL[20].

The OWL is a language used to define and instantiate web ontologies, it's based on OIL and DAML+OIL languages and it's now a W3C recommendation.

An OWL ontology includes class descriptions and their properties and relationships. The language was created for use in applications that need to process the information like in this case to process all the food\nutrition data. The language allows the interpretation of web information in an easier way than XML and RDF since it has additional semantics. It's a very important language for web semantics so being used more and more in a huge number of applications. OWL was based on XML so the information can be exchanged between different computers, different operating system and different programming languages, and its easily read by humans. There is three type of sub-languages for OWL: OWL Lite, OWL DL and OWL Full.

- OWL Lite: Supports users that want a hierarchical classification's and simple restrictions. Only allows cardinality restrictions of 0 and 1, so has less formal complexity then OWL DL.
- OWL DL: For users that want maximum expressivity and still makes it conclusive and computable. It includes all constructions of the language, but just some can be used with restrictions. It's called DL because of the description language it uses.
- OWL Full: Has maximum expressivity and syntactic freedom like in RDF without any computable guaranty so it's impossible for any inference software to fully use it.

All of this sub languages are an extension of his predecessor, so every valid OWL Lite it's a valid OWL DL but an OWL DL can be a not valid OWL Lite. The choice of the sub language depends on the project needs, in this case we will use DL since we want maximum expressivity but we need it to be computable or we won't be able to use inference on it.

The web ontology language was created to be able to describe classes, and relations between them so it can be interpreted by software.

Since the goal of this project is to be able to make menu recommendations using the food and nutrition knowledge base we will need a recommender engine (inference machine), the recommender engine will interpret the ontology data in OWL (Web Ontology Language) format

that is a standard ontology language designed for processing web information. Since OWL is written in XML and so, it can be exchanged between different computers and different applications. For inferring the knowledge base and getting all information the system will need it will be used SWRL.

2.4.4. *SWRL (Semantic Web Rule Language)*

The SWRL is a language for web semantics that express rules, combining the OWL language with a rule language. The rules are express from the OWL concepts (classes, properties, individuals and literals). It's built on the same foundation as OWL and has strong formal guarantees for inferences.

Example: `Food(pizza) -> hasIngredient(pizza,Cheese)`

All pizzas have the ingredient cheese.

2.4.5. *SQWRL (Semantic Query Web Rule Language)*

There's a big number of languages made for querying RDF and OWL. The standard for querying RDF is SPARQL, which can also be used to query OWL since OWL is serialized as RDF. But since SPARQL has no understanding of OWL, it operates only in RDF serialization and doesn't use the language constructs, so for each ontology developed in a different tool the query would has to look different, so SPARQL it's not a natural fit for querying OWL.

For querying the OWL in this project we will use the SQWRL (Semantic Query-enhanced Rule Language), that is a standard for SWRL language and treats the SWRL rule as a query, it replaces the rule event for a return value. No external extensions are needed to use SWRL, so all SWRL can be used with SQWRL. The SQWRL is a simple but expressive language that provides a big array of operators to easily query whatever we need from the knowledge base.

The most important operator in SQWRL is the `sqwrl:select`, and builds a table with the selected arguments. In the next example we get all main dish, and correspondent ingredients.

`Main_dish(d?) ^ hasIngredient(?d,?f) -> sqwrl:select(?d,?f)`

Another important operator is the `swrl:orderBy`, that orders in ascendant way the selected argument (`orderbydescended` for descendent way). In the next example we get the same result as before but order by ingredient.

Main_dish(d?) ^ hasIngredient(?d,?f) -> sqwrl:select(?f)

There is some other operators that are going to be used in this system and are all explained in the development chapter.

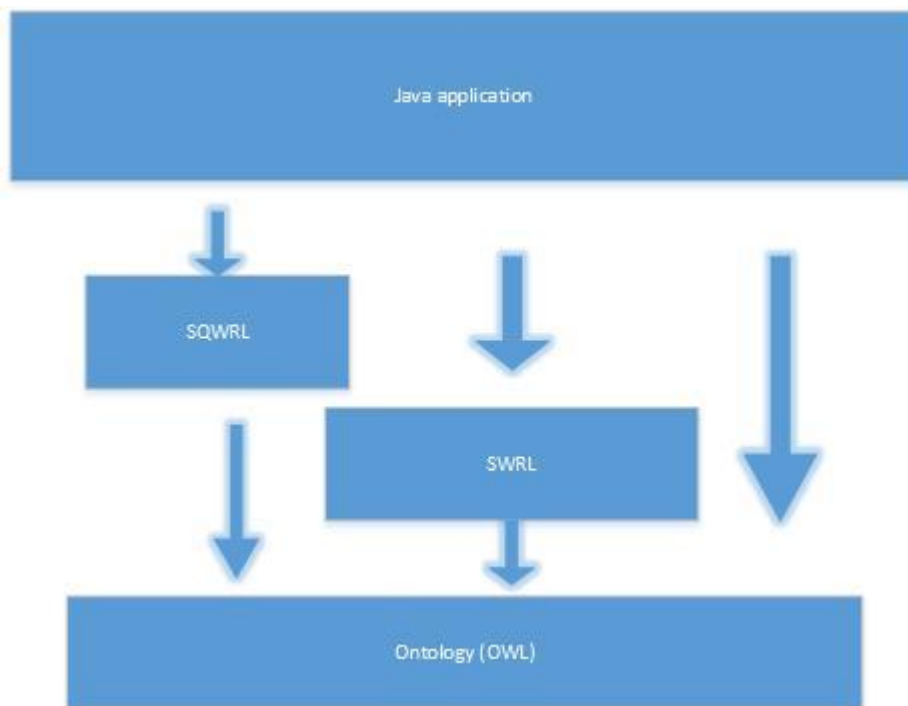


Figure 6: Java application and OWL interaction

2.4.6. *Protégé*

For the development of the unified sports and nutrition ontology the software protégé will be used, a free and open-source platform that allows users to build an ontology in OWL and most used software for this kind of projects with a lot support.

Protégé is based on Java, is extensible, and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development.

Protégé is supported by a community of developers and academic, government and corporate users, who are using Protégé for knowledge solutions in areas as diverse as biomedicine, intelligence gathering, and corporate modelling and of course engineering.

And for the SQWRL, we will use the SWRLTab plugin from protégé OWL that gives a graphical interface to test the queries as also a Java API to execute the queries in the java application.

CHAPTER 3

SYSTEM ANALYSIS

3.1. System Overview

The objective of this system is for any weightlifting athlete, to depending on his personal information (age, gender, weight etc...), personal taste (favourite food, favourite beverages) and training needs to get the best possible food menu with all necessary nutrients.

As it's showed in figure 7 , the user (the athlete or his coach) will insert all his personal data and send it to the recommendation engine, and if there's a sensor available the recommendation engine will collect the sensor data. Both the insertion data application and the recommendation engine will be developed in java, but since java doesn't have direct access to external hardware, the application that will access the sensor will be developed in c++ and then JNI will be used to communicate between the Java and c++ . Then the engine will call the inference agent (Protégé-OWL API) that will query the ontology for all need data using SWRL.

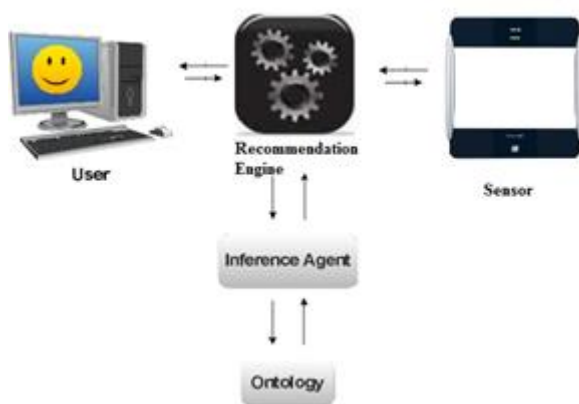


Figure 7: Overview of the system

3.2. System Requirements

The system requirements are:

- The user must be able to enter his personal info
- The user must be able to choose his food preferences
- The user must be able to choose using a sensor or not

- The user must be able to choose favourite and non-favourite items for his menus
- The user must be able to choose favourite and non-favourite process types
- The user must be able choose a beverage
- The system must give back the ideal items\food for the user specific profile
- The system must be able to get the sensor measurements
- The system must be able to calculate TEE with or without a sensor
- The system must allow the user to choose is items for the daily menus
- The system must report the chosen menus and all personal information from the user

3.3. Menus calculation

Since we want to calculate in the most accurate way possible the menu for every day of weightlifter some personal information will be needed. For the calculation of which menu is better for each time of the day we will need the total energy expenditure (TEE) for 1 day as said before in this paper.

$$\text{TEE} = (\text{RMR} \times \text{General Activity}) + (\text{RMR}_{(\text{per hour})} \times \text{METs})$$

Equation 3.1

3.3.1. Activity Factor

As we can see in equation 3.1 one of the values we need must be given by the athlete, the general activity. As said before in this text there's 6 different types of level of activity person:

Activity Factor		
Level of Activity	Male	Female
Sedentary	1.2	1.2
Very Light Activity	1.3	1.3
Light Activity	1.6	1.5
Moderate Activity	1.7	1.6
Heavy Activity	2.1	1.9

Exceptional Activity	2.4	2.2
-----------------------------	-----	-----

Table 1: Level of activity

A person is considered sedentary, when his movement during a day is minimum, passes most of his life mainly sitting or lying, as his activities are mostly watching television and reading etc. A person is considered to have Very Light Activity when he has an office work that he is sitting for 8 or more hours and his physical activity are just walking to office at very moderate speed. A person with light activity has the same type of life as a person with a very light activity but walks more and practices some activities like gold, do laundry, gold and ping pong.

A person is considerate to have a Moderate Activity when his job has some light labour and/or weekly does some activities like walking are fast speed, carrying a load, tennis, cycling dancing etc.. To have a heavy activity a person must be full time athlete, or very hard labour jobs like agriculture, military duty's, and mines and steel works etc..., and practices some heavy activities like walking with a load up hill, team sports (Football, Basketball, Handball) and climbing.

A person is considerate extremely active when is job is extremely hard work, like construction jobs, lumberjacks coal miners, and some jobs in very dangerous and physically demanding places, and/or is full time athlete with very big strength training.

3.3.2. *Metabolic Equivalent of Task*

Other necessary value for the calculation of TEE is the METs (Metabolic Equivalent Task) is the measure to express the energy cost of physical activities and its defined by what stage of training the athlete is.

The different type of training stages and the subsequent METs are:

Training Stage	Met's Value
General Preparation	x 5
Specific Preparation	x 4
Competition Preparation	x 3
Transition Preparation	x 1

Table 2: Table of Training Stage and the Met's values

The other value we need for TEE calculation is the RMR and there is two was of calculating it, one using a measuring device that can give us the RMR directly and the other one using a specific equation that will give us the approximate value.

3.3.3. *Calculation of RMR using specific equation*

If the user chooses not to use an external sensor that give us the RMR we will need the user to insert his weight, his height, his gender and this age.

$$RMR(Male) = 66.47 + 13.75 (weight) + 5 (height) - 6.79(age)$$

3.2- Equation for RMR calculation in a male user

$$RMR(Female) = 655.1 + 9.56 (weight) + 1.85 (height) - 4.68(age)$$

3.3- Equation for RMR calculation in a female user

3.3.4. *Other needed input variables*

There's another variables we will need so the menus accuracy will be better, one of them is if the user wants to lose, maintain or gain weight. Has we can see in table 3.x. if he athlete wants to maintain is weight the calorie intake must be the same as the TEE, but if he wants to lose weight the calorie intake must be less than the TEE and if wants to gain the calorie intake must be bigger than the TEE.

Maintain Weight	Calorie Intake = TEE
Increase Weight	Calorie Intake > TEE
Decrease Weight	Calorie Intake < TEE

Table 3: Weigh Goal

For weight increase and decrease the TEE should be moderately decrease and increase by 500kcal/d as showed in table 4.4.

Gain Weight	Maintain Weight	Lose Weight
TEE = Tee + 500	TEE = TEE	TEE = TEE - 500

Table 4: Table of TEE for gain, maintain and lose weight

The other parameter the user needs to add it's his name, it won't affect the menus but will be necessary for the report saving.

3.3.5. *Percentage of TEE per meal*

Since we will get the TEE for one day and an athlete must have 7 meals per day we will have to divide the TEE per each meal. Since dinner and lunch are the biggest meals they will have the biggest amount of energy needed, and breakfast have a big amount since its one if not the most important meal of the day. During the exercise an athletes should ingest 240kcal and with 60g of Cho normally in liquids (energy drinks most likely). Before and during training the athlete should ingest small snacks of about 10% before the training and 12% after the training.

Since it's almost impossible to get perfect values of TEE for each menu (for example a person need a breakfast of 402.34kcal, its possible there is none on the list with that energy) we have to give a margin of energy per meal. That margin will be $TEE_min = (50 - (TEE * \%Meal))$ and $TEE_max = (50 + (TEE * \%Meal))$ in the future if more menus are added this value can be diminish so the TEE for each menu can be even more accurate.

Meal	Total Energy	Carbohydrates	Protein
Breakfast	16% of total energy	Not defined	Not defined
Before Training	10% of total energy	50g	5-10g
During Exercise	Always 240kcal	60g	Not defined
After Training	12% of total energy	1g * weight	0.5g * Weight

Lunch	20% of total energy	Not defined	Not defined
Before Training	10% of total energy	50g	5-10g
During Exercise	Always 240kcal	60g	Not defined
After Training	12% of total energy	1g * weight	0.5g * Weight
Dinner	20% of total energy	Not defined	Not defined

Table 5: Meals total Energy

3.3.6. *Necessary Input values*

As we can see in the table 7 some of the input values won't be needed for RMR\TEE equations, but since all sensors that can measure LBM need some athlete information for the LBM calculation the user will always have to add all this variables.

Input	If no sensor available	Sensor Available
Name	Not necessary	Not necessary
Age	Necessary	Not necessary
Weight	Necessary	Not necessary
Height	Necessary	Not necessary
Training Stage	Necessary	Necessary
General Activity	Necessary	Necessary
Keeping Weight	Necessary	Necessary
Number of hours of training	Necessary	Necessary

Table 6: Input values needed

3.4. Selection of Sensor and communication Protocol

Since we needed some measurements that aren't always the same (age, genre, height and some other are constant.) we will need some measurement device that is able to get them. The measurements we need for this project that are not constant and can be measure with a

sensor are: Weight and LBM (Lean Body Mass, Body Fat)). We need also a device that could connect to the application directly and if possible wireless.

After some research, the best option was a scale that measures both the needed parameters, the scale is Tanita BC1000. It connects to the computer via ANT+ a wireless protocol.



Figure 8: Tanita bc 1000

3.5. ANT+ Protocol

The ANT+ is a wireless sensor protocol with low power consumptions, ideal for sport projects as this one. Since its power consumption is so low, most of the big sports hardware companies are starting to use it, as Garmin etc..., it runs in the 2.4GHz ISM Band, it's efficient and easily handles peer to peer, provides accurate data communication and adaptive network operations, it's highly optimized and it's easy to use with low cost systems. ANT+ is interoperable so all ANT+ products from different brands work together and are compatible so in the future it's possible to expand this system for more external sensors using the ANT+ protocol.

This project will communicate with Tanita scale via ANT+ pen drive, so the java application will be the master and the scale will be the slave, that will always be waiting for the application to send a wakeup call (since its ultra-low power the scale ant+ will always be on waiting for profile sent).

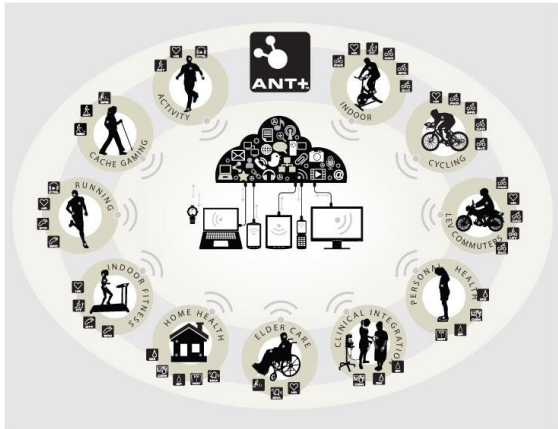


Figure 9: Ant+ in Sports domain

3.6. Communication with the Scale

Since the java cannot access the sensor directly we will create a c++ application that will communication and get the values from the tanita. The PC application will be the slave, and the Weight Scale will be the master waiting the channel to be established by the application and a profile sent to start broadcasting the data back to the application.

3.6.1. Channel Communication

Communication between the master and the slave depends on what channel is used, how is the channel configured, which direction is the data sent and what type of data. In this application the ANT+ implementation use bidirectional channels (we send information to the scale and get information back), is independent and synchronous. So when the master (the scale) will always have search window, and will always transmit a message in a predefined timeslot, as we can see in the figure 3.x.

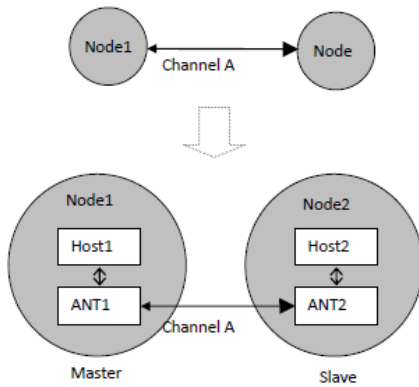


Figure 10: Channel communication

3.6.2. *Establishing a channel*

For a channel to be established, the slave and the master must have a common knowledge on channel configuration, some parameters have no default value (network key, frequency, channel period, Tx power and search timeouts) when the rest just need to be modified if a different value is desired. By default the network key is assigned to 0 since it's the public network key, and in this case we need the network to be public and not private.

After setting the network, we need to assign the channel type, being it slave or master and since this application will be the slave we must assign the channel type as slave.

Next we must set the Channel ID, for that we need to specify the device number, the device type and the transmission type.

Next after setting the channel id we must set the frequency, the channel period, the tx power and the search timeouts.

The next and final step is opening the channel, and when open and the channel established the master will start transmitting an 8 byte data packets in the specified time slot, when the slave will start searching for the master that matches the channel id that was set, and when located the connection will be established. If the master is not found the slave channel will close.

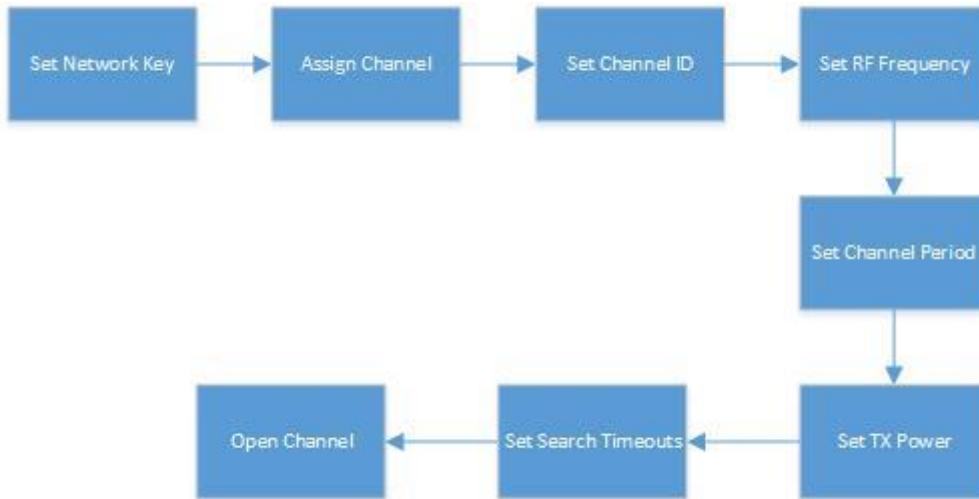


Figure 11: Establish a channel between master and slave

3.6.3. Channel Configuration

For the configuration of the channel for the weight scale as said before and showed in the table 3.x we need to set various parameters the first being the channel type, network key, channel id, rf frequency, channel period, tx power and search timeouts.

Network Number – An 8 byte number that identifies the network and provides security and control of the network. It's given in the www.thisisant.com site and its licensed by them.

Channel Type – Specifies the type of communication that will occur in the channel. As we see in the table 3.x, it's a 8 bit fields with range from 0 to 255, and in this system case since it's a the application is a slave channel and bidirectional since the data will flow in the direction of the scale when we send the profile data and in the direction of the application. So we must set the channel type with the value 0x00.

Value	Description
0x00	Bidirectional Slave Channel
0x01	Bidirectional Master Channel
0x02	Shared Bidirectional Slave Channel
0x04	Shared Bidirectional Master Channel
0x05	Slave Receive Only Channel
0x06	Master Transmit Only

Table 7: Channel type configurations

RF Frequency – The ANT+ network allows the use of 125 unique RF operating frequencies, the rf frequency is a 8 bit field with range from 0 to 124 and each specific device operates at different frequencies bet that 0 to 124 range.

Tanita operates at 57 channel so we must use 0x39 for rf configuration.

Channel ID – The channel id is the most basic descriptor of a channel and used for device paring. The channel id is a 4 byte value with 3 fields:

- **Transmission Type** – It’s an 8 bit field that defines the type of transmission of the device. Since our transmission is reserved we must set transmission type to 0 for pairing search.

Bits	Description
0-1	00:Reserved 01:Independent Channel 10: Shared Channel using 1 by address 11: Shared Channel using 2 byte address
2	0: Global page not used 1: Global page used
3	Undefined
4-7	Optional

Table 8: Transmission type configurations

- Device Type – It's an 8 bit field that differentiate between different devices. The device type in a weight scale in ANT+ is the 119 (0x77).
- Device number – Unique 16 bit field for each device type. We will set the device number to 0 to allow wildcard matching.

Channel Period - The channel period is the rate of data packets send by the master, in the scale the data is transmitted every 8192/32768 so the channel period will be set to 8192 counts.

3.6.4. Channel Configuration for a weight scale

We can see in the next table the full configuration that will be used to communicate and established a channel with the tanita device.

Parameter	Set Value
Channel Type	0x00
Network Key	Check www.thisisant.com for this
RF Frequency	57
Device Type	119
Device Number	0
Channel Period	8192

Transmission Type	0
--------------------------	---

Table 9: Channel configuration for a weight scale

3.6.5. *ANT+ Messages Payload Format*

The ANT+ messages have a payload of 8 bytes, being the first the data page number and the rest specific data for the sensor. The data page can be main data and common data, the main data it's the measurements and calculations done by the weight scale, when the common pages send information about the device itself, like manufacture identification and battery voltage.

3.6.6. *Data Page 58 – User Profile*

To wake up the scale and start the measuring we must send it a user profile, we can see in the table 3.x the data page 58.

Byte	Description	Length	Value
0	Data page number	1 Byte	0x3a (58)
1	User Profile LSB	2 Bytes	0xFF
2	User Profile MSB		0xFF
3	Capabilities Field	1 Byte	0xFF
5	Gender	1 Bit	0 = Female , 1 = Male
	Age	7 Bits	Age = 0 - 127
6	Height	1 Byte	Height in cm – 0 to 255 cm

7	Descriptive Bit field	1 Byte	Activity Factor (0-7)
----------	-----------------------	--------	-----------------------

Table 10: Table of user profile data page

3.6.7. *Date Page 1 - Body Weight*

After the user profile is sent the scale will turn on and person must step on her for measurement's when it will blink red it means the measurement is complete and the scale will send us 4 pages, the first one the Body Weight page.

Byte	Description	Length	Value
0	Data page number	1 Byte	0x3a (58)
1	User Profile LSB	2 Bytes	0xFF
2	User Profile MSB		0xFF
3	Capabilities Field	1 Byte	0xFF
5	Reserved	1 Byte	0xFF
	Reserved	1 Byte	0xFF
6	Body Weight LSB	2 Bytes	MSB, LSB / 100
6	Body Weight MSB		

Table 11: Date page 1

3.6.8. Page 2 – Body Composition

In this page we can get the body fat percentage and the hydration percentage, for this page a valid user profile must be sent since the scale will use those parameters for this calculations

Byte	Description	Length	Value
0	Data page number	1 Byte	0x3a (58)
1	User Profile LSB	2 Bytes	0xFF
2	User Profile MSB		0xFF
3	Reserved	1 Byte	0xFF
4	%Hydration LSB	2 Bytes	%Hydration
5	%Hydration MSB		
6	% Body fat LSB	2 Bytes	% Body fat
7	% Body fat MSB		

Table 12: Date page 2

3.6.9. Date Page 3 – Metabolic Information

In data page 3 we get the metabolic rate and recommended daily caloric intake of the user. Also needs user profile information for calculations.

Byte	Description	Length	Value
0	Data page number	1 Byte	0x3a (58)
1	User Profile LSB	2 Bytes	0xFF
2	User Profile MSB		0xFF
3	Reserved	1 Byte	0xFF
4	Active Metabolic Rate LSB	2 Bytes	Active Metabolic Rate

5	Active Metabolic Rate MSB		
6	Basal Metabolic Rate LSB	2 Bytes	Basal Metabolic Rate
7	Basal Metabolic Rate MSB		

Table 13: Date page 3

3.6.10. Date Page 3 – Body Mass Composition

In data page 4 we get the Bone and Muscle composition. Also needs user profile information for calculations.

Byte	Description	Length	Value
0	Data page number	1 Byte	0x3a (58)
1	User Profile LSB	2 Bytes	0xFF
2	User Profile MSB		0xFF
3	Reserved	1 Byte	0xFF
4	Reserved	1 Byte	0xFF
5	Muscle Mass LSB	2 Bytes	Muscle Mass
6	Muscle Mass MSB		
7	Bone Mass	1 Byte	Bone Mass

Table 14: Date page 4

3.7. User Cases Diagram for the enter menu

As we can see in the User Cases Diagram the user must insert all the information\parameters, so he can move forward, he also must decide if he wants to use the

scale and directly go to picking a meal menu avoiding the picking favourite ingredients and other parameters page.



Figure 12: Enter Personal information interface page Use Cases

CHAPTER 4

PROJECT DESIGN

4.1. Ontology Design

Here all concepts individual and proprieties of the ontology will be design.

4.1.1. *Modelling of the Ontology*

The development of the ontology started by defining the four main elements of ontology the classes or concepts, the individuals, the proprieties and all the relationships.

Since it was decided to start with only weightlifting, it was used a top-down approach starting with the definition of the most general concepts in the domain and then subsequent the specialization of those concepts.

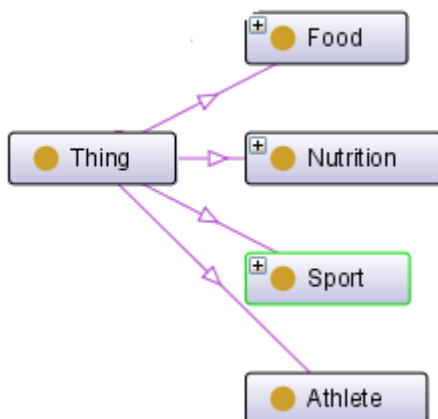


Figure 13: Ontology main concepts

The Athlete concept: The athlete class represents the concept of the athlete profile with the athlete information all the necessary information about the needed personal data like height, weight, age, etc...

The Food concept: The food class is the root of this problem and represents the concept of the food which will have multiple subclasses.

The Nutrition concept: The nutrition concept represents all the nutrition needs of an athlete and all the nutrients present in an ingredient/food item.

- **Nutrient Type** – all nutrients presented in all ingredients and food items are here listed.
- **Nutrition Level** – the level of nutrients per ingredient and food item and represented here, like low carbohydrates and high level of fat.
- **Nutrition Goal** – the amount of nutrients that meet the requirement.
- **Nutrition Plan** – special plan that athletes need, like maintain weight/ increase muscle or decrease weight/maintain muscle.

The Sports concept: The sport concept represents athletes characteristic which affect their nutrition need.

Anthropometric characteristic – body type of athletes consist of endomorph, mesomorph, ectomorph which corresponding to body fat percentage of athletes.

Periodization of training – the systematic planning of athletes training consists of preparation phase, specific phase, competition phase, transition phase. In each phase, the athlete's energy need is different.

This concept won't be fully used in this project since our focus is in nutrition, but will be added for helping the future work.

4.1.2. *Ontology classes and subclasses*

The main concept food will have 4 subclasses:

- **Food Group** - this subclass will be divided in 5 groups of food like the Food Pyramid, all the 5 groups will have subclasses for even more specific type of food (e.g., in meat group there are different types of meat like beef or pork). This is the biggest concept where all the ingredients of all the menus will be. Inside this class it will be the subclasses Group 1, Group 2, Group 3, Group 4 and Group 5.

In the Group 1 will be the Aquatic, Egg, Meat, Milk and Nutshells ingredients, this is the biggest group and with more members. This classes will be divided into even smaller sub classes, Aquatic will be divided into Fish and shellfish, the Egg and Meat

classes into Poultry and Non Poultry, the Milk Products into butter, cheese, drinking milk, condensed milk and yogurts, and Nutshell into product and non-product. All this sub-classes will have even more sub-classes.

The Group 2 class will contain the cereals and Starchy subclass with product and non-product cereals and starchy.

The group 3 will be the vegetables class where it will be divided into energy vegetables and non-energy vegetables.

In group 4 there will be the fruits product and no product and the 5th and last group will be the fat group with all the oils and toppings.

Dividing all this into several groups will make the knowledge base more accurate and easier to navigate and query.

- **Food Menu** – this subclass represents the different type of food items like main dish, dessert snacks or beverages. Inside these subclasses will be all available menus\items divided in those 3 categories. In the main dish subclass all will be all the menus that are best suited for dinner\lunch or any big meal (for example rice or pizza). The dessert snacks will present all menus for breakfast and before and after training (for example a croissant or cookies). The last one area where all beverages will be (like tea, coca cola)
- **Process Type** – represents how that food item was cooked, there will be different types of processing.
- **Type of Meals** – this subclass represents the food item advised time of ingestion, e.g., during dinner, lunch, breakfast or during training sessions). All food items can have more than one type.

4.1.3. *Properties*

The properties or attributes are related to individuals or class, as they are something that define or explain them. There are two types of properties: *datatype* used to assign a value to a property or class, or *object* type where one object can be attributed to other.

Object properties:

- **hasProcessType:** This property attribute to a specific menu a type of food process, so it makes sure every menu has a food process.
 - Domain: Beverages, Dessert_snack, Main_dish
 - Range: Process_Type
 - Example: Pizza hasProcessType baked

- **hasIngredient:** This property attribute a specific ingredient to a menu, making sure all menus has ingredients (one or more).
 - Domain: Beverages, Dessert_snack, Main_dish
 - Range: Food_Group
 - Example: Pizza hasIngredient Cheddar_Chesse

- **hasNutrient:** This property attribute a specific nutrient to a menu or ingredient, making sure all menus and ingredient have specific nutrients.
 - Domain: Beverages, Dessert_snack, Main_dish
 - Range: Type_of_nutrients
 - Example: Cheese hasNutrient Carbohydrates

- **hasNutritionLevel:** This property attribute a specific nutrition level to a menu or ingredient, making sure all menus and ingredient have nutrition levels.
 - Domain: Beverages, Dessert_snack, Main_dish

- Range: Nutrition_Level
 - Example: Tomato hasNutritionLevel Low_Fat
- **hasTypeofMeal:** This property attribute a type of meal to a menu, making sure all menus have meal types.
 - Domain: Beverages, Dessert_snack, Main_dish
 - Range: Type_of_Meal
 - Example: Pizza hasProcessType Dinner

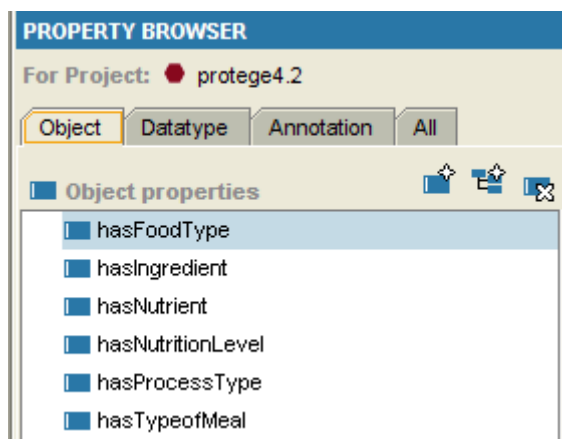


Figure 14: object properties

Datatype properties:

- hasCalcium
- hasCarbohydrates
- hasCholesterol
- hasFat
- hasIron

- hasMagnesium
 - hasPotassium
 - hasProtein
 - hasSodium
 - hasTotalEnergy
 - hasVitamin_B1
 - hasVitamin_B12
 - hasVitamin_B2
 - hasVitamin_B6
 - hasVitamin_C
- All these properties have the same range and domain
- Domain: Beverages, Dessert_snack, Food_Group
- Main_dish, Type_of:nutrients
- Range: Float
- Example: Coffe hasCalcium 170.0g

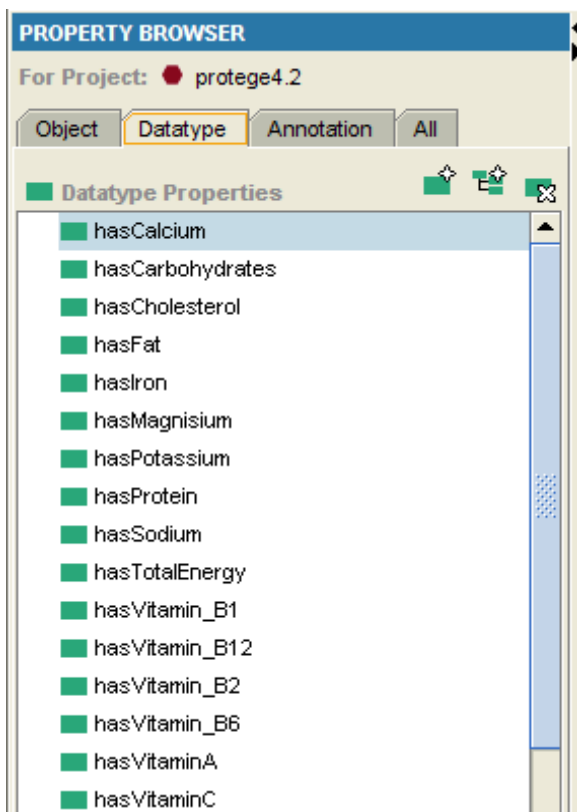


Figure 15: datatype properties

4.1.4. *Individuals*

Individuals are the basic components of an ontology. The individuals may be concrete concepts like a specific menu or an ingredient or an abstract one like numbers as calories in a menu. Since the number of individuals are very big (all menus, ingredients etc...), just some examples will be showed. All the lowest subclasses have at least 1 individual).

Example:

Process Type	Beverages	Type of Meal	Type of Nutrient
Baking	Apple Juice	After Training	Calcium
Boiling	Coffee	Lunch	Iron
Frying	Coca cola	Breakfast	Protein
Roasting	Fanta	Dinner	Sodium
Smoking	Ice Tea	Before Training	Fat

Table 15: Available Process types individual's examples

4.1.5. *Individuals Forms*

Individuals Forms in protégé is a way to enter instance data, the forms contain the data entry field for each propriety attached to a class. There's various types of data entry fields, for texts for integers for cardinality etc. This form will be used to enter all the food and ingredients and all the proprieties attached with them.

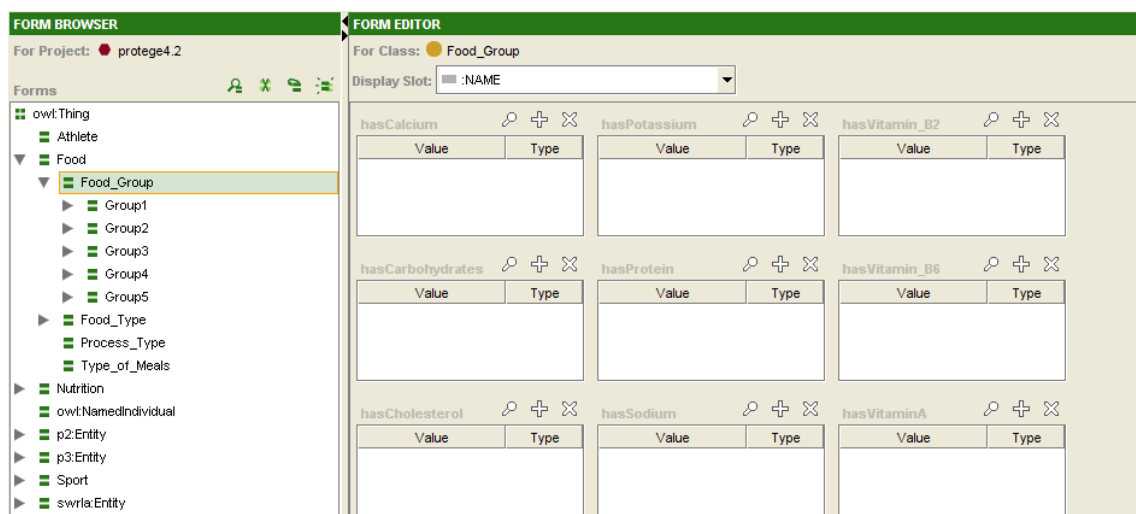


Figure 16: Protégé individual form

4.2. Enter and report interface application analyses

This application will be developed in java and has the objective of getting all the needed information and personal preferences from the user, request sensor application data and query the knowledge base for the nutritional menus all and subsequent information.

4.2.1. Application Overview

In the image 15 we can see that when we run the application, a menu for the personal information will open, after submitting the data the system if external sensor is enable must acquire the sensor data and move to the next page. Since there's the option of skip the personal food and nutrition choices page the user will be forward directly to the pick a menu page. In this page all the menus\items that are adequate for that athlete will be available for the user to choose what menus he wants. After choosing all the menus for the different meals of the day the report with the choosing options and all athlete information will be showed. This is just a small overview about the java application, all application subsystems and user interface will be explained in detail forward in this paper.

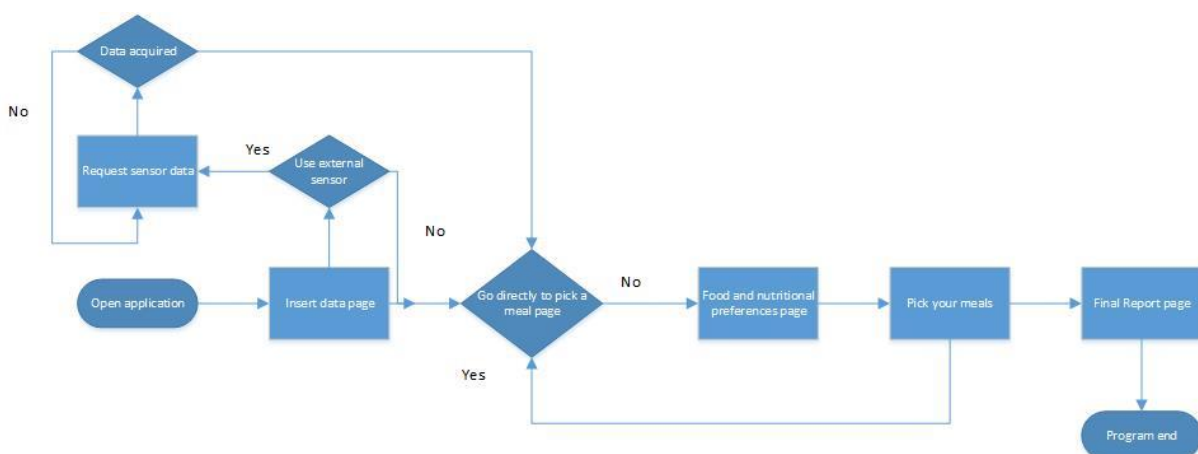


Figure 17: Application flow chart

4.3. Enter personal information and report interface application subsystems design

4.3.1. *Enter Personal Information interface page subsystem*

In the personal interface subsystem, all data about the athlete and all TEE calculation will be made, when the user starts the application all interface will be draw, and will be waiting for the click the button event, when the event happens it will check if all the inserted parameters are correct and not outside of the correct and allowed range (like in the table 8)

Parameter	Value Min	Value Max
Age	10	125
Weight	35	200
Height	120	240

Table 16: Max and Minimum values of the inserted parameters

If all entered parameters are inside the allowed range, the system will then check if the user wants to use sensor or not, if he does it will calculate the user profile data, and send it to the application that will control the sensor application. Then it will wait for incoming messages, if the message is an error it will go back to the start if for the contrary it's a valid value it will make all calculations. The calculations when the sensor is used are just the calculation of TEE and the weight goal, since the scale will give us directly the RMR.

If the user didn't choose to use sensor the application will make bigger calculations, since he has to calculate the RMR. After all the calculations are done the system will check if the user wanted to skip the food preference page if yes will go directly to pick a meal page if not will go to food preferences page.

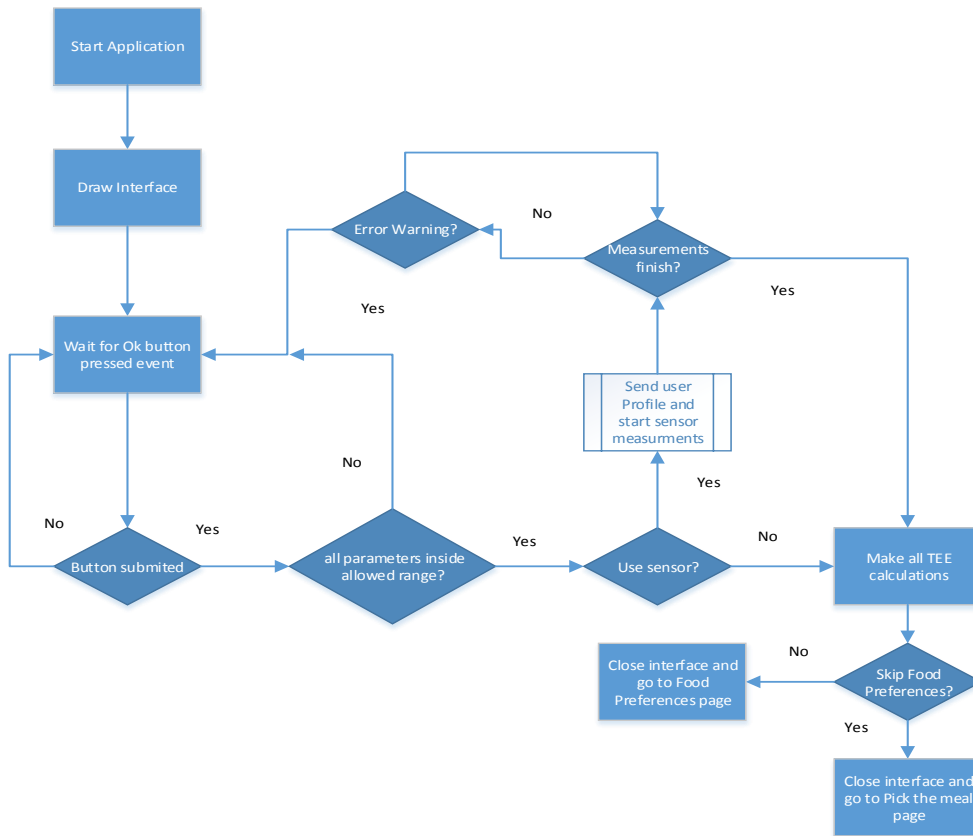


Figure 18: Enter personal information flowchart

4.3.2. Food Preferences Subsystem

In this page the user can choose the food preferences, it will have 5 combo boxes the first two with all the food group available, and if one of those combo box are triggered (the user can choose the food type they want to see for example meat) the system will connect to the ontology and query all ingredients from the chosen food group and then draw another combo box with all those ingredients. The other three original combo boxes will be all process types and all beverages. If the button ok is pressed the system will construct all final queries with the constrains from the combo boxes (if any of them is used), like if the person want some beverage the TEE will be subtracted with the beverage and if the users wants a specific process type the system will query food with the needed TEE and that process type (that can cause the given menus to be a lot smaller). After that the user will be redirect to the pick a meal interface menu.

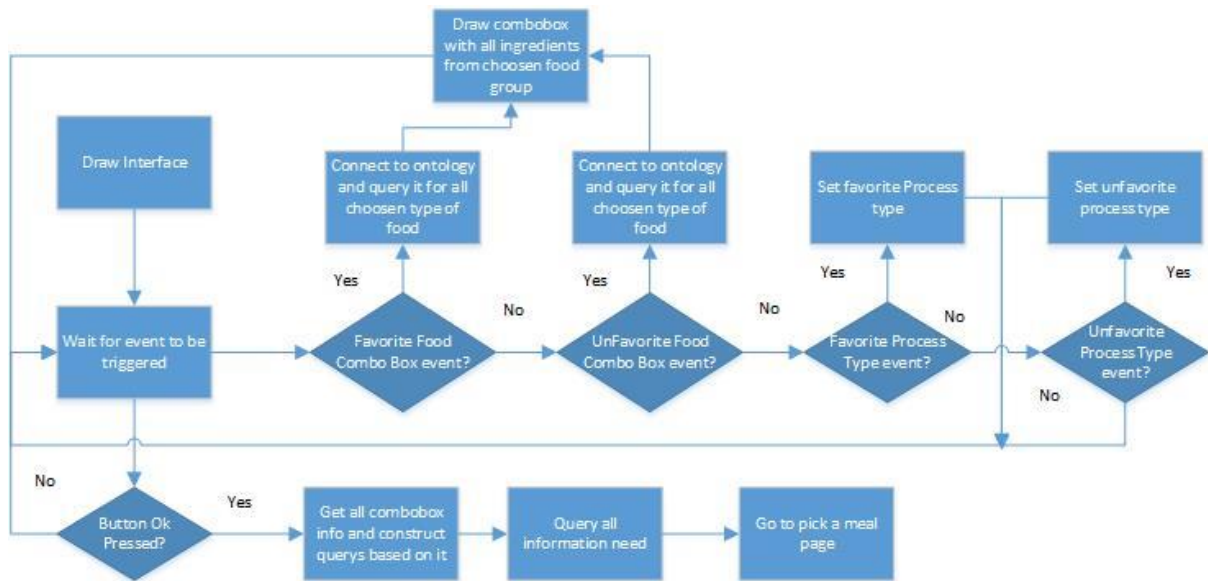


Figure 19: Food preferences subsystem

4.3.3. *Pick a meal subsystem*

In this subsystem the user will be allowed to pick menu\meal for the 9 meals of the day (Breakfast, Before Training, During Training, After Training, Lunch, Before Training, During Training, After Training and Dinner). There will so be 9 combo boxes each with items with the range of TEE , TEE\2 and TEE\3 so the user can make is own menus (for example he can have 1 dose of rice + 1 hamburger + 1 egg if all of them have the range of TEE\3) for each of the 9 meals. The user will first choose the 9 first items if all of them are within TEE no other combo boxes will appear , if he picks one item with TEE\2 another box with TEE\2 items only will appear so he can do TEE\2 + TEE2 = TEE. IF he picks one item with is total energy equal to TEE\3 the combo box 2 will be create and also a combo box 3 and both will only have items with TEE\3 because TEE + TEE\3 > TEE and TEE\3 + TEE\2 > TEE,. We talk here about combo box 2 and 3 but this happens for each of the 9 menus so it will be possible that 18 additional combo boxes will be created, and the combo boxes are just the mechanisms for choosing the items for the final meals)

Value	Combo box 1	Combo box 2	Combo box 3
TEE	Available	Not Available	Not Available
TEE\2 + TEE\2	Available	Available	Not Available
TEE3 + TEE\3 + TEE\3	Available	Available	Available

Table 17: Combo boxes

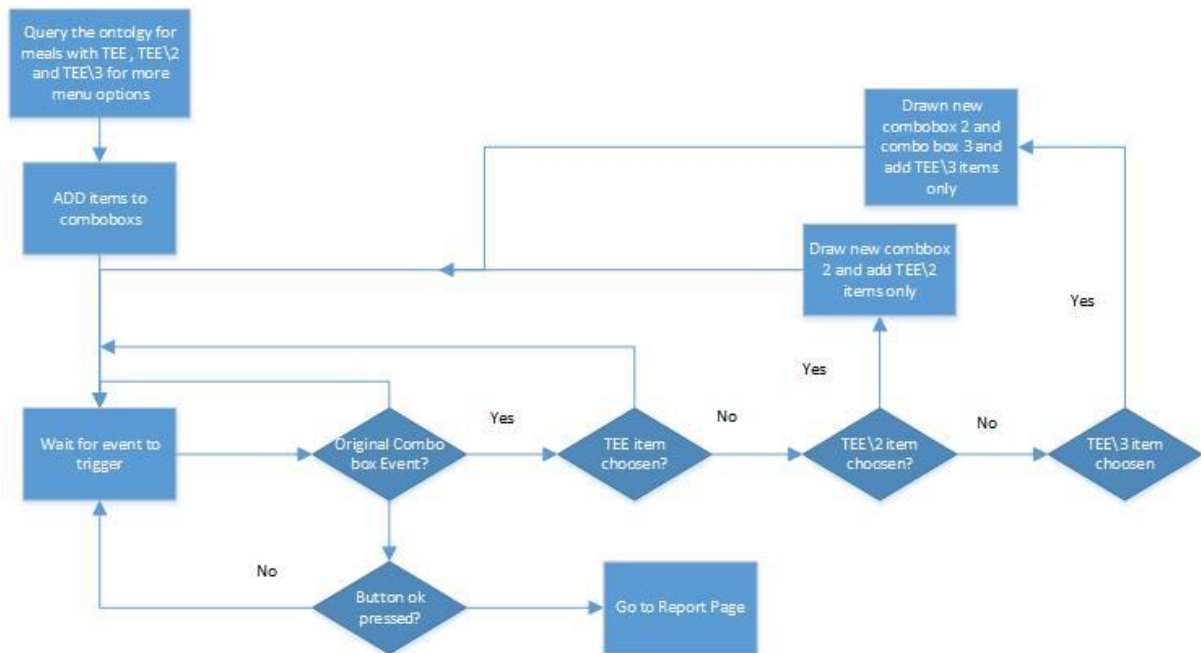


Figure 20: Pick a menu subsystem flow chart

4.3.4. Queries Subsystem

Every time the application ask for a query, it has first to create the OWL model with the given owl URI (in this project it is saved in an online URL, then create the query engine factory and it will run it, after receiving the valour's (takes some time since the application must go online and “download” the ontology) it saves them and return them. All of this will work using the SQWRL TAB API for java.

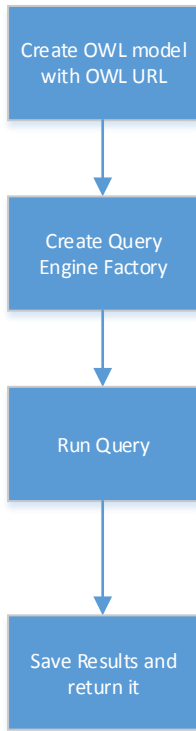


Figure 21: Query's subsystem flow chart

4.3.5. *Sensor application overview*

In the image 22 we see the flowchart of sensor system, when we call this system from the java, the first step that will happen is the setting of the channel establishment parameters and send it, after the open slave channel request will be sent, when the channel is established the next step is the sending of the user profile with the profile parameters from the java part, then the application will wait for some RX message to be received if none is received during the search timeout the channel will be closed and an error warning will be sent to java, if a message is receive will be interpreted as Page 1 ,2 , 3 or 4 (the other pages are ignored since they are not needed here), and the values will be set, and when we received all the pages the channel will be closed and the data sent to the java interface application.

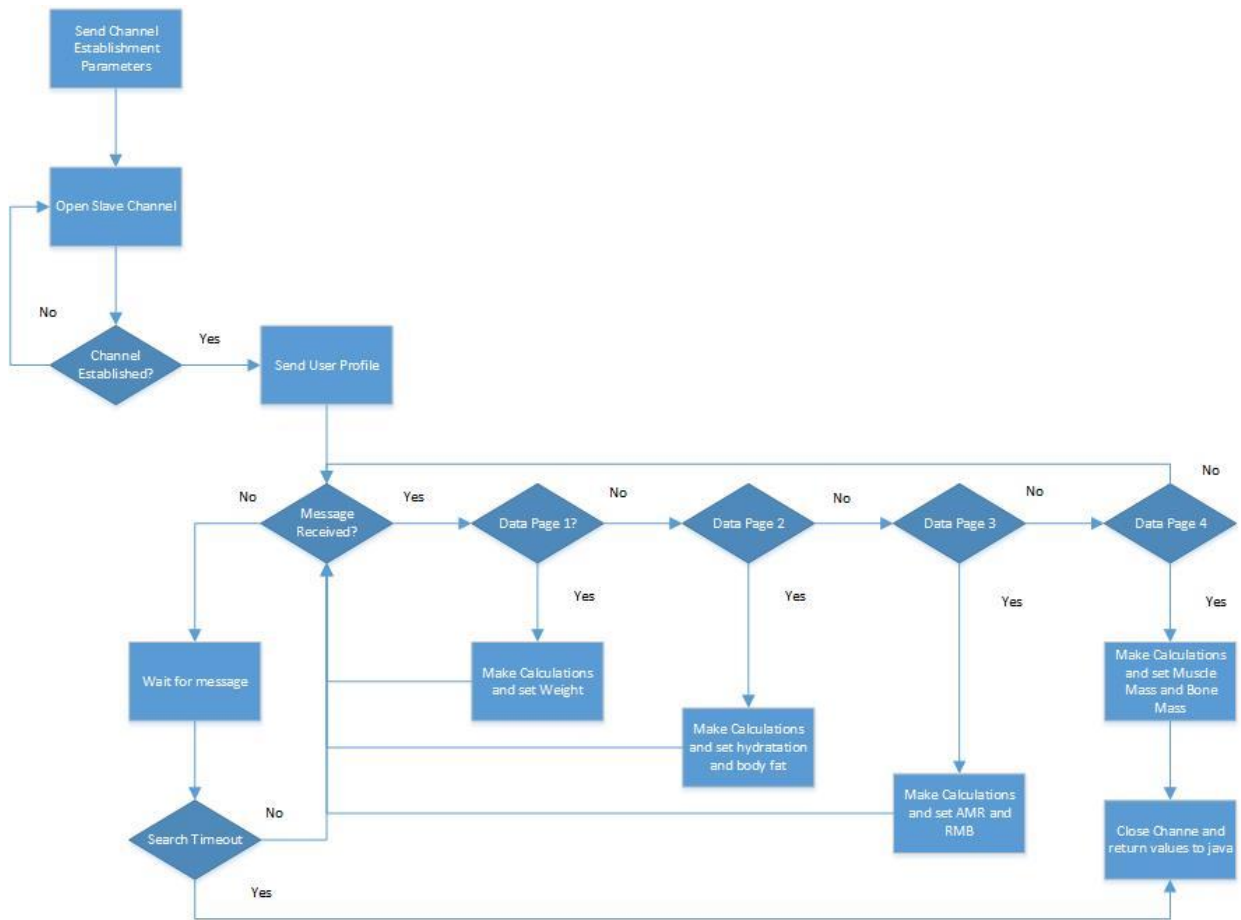


Figure 22: Sensor application flowchart

CHAPTER 6

PROJECT IMPLEMENTATION

5.1. Java Application

5.1.1. *Interface Main Page*

In the interface main page, the user will enter all the personal info, and choose if he wants to use the sensor and skip food preferences.

The screenshot shows a 'Personal Profile' form with the following fields and options:

- Name:
- Age:
- Gender:
- Weight: kg
- Height: cm
- Level of general activity:
- Number of training hours:
- Training Phase:
- Weight Variation:
- Use Scale
- Skip Food Preferences
-

Figure 23: Main Page Interface

5.1.2. *Food Preferences*

In this page the user will choose the food preferences and process type he likes the most and if he wants the menu with a beverage and how much beverages he wants (maximum of two for dinner and lunch). If the user wants the menu with beverage the system will subtract the Energy value of the chosen values at the final TEE.

Food Preferences

Non-Favorite Ingredients:

Favorite Ingredients:

Favorite Process Type:

Non-Favorite Process Type:

Number of Beverages:

Beverage:

Figure 24: Food preferences interface

When this page is submit and before starting the query the system must know what options the user chose and what not, so we will call a method to do the combinations and decide with query we will use. Depending the value of this equation the system will choose the appropriate query. The cb1, cb2,cb3 and cb4 in this equation are if any of this options are chosen and if yes that variables will be true (if a favourite ingredient was chosen the cb2 bool will be true).

$$int\ combination = (cb1 ? 8 : 0) + (cb2 ? 4 : 0) + (cb3 ? 2 : 0) + (cb4 ? 1 : 0);$$

5.1.3. *Pick your daily meals*

In this page the user will choose 9 daily meals, and will have 3 choices, just 1 item, 2 or 3 items (during training the athlete just need an energy drink). When the user picks for example the first item of the lunch if the item is in the range of the Final TEE no other items will be available but if for example, the user picks an item half of TEE he will be able to be two and if 3 times the TEE three items, like this he will always have more food choices.

The screenshot shows a web application window titled "Design Preview [ResultsPage]". The main content area is titled "Choose your meals for today". It features a table with three columns: "First Item", "Second Item", and "Third Item". The rows represent different times of the day: Breakfast, Before Training, During Training, After Training, Lunch, Before Training, During Training, After Training, and Dinner. Each cell in the table contains a dropdown menu. The "First Item" column has "Item 1" selected in all rows. The "Second Item" and "Third Item" columns have "None" selected in all rows. A "Submit" button is located at the bottom center of the form.

	First Item	Second Item	Third Item
BreakFast	Item 1	None	None
Before Training	Item 1	None	None
During Training	Item 1		
After Training	Item 1	None	None
Lunch	Item 1	None	None
Before Training	Item 1	None	None
During Training	Item 1		
After Training	Item 1	None	None
Dinner	Item 1	None	None

Figure 25: Choose your meals for today interface

5.1.4. Report Page interface

All the Information about the user can be seen in this page, that includes all personal information the user inserted, all the data from the sensor and all the menus he picked.

User Profile				Sensor Data			
Name:	Not Available	Height:	Not Available cm	Hydration:	Not Available	Muscle Mass:	Not Available
Gender:	Not Available	Activity Factor:	Not Available	Body Fat:	Not Available	Bone Mass:	Not Available
Age:	Not Available years	Training Phase:	Not Available	Basel Metabolic Rate:	Not Available	Active Metabolic Rate:	Not Available
Weight:	Not Available kg	Weight Goal:	Not Available				

Menus for today

Breakfast:	Not Available
Before Training:	Not Available
During Training:	Not Available
After Training:	Not Available
Lunch:	Not Available
Before Training:	Not Available
During Training:	Not Available
After Training:	Not Available
Dinner:	Not Available

Figure 26: Report page interface

5.1.5. TEE calculation

After the user insert all his information, if the check box Use Sensor is not true (user doesn't want to use the external sensor), the system will start by calculating the RMR since when using the sensor the RMR will be calculated there. The system will then start by checking if the user is a female and if yes calculate the RMR for a female, if it's a male it will calculate the RMR for a male user.

```
if(use_sensor == false)
{
    if(sex == "female")
    {
        rmr = (float) (655.1 + (9.56*weight) + (1.85*height)- (4.68*age));
    }
    else
        rmr = (float) (66.47 + (13.75*weight) + (5*height)- (6.76*age));
}
```

```
}
```

After calculating RMR we must calculate the TEE:

```
tee = (float) ((rmr * act_factor) + ((rmr/24) * mets * hours));
```

Then we must check the user weight goal and calculate the TEE depending on this option:

```
switch (weight_goal) {  
    case "Increase Weight":  
        tee = tee + 500;  
        break;  
    case "Decrease Weight":  
        tee = tee - 500;  
        break;  
}
```

5.1.6. *Items TEE calculation*

We must calculate the percentage of TEE for each meal, for breakfast the percentage is 18% of the TEE as we can see the code above, we will then calculate that percentage for the TEE , TEE\2 and TEE\3 for more items options. The percentage of Before Training meal is 10%, the after training is 12% and the lunch and dinner are 20% of the total TEE and are calculated in the same way as the breakfast.

```
//breakfast  
double b_tee_1 = (tee*0.18) + 50;  
double b_tee_12 = b_tee_1/2;  
double b_tee_13 = b_tee_1/3;  
double b_tee_2 = (tee*0.18) - 50;  
double b_tee_21 = b_tee_2/2;  
double b_tee_23 = b_tee_2/3;
```

5.1.7. User Profile Calculation

If the user chooses to use the sensor we must calculate the string to send to the sensor as we can see in the table 3.x. We can see in the extract of code above that we first start to construct the string with the values that are constant the 0x3a being the profile page number, the 0x10,0x00 the user profile identification, the 0x02 the capabilities byte and the 0xFF is a reserved byte.

```
String userprofile = "{0x3a,0x10,0x00,0x02,0xFF,0x";
```

Then we will create the terminator char (string f) and the aux to complement the final string the x. After we will parse all from the text fields and make calculations for the byte age and gender. (female is 0x00 and male 0x01 so we must add 127 if female and 128 if male).

```
String x = ",0x";  
String f = ",}";  
int ageaux = Integer.parseInt((jTextField2.getText());  
int heightaux = Integer.parseInt((jTextField5.getText());  
String factor = "";  
String aux = "0";  
if("Female".equals(gender.getSelectedItem()))  
{  
    ageaux = ageaux + 127;  
}  
else  
    ageaux = ageaux + 128;
```

After calculation that byte and getting the activity factor we must convert the values from decimal to hexadecimal and make the string concat to construct the final string.

```
aux = Integer.toHexString(ageaux);
```



```

userprofile = userprofile.concat(aux);
userprofile = userprofile.concat(x);
aux = Integer.toHexString(heightaux);
userprofile = userprofile.concat(aux);
userprofile = userprofile.concat(x);
userprofile = userprofile.concat(factor);
userprofile = userprofile.concat(f);

```

5.1.8. Queries

In this system we will use multiple queries, here it will be showed the most important ones, since the others will depend always on the athlete preferences so are never static. We will use the SQWRL Collections to some of our queries since it supports a big degree of closure without violating OWL.

Some of the queries that won't ever change are the food and beverages information since the user when wanting to choose is preferred food will always need that information

Querying all available beverages:

```

querymenu", "Beverages(?f)  $\wedge$  hasTotalEnergy("+bev+", ?g)  $\rightarrow$  sqwrl:select(?g)"

```

Querying all available food from specific food group:

```

"querymenu", ""+Food_group+"(?f)  $\wedge$  hasTotalEnergy(?f, ?c)  $\rightarrow$  sqwrl:select(?f,?c)  $\wedge$ 
sqwrl:orderBy(?c)"

```

The rest of the queries are always changing for example if the user doesn't have any request in favourite type of ingredients, the query will depends only on the TEE and the time of the meal, in this next query, we are asking from all dessert snacks that are between the minimum and maximum range of the TEE and ordering it by energy, but since we will always show the

items with TEE, TEE\2 and TEE\3 we will execute 3 times this query but with different TEE. For querying the rest of the time meals will be the same just the TEE will be different.

"Dessert_snack(?f) \wedge hasTotalEnergy(?f, ?c) \wedge swrlb:lessThan(?c, "+b_tee_1+") \wedge swrlb:greaterThan(?c, "+b_tee_2+") \rightarrow sqwrl:select(?f, ?c) \wedge sqwrl:orderBy(?c)";

But if the user request in the food type some favourite ingredient or non-favourite the system will query for the items with that type of ingredient or process type or avoid it if non favourite . In this next query we want a specific process type “c3”, so the result of this query will be all main dishes between the tee range and with that type of process, instead if we wanted a type of ingredient we would have hasIngredients and the specific ingredients as “c3”.

"Main_dish(?f) \wedge hasTotalEnergy(?f, ?c) \wedge swrlb:lessThan(?c, "+ld_tee_1+") \wedge swrlb:greaterThan(?c, "+ld_tee_2+") \wedge hasProcessType(?f, "+c3+") \rightarrow sqwrl:select(?f, ?c) \wedge sqwrl:orderBy(?c)";

The most complex queries we have is when the user wants a main dish with a specific type of ingredient, and doesn't have one specific ingredient and wants a specific process type. In this example we want some main dish with ingredient mushroom and without tomato and with the process type baking.

Main_dish(?p) \wedge hasIngredient(?p, ?d) \wedge sqwrl:makeSet(?s, ?d) \wedge sqwrl:groupBy(?s, ?p) \wedge sqwrl:notElement(Tomato, ?s) \wedge sqwrl:element(Mushroom, ?s) \wedge hasProcessType(?p, Baking) \wedge hasTotalEnergy(?p, ?f) \wedge swrlb:lessThan(?f, "+ld_tee_1+") \wedge swrlb:greaterThan(?f, "+ld_tee_2+") \rightarrow sqwrl:select(?p, ?f)

5.2. Sensor data acquiring Implementation

For the implementation of sensor data acquiring we will use the ANT Windows Library Package from the www.thisant.com website copyright © from DynasStream Innovations. This

package contains some libraries that help in the development of application with communication with ANT platforms, like connecting with the ANT USB stick and handle the channel connection and serial communication.

We will use more specifically the ANT_LIB for C++, that includes the serial driver for USB stick communication, ANT message framing and the serial messages exchanged between pc and the ANT platform.

5.2.1. *Establishing a channel*

After receiving from the java the User profile, the baud rate and the USB device number, the application will with creating the serial object so we can connect with the ANT USB stick.

```
pclSerialObject = new DSISerialGeneric(); //the function DSISerialGeneric creates a serial object
```

Then we initiate that object with the baud rate and USB device number, and create the message\framer object

```
pclSerialObject->Init(BAUDRATE, USBDeviceNumber);  
pclMessageObject = new DSIFramerANT(pclSerialObject);
```

When the serial object is initiated we already have connection with the USB stick, now we just need to setup the channel for communication with the scale.

Setting Network Key – `pc1MessageObject->SetNetworkKey(USER_NETWORK_NUM, NetworkKey, MESSAGE_TIMEOUT);`

Parameters	Type	Description	Value Used
User Network Number	UCHAR	The Network Number	0
Network Key	UCHAR[8]	The network 8 byte number	0xB9, 0xA5, 0x21, 0xFB, 0xBD, 0x72, 0xC3, 0x45

Table 18: Setting Network key table

When the Network key is set we will get a message with an event report (the events and messages must be configured) and since our channel type is slave we will assign it if the network key was set correctly. The `stMessage` is the buffer that receives incoming messages.

```

switch(stMessage.aucData[1])
{
    case MESG_NETWORK_KEY_ID:
    {
        if(stMessage.aucData[2] != RESPONSE_NO_ERROR)
        {
            printf("Error configuring network key: Code %d\n",
stMessage.aucData[2]);
            break;
        }
        printf("Network key set.\n");
        printf("Assigning channel...\n");
        if(ucChannelType == CHANNEL_TYPE_MASTER)
        {
            bStatus = pc1MessageObject-
>AssignChannel(USER_ANTCHANNEL, PARAMETER_TX_NOT_RX, 0, MESSAGE_TIMEOUT);
        }
        else if(ucChannelType == CHANNEL_TYPE_SLAVE)
        {
            bStatus = pc1MessageObject->AssignChannel(USER_ANTCHANNEL, 0, 0,
MESSAGE_TIMEOUT);
        }
        break;
    }
}

```

Parameters	Type	Description	Value used
Channel number	UCHAR	Number of the channel	0
Channel Type	UCHAR	Type of Channel used	0

Table 19: Assign a channel function table

If the channel is correctly assign we will get an event to set the channel id, assign the device number, the device type and the transmit type.

```

case MSG_ASSIGN_CHANNEL_ID:
{
    if(stMessage.aucData[2] != RESPONSE_NO_ERROR)
    {
        printf("Error assigning channel: Code 0%d\n", stMessage.aucData[2]);
        break;
    }
    printf("Channel assigned\n");
    printf("Setting Channel ID...\n");
    bStatus = pc1MessageObject->SetChannelID(USER_ANTCHANNEL,
USER_DEVICENUM, USER_DEVICETYPE, USER_TRANSTYPE, MESSAGE_TIMEOUT);
    break;
}

```

Parameters	Type	Description	Value Used
Channel Number	UCHAR	Channel Number	0
Device Number	USHORT	Device Number	0 (match any device)
Device Type MSB	UCHAR(1bit)	Paring Request	0
Device Type LSB	UCHAR(7bit)	Device Type	119
Transmission Type	UCHAR	Transmission Type	0

Table 20: Assign Channel ID function Table

After setting channel we must set the Channel Frequency and Channel Period. If no error happens, we will also open the channel since all need parameters were already set.

```

case MSG_CHANNEL_ID_ID:
{
    if(stMessage.aucData[2] != RESPONSE_NO_ERROR)
    {
        printf("Error configuring Channel ID: Code 0%d\n",
stMessage.aucData[2]);
        break;
    }
    printf("Channel ID set\n");
    printf("Setting Radio Frequency...\n");
    bStatus = pc1MessageObject->SetChannelRFFrequency(USER_ANTCHANNEL,
USER_RADIOFREQ, MESSAGE_TIMEOUT);
    break;
}

```

Parameters	Type	Description	Value used
Channel Number	UCHAR	The channel to be assigned	0
Channel RF Freq.	UCHAR	The channel frequency	57

Table 21: Setting Channel Frequency function Table

```

case MSG_CHANNEL_RADIO_FREQ_ID:
    {
        if(stMessage.aucData[2] != RESPONSE_NO_ERROR)
        {
            printf("Error configuring Radio Frequency: Code 0%d\n",
stMessage.aucData[2]);
            break;
        }
        printf("Radio Frequency set\n");
        printf("Opening channel...\n");
        bBroadcasting = TRUE;
        bStatus = pclMessageObject-
>SetChannelPeriod(USER_ANTCHANNEL,8192,MESSAGE_TIMEOUT);
        bStatus = pclMessageObject->OpenChannel(USER_ANTCHANNEL,
MESSAGE_TIMEOUT);

        break;
    }

```

Parameters	Type	Description	Value used
Channel Number	UCHAR	The channel number	0
Message Period	USHORT	Channel messaging Period	8192

Table 22: Setting Channel Period function table

After all this if no error happens the channel is open and ready to communicate with the sensor, if in any of this steps an error happens all will start from the beginning and try to established the channel again.

5.2.2. *Send User Profile*

After the channel is established we must send the profile so the will be ready for start the measurements. We will send the user profile information message via broadcast using the broadcast function given by the ANT library, the payload of the message is 8 bytes.

```

UCHAR msg[] = USER_PROFILE;
pclMessageObject->SendBroadcastData(USER_ANTCHANNEL, msg);

```

Parameters	Type	Description	Value used
Channel Number	UCHAR	The channel the data is from	0
User Profile message	UCHAR[8]	The user profile from java	Check chapter 3.x

Table 23: Send Broadcast message table

5.2.3. *RX Message Received*

After connecting to the channel the scale will start sending messages (common pages mostly), so we will have to interpret them to get the correct measurements.

In the case a RX Message is Received an event MESH_EXT_BURST_DATA_ID will happen where we will interpret the receiving message. In the extract of code above we can see that when a message is received, the application will check first if it's different from the common pages address and then check what page it is and interpret the results. In the code above we check what page was received (so avoiding common pages) and if the page is invalid and or still computing then we interpret the data received and get the final values of the measurements.

```

if(stMessage.aucData[ucDataOffset + 0] == 0x01)
{
    int LSB = stMessage.aucData[ucDataOffset + 6];
    int MSB = stMessage.aucData[ucDataOffset + 7];

    int aux = (MSB<<8) | LSB;
    double weight = aux/100;
}
if(stMessage.aucData[ucDataOffset + 0] == 0x02 &&
stMessage.aucData[ucDataOffset + 6] != 0xFF && (stMessage.aucData[ucDataOffset + 0] !=
0xFE ||stMessage.aucData[ucDataOffset + 0] != 0xFF) )
{
    int LSB = stMessage.aucData[ucDataOffset + 4];
    int MSB = stMessage.aucData[ucDataOffset + 5];
    int LSB2 = stMessage.aucData[ucDataOffset + 6];
    int MSB2 = stMessage.aucData[ucDataOffset + 7];

    int aux = (MSB<<8) | LSB;
    int aux2 = (MSB2<<8) | LSB2;
}

```

```

        double hydratation = aux/100;
        double bodyfat = aux2/100;
    }

    if(stMessage.aucData[ucDataOffset + 0] == 0x03&&
stMessage.aucData[ucDataOffset + 6] != 0xFF && (stMessage.aucData[ucDataOffset + 0] !=
0xFE ||stMessage.aucData[ucDataOffset + 0] != 0xFF))
    {
        int LSB = stMessage.aucData[ucDataOffset + 4];
        int MSB = stMessage.aucData[ucDataOffset + 5];
        int LSB2 = stMessage.aucData[ucDataOffset + 6];
        int MSB2 = stMessage.aucData[ucDataOffset + 7];

        int aux = (MSB<<8) | LSB;
        int aux2 = (MSB2<<8) | LSB2;
        double AMR = aux/4;
        double BMR = aux2/4;
    }
    if(stMessage.aucData[ucDataOffset + 0] == 0x04)
    {
        int LSB = stMessage.aucData[ucDataOffset + 5];
        int MSB = stMessage.aucData[ucDataOffset + 6];
        int bone = stMessage.aucData[ucDataOffset + 6];

        int aux = (MSB<<8) | LSB;
        double MM = aux/100;
        double Bone_Mass = bone/10;
    }

```


5.3. JNI

Since the java can't communicate with external devices we will use java native interface, JNI it's a programming framework that allows that the virtual machine of java access library coded in a different type, allowing applications to run in java.

JNI allows the writing of native methods to handle situations like in this system where the application is not fully written in java. We have to make the java virtual machine to be aware of the C++ functions we want to call and load the library (our C++ application).

```
System.loadLibrary("sensor");
```

Then we will compile the java program with javac, the program won't still work because the library is still not created, so we must compile our java application with javah so we can generate the header file that will contain the functions that will be present in the C++.

```
JNIEXPORT void JNICALL Java_Sensor_Values  
(JNIEnv * env, jobject obj)
```

After we need to compile and create the native library, this is system depend so the extension change from system to system. (.dll for windows, .so for linux

CHAPTER 6

RESULTS

In this chapter we will test and show results of all our system, explain in detail how to use the interface.

6.1. Interface Results

It will be test different results for different types of users , we will use for this testing a female and a male user that are professional athlete and a casual male and female user so we can see the difference in nutrition between them. We will first start by testing the interface without using the sensor and after with the sensor to check the differences between the calculate RMR and the measured RMR. Since we won't be able to test on two real Olympic athletes the measurements with the scale will only happen with a female and male casual athletes.

6.1.1. *Testing profiles*

We will use for the system testing 4 different profiles, the female and male A are not professional weightlifters and are here to compare the values with the female B and male B that is the profile of two Olympic medal winners in weightlifting.

The female A is a casual athlete with 23 age

The Male A is a casual athlete with 26 years old and 77 kg, trains around 1 hour a day and is activity level is light in comparison with the Male B an Olympic athlete with 31 years 80 kg and trains 4 hours a day an exceptional activity.

User	Age	Weight	Height	Training Hours	Level Activity	Training Phase
Female A	23	66	172	1	Moderate Act.	Comp.
Female B	22	56	160	4	Exceptional Act.	Comp.
Male A	26	77	180	1	Light Act.	Comp.
Male B	31	80	172	4	Exceptional Act.	Comp.

Table 24: User profiles for testing the system

6.1.2. Profile Male A testing

We will start inserting all the wanted data like we can see in the figure 27, the sensor won't be used in this first test and we will skip food preferences testing after. With this parameters the TEE will be 3190 Kcal.

Personal Profile

Name: MALE A

Age: 26

Gender: Male

Weight: 77 kg

Height: 180 cm

Level of general activity: Light Activity

Number of training hours: 1

Training Phase: Competition

Weight Variation: Maintain Weight

Use Scale

Skip Food Preferences

Ok

Figure 27: Male A User Profile

Since the TEE is 3690KCal the application will now calculate the possible items for the 9 meals between the range of TEE, twice the TEE and three times the TEE. So we will have the option of items between 524 and 624, 262 and 362, 174 and 274 for Breakfast; 269 and 369,

134 and 184, 89 and 123 for before training; 332 and 432, 166 and 266, 110 and 144 for after training; 588 and 688, 269 and 369, 162 and 262 for dinner and lunch; As we can see in the image above using the 3 times TEE gives a lot more options and makes this system a lot more flexible for the user to pick what items he wants to eat, during the training for example just that chosen item is enough for his ideal during training needed calories but for example in the dinner he needs that 3 items to complete the ideal energy needs and in lunch those 2 items are enough.

The screenshot shows a web application window titled "Choose your meals for today". The interface is organized into three columns: "First Item", "Second Item", and "Third Item". The rows represent different meal occasions. The "During Training" row only has a dropdown for the "First Item". The "Lunch" and "Dinner" rows have dropdowns for all three items. A "Submit" button is located at the bottom center of the form.

	First Item	Second Item	Third Item
BreakFast	Cheesecake	Bread_rice_bean_filled	None
Before Training	Compufl_salted_baked	None	None
During Training	Energy Drink 33cl		
After Training	Bread_chocolate_top	None	None
Lunch	Khao-Moo-Daeng__Rice_with_roastred_red_p...	Pla-Insee-Tod__Mackeral_spanish_salted_drie...	None
Before Training	Snack_crisspy_rice_flour_baked_fried	None	None
During Training	Energy Drink 33cl		
After Training	Cake_fruit	None	None
Dinner	Ho-mok-Pla-chon-Sai-Bai-Yor__Fish_with_cur...	Tom-Kha-Kai	Ho-mok-Pla-chon-Sai-Bai-Yor__Fish_with_curry...

Figure 28: Choose a meal for Profile Male A example

When we submit we get the report with all profile information (in this example we didn't use the sensor), and all the menus the user choose.



Figure 29: Report for user Male A

6.1.3. Profile Male B

In this profile testing we use the information of an Olympic weightlifter and his TEE was of 5269 Kcal a lot higher level in comparison with the user MALE B so we can notice that the more hours of training and the heavier the activity the more kcal the user will need.

The screenshot shows a 'Personal Profile' form with the following fields and values:

- Name: MALE B
- Weight: 80 kg
- Age: 31
- Height: 172 cm
- Gender: Male
- Level of general activity: Exceptional Activity
- Number of training hours: 4
- Training Phase: Competition
- Weight Variation: Maintain Weight
- Use Scale:
- Skip Food Preferences:

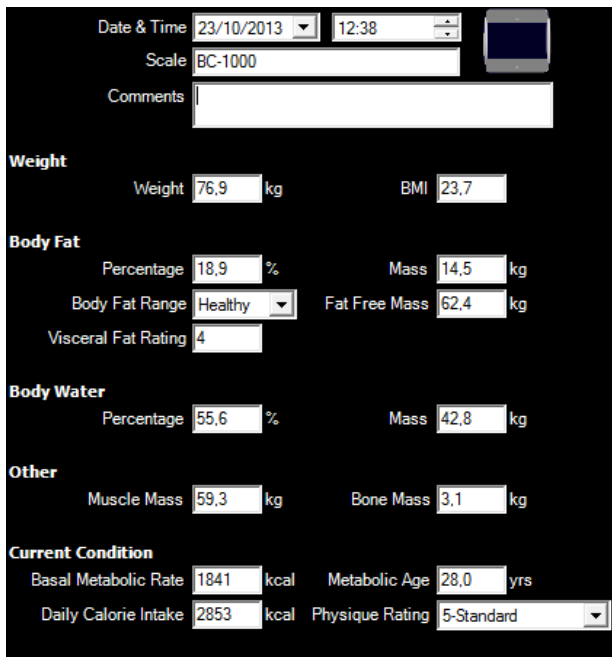
An 'Ok' button is located at the bottom center of the form.

Figure 30: Male B profile information

6.1.4. Profile Male A Sensor Results

We will compare the values from that we get from this system with the value from the official software from this system and check if the values are the same or if there is some error.

As we can see in the two testes above the values are the same or almost the same, so the system is receiving the values with 100% accuracy. We can see that the daily calorie intake in the official software is lower (3690kcal > 2853kcal) since our software was development thinking about high level athletes so takes in account the competition phase and training phase and daily hours as this software doesn't that in account making this system more viable for a professional athlete.



The screenshot displays the software interface for a Tanita scale. At the top, there are fields for 'Date & Time' (23/10/2013), 'Scale' (BC-1000), and 'Comments'. Below this, the data is organized into several sections:

- Weight:** Weight 76.9 kg, BMI 23.7
- Body Fat:** Percentage 18.9 %, Mass 14.5 kg, Body Fat Range Healthy, Fat Free Mass 62.4 kg, Visceral Fat Rating 4
- Body Water:** Percentage 55.6 %, Mass 42.8 kg
- Other:** Muscle Mass 59.3 kg, Bone Mass 3.1 kg
- Current Condition:** Basal Metabolic Rate 1841 kcal, Metabolic Age 28.0 yrs, Daily Calorie Intake 2853 kcal, Physique Rating 5-Standard

Figure 31: Original tanita scale software

Sensor Data			
Hydration:	55.6%	Muscle Mass:	49.3 kg
Body Fat:	18.9%	Bone Mass:	3.1 kg
Basel Metabolic Rate:	Not Available	Active Metabolic Rate:	1849.46 cal

Figure 32: System sensor measurements

6.1.5. Profile FEMALE A

We will start inserting all the wanted data like we can see in the figure 5.x, the sensor won't be used in this first test and we will skip food preferences testing after. With this parameters the TEE will be 2769kcal.

The screenshot shows a 'Personal Profile' dialog box with the following fields and options:

- Name: FEMALE A
- Age: 23
- Gender: Female (dropdown)
- Weight: 66 kg
- Height: 172 cm
- Level of general activity: Moderate Activity (dropdown)
- Number of training hours: 2
- Training Phase: Competition (dropdown)
- Weight Variation: Maintain Weight (dropdown)
- Use Scale (selected radio button)
- Skip Food Preferences (unselected radio button)
- Ok button

Figure 33: FEMALE A profile information

	First Item	Second Item	Third Item
BreakFast	Krong_krang_krob_kem	None	None
Before Training	Cookie_chocolate_chip	None	None
During Training	Energy Drink 33cl		
After Training	None	None	None
Lunch	Guay-Teaw-Rad-Na-Moo__Fried_rice_noodle_...	Pad-Tua-Fakya-Sai-Moo__Saute_green_yard_...	Laab-Nuea-E-san__Beef_cooked_northeastern...
Before Training	Kanom_Toa_Kearw	Pancake	None
During Training	Energy Drink 33cl		
After Training	Luk_Chups	None	None
Dinner	Yum-Moo-Yor__Sour_steamed_pork_sausage_...	Khao_phat_Moo_Sai_Khai__Fried_rice_with_p...	None

Figure 34: Choose a meal for Profile FEMALE A example

User Profile		Sensor Data		
Name: FEMALE A	Height: 172 cm	Hydration: 60.3%	Muscle Mass: 53.2 kg	
Gender: Female	Activity Factor: Moderate Activity	Body Fat: 15.2%	Bone Mass: 3.1 kg	
Age: 23 years old	Training Phase: Competition	Basel Metabolic Rate: 1656 kcal	Active Metabolic Rate: 1678 kcal	
Weight: 66 kcal	Weight Goal: Maintain Weight			

Menus for today	
Breakfast:	Krong_krang_krob_kem
Before Training:	Cookie_choc...
During Training:	Energy Drink 33cl
After Training:	None
Lunch:	Guay-Teaw-Rad-Na-Moo__Fried_rice_noodle_dark_saysauce_topped_with_fried_Chinese_kale_pork + Pad-Tua-Fakya-Sai-Moo__Saute_green_yard_long_bean_and_pork + Laab-Nuea-E-san__Beef_cooked_north...
Before Training:	Kanom_Toa_Kearw + Pancake
During Training:	Energy Drink 33cl
After Training:	Luk_Chups
Dinner:	Yum-Moo-Yor__Sour_steamed_pork_sausage_salad + Khao_phat_Moo_Sai_Khai__Fried_rice_with_pork_and_egg

Figure 35: Report for user FEMALE A

6.1.6. Profile FEMALE B

In this profile testing we use the information of an female Olympic weightlifter and her TEE was of 3597 kcal a lot higher level in comparison with the user FEMALE A even if the female A was heavier and higher but a big margin (more than 10cm and 10kg) the kcal is still

a lot superior, so the level of activity and the hours of training really change the value of TEE a lot.

The screenshot shows a 'Personal Profile' window with the following fields and values:

- Name: FEMALE B
- Age: 22
- Gender: Female
- Weight: 56 kg
- Height: 160 cm
- Level of general activity: Exceptional Activity
- Number of training hours: 4
- Training Phase: Competition
- Weight Variation: Maintain Weight

There are also two radio buttons: 'Use Scale' (selected) and 'Skip Food Preferences'. An 'Ok' button is at the bottom.

Figure 36: Male B profile information

6.1.7. Profile Female A Sensor Results

We will compare the values from that we get from this system with the value from the official software from this system and check if the values are the same or if there is some error, but again the values were almost the same making the sensor reading very accurate.

Sensor Data			
Hydration:	60.3%	Muscle Mass:	53.2 kg
Body Fat:	15.2%	Bone Mass:	3.1 kg
Basel Metabolic Rate:	1656 kcal	Active Metabolic Rate:	1678 kcal

Figure 37: Application report sensor data

Measurement	
Date & Time	23/10/2013 10:34
Scale	BC-1000
Comments	
Weight	
Weight	66.0 kg
BMI	22.6
Body Fat	
Percentage	15.2 %
Mass	10.0 kg
Body Fat Range	Healthy
Fat Free Mass	56.0 kg
Visceral Fat Rating	2
Body Water	
Percentage	60.3 %
Mass	39.8 kg
Other	
Muscle Mass	53.2 kg
Bone Mass	2.8 kg
Current Condition	
Basal Metabolic Rate	1656 kcal
Metabolic Age	16.0 yrs
Daily Calorie Intake	2947 kcal
Physique Rating	5-Standard

Figure 38: Original software female A profile data

6.1.8. Food preferences testing

For this test we will suppose the user MALE A has the preference for the ingredient Tomato (this system just allows the preferences for dinner and lunch since it's not healthy to ate every meal with the same ingredient), the only options for lunch and dinner are items with the ingredient tomato, that reduces our options a lot since a lot of the items doesn't have tomato in their constitution.

Food Preferences

Non-Favorite Ingredients:

Favorite Ingredients:

Favorite Process Type:

Non-Favorite Process Type:

Number of Menus:

Beverage:

Figure 39: User favourite ingredient tomato

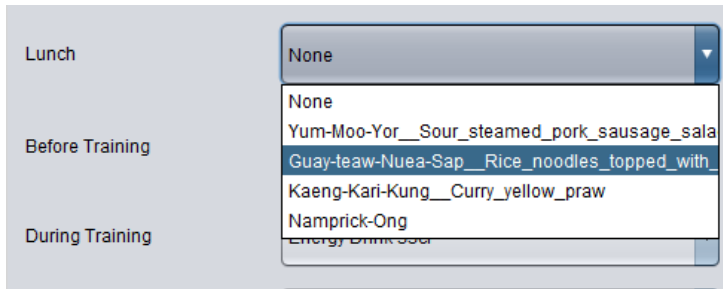


Figure 40: Items with ingredient Tomato

If in the other hand the user non favourite ingredient is Tomato, no item with tomato will be appear so the user can always avoid their non-favourite ingredients (also since there's more food without tomato then with tomato the number of available menus are bigger) it works the same way if the user wants item with a specific process type or without it.

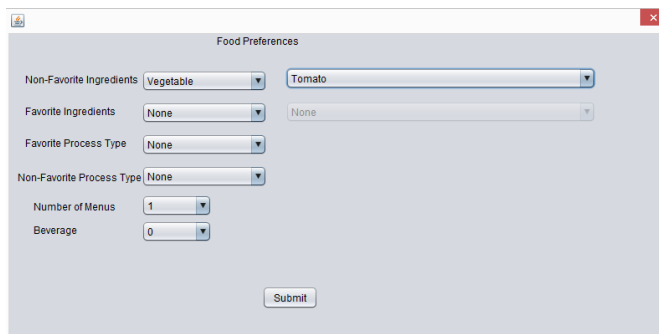


Figure 41: User Non favourite ingredient tomato

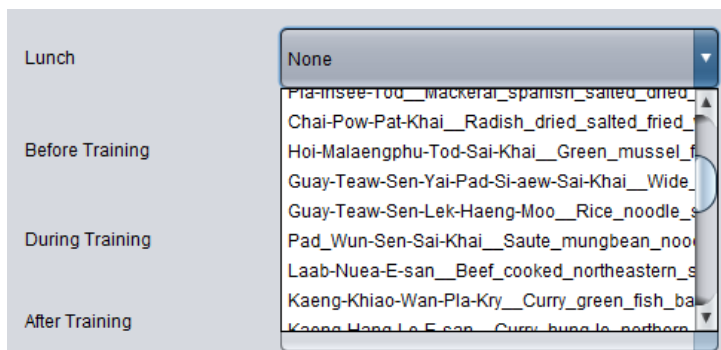


Figure 42: Items without Tomato ingredient

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this project, it was concluded that an ontology is a viable way to approach this kind of problems when you need to have a knowledge base with a lot concepts and relationships between them, in this case all food and nutrition proprieties. Still the performance in an ontology model is considerable lesser than in a DB schema making the project a bit slow when querying big number of data. But since in an ontology the data can be reused in other system opposite to DB schema where the database it's rarely reused, and the ontology can be upgraded with new types of sports other than weightlifting, the use of an ontology knowledge base in this project is very successful.

There's still a lot room for improving in this project, starting by using a food knowledge and a more suited for Europe since most of the food in this project are not usual in the European menus. Another possible improving is connecting the profile information with a SQL database, so the user doesn't need to insert the data every time he/she uses the system.

Another possible future work is creating an ontology for biomechanics of weightlifting, since the weightlifting techniques can be improved with help of the physic laws.

With the help of a biomechanics ontology and the use of specialize sensors the athlete can have a big help during trainings like lifting. On starting position the athlete must assume a position that he can lift the barbell with maximum the acceleration with the minimum effort. The shoulders must be vertically lined up with the bar, the elbow aligned with the knee. There is an angle on the knees with approximately 80o, depending on athlete height. The ontology can with the help of the sensors, giving up warnings in real time to all those parameters, like your knees are in the wrong position, during the lifting that can improve the athlete performance. This ontology can be used by merging with the nutrition and food ontology, to create a system that will give the menus to athletes, and still help them greatly during the training, so we can get to his peak performance.

In conclusion this project was successful experience into ontology systems, and that ontology it's a valid technology to system like this.

REFERENCES

- [1] Weightlifting eating for your sport [Online]. Available: www.nutrition.nestle.co.nz
- [2] G. Slater and S. M. Phillips, "Nutrition guidelines for strength sports: sprinting, weightlifting, throwing events, and bodybuilding," *Journal of sports sciences*, vol. 29 Suppl 1, pp. S67-77, 2011.
- [3] A. Storey and H. K. Smith, "Unique aspects of competitive weightlifting: performance, training and physiology," *Sports Med*, vol. 42, pp. 769-90, 2012.
- [4] M. Hassapidou, "Dietary assessment of five male sports teams in Greece," *Nutrition and food science.*, vol. 31, pp. 31-34, 2001.
- [5] J. D. Chen, J. F. Wang, K. J. Li, Y. W. Zhao, S. W. Wang, Y. Jiao, and X. Y. Hou, "Nutritional problems and measures in elite and amateur athletes," *Am J Clin Nutr*, vol. 49, pp. 1084-9, 1989.
- [6] D. H. Fudholi, N. Maneerat, and R. Varakulsiripunth, "Ontology-based daily menu assistance system," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, 2009, pp. 694-697.
- [7] C. Snae and M. Bruckner, "FOODS: A Food-Oriented Ontology-Driven System," in *Digital Ecosystems and Technologies, 2008. DEST 2008. 2nd IEEE International Conference on*, 2008, pp. 168-176.
- [8] L. Chang-Shing, W. Mei-Hui, L. Huan-Chung, and C. Wen-Hui, "Intelligent ontological agent for diabetic food recommendation," in *Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on*, 2008, pp. 1803-1810.
- [9] C. Jaime, D. David, G. Valeria, L. Loredana, and T. Valentina, "An Example of Food Ontology for Diabetes Control," ed, 2005.
- [10] N. Suksom, Buranarach, M., Thein, Y.M., Supnithi, T., and Netisopakul, P., " A Knowledge-based Framework for Development of Personalized Food Recommender System," in *the 5th International Conference on Knowledge, Information and Creativity Support Systems (KICSS2010)*, 2010.

- [11] T. Stellingwerff, R. J. Maughan, and L. M. Burke, "Nutrition for power sports: middle-distance running, track cycling, rowing, canoeing/kayaking, and swimming," *J Sports Sci*, vol. 29, p. 28, 2011.
- [12] M. M. Manore, N. L. Meyer, and J. Thompson, *Sport Nutrition for Health and Performance*, 2 ed.: Human Kinetic, 2009.
- [13] B. E. Ainsworth, W. L. Haskell, M. C. Whitt, M. L. Irwin, A. M. Swartz, S. J. Strath, W. L. O'Brien, D. R. Bassett, Jr., K. H. Schmitz, P. O. Emplaincourt, D. R. Jacobs, Jr., and A. S. Leon, "Compendium of physical activities: an update of activity codes and MET intensities," *Med Sci Sports Exerc*, vol. 32, pp. S498-504, 2000.
- [14] N. Noy and D. McGuinness, "Ontology Development 101: A Guide to Creating Your First Ontology," 2001.
- [15] Protégé [Online]. Available: <http://protege.stanford.edu/>
- [16] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, and M. Dean, "{SWRL: A Semantic Web Rule Language Combining OWL and RuleML}," 2004.
- [17] K. Clark, M. Grove, E. Sirin, H. Pérez-Urbina, P. Klinov, and E. Rodríguez-Díaz. Pellet: OWL 2 Reasoner for Java [Online]. Available: <http://clarkparsia.com/pellet/>
- [18] C. Smith and E. Friedman-Hill. Jess, the Rule Engine for the Java Platform [Online]. Available:
<http://www.jessrules.com/jess/index.shtml>
- [19] Michael Uschold, PHD Semantic Arts
- [20] Rajiv Pandey and Dr. Sanjay Dwivedi, "Ontology Description using OWL to Support Semantic Web Applications ", 2011
- [21] Denis Shapovalenko, Available:
http://shapovalenko.typepad.com/denis_shapovalenko/2008/03/mini-research-s.html", 2008

