ONTOLOGY-BASED NUTRITION PLANNING ASSISTANCE SYSTEM FOR HEALTH CONTROL

DHOMAS HATTA FUDHOLI

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRONICS ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2010

KMITL-2010-EN-M-040-041

หัวข้อวิทยานิพนธ์	ระบบช่วยเหลือการวางแผนทางโภชนาการเพื่อควบคุมสุขภาพโดยใช้ออนโทโลยี
นักศึกษา	นาย โดมัส ฮัตตะ ฟูโดลิ
รหัสนักศึกษา	51060419
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมอิเล็กทรอนิกส์
พ.ศ.	2553
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ. ดร. รัตติกร วรากูลศิริพันธุ์

บทคัดย่อ

แรงจูงใจที่จะนำองค์ความรู้ทางโภชนาการเพื่อการควบคุมสุขภาพของมนุษย์ ทำให้เราได้นำเสนอ "ระบบช่วยเหลือการวางแผนทางโภชนาการโดยใช้ออนโทโลยี (Ontology-based Nutrition Planning Assistance System (ONPAS))" โดยมีจุดประสงค์เพื่อให้ ONPAS เป็นระบบให้การช่วยเหลือมวลมนุษยชาติในการรักษาสุขภาพด้วยการมีโภชนาการที่เหมาะสม ซึ่งได้จากการรับสารอาหารที่พอเหมาะกับความต้องการของร่างกาย ทั้งนี้ ONPAS จะให้คำแนะนำรายการอาหารประจำวันประกอบไปด้วยสารอาหารที่ตอบสนองความต้องการทางด้านพลังงานในแต่ละวันที่เหมาะสมกับบุคคลนั้นๆ ONPAS ได้ถูกสร้างขึ้นในรูปแบบการประยุกต์เป็นเว็บที่มีความหมาย (Semantic Web) โดยใช้ออนโทโลยี ซึ่งเป็นเทคโนโลยีใหม่ที่ประยุกต์ใช้กับเว็บได้ พื้นฐานของออนโทโลยี คือเทคโนโลยีการสร้างมโนภาพที่จำลองเหตุการณ์จริง ดังนั้นจึงสามารถใช้หลักการของออนโทโลยีในการสร้างฐานข้อมูลทางโภชนาการของ ONPAS ที่แสดงในรูปของคำศัพท์ธรรมดา สามารถใช้อ้างอิงได้ในชีวิตประจำวัน และมีการกำหนดฟังก์ชันต่างๆ ที่ช่วยให้คำแนะนำในการเลือกหรือแยกแยะรายการโภชนาการที่เหมาะสมได้อย่างมีประสิทธิภาพ นอกจากนั้นยังได้มีการนำหลักการของฟัซซี่มาใช้ร่วมกับออนโทโลยี เพื่อกำหนดค่าการตัดสินเลือกองค์ประกอบของคำแนะนำที่เหมาะสม จึงเป็นผลลัพธ์ที่ระบุได้ว่า ONPAS จะเป็นระบบให้คำตอบด้านโภชนาการที่เหมาะสมกับการควบคุมสุขภาพของบุคคลในชีวิตประจำวัน ได้แก่ จำนวนแคลลอรีที่จำเป็น รายการอาหารที่ถูกหลักโภชนาการ การบริหารจัดการน้ำหนักตัวที่ถูกต้อง และการกำหนดตารางรายการอาหารที่เหมาะสมต่อการรับประทานเพื่อการรับสารอาหารที่ถูกต้อง

Thesis Title	Ontology-based Nutrition Planning Assistance System for Health Control
Student	Mr. Dhomas Hatta Fudholi
Student ID	51060419
Degree	Master of Engineering
Program	Electronic Engineering
Year	2010
Thesis Advisor	Assoc. Prof. Dr. Ruttikorn Varakulsiripunth

ABSTRACT

The motivation to bring out the nutritional knowledgement into human's health control leads us to approach "Ontology-based Nutrition Planning Assistance System (ONPAS)". ONPAS aims to be one assistance system that can help the humanity to maintain their health by achieving the optimal nutritional status. Optimal nutritional status can be achieved by appropriate balancing the nutrient intake with the nutrient requirement. ONPAS will suggest daily menu as nutrient intake that balancing the daily calorie needs as nutrient requirement of person. Ontology as the foundation of the new web technology is used to construct ONPAS into a Semantic Web application. The fundamental of Ontology is the conceptualization of knowledge in the real world's domain. Ontology makes the data and information in ONPAS can be published in the common vocabulary and ready for inference. As an assistance system, ONPAS also provides recommendation support to give more proactive function. This recommendation support is dedicated for specifying and categorizing the suggested nutritional menu. Furthermore, fuzzy concept is brought in ontology to handle the fuzziness value for selecting the appropriate recommendation. As the result, ONPAS is implemented to provide optimal nutritional solution for personal daily health control; such as calorie need information, appropriate menu recommendation, suitable weight management program and appropriate menu table creation to support the optimal nutrient intake.

ACKNOWLEDGMENTS

Alhamdu lillahi rabbil 'alamin, all the praises and thanks only for Allah. *Wassalatu wassalamu `ala Rasulillah*, peace be upon the prophet Muhammad, his family, his companion and who follow him until the 'judgment day'.

First of all, I would like to express my deep gratitude to my thesis advisor Associate Professor Dr. Ruttikorn Varakulsiripunth for all of his advice in undergoing my master program at KMITL.

My sincere gratitude goes also to Dr. Noppadol Maneerat and Professor Dr. Yasushi Kato (Sendai National College of Technology) for giving me suggestion and encouragement to improve my knowledge and research skills.

I will also recognize and give a special thanks to Japan International Cooperation Agency (JICA) under AUN/Seed-Net Project for their financial supports and chances to contribute in the international conferences. It also goes to the staffs of the AUN/Seed-Net Project.

My appreciation will also goes to all lecturers in Department of Electronics of KMITL and also the staffs for their lectures and supports in completing my study.

Last but not least, my best regards and thanks is sent to my parents, my brothers and sisters, all of my best friends, all lab members, Thai colleagues, Indonesian colleagues and everyone that always support me.

Studio 1201 - Ladkrabang, April 2010

Dhomas Hatta Fudholi

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

Ontology is an explicit specification of a conceptualization [1]. It can be used to explain about a concept in some domains, properties and attributes, and roles. Ontology has been proved to be very useful in sharing concepts across applications in an unambiguous way and also conceived as a good mechanism to describe the shareable and common understanding in a domain concept. This statement is strengthened by Jason Cardoso in [2]. Figure 1.1 shows the reasons and percentage for the application usage of ontology according to Jason Cardoso's study.

Semantic Web is getting popular and performs information in the web with semantic (meaning). Ontology is one of Semantic Web development methodology. By using the ontology in the Semantic Web, users can take the advantages of the following two features: (i) data is published using common vocabulary and grammar; and (ii) the semantic description of data is preserved in ontologies and ready for inference [3].

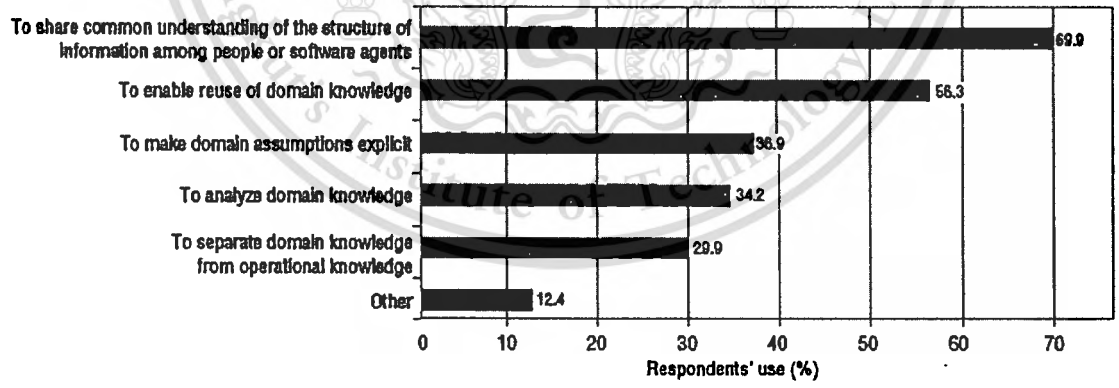


Figure 1.1 Reasons and percentage of ontology application [2].

An individual's nutritional status reflects the degree to which physiologic needs for nutrients are being met. Figure 1.2 shows optimal nutritional status achieved when the nutrient intake and the nutrient requirements is balanced. When adequate nutrients are consumed to support the physical situation of the human body and any increased metabolic demands, the person develops an optimal nutritional status. This status promotes growth and development, maintains general health, supports activities of daily living, and helps protect the body from disease and illness [4]. States of nutritional deficiency or excess occur when the nutrient intake is not balanced with specific requirements for optimal health [4].

Society should know what they have to eat in order to develop an optimal nutritional status by knowing the nutrient that they need. By taking the ontology method in the Semantic Web application and also the nutrient domain with its issue, an ontology based system is developed in this research. The system name is Ontology-based Nutrition Planning Assistance System (ONPAS). ONPAS is developed to assist the society to get the daily nutritional menu which is suitable for them. By giving the suitable daily menu with a balanced nutrient, ONPAS can be one way to help the society to maintain their healthiness.

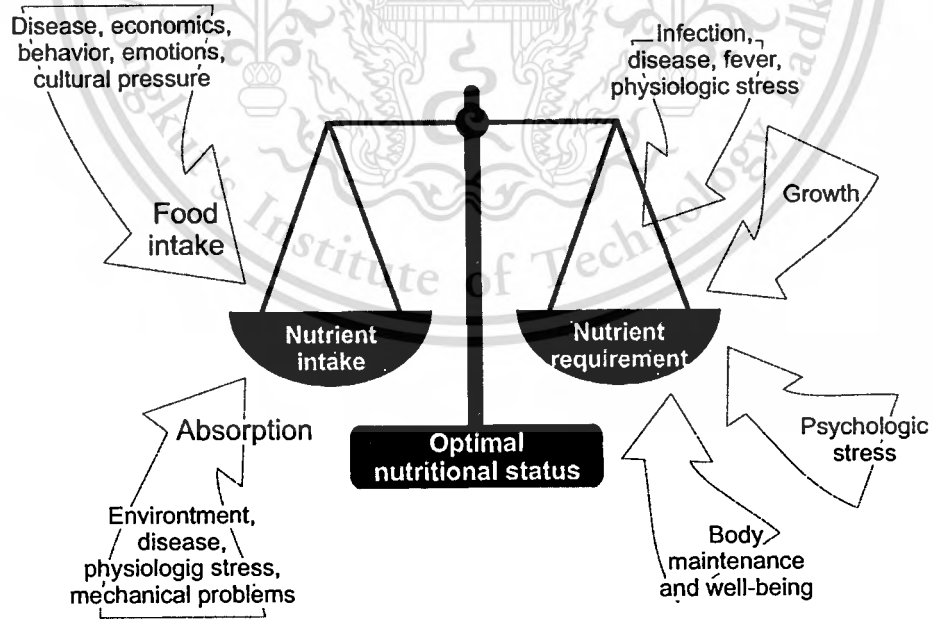


Figure 1.2 Optimal nutritional status viewed as a balance between nutrient intake and nutrient requirements [4].

A system that was integrated with recommendation support had shown the ability to provide a more personalized and proactive functions [5]. ONPAS also gives a recommendation support by adding preference into the suggestion. This recommendation support uses a fuzzy ontology concept. Fuzzy ontology concept is used to bring out the vague and imprecise value handling in the ordinary ontology concept. Fuzzy ontology concept is also used to make the recommendation to be more specific.

1.2 Objective and Scope of Research

The goal of this research is to propose and build the prototype of an Ontology-based Nutrition Planning Assistance System (ONPAS) as a health control system. This system takes the nutrition domain and the problem mentioned in the background and motivation. ONPAS is built using ontology method and it is built as a Semantic Web application. ONPAS Semantic Web is developed to be able to support the following items:

- Provides daily calorie needs information for a user. The user will be requested to fill some personal data information.
- Provides menu suggestion which is daily menu to the user regarding to the user's daily calorie needs.
- Provides menu recommendation to the user if the user prefers to get recommendation from their menu suggestion.
- Provides weight management program for user by balancing the daily calorie needs. This program will be available when the weight of a user is in abnormal condition.
- Provides nutritional menu creation feature. User can make his/her own nutritional menu regarding to their daily calorie needs.
- Provides a feature to make a nutritional menu table as a collection of daily menu for maximum 7 days in a table.

The food database that is used in this system is limited to the Thai food database as the case study for prototype of ONPAS. The information of the food is taken from *ASEAN Food Composition Tables* [6] and *Thai Food Composition Tables* [7]. These two composition tables contains of nutrient detail of Thai food and dish.

1.3 Research Methodology

The methodology to do this research is regarded to the procedure shown in Figure 1.2. Accordingly, there are 4 steps in developing ONPAS Semantic Web application as shown in Figure 1.3; those are analysis, design, development and testing.

Analysis is the first step to run this research. The domain analysis and the requirement analysis are covered in this step. As mentioned in the previous sub chapter, ONPAS will take nutrition domain and it is built to assist the society especially in getting appropriate daily menu.

The second step is design. In this step, ONPAS components are designed starting from the system architecture, ontology design, and finally the algorithm of each function in ONPAS.

The third step is development process which comes after designing all off the ONPAS components. Applying in the Semantic Web by using programming languages and also supporting framework will do this development process.

The last step is testing. This step is needed to test ONPAS and fix the problem and bug which are found.

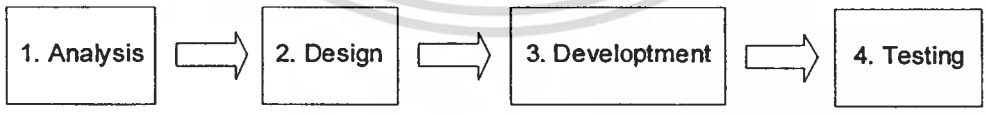


Figure 1.3 Research methodology.

1.4 Thesis Outline

This thesis is organized as follows:

Chapter 2 gives all of literature review which is used to run the research. Daily nutrient needs are one of the most importance literatures. This chapter also explains about Semantic Web and the ontology. These two things are the core of the system development method. Another thing that is necessary to be explained is the Jena ontology framework and the concept of fuzzy logic and fuzzy ontology. All of the review will give most of the knowledge to develop ONPAS.

Chapter 3 shows the system design and development. This chapter covers the concept and application design of ONPAS. Start from the general system architecture, ontology design, until the Semantic Web application design. There is also the detail information of the program flow. The ONPAS Semantic Web will is categorized into 4 modules. The modules are Personal Data Calculation modules, Menu Suggestion and Recommendation modules, Menu and Dish Creation modules and User and Admin module. This chapter explains the details of each module. The description, the function and the development design of the module is described.

Chapter 4 contains of the ONPAS Semantic Web application implementation in general. This chapter also shows the detail implementation in each module. Start from the use of the ontology framework until the final result of each function. Finally, there is a system evaluation which includes evaluation in research objective, and evaluation in ONPAS Semantic Web.

Chapter 5 is conclusions of the research and future work that includes the suggestions to develop this system to be a better system and more useable system.

Chapter 2

Literature Review

2.1 Daily Nutrition Need

2.1.1 Energy Requirement

The energy requirement of an individual, in a state of desirable equilibrium, is equal to the energy expenditure. In some clinical situations, where an improvement in nutritional status may be advisable, the energy requirement may be set at a higher level than the energy expenditure in order to produce, temporarily, a positive energy balance [8]. One of the methodologies to measure daily energy expenditure is by multiplying Basal Metabolic Rate (*BMR*) with Activity Factor (*AF*) which will be dependent on the degree and duration of physical activity or Physical Activity Level (*PAL*) [8]. The daily energy expenditure will be represented as Total Energy Expenditure (*TEE*) and (2.1) is the equation to calculate *TEE*.

$$TEE = BMR \times AF \tag{2.1}$$

TEE : Total Energy Expenditure (kcal/day),

BMR : Basal Metabolic Rate (kcal/day),

AF : Activity Factor.

The report from the 1985 Food and Agriculture Organization (FAO) / World Health Organization (WHO) / United Nations University (UNU) expert consultation used a set of equations derived mostly from studies in Western Europe and North America which is Schofield equation to calculate *BMR* based on human's age [9]. The Basal Metabolic Rate is the minimum rate at which the body uses energy at complete rest. The *BMR* equations described in Table 2.1 were derived by using linear regression models. The statistical data of calorie need versus human's weight was randomly collected and plotted in accordance with the range of their ages. The least-mean-square-error and curve fitting technique were introduced in order to define the most appropriate linear equation that can represent the relation between *BMR* versus human's weight. The equation to be

defined is $BMR = aW \pm b$ where W is the variable of human's weight, a is slope and b is BMR-intercept. Curve fitting using first degree polynomial was processed by which the equation $BMR = aW \pm b$ would best fit to given data point. As the conclusion, the optimal values of a and b were obtained related to the range of human's ages and gender as shown in Table 2.1. For instance, a 20 years old man will use the fourth equation in the Table 2.1 which is $(15.057 \times W) + 692.2$ to calculate his BMR and change the W variable with his weight in kilogram.

Activity Factor (AF) is a factor used in the equation to determine energy needs that takes into consideration. In every range of age activity factor would be categorized in different level and value. [8] and [9] will be the reference to describe the activity factor in each age category.

The activity factor for children and adolescents was studied with doubly-labeled water, heart rate monitoring and time allocation. Table 2.2 shows different AF allocation for different age in children and adolescents with different level of activities. The level of activities such as light, moderate and heavy is depended on their physical lifestyle.

Table 2.1 Equations for Estimating BMR from Body Weight (W) in kilogram

Age (Years)		*BMR*: kcal/day
Male	< 3	(59.512 × W) – 30.4
	3–9.9	(22.706 × W) + 504.3
	10–17.9	(17.686 × W)+ 658.2
	18–29.9	(15.057 × W) + 692.2
	30–59.9	(11.472 × W) + 873.1
	≥ 60	(11.711 × W) + 587.7
Female	< 3	(58.317 × W) – 31.1
	3–9.9	(20.315 × W) + 485.9
	10–17.9	(13.384 × W) + 692.6
	18–29.9	(14.818 × W) + 486.6
	30–59.9	(8.126 × W) + 845.6
	≥ 60	(9.082 × W) + 658.5

Children and adolescents with light physical lifestyles are children and adolescents who every day spend several hours at school or in sedentary occupations; do not practise physical sports regularly; generally use motor vehicles for transportation; and spend most leisure time in activities that require little physical effort, such as watching television, reading, using computers or playing without much body displacement.

Children and adolescents with heavy lifestyles that are more active than average, are children and adolescents who every day walk long distances or use bicycles for transportation; engage in high energy-demanding occupations, or perform high energy-demanding chores for several hours each day; and/or practise sports or exercise that demand a high level of physical effort for several hours, several days of the week.

Children and adolescents with habitual physical activity that is more strenuous than the examples given for a light lifestyle, but not as demanding as the examples for vigorous lifestyle, would qualify in the category of average or moderate physically active lifestyles.

As an example, 10 years old boy that has a light physical activity will be given the *AF* value that is equal to 1.54. This *AF* value and the *BMR* value resulted from the equation shown in Table 2.1 are then used to calculate the *TEE* of the boy by using equation (2.1).

Table 2.2 Activity Factor of Children and Adolescents

Age (years)	Sex	Light	Moderate	Heavy
1-5	Male, Female	1.44	1.61	-
6-13	Male	1.54	1.75	1.96
14-18	Male	1.60	1.82	2.04
6-13	Female	1.48	1.68	1.88
14-18	Female	1.46	1.66	1.86

The FAO/WHO/UNU Expert Consultation suggested the average daily physical activity of adults whose occupational work is classified as light, moderate, and heavy. It is shown in Table 2.3.

Adult people with light activity are people who have occupations that do not demand much physical effort, are not required to walk long distances, generally use motor vehicles for transportation, do not participate in sports regularly, and spend most of their leisure time sitting or standing, with little body displacement (e.g. talking, reading, watching television, listening to the radio, using computers).

People with moderate lifestyle are people that occupations that are not strenuous in terms of energy demands, but involve more energy expenditure than that described for light or sedentary lifestyles. Alternatively, they can be people with sedentary occupations who regularly spend a certain amount of time in moderate to vigorous physical activities, during either the obligatory or the discretionary part of their daily routine.

People with heavy lifestyle are people that engage regularly in strenuous work or in strenuous leisure activities for several hours. Examples are women with non-sedentary occupations who swim or dance an average of two hours each day, or non-mechanized agricultural labourers who work with a machete, hoe or axe for several hours daily and walk long distances over rugged terrains, often carrying heavy loads.

As an example, 25 years old man that has a moderate physical activity will be given the *AF* value that is equal to 1.78. This *AF* value and the *BMR* value resulted from the equation shown in Table 2.1 are then used to calculate the *TEE* of the man by using equation (2.1).

Table 2.3 Activity Factor of Adult (18 – 64 years)

	Light	**Moderate**	**Heavy**
Male	1.55	1.78	2.10
Female	1.56	1.64	1.82

The *AF* for older individuals was studied form older people in the mean ages of 64-74 years. Goran&Poehlman (1992), Roberts et al (1993), and Pannemans&Westerterp (1995) study using the doubly labeled water method result the average *AF* for older individuals which is shown in Table 2.4.

Table 2.4 Activity Factor of Older Individuals

Male	Female
1.77	1.71

2.1.2 Population Nutrient Intake Goals

Population nutrient intake goals represent the population average intake that is judged to be consistent with the maintenance of health in a population. Health is marked as a low prevalence of diet-related diseases in the population. Seldom there is a single "best value" for such a goal. Instead, consistent with the concept of a safe range of nutrient intakes for individuals, there is often a range of population averages that would be consistent with the maintenance of health. If existing population averages fall outside this range, or trends in intake suggest that the population average will move outside the range, health concerns are likely to arise [4].

The population nutrient intake goals for consideration establish dietary recommendations for the prevention of diet-related chronic diseases. The recommendations are presented in Table 2.5 and created by the WHO [10]. These nutrient intake recommendations are expressed in numerical terms and focus in the energy-supplying macronutrients. The proportion of calorie and the weight of each macronutrient are 4 kilocalories for 1 gram carbohydrate, 4 kilocalories for 1 gram protein and 9 kilocalories for 1 gram of fat [4].

Table 2.5 Ranges of Population Nutrient Intake Goals

Dietary Factor	Goal (% of total energy)
Total Fat	15-30%
Total Carbohydrate	55-75%
Protein	10-15%

2.1.3 Body Mass Index (*BMI*)

The Body Mass Index (*BMI*) is the body composition according to the relationship between weight and height [11]. The body composition number from *BMI* shows the body fatness. *BMI* for adult can be evaluated by equation (2.2).

$$BMI = \frac{weight(kg)}{height^2(m^2)} \tag{2.2}$$

BMI status of a person basically divided as underweight, normal and overweight. Table 2.6 gives the detail classification of person's weight based on *BMI* calculation.

Table 2.6 Classification of Weight based on *BMI* [10]

Classification	*BMI* (kg/m^2)
Underweight	< 18.5
Normal	18.5 – 24.9
Overweight	≥ 25.0
- Pre-obese	25.0 – 29.9
- Obese class I	30.0 – 34.9
- Obese Class II	35.0 – 39.9
- Obese Class III	≥ 40.0

By knowing the *BMI* status, a person can do some weight management by modifying the total of food that is consumed. People can gain or lose 0.5-1 kg of body weight in a week by increasing or decreasing 500-1000 kilocalories in a day from the total daily calories [12]. The range of body weight balance is the safest but slow way to manage body weight from food consumption. Extreme diet can effect in one's health to nutrient deficiencies and psychological disturbance [13].

2.2 Semantic Web

2.2.1 Web Evolution

The first Web (assumed as Web 1.0) has changed the world. It connects people in different places and share many kinds of information around the world.

The second wave of the Web, Web 2.0, has more active than the first generation of Web. It is not only share the information but give the ability to interact more. The interaction in the Web world is not only between the user and the information itself, but also the user and other user. This growth enables social network, blogging and also web services in the Web technology.

Web 3.0, the Semantic Web, is the third major wave of the Web. In this era, real world knowledge is mapped in the Web. By mapping the real world knowledge, the 3rd generation of the web shows more semantic (meaningful) information rather than just ordinary information based on syntax.

Figure 2.1 shows the evolution of the Web technology. We can see the clear progression of technology from the Personal Computing era, to the first Web 1.0 of pages and documents, to the Web 2.0 era of social networking, and to the Web 3.0 era of the Semantic Web and data networking.

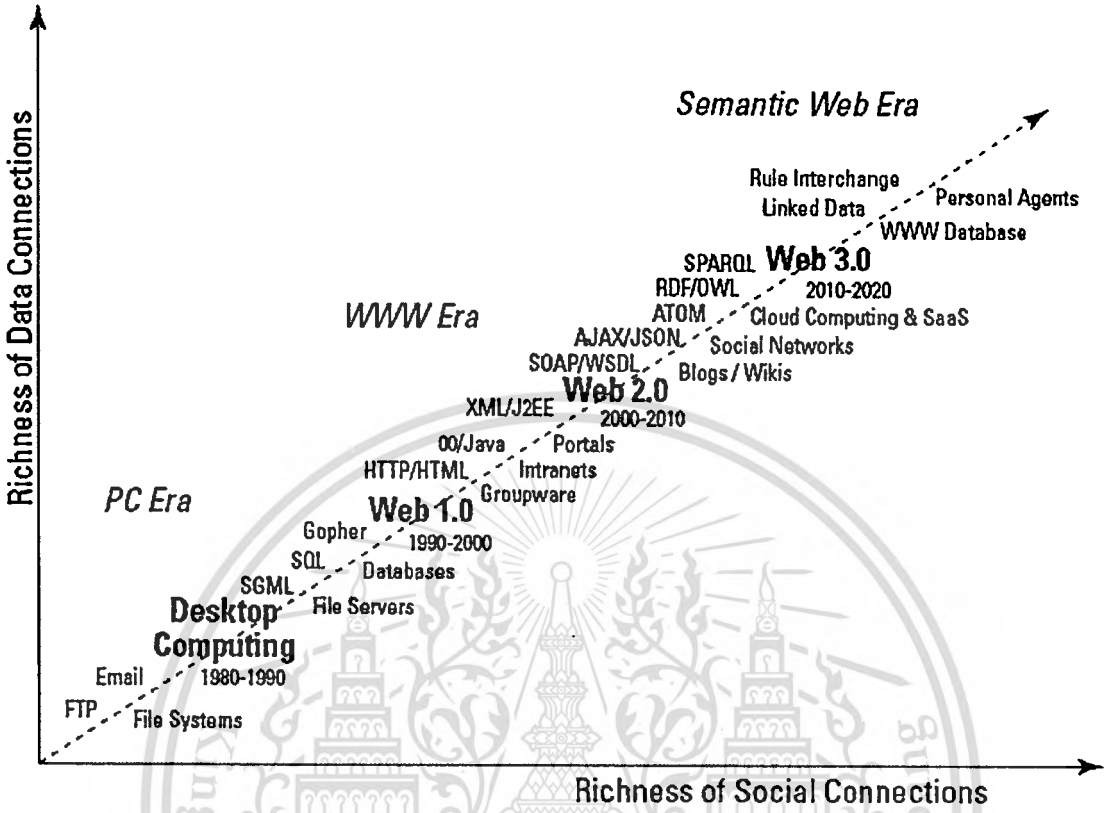


Figure 2.1 The evolution of Web technology [14].

2.2.2 Semantic Web Definition

A definition for the Semantic Web begins with defining *semantic*. That is what John H. et al firstly said to define of the Semantic Web on their book [15]. They give a great explanation regarding to the definition of Semantic Web that is stated, in brief, below.

Semantic simply means *meaning*. Meaning enables a more effective use of the data. Meaning is usually unavailable in the information. The meaning of information can be achieved by giving more complex programming instruction. Semantics give a keyword symbol useful meaning through the establishment of relationships. By adding relationships between information it will expose semantics. The formal standards of grammar and language help incorporate meaning, or semantics. As this contextual web of grammar rules and language terms expands through relationships, the semantics are further enriched.

The Semantic Web is simply a web of data described and linked in ways to establish context or semantics that adhere to defined grammar and language constructs. The Semantic Web addresses semantics through standardized connections to related information. This includes labeling data unique and addressable. The flexibility of a web form enables connections to all the necessary information, including logic rules. Semantic Web applications typically use many ontologies, each chosen for a required information area.

The fundamental building block of the Semantic Web is a statement. In the Semantic Web, statements describe concepts, logic, restrictions, and individuals. The statements share the same standards to enable sharing and integration, which take advantage of the semantics.

2.3 Ontology

2.3.1 Ontology Definition

Asunción Gómez-Pérez et al. collects and describe many definitions of ontology in [16]. One of the first definitions of ontology was given by Neches and colleagues (1991) [17]. They define an ontology as follows:

> *An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary.*

This descriptive definition tells us what to do to build an ontology, and gives us some vague guidelines: this definition identifies basic terms and relations between terms, identifies rules to combine terms, and provides the definitions of such terms and relations.

A few years later, Gruber (1993) [1] defined an ontology as follows:

An ontology is an explicit specification of a conceptualization.

The definition stated by Gruber became the most quoted in literature and by the ontology community. Based on Gruber's definition, many definitions of ontology were proposed. Borst (1997) [18] modified slightly Gruber's definition as follows:

Ontologies are defined as a formal specification of a shared conceptualization.

Gruber's and Borst's definitions have been merged and explained by Studer and colleagues (1998) [19] as follows:

An ontology is a formal, explicit specification of a shared conceptualization. Conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group.

By knowing few descriptions about ontology above, then we can conclude that ontology is a knowledge based mapping from a real world into a specific specification which shows the concept, relationship and interaction between the concepts in the domain.

2.3.2 Component of Ontology

Gruber in [1] identified five kinds of components: classes, relations, functions, formal axioms and instances.

- **Classes**

Classes represent concepts, which are taken in a broad sense. Classes in the ontology are usually organized in taxonomies through which inheritance mechanisms can be applied. For instance, we can represent a taxonomy of shopping places (mall, minimarket, etc.) or clothing (suit, trouser, etc.). Classes can represent abstract concepts (intentions, feelings, etc.) or specific concepts (people, building, etc.).

- **Relations**

Relations represent a type of association between concepts in the domain. They are formally defined as any subset of a product of n sets, that is: $R \subset C1 \times C2 \times ... \times Cn$. Ontologies usually contain binary relations. The first argument is known as the domain of the relation, and the second argument is the range. For instance, the binary relation *Subclass-Of* is used for building the class taxonomy. The example of the taxonomy is *Squid* is a subclass of *Water Animal*.

Binary relations are sometimes used to express concept attributes (aka slots). Attributes are usually distinguished from relations because their range is a datatype, such as string, integer, etc., while the range of relations is a concept.

- **Functions**

Functions are a special case of relations in which the n-th element of the relation is unique for the n-1 preceding elements. This is usually expressed as: $F: C1 \times C2 \times ... \times Cn\text{-}1 \rightarrow Cn$.

- **Formal Axioms**

Formal axioms serve to model sentences that are always true. They are normally used to represent knowledge that cannot be formally defined by the other components. In addition, formal axioms are used to verify the consistency of the ontology itself or the consistency of the knowledge stored in a knowledge base. Formal axioms are very useful to infer new knowledge.

Formal axioms can also be domain independent. We are now going to redefine the relation connects in a domain independent way.

- Instance

Instances are used to represent elements or individuals in an ontology.

2.3.3 Ontology Language

Ontology need to be expressed in the real notation. Few components that construct ontology are XML (eXtensible Markup Language), RDF (Resource Description Framework), RDFS (Resource Description Framework Schema) and OWL (Web Ontology Language). SPARQL will also be explained as an RDF query language.

XML is the basis of ontology languages. Ontology languages are stored as XML form. The most used ontology language is OWL and RDFS. Figure 2.2 shows the percentage of correspondent that used some ontology languages. This survey was done by Jorge Cardoso in his paper [2].

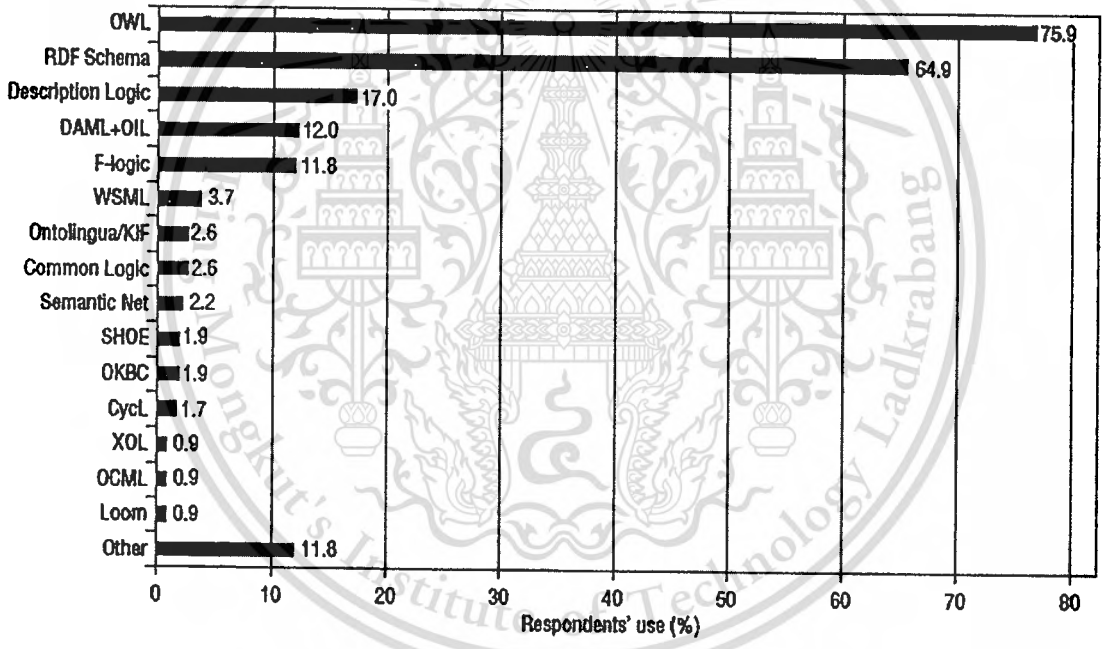


Figure 2.2 Ontology languages currently used [2].

- **XML**

XML is a collection of data. It gives a flexible syntax to perform the structured document. XML basically used to store and transport the data [20]. XML become the basic of declaring ontology component since it can store the data and has flexibility in declaring the data. The following script is an example of an XML data:

```
<?xml version="1.0"?>
<note>
    <to>Mark</to>
    <from>Lisa</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

XML constructed by binding the information using the <tags>. The <tags> always be in pair, so there will be an open <tags> and closing </tags>.

- **RDF**

RDF represents the information in Semantic Web as a set of assertions called *statement*. Statement made up of three parts: subject, predicate, and object [15]. Because of the three parts, statement can be referred as *triples*. The three elements of a statement have meanings that are analogous to their meanings in normal English grammar. The subject of a statement is the thing that statement describes, and the predicate describes a relationship between the subject and the object.

For instance, we have these sequences of sentence:

`Mark has wife named Lisa. Mark works in O Company.`

Figure 2.3 is graphical representation of above information. The subject and the object are represented as nodes. The predicates are represented as edges. The nodes of an RDF graph are the subjects and the objects of the statements that make up the graph. There are two kinds of nodes: resources and literals. Literals represent concrete data values like numbers or strings and cannot be the subjects of statements, only the objects. Resources, in contrast, represent everything else, and they can be either subjects or objects. In RDF, resources can represent anything that can be named. A resource is, in fact, nothing but a name that represents an object, act, or concept. Resource names take the form of Internationalized Resource Identifiers (IRI).

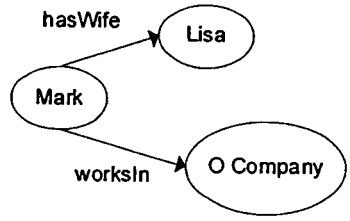


Figure 2.3 Example of graph representation from statements.

IRI provides a foundation for a data-sharing infrastructure because they all exist within a single universal namespace. This means that every statement with a named resource as its subject unambiguously describes that particular resource, regardless of where the statement is asserted. By using IRI to name the resource, the resource will be unique.

Prefix is defined as a variable which contains some IRI. For instance, the following example shows `rdf` as prefix from an IRI `"http://www.w3.org/1999/02/22-rdf-syntax-ns#"`:

```
rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

Predicates, also called *properties*, represent the connections between resources; predicates are themselves resources. Predicates also are represented as IRIs.

One special type of predicate defined by RDF is `type`. The `rdf:type` predicate is used to group resources together. In Figure 2.4, the resource Mark is associated via the `rdf:type` predicate with a resource that represents the notion of a Person.

The RDF/XML is the XML representation for RDF. The following XML is an example of RDF/XML representation from Figure 2.3 :

```
<rdf:RDF
xmlns:people="http://example.com/people#"
xmlns:company="http://example.com/company/company#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ext="http://example.com/ont/extension#">
<rdf:Description rdf:about="http://example.com/people#Mark">
<ext:hasWife rdf:resource="http://example.com/people#Lisa"/>
</rdf:Description>
<rdf:Description rdf:about="http://example.com/people#Mark">
<ext:worksIn rdf:resource="http://example.com/company#O Company"/>
</rdf:Description>
</rdf:RDF>
```

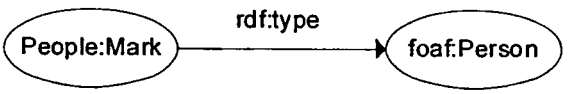


Figure 2.4 Mark is designated as being the type of `foaf:Person`.

- **RDFS**

RDF Schema, RDF's vocabulary description language, is a semantic extension of RDF [21]. It provides mechanisms for describing groups of related resources and the relationships between these resources. Resources are used to determine characteristics of other resources, such as the domains and ranges of properties.

Two tables below present an overview of the vocabulary of RDF and RDF Schema. Table 2.7 shows the RDF classes and Table 2.8 shows the RDF properties.

Table 2.7 RDF Classes

Class name	Comment
rdfs:Resource	The class resource, everything.
rdfs:Literal	The class of literal values, e.g. textual strings and
rdf:XMLLiteral	integers.
rdfs:Class	The class of XML literals values.
rdf:Property	The class of classes.
rdfs:Datatype	The class of RDF properties.
rdf:Statement	The class of RDF datatypes.
rdf:Bag	The class of RDF statements.
rdf:Seq	The class of unordered containers.
rdf:Alt	The class of ordered containers.
rdfs:Container	The class of containers of alternatives.
rdfs:ContainerMembershipProperty	The class of RDF containers.
	The class of container membership properties, rdf:_1,
rdf:List	rdf:_2, ..., all of which are sub-properties of
	'member'.
	The class of RDF Lists.

Table 2.8 RDF Properties

Property Name	Comment	Domain	Range
rdf:type	The subject is an instance of a class.	rdfs:Resource	rdfs:Class
rdfs:subClassOf	The subject is a subclass of a class.	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	The subject is a subproperty of a property.	rdf:Property	rdf:Property
rdfs:domain	A domain of the subject property.	rdf:Property	rdfs:Class
rdfs:range	A range of the subject property.	rdf:Property	rdfs:Class
rdfs:label	A human-readable name for the subject.	rdfs:Resource	rdfs:Literal
rdfs:comment	A description of the subject resource.	rdfs:Resource	rdfs:Literal
rdfs:member	A member of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:first	The first item in the subject RDF list.	rdf:List	rdfs:Resource
rdf:rest	The rest of the subject RDF list after the first item.	rdf:List	rdf:List
rdfs:seeAlso	Further information about the subject resource.	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	The definition of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:value	Idiomatic property used for structured values (see the RDF Primer for an example of its usage).	rdfs:Resource	rdfs:Resource
rdf:subject	The subject of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:predicate	The predicate of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:object	The object of the subject RDF statement.	rdf:Statement	rdfs:Resource

- **OWL**

OWL is a vocabulary extension of RDF. OWL assigns an additional meaning to certain RDF triples. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to represent machine interpretable content on the Web [22].

OWL uses the ability of RDF to define the class. OWL is also uses the ability from RDFS to make the hierarchy of class. OWL simplified the way it define the class. OWL has extension in declaring the class. Class in OWL can be designed as a combination of logical term such as *unionOf*, *complementOf* and *intersectionOf* from other class.

OWL can also define a property and provides the property with domain and range value. Property in OWL can be a data type property like numbers and string, or can be an object property. The property in OWL can be assigned as a symmetric, transitive, functional or inverse from other property. OWL can also provides some value restriction to the property.

Individual in OWL can be expressed as the member of a class. Individual in OWL derived clearly its property and its property value.

The ability of the OWL is beyond RDF. In example, by using RDFS we can:

- Declare classes, such as 'country', 'people' and 'car.
- Declare that 'teacher' is the sub-class of 'people'.
- Declare that 'Indonesia' and 'Thailand' is the member of 'country' class.
- Declare 'has nationality' as a property that connects 'people' class as domain to 'country' class as range.
- Declare 'age' as a property with 'people' as domain class and integer data type as the range.
- Declare 'Mark' as a member of 'Indonesia' class and has 'age' value '25'.

By using OWL, we also can:

- Declare that 'country' and 'people' is a disjoint class.
- Declare that 'Indonesia' and 'Thailand' as a different individual.
- Declare 'has citizen' as the inverse of 'has nationality' property.
- Declare 'no country' class for any 'people' that doesn't has value in the 'has nationality' property.

OWL is also declared as XML document. OWL is easier to use since it simplifies RDF. The following scripts result the same value:

Written in OWL :

```
<owl:class rdf:ID="Country"/>
```

Written in RDF/XML:

```
<rdf: Description rdf:about="#Country">
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
<rdf: Description>
```

- **SPARQL**

SPARQL is a query language for RDF. SPARQL can be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports extensible value testing and constraining queries by source RDF graph. The results of SPARQL queries can be results sets or RDF graphs [23].

The following example is SPARQL language which query any individual (?x) which has name ?name:

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE
   { ?x foaf:name ?name }
```

2.3.4 Protégé

Protégé is software which is developed in Stanford Medical Informatics [24]. It is built using Java programming language. It supports OWL (Web Ontology Language) that intends to interpret the web information into machine-readable content with semantics. Protégé has GUI (Graphical User Interface) that makes it easier to use. Figure 2.5 is the GUI of Protégé.

Protégé is software which is used to build ontology. In brief, Protégé can:

- Create ontology in OWL, RDF and XML.
- Define classes, properties, individual, including its axioms and functions.
- Create form to input the data.
- Draw ontology graph.
- Define knowledge base rules and do the inference by adding some plug-in.
- Built-in SPARQL tab to query the data.

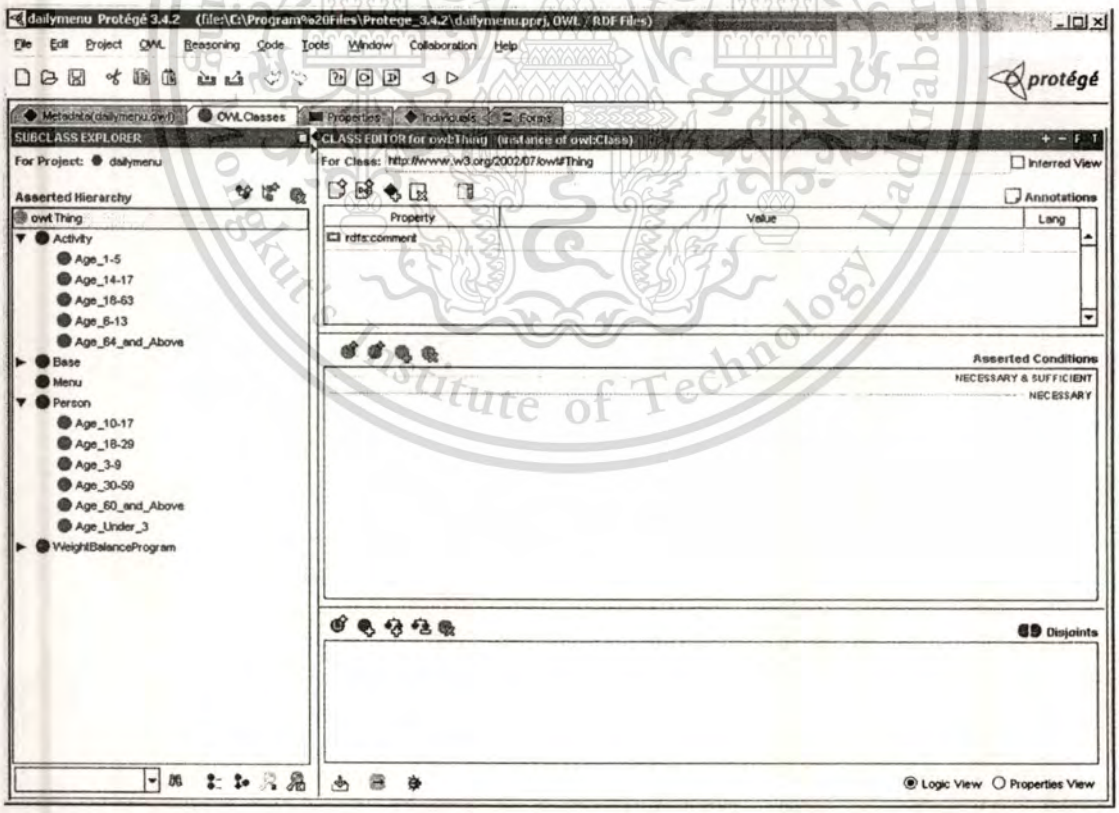


Figure 2.5 Protégé GUI.

2.4 Jena – Ontology Framework

Jena is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. It translates the statements and constructs of the Semantic Web into useful programming artifacts such as Java classes, objects, methods, and attributes [15].

The Jena Framework maintains a consistent treatment of the Semantic Web through its use of Java classes and variables. Table 2.9 illustrates the main Java classes and interfaces and connects them to their corresponding Semantic Web component.

Table 2.9 Semantic Web and Jena Framework Comparison [15]

Artifact	Semantic web	Jena Java Class	Notes
Subject, predicate, object	URI	Resource, Property	A resource can be a subject, object, or predicate.
Statement	Statement	Statement	Special consideration for reified statements.
Data	Ontology and instance data	Graph and Model	Graphs are a basic building block for models. They both may contain ontology and instance data.
Query and results	SPARQL and Semantic Web data	Query and ResultSet	Analogous to relational databases.
Reasoner	Reasoner	Reasoner	Allow multiple internal and external reasoners.
Rules	SWRL (Semantic Web Rule Language)	Reasoner	Rule support determined by specific reasoner.
Event notification	Not applicable	ObjectListener	Enable event-driven processing.

The Jena Framework employs the following major Java classes [15]:

- `Resource` – A class representing an element contained within a statement such as a subject, predicate, or objects. This is analogous to an RDF resource.
- `Statement` – A Semantic Web triple containing a subject, predicate, and object. The Statement class allows simple interrogation of its containing components.
- `Graph` – Basic method for maintaining Semantic Web data. A graph allows basic add, delete, find, and contain operations.
- `Model` – A model builds on the basic graph to offer rich interactions with Semantic Web data. Your applications read, write, reason, and query Semantic Web data through access to the Jena model. The model forms the actual knowledgebase. There exist several types of model classes based on the application's needs for expressivity and reasoning. These classes include `Model` for basic RDF and `OntModel` for OWL.
- `Query` and `ResultSet` – The query employs SPARQL with results returned as `ResultSet`. Your application iterates through the `ResultSet` matching on the variables used in the query.
- `Reasoner` – Contains the reasoner processing via either internal or external reasoning. Internal refers to the framework's capabilities itself, whereas external enables third-party reasoners access to the knowledgebase.

Jena framework contains API (Application Programming Interface) which is used to build the ontology model and populate it. The API includes all six major classes that mentioned before; `Resource`, `Statement`, `Graph`, `Model`, `Query` and `ResultSet`, and `Reasoner`.

2.4.1 Building an Ontology Model Using Jena's RDF API

Jena provides the RDF API that is works under `Model` class. The following examples show RDF API functions that can be used to create and populating ontology.

a. Create an empty model. The script below is an example of creating an empty model :

```
Model model = ModelFactory.createDefaultModel();
String ns = new String("http://www.example.com/example#");
```

A `Model` named `model` is created. The second line of above script is used to define namespace for the model, since the ontology is identified by a namespace. The namespace itself is an IRI.

b. Read model from external source (file/database). Besides creating an empty model, Jena has a support to read an external model from file or database. The following script show an example how to read a model from external source :

From file

```
String fileURI = "file:myRDF.rdf"; //file location
Model modelFromFile = ModelFactory.createDefaultModel();
modelFromFile.read(fileURI);
```

From database (using MySQL database)

```
try {
Class.forName("com.mysql.jdbc.Driver"); // Load MySQL driver
}
catch(ClassNotFoundException e) { ... }

IDBConnection conn = new DBConnection("jdbc:mysql://localhost/jenadb",
"user", "pass", "MySQL");
ModelMaker maker = ModelFactory.createModelRDBMaker(conn);

Model dbModel = maker.openModel("http://www.example.com/example",true);
```

c. Create resource. The following script is an example of creating two resources, `mark` and `lisa`:

```
Resource mark = model.createResource(ns + "Mark");
Resource lisa = model.createResource(ns + "Lisa");
```

d. Create property. For instance, the `hasWife` and `hasHusband` properties can be created by the following example script:

```
Property hasWife = model.createProperty(ns, "hasWife");
Property hasHusband = model.createProperty(ns, "hasHusband");
```

e. Associate resources. There are two ways of associating resources through properties; direct and with a `Statement`. The following script is an example of creating an association of `lisa` and `mark`:

Associate directly

```
mark.addProperty(hasWife, lisa);
```

Associate with a `Statement`

```
Statement husbandStmt = model.createStatement(lisa, hasHusband, mark);
model.add(husbandStmt);
```

f. List statements. Jena has ability to list statements as triples. The following script is an example of iterating statements:

```
StmtIterator iter = dbModel.listStatements();
while(iter.hasNext()) {
       Statement stmt = (Statement)iter.next();
       System.out.println(stmt.asTriple().toString());
       }
```

2.4.2 Building an Ontology Model Using Jena's OWL API

OWL is also fully supported by Jena. The OWL API in Jena has some differences with RDF API since the OWL is the extension of RDF.

OWL API in Jena defines `Resources` in more specific term as a `Class` or an `Individual`. Classes can be designed in a subclass and superclass relations. An equivalency or disjointness is also can be assigned to the classes. Properties are defined more specific, as datatype properties or as object properties. Constrains of property can be assigned too, such as `AllValueFrom` and Cardinality restriction.

The following task can be done by using OWL API in Jena:

a. Create an empty ontology model. The script below is an example to create an empty model using OWL API.

```
OntModel ontModel = ModelFactory.createOntologyModel();
String ns = new String("http://www.example.com/onto1#");
String baseURI = new String("http://www.example.com/onto1");
Ontology onto = ontModel.createOntology(baseURI);
```

b. Create classes. The following script creates three classes. They are 'Person', 'MalePerson' and 'FemalePerson':

```
OntClass person = ontModel.createClass(ns + "Person");
OntClass malePerson = ontModel.createClass(ns + "MalePerson");
OntClass femalePerson = ontModel.createClass(ns + "FemalePerson");
```

c. Adding subclass relation. As mentioned before, OWL API can make subclass hierarchy to the classes. 'MalePerson' class and 'FemalePerson' class is assigned as subclasses of 'Person' by using the script below:

```
person.addSubClass(malePerson);
person.addSubClass(femalePerson);
```

d. Adding class restriction. Since a person cannot have two genders, the female person must be disjoint with the male person. The following script add disjoint class restriction to 'MalePerson' and 'FemalePerson' classes:

```
malePerson.addDisjointWith(femalePerson);
femalePerson.addDisjointWith(malePerson);
```

e. Create datatype properties. Datatype property 'hasAge' is created by the script below:

```
DatatypeProperty hasAge = ontModel.createDatatypeProperty(ns+"hasAge");
hasAge.setDomain(person);
hasAge.setRange(XSD.integer);
```

'hasAge' property has domain 'Person' class and it has range an integer data type.

f. Create object properties. The following script is used to create 'hasWife' and 'hasHusband' object properties and assign them with domain and range from 'Person' class.

```
ObjectProperty hasWife = ontModel.createObjectProperty(ns + "hasWife");
hasWife.setDomain(person);
hasWife.setRange(person);

ObjectProperty hasHusband = ontModel.createObjectProperty(ns +
"hasHusband");
hasHusband.setDomain(person);
hasHusband.setRange(person);
```

g. Create properties restriction. The script below is an example of `AllValueFrom` restriction for 'hasWife' object property, so that 'hasWife' object property can only have values from 'femalePerson' class:

```
AllValuesFromRestriction onlyFemalePerson =
ontModel.createAllValuesFromRestriction(null, hasWife, femalePerson);
```

h. Create `Individual`. The script below is an example of creating `Individual`:

```
Individual mark = malePerson.createIndividual(ns + "Mark");
Individual lisa = femalePerson.createIndividual(ns + "Lisa");
```

i. Create Literal and Statement. Literal is an exact valued resource. The following script is creating a literal then creating statements:

```
Literal age25 = ontModel.createTypedLiteral("25", XSDDatatype.XSDint);

Statement markIs25 = ontModel.createStatement(mark, hasAge, age25);
ontModel.add(markIs25);

Statement wife = ontModel.createStatement(mark, hasWife, lisa);
Statement husband = ontModel.createStatement(lisa, hasHusband, mark);
ontModel.add(wife);
ontModel.add(husband);
```

2.4.3 Jena's Reasoning

Jena has a built in reasoner. Reasoner or reasoning engine is a system that infers new information based on the content of the knowledgebase. To interpret semantics and realize the enriched information, a knowledgebase needs to apply an inference component. Jena has a rule based reasoner. The rule that is used in Jena reasoned is Jena rules.

A Jena rule contains a list of premises (the *if* clause) and a list of conclusions (the *then* clause). Each rules can optionally have rule name. For instance, the rule below named `hasBrotherInLaw`, it has some premises which are if a person has a brother and has a wife, so the wife will have a brother in law the person's brother.

```
[hasBrotherInLaw:
       (?m reasoner:hasBrother ?b)
       (?m reasoner:hasWife ?f)
->     (?f reasoner:hasBrotherInLaw ?b)]
```

The rule above can be inferred into the following OWL data:

```
.......
  <rdf:Description rdf:about="http://www.example.com/ontol#Mark">
    <j.0:hasBrother rdf:resource="http://www.example.com/ontol#John"/>
    <j.0:hasWife rdf:resource="http://www.example.com/ontol#Lisa"/>
    <j.0:hasAge
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">25</j.0:hasAge>
    <rdf:type rdf:resource="http://www.example.com/ontol#MalePerson"/>
  </rdf:Description>
<rdf:Description rdf:about="http://www.example.com/ontol#Lisa">
    <rdf:type rdf:resource="http://www.example.com/ontol#FemalePerson"/>
    <j.0:hasHusband rdf:resource="http://www.example.com/ontol#Mark"/>
</rdf:Description>
.......
```

And resulting an addition of an inferred property as shown in OWL data below:

```
.......
<rdf:Description rdf:about="http://www.example.com/onto1#Lisa">
    <j.0:hasBrotherInLaw rdf:resource="http://www.example.com/onto1#John"/>
    <rdf:type rdf:resource="http://www.example.com/onto1#FemalePerson"/>
    <j.0:hasHusband rdf:resource="http://www.example.com/onto1#Mark"/>
</rdf:Description>
.......
```

Jena rules provide a basic rule syntax to expand the expressiveness of an ontology. Jena and Jena rules support several built-in functions that cover test for type of object, equality, math, string concatenation, string search, time, print, and list manipulation.

2.4.4 Jena's Query

SPARQL is fully supported in Jena. Jena uses the ARQ engine for the processing of the SPARQL queries. There are six basic classes in Jena API framework in ARQ; `Query` that represents a single SPARQL query, `Dataset` is the knowledge base on which queries are executed, `QueryFactory` that can be used to generate `Query` objects from SPARQL strings, `QueryExecution` that provides methods for the execution of queries, `ResultSet` contains the result obtained from an executed query, and `QuerySolution` that represents a row of query results.

2.5 Fuzzy Expert System and Fuzzy Ontology

2.5.1 Fuzzy Expert System

An expert system is a system which consists of a set of rules that are developed in collaboration with an expert. Fuzzy expert system is an expert system that use fuzzy concept. Basically, fuzzy expert system can be built in five steps. Those five steps are defining fuzzy sets, defining fuzzy rules, relating observation to fuzzy sets, evaluating fuzzy rules, and defuzzification [25].

The first step is defining fuzzy sets. Fuzzy sets were introduced firstly by Zadeh [26] to deal with a vague concept like fast, tall, big and the like. Silvia C. et al gave some short and compact description in fuzzy sets [27]. Let us consider a nonempty set of objects U, called the universe. A *fuzzy set* is defined as a [0, 1]–valued function on U, $f : U \rightarrow [0, 1]$. Given an object $x \in U$, f(x) represents the membership value of x to the set f . The definitions give a basic knowledge in how a fuzzy can handle the vague and imprecise things with giving a degree into the value.

Defining fuzzy value can be done in two models, linguistic and precise. Table 2.10 shows an example of defining fuzzy value. Precise fuzzy modeling is used to get a good accuracy. The main objective using linguistic modeling is to obtain fuzzy value with a good interpretability.

Table 2.10 Modelling of Fuzzy Value

Linguistic	little	enough	moderately	very	totally
Precise	0.2	0.4	0.6	0.8	1.0

The second step is to define a set of fuzzy rules. In fuzzy rules, the rules are expressed in vague language terms and do not define cut-off points or thresholds, but rather use subjective terms such as "high" and "low" [25]. It is more natural to express the knowledge in a simple way. The example of fuzzy rules is "IF the rate is high AND the price is low THEN the recommendation is high".

The third step is relating observations to fuzzy sets. It means that the sets of data is examined and converted into fuzzy membership values regarding to the membership functions.

The fourth step is to evaluate each case for the fuzzy rules by applying the fuzzy input values into the rules. Fuzzy rules can use logical operation such as OR and AND. OR fuzzy operator takes the maximum value and AND fuzzy operator will take the minimum value of the related sets.

Finally, we do the defuzzification to obtain the final crisp values. The way to get the final crisp value is by evaluating the result of each value from fuzzy rules evaluation in the defuzzification function.

2.5.2 Fuzzy Ontology Concept

A fuzzy ontology that is described by Silvia C. et al [27] is an ontology extended with fuzzy values. The basic idea is by putting fuzzy value in the ontology relation. Fuzzy ontology can be assigned through the following two functions:

f: (*Concepts* $\cup$ *Instance*) $\times$ *Properties* $\rightarrow$ *Property_Value* $\times$ [0,1] , e.g: f(*MenuA*, *taste*) $\rightarrow$ (*sweet*, 0.8).

g : (*Concepts* $\cup$ *Instances*) $\times$ (*Properties* $\cup$ *Property_Value*) $\rightarrow$ [0,1], e.g: g(*MenuA, cheap*)$\rightarrow$ 0.3.

Chapter 3

System Design and Development

3.1 System Architecture Design

Ontology-based Nutrition Planner Assistance System (ONPAS) is a Semantic Web application. Figure 3.1 shows the system architecture. The system architecture starts from the user as the client of the system that uses the internet as the media to access ONPAS. There are two main servers, Tomcat server is the web server and the MySQL database server. MySQL database is used to store the data besides the OWL files. ONPAS is the core of the applications. It provides the user interface for the data exchange algorithm and the data processing algorithm. ONPAS is built using Jena as the ontology framework. Jena is a framework with API to build and manage the ontology. Jena has reasoner and provides an ability to query the data in ontology.

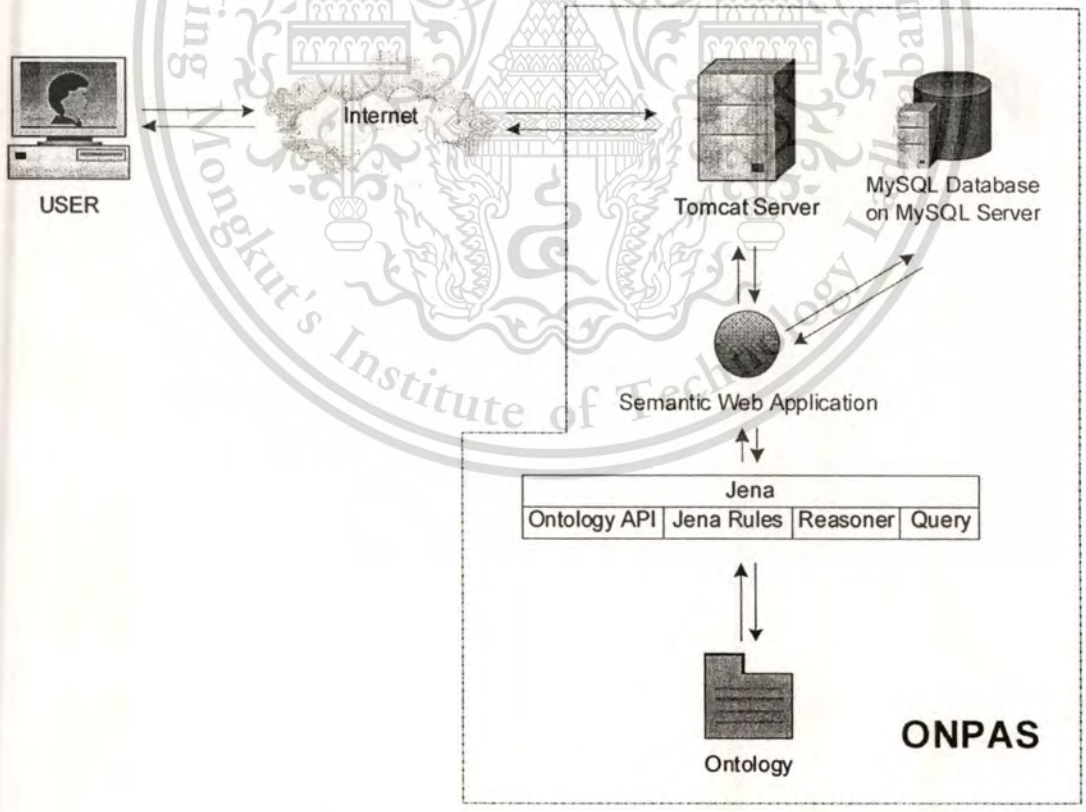


Figure 3.1 ONPAS System Architecture.

ONPAS is developed using the following tools:

a. Apache Tomcat 6.0.20 is the web server that is used by ONPAS. Tomcat server supports the web application that is developed using JSP.

b. Java 1.6.0_17 is the SDK (Software Development Kit) for Java language. JavaServer Pages which is the Java standard language to build web application is used to develop ONPAS.

c. MySQL database 5.0.51b with phpMyAdmin Database Manager Version 2.10.3. MySQL database is used to store some information and data of ONPAS.

d. Jena Semantic Web framework 2.5.6 is the ontology framework which built in Java language. Jena is used as the framework to build and manage the ontology model in ONPAS. Jena framework can construct, add, delete, and also infer ontology by its reasoner and its Jena rules.

e. Flock 2.5.6, a web browser, is used to test the ONPAS.

f. NetBeans IDE 6.8 is application development software which has Integrated Development Environment. NetBeans is free and powerful builder for Java standards and platform.

g. Adobe Photoshop CS3, a graphics design software, is used to design the graphical environment of ONPAS application.

h. Concept Draw Pro 8 is used to design the flowchart and diagram.

3.2 ONPAS Semantic Web Application Design

3.2.1 Use Case

Use case diagram of ONPAS in Figure 3.2 shows the actor who is involved in this application. The diagram also describes the function and the privileges of each actor.

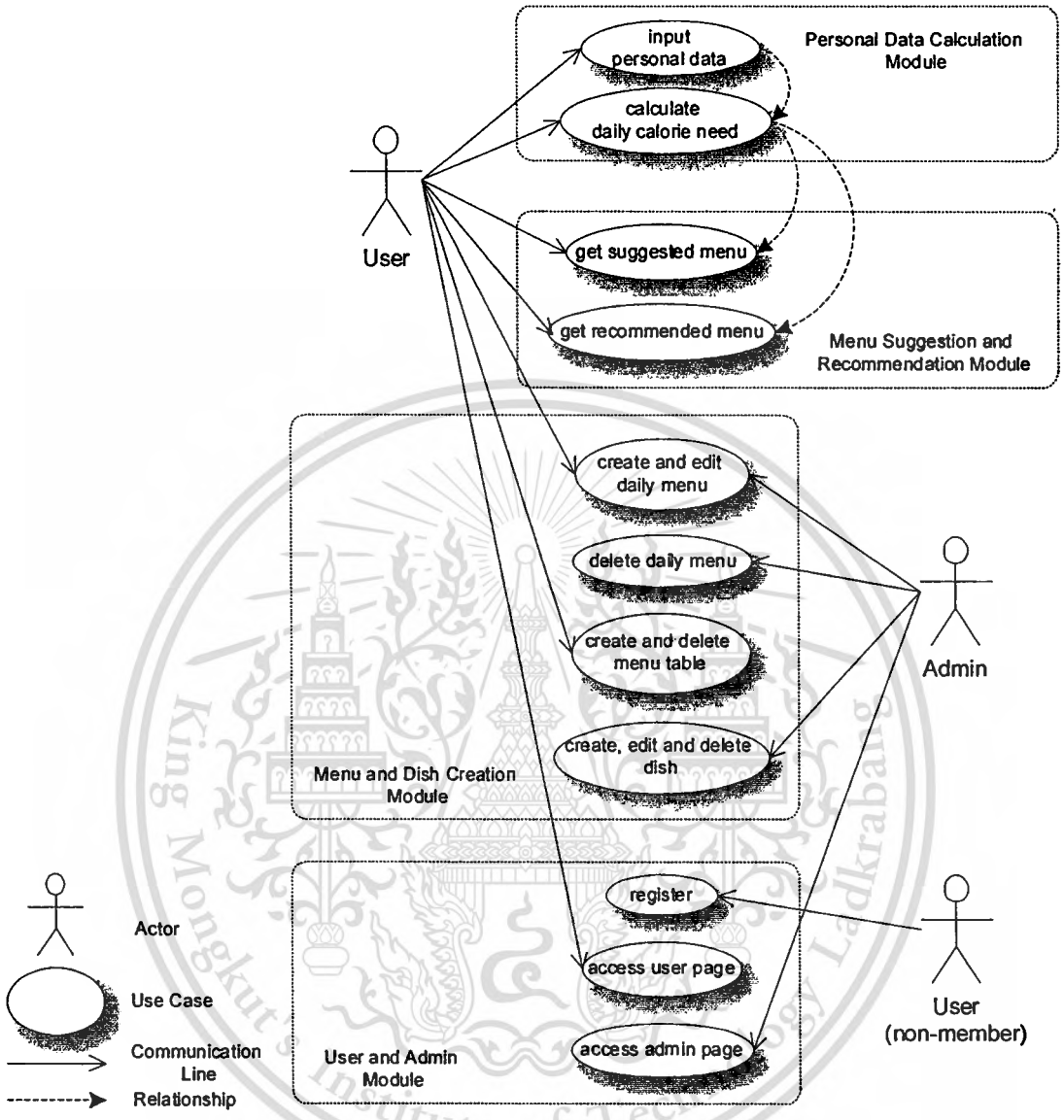


Figure 3.2 Use case diagram of ONPAS.

The use case diagram of ONPAS is explained in detail below.

Use Case: Ontology-based Nutrition Planner Assistance System Semantic Web Application

Use Case type: General system analysis

Main actor: User, non-member User, and Admin

Actor role:

- User has privileges to use most of features in ONPAS. User can calculate his/her own personal daily calorie needs, get daily menu suggestion, make menu table and review all of his/her menu creation.
- A non-member User has to register at first to get the same privileges of the User.
- Admin has full privileges in ONPAS management.

Description:

The functions in ONPAS are categorized in modules. Those are Personal Data Calculation, Menu Suggestion and Recommendation, Menu and Dish creation, and User and Admin module.

Personal Data Calculation module includes the input process of personal data and the calculation of the daily calorie needs of user. The calculation process will be executed after the user input all required personal data.

Menu Suggestion and Recommendation module has two main processes. The processes are giving the daily menu suggestion and the daily menu recommendation. These processes will be done after calculating the daily calorie needs from Personal Data Calculation module.

Menu and Dish Creation module contains some functions to create and manage dishes, menus and menu tables.

User and Admin module is mostly an interface form for user and admin management.

3.2.2 Program Flow

ONPAS has main program flow as shown in Figure 3.3 and Figure 3.4. These two figures represent the program flow of ONPAS as activity diagram.

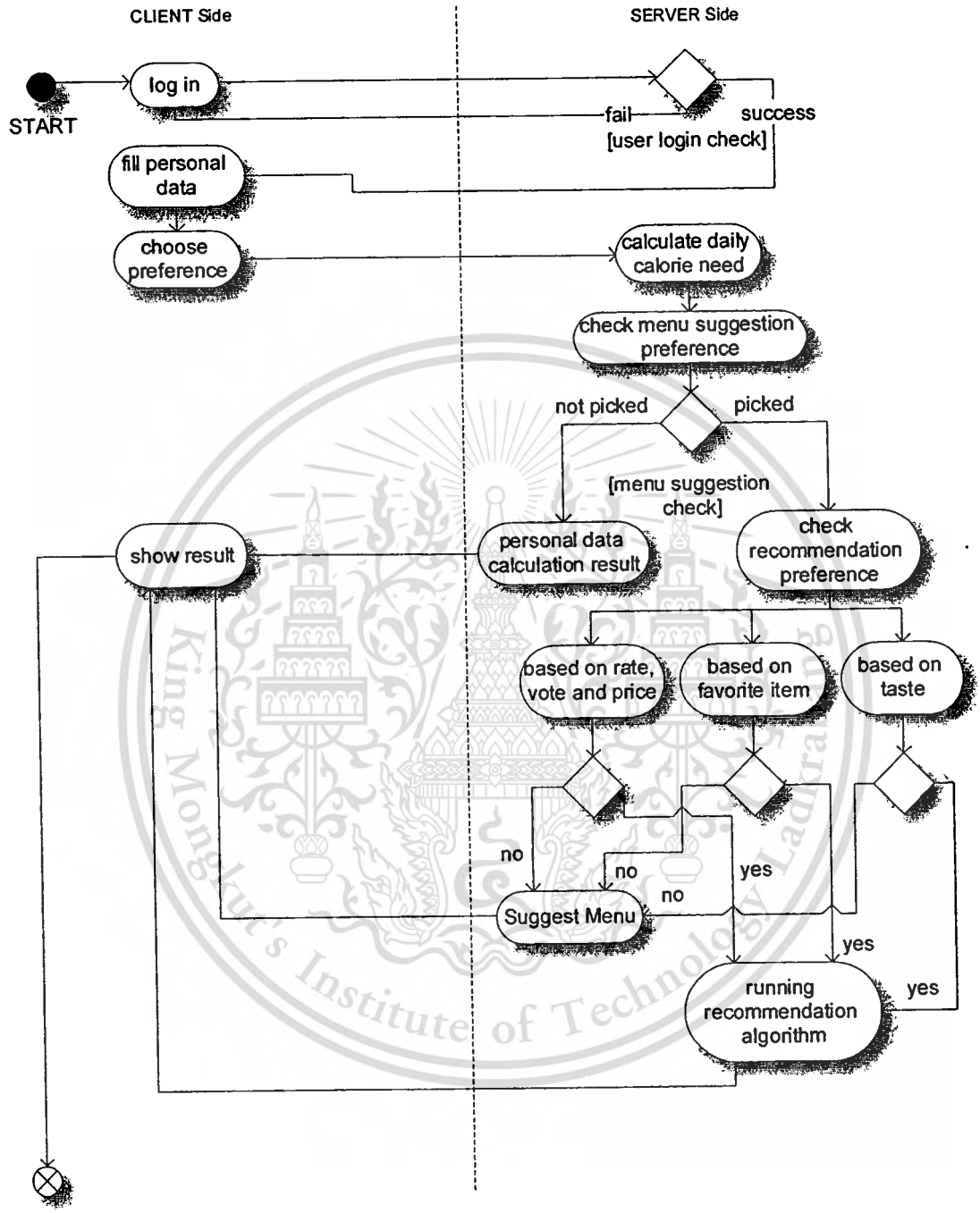


Figure 3.3 Activity diagram of ONPAS (part 1).

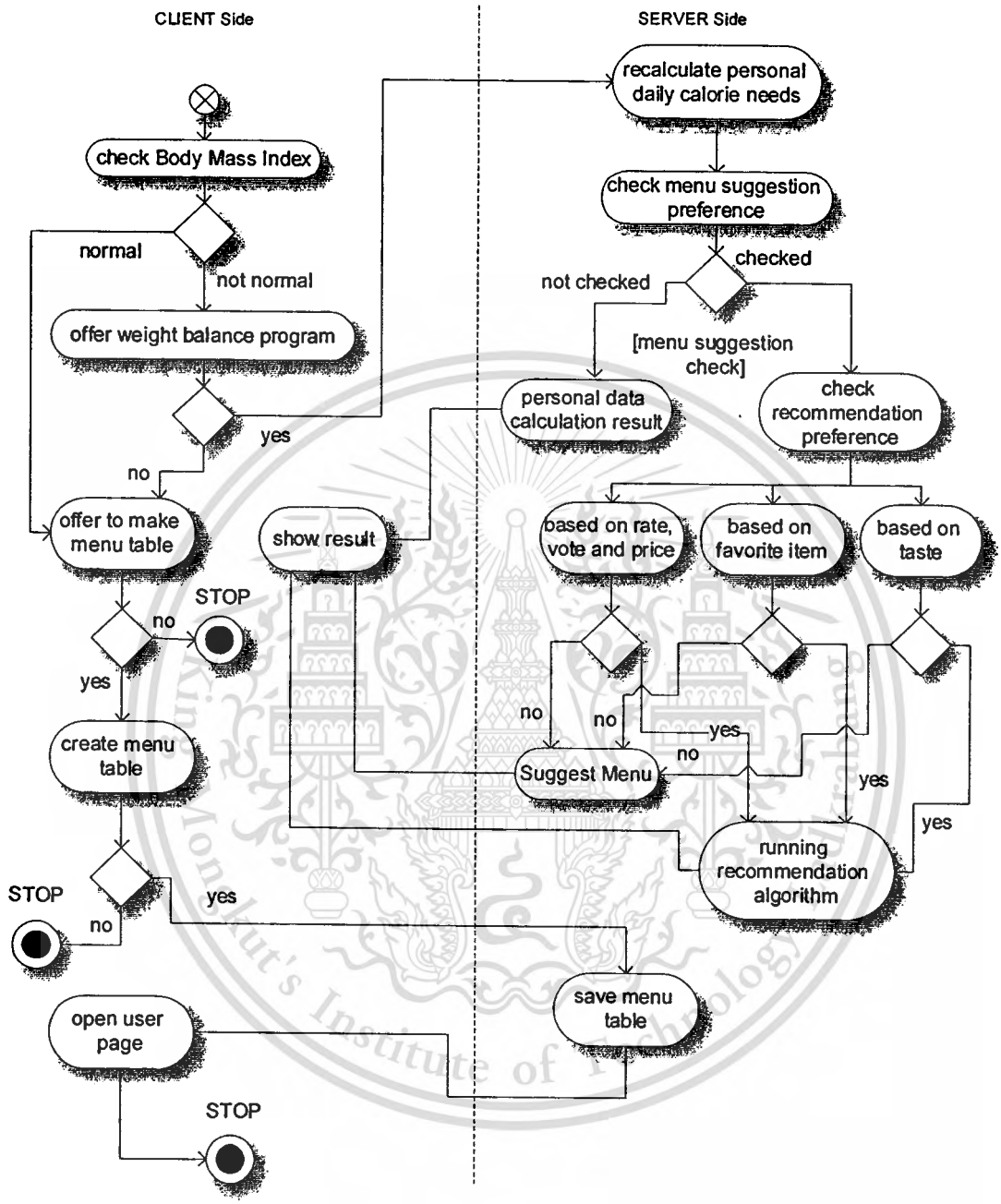


Figure 3.4 Activity diagram of ONPAS (part 2).

From the starting point in the Figure 3.3, user has to log in into ONPAS at first. Next, the user can calculate his/her personal data and get suggestion and recommendation for his/her daily menu. The steps of the process are shown in Figure 3.3.

Figure 3.4 continues the process from Figure 3.3. Figure 3.4 shows the steps from recalculating personal daily calorie needs because of the weight balance program that the user choose until creating menu and menu table.

The detail of each process will be explained in each module in the next sub chapter.

3.2.3 User Interface Design

The user interface of ONPAS is made simple but interactive. There are five components in the user interface design depicted in Figure 3.5. The banner is located at the top of the interface as the title or name. The content of the web will be located in the center. The content changes dynamically depend on what part of the web that is accessed. On the left part, there is a web navigation which contains links to main web pages in ONPAS. Admin and user menu part on the right is used to show the navigation link to admin and user pages. Finally, the footer is used as the place to put the signature.

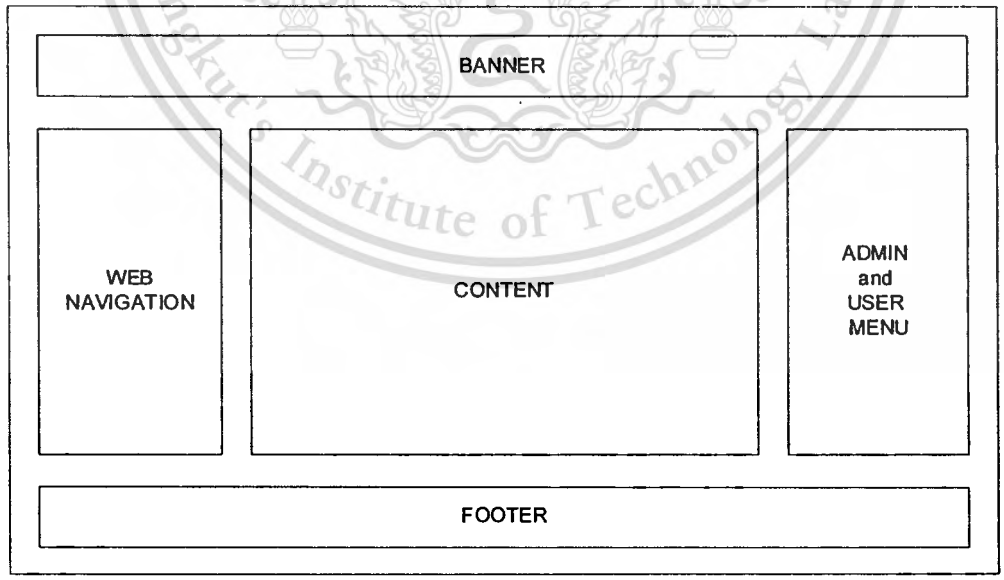


Figure 3.5 User interface design of ONPAS.

3.3 Personal Data Calculation Module

3.3.1 Module's Description

Personal Data Calculation module has the following functions and usages:

a. Input personal data. By showing the user interface in the form, user is asked to fill personal data. The user personal data is used to calculate his/her daily calorie needs by using equation (*2.1*). In equation (*2.1*), *TEE* comes from the multiplication of *BMR* and *AF*. *BMR* can be achieved if the user provides the information of gender, age and weight. *AF* can be gotten by knowing the activity and the gender of the user. The height of user is needed to know his/her *BMI* and *BMI* status. The name of the user will be the compulsory information that must be added too, and the system provides the user's name automatically from the user's data in the database.

b. Calculate the daily calorie need. The calculating process will be run after user fill all needed information (personal data). The daily calorie needs calculation will give the Activity Factor value by inferring the activity of the user. The next step is *BMR* calculation and finally the *TEE* will be gotten. All of calculation results will be shown in the web page after calculation processes.

c. Calculate *BMI* and offer weight balance program. The *BMI* equation in the literature review (chapter 2) is the calculation for adult only (age > 18 years). If an adult ask the system to do the calculation, the system also gives additional information regarding to his/her *BMI* value and the *BMI* status such as underweight, normal and overweight. If abnormal *BMI* status is shown, user can select a weight balance program to gain or lose some weight, and then the system will recalculate the daily calorie needs of the user.

The personal data will be stored in the OWL file. Each user has his/her personal data file.

3.3.2 Module's Program Flow

The program flow of Personal Data Calculation module is depicted in Figure 3.6. It starts from input the personal data, infers Activity Factor value, calculates Basal Metabolic Rate, calculates Total Energy Expenditure and finally shows the personal data and calculation result. Especially for adult, there will be *BMI* calculation and weight balance program will be offered if *BMI* is abnormal.

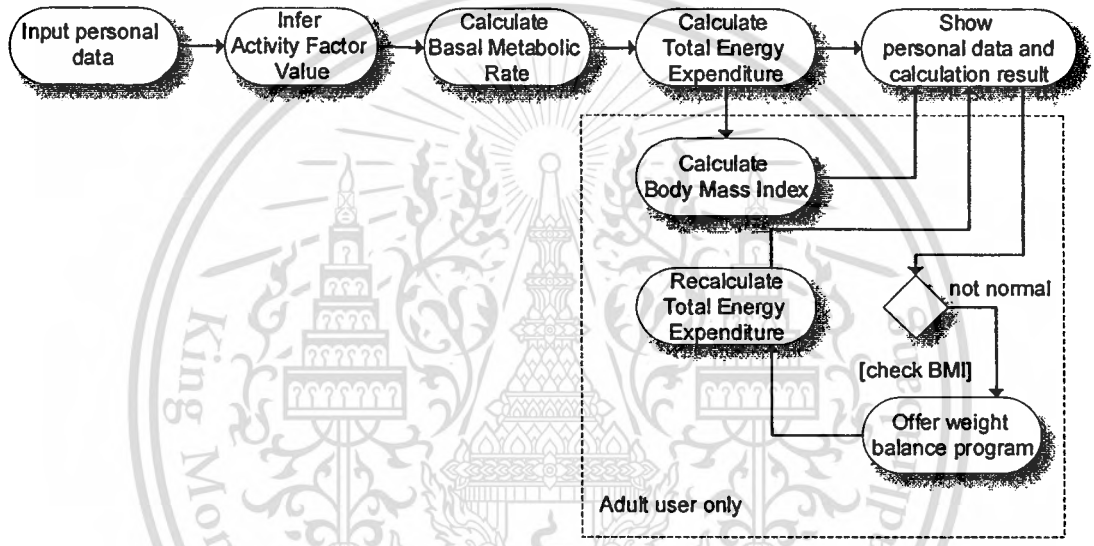


Figure 3.6 Personal Data Calculation module program flow.

3.3.3 Personal Data Calculation Ontology

Personal Data Ontology is the ontology of user's personal data. This ontology is importance to keep the information of personal data so that the system can infer a new property and calculate the daily calorie needs of a user. Figure 3.7 shows the ontology of Personal Data Calculation module.

`Person` class is categorized into six subclasses based on the person's age. This categorization is based on the Schofield *BMR* equations. The categorization is also applied in `Activity` class, since the *AF* value is different in each group of people based on age. The classification is explained in chapter 2.

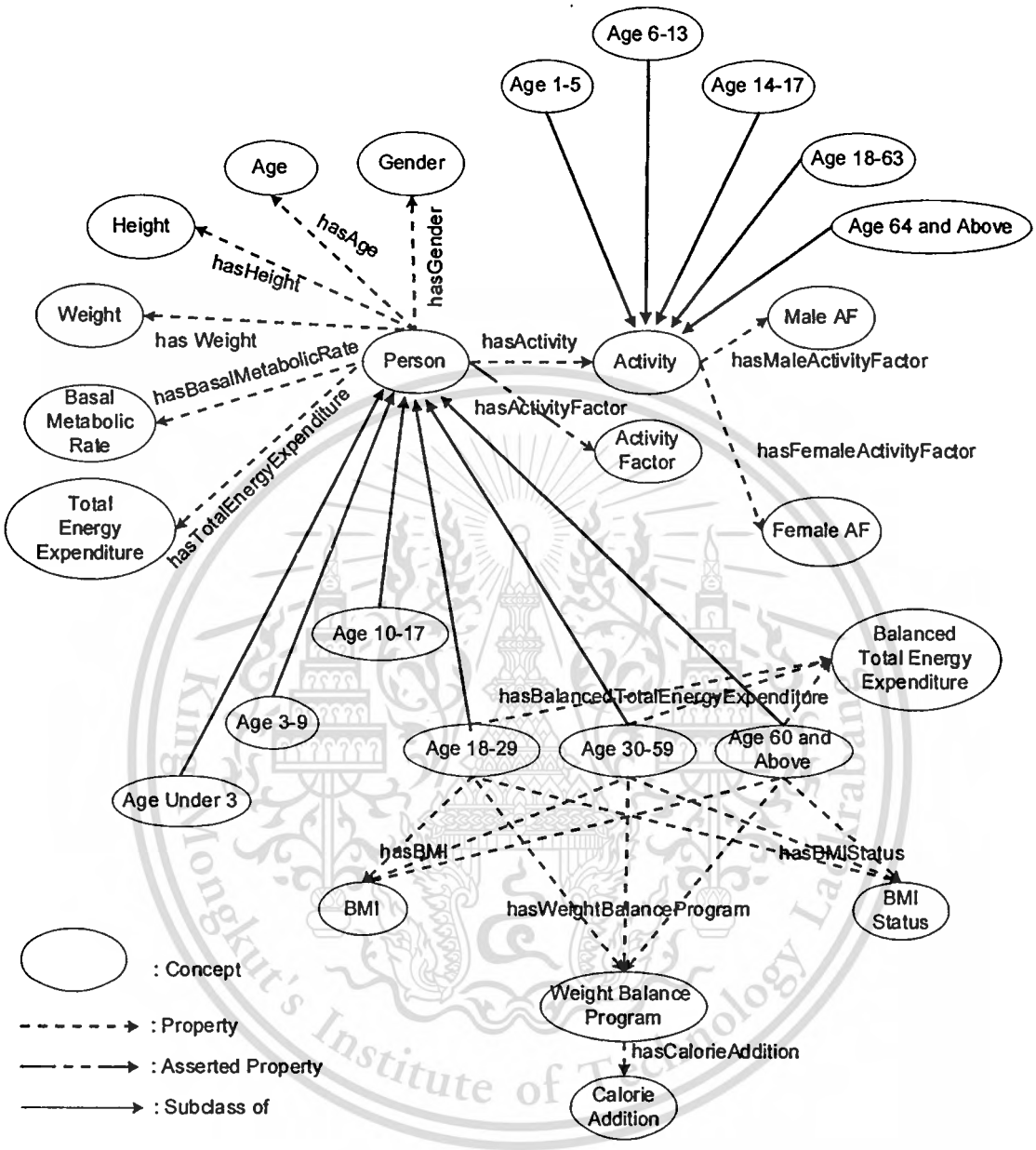


Figure 3.7 Personal Data Calculation ontology.

The ontology graphs in Figure 3.7 shows the relationship of concept to another concept. It consists of all factors for daily calorie needs calculation. This ontology graph performs a triplet in each node which is connected by each property (relation). For instance, there is `Person` class with property `hasAge` and property value an `Age`. `Person` is the subject, `hasAge` is the predicate and the value of `Age` is the object.

`Person` will have `hasActivityFactor` property after the infer *AF* value process. This process will infer an asserted property, which is `hasActivityFactor`, after the reasoner executes the knowledge base that is made. The knowledge base matches the type of activity and the gender that a person have with the ontology model that is made. In short, if the gender of a person is male, then the *AF* value will be taken from the property value of `hasMaleActivityFactor`.

When a user picks a weight balance program, the system will recalculate the *TEE* of the user. The result of the new *TEE* (balanced *TEE*) will not be stored as the property value of `hasTotalEnergyExpenditure` property. The balanced *TEE* will be stored in the personal data ontology as the property value of `hasBalancedTotalEnergyExpenditure` property.

3.4 Menu Suggestion and Recommendation Module

3.4.1 Module's Description

Menu Suggestion and Recommendation module has two main usages; suggest menu and recommend menu.

Suggested menu is a daily menu which has the same rounded total calorie as the rounded person's total daily calorie needs (*TEE*). The total calorie of a daily menu and a person's *TEE* has float data type. The system round the total calorie of menu the person's *TEE* by 100 to make it easier but do not make large differences. The suggested menu is filtered by the person's abstinence. Abstinence is food item which are prohibited for a person to eat. It can be in the case of allergy or another prohibition rules.

Menu recommendation feature from this module will be categorized in three kinds; Menu recommendation based on rate, vote and price, Menu recommendation based on taste, and Menu recommendation based on favorite items. These three recommendation features will be explained in details on subchapter 3.4.5, 3.4.6 and 3.4.7 respectively.

3.4.2 Menu Suggestion Ontology

Menu Suggestion ontology has two main classes; `Person` and `Menu`. The design of Menu Suggestion ontology is shown in Figure 3.8. In the `Person` class, there is a property that has not been mentioned before in the Personal Data Calculation ontology. The property is `hasAbstinence`. `hasAbstinence` is used in the menu suggestion process. This property contains the prohibited food items of the user. `Menu` class has two importance properties which will be used in the inference process to suggest some menus to the user. `hasCalorie` property contains the information of total calorie that a menu has, and `hasMenuItem` property mentions all items which is contained in the menu.

There are two asserted properties in this ontology. `hasProhibitedMenuforAbistinence` and `hasMenu` is created by inferring some knowledge bases. The knowledge bases that infer these two asserted properties will be explained in subchapter 3.4.3.

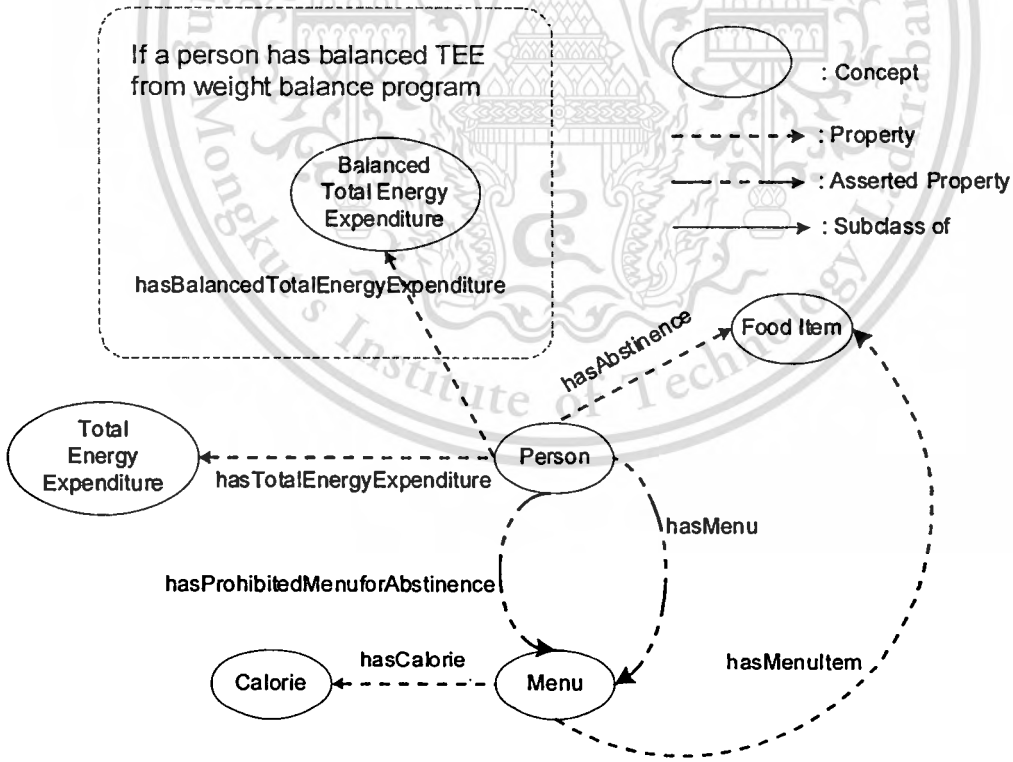


Figure 3.8 Menu Suggestion ontology.

3.4.3 Knowledge Base of Menu Suggestion

There are two main ideas of giving the menu suggestion. The first one is suggesting daily menu which has the same total calorie with the user's daily calorie needs. The other is filtering the suggested menu from the first idea regarding to the user's abstinence. The second idea will separate the menu which contains food item that is prohibited for the user.

The representation in if-clause sentences for these two ideas gives more understanding explanations. The two clauses are:

- If person `x` has Total Energy Expenditure `y` kilocalorie (kcal) and menu `z` has total calorie `y` kcal then person `x` has suggested menu `z`.
- If person `r` has abstinence in `s` and menu `t` contains of `s` then person `r` is prohibited to have menu `t`.

`x`, `y`, `z`, `r`, `s`, and `t` are variables. These two if-clauses will be the knowledge bases to suggest menu for the user. When that knowledge is inferred by the reasoner, these two clauses will produce two new asserted properties; `hasMenu` and `hasProhibitedMenuforAbstinence` respectively.

3.4.4 Menu Recommendation Idea

A system that is integrated with recommendation support can provide a more personalized and proactive functions. Therefore, ONPAS has a recommendation feature to support the decision processes. The main idea is to give the user more specific daily menu from his/her suggested menu.

Figure 3.9 shows the idea to make recommendation feature in ONPAS. A person will get suggested menu from menu suggestion process. There will be so many menus suggested by the process. Therefore, recommendation processes give more preference to the person to get more specific menu. ONPAS has three categories in giving recommendation. The three categories are menu recommendation based on rate, vote and price, menu recommendation based on taste and menu recommendation based on favorite item. These three recommendation categories can be

combined each other to get more and more specific menu which is preferred by the user. The complete explanation and concept of these three recommendation categories will be described on subchapter 3.4.5, 3.4.6 and 3.4.7 respectively.

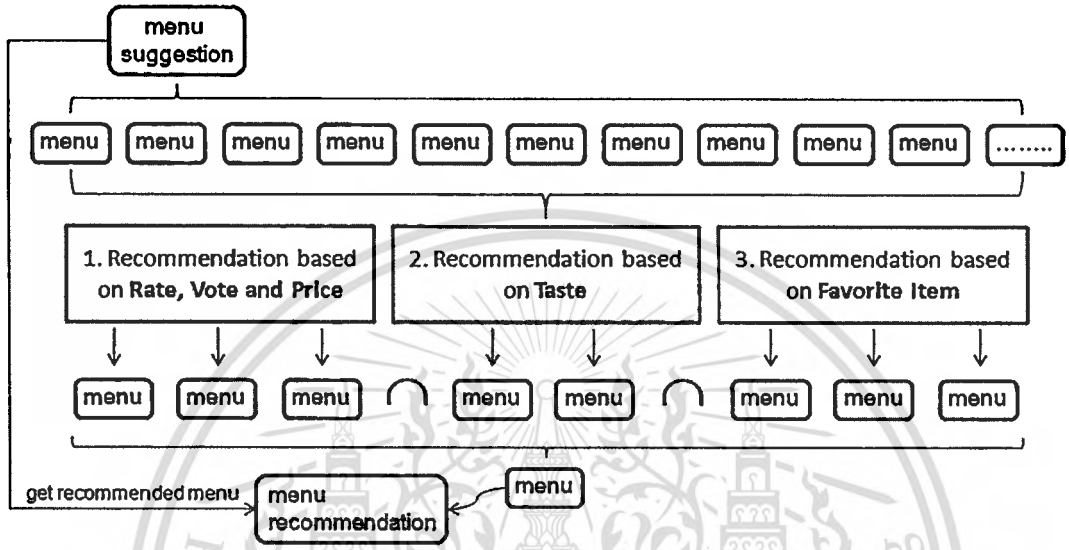


Figure 3.9 Menu recommendation idea.

3.4.5 Menu Recommendation based on Rate, Vote and Price

Three factors that will be involved in the recommendation are rate, vote and price. These three factors are belong to `Menu` class. The rate of menu means the point that menus get from user appreciations. The user gives appreciation to a certain menu by giving rating score to the menu. The vote of menu is the total amount of rate which is given by the user. The price of menu is the value of average amount of money to make a certain menu.

Menu recommendation based on rate, vote and price uses the method of fuzzy ontology. Fuzzy ontology method is used since ONPAS is an ontology-based system and fuzzy logic is needed in this recommendation feature. There are three factors in this recommendation feature. Rate, vote and price factors will have membership degree level. There will be knowledge based rules to obtain the final recommendation value. Fuzzy concept is suitable to bring rate, vote and price factors into relation and results the value of recommendation.

The following explanation will describe the component of this menu recommendation.

a. Ontology of menu recommendation based on rate, vote and price.

As mentioned before, this recommendation uses the fuzzy ontology method. Figure 3.10 shows the fuzzy ontology design of menu recommendation based on rate, vote and price. There are three new concepts; `Rate`, `Vote` and `Price`. The fuzzy value of `Vote` and `Rate` will be categorized to `Low` and `High`. The fuzzy value of `Price` will be categorized to `Cheap` and `Expensive`. All fuzzy value will have *membership degree* (*md*) value which has range value [0-1]. The membership degree value shows that the fuzzy concept is applied in the ontology.

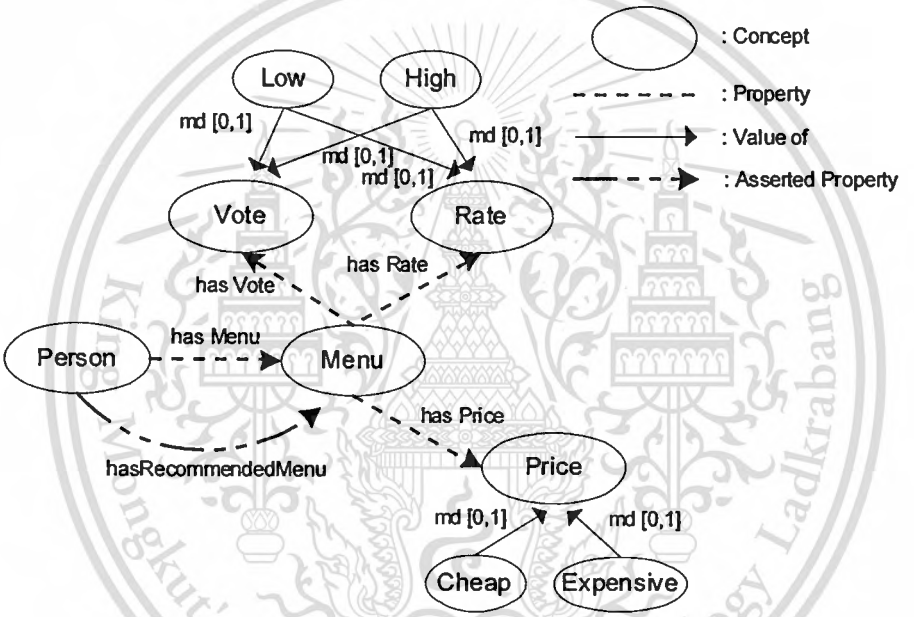


Figure 3.10 Ontology of menu recommendation based on rate, vote and price.

b. Value calculation of `Rate`, `Vote` and `Price`.

The way to get the value of `Rate` is by using the five stars rate, i.e. 0, 1, 2, 3, 4 and 5 stars, which will equal to assigned *md* value, 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0, respectively. The value of `Rate` in a menu is the average *md* value. The vote means how many user give votes (appreciation) for a menu, but the value of `Vote` class in a menu is the total amount of vote that a menu has divided by total amount of vote that all suggested menu have. The calculation of `Rate` and `Vote` is shown in (3.1) and (3.2) respectively as given below. Finally, the cost or price of the menu will be used as the value of `Price`.

$$Rate = \frac{\sum_{i=0}^{5} a_i (0.2) i}{\sum_{i=0}^{5} a_i} \quad (3.1)$$

$$Vote = \frac{\sum_{i=0}^{5} a_i}{\sum_{j=1}^{n} \sum_{i=0}^{5} a_{ji}} \quad (3.2)$$

a_i is the amount of i star(s) in a suggested menu, a_{ji} is the amount of i star(s) in suggested menu j, and n is the amount of suggested menu.

c. Fuzzy ontology sets

Figure 3.11 shows the trapezium type membership function which is used to define the *md* of `Price`, `Rate` and `Vote`. The x-axis is the concept's real value of `Price`, `Rate` and `Vote`. The y-axis is the *md* of fuzzy set. f_{cheap}, $f_{expensive}$, f_{low}, and f_{high} are the membership functions for cheap price, expensive price, low rate or low vote and high rate or high vote, respectively. Variable a and b in f_{cheap} and $f_{expensive}$ are the limitation value of cheap and expensive price. Variable a and b in f_{low} and f_{high} are the limitation value of low and high in rate and vote.

Variable a and b are declared as the basis of rate, vote and price membership degree function. These bases can be changed by Admin in the administration area of ONPAS Semantic Web.

Ontology is constructed as triplets. It does not have any feature to define *md* directly yet. It can only define the concept, property and property value as the subject, predicate and object. Fuzzy ontology functions will be declared by putting the property value along with the property, and make the *md* as the property value. For example, a fuzzy ontology function *f*(*Menu2000A*, *price*) = (*cheap*, 0.7) means that the individual *Menu2000A* has *hasCheapPrice* property with property value equal to 0.7 and this value is the *md* of *cheap*.

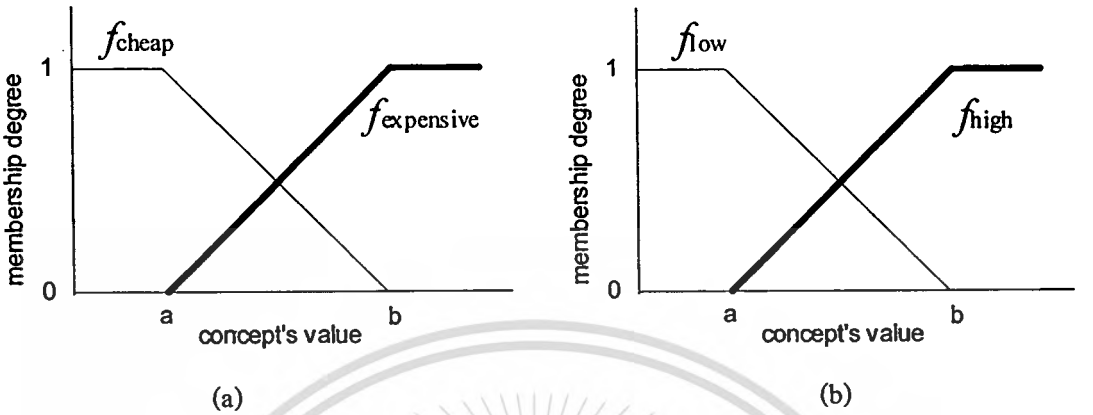


Figure 3.11 (a) Fuzzy set membership function for *Price*,
(b) Fuzzy set membership function for *Rate* and *Vote*.

d. Fuzzy rules

Menu recommendation based on rate, vote and price needs some fuzzy rules to get the value of recommendation. Table 3.1 shows eight fuzzy rules which are used. AND operator is used in every rule. The way to specify the rules from Table 3.1 can be explained as follows. For example, **Fuzzy Rule No.8** declares that high recommendation is given to menu if the menu has high vote, high rate and cheap price.

Table 3.1 Fuzzy Rules for Menu Recommendation based on Rate, Vote and Price

Fuzzy Rule No.	1	2	3	4	5	6	7	8
Rate	L	L	L	L	H	H	H	H
Vote	H	H	L	L	L	L	H	H
Price	E	C	E	C	E	C	E	C
Recommendation	LOW				MEDIUM			HIGH

*L : Low, H : High, C : Cheap, E : Expensive

e. Defuzzification

Defuzzification process is needed to get the crisp value of recommendation. We use a singleton function for total recommendation value defuzzification as shown in Figure 3.12. The degree of recommendation is divided into three degrees; low (20 points), medium (50 points) and high (80 points).

Calculating *Total Recommendation Value (TRV)* can be done by weighted average calculation in equation (3.3). The equation needs the result of low, medium and high recommendation value from evaluating fuzzy rules.

The recommended menu based on rate, vote and price is a menu which has *TRV* equals or more than *x*. *x* is a variable defined by the Admin as the border or limitation. The value of *x* can be changed by the Admin from the administration area in ONPAS Semantic Web.

$$TRV = \frac{(lrv \times 20) + (mrv \times 50) + (hrv \times 80)}{lrv + mrv + hrv} \quad (3.3)$$

TRV : Total Recommendation Value,

lrv : low recommendation value ,

mrv : medium recommendation value,

hrv : high recommendation value.

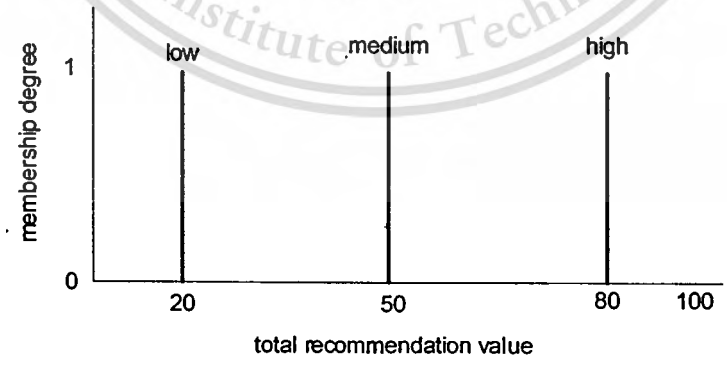


Figure 3.12 Defuzzification function of total menu recommendation points.

3.4.6 Menu Recommendation based on Taste

The second menu recommendation is based on taste only. The ontology design of the recommendation is shown in Figure 3.13. `Taste` class has five kinds of value which also has a *membership degree* (*md*) in each value. The five kinds of value are `Sweet`, `Bitter`, `Salty`, `Sour` and `Spicy`. Fuzzy concept and linguistic fuzzy value model is used. By defining *md* in taste, the recommendation gives more precise result, since a very spicy menu is different from a little bit spicy menu.

Taste's *md* can be assigned directly by putting some values. For example, *Menu2000A* has very sweet taste. Based on fuzzy value's modeling in Table 2.10, the function can be written as follows, $f(Menu2000A,taste) = (sweet,0.8)$. This recommendation process does not need a fuzzy rules and defuzzification because it only depends on one factor; taste.

Tolerance value is added when querying menu with preferred taste. It extends the query. It also can give more choices even if there is no menu which has `Taste`'s *md* exactly matches with user's request.

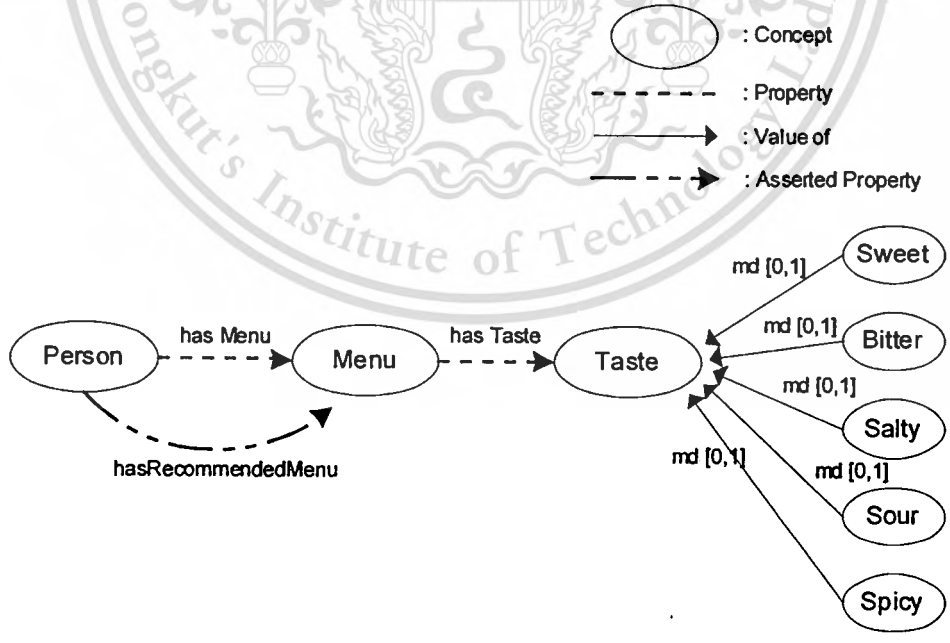


Figure 3.13 Ontology of menu recommendation based on taste.

3.4.7 Menu Recommendation based on Favorite Item

A user could have some favorite food items when picking a menu. The recommendation process will give the person some recommended menu based on his/her favorite items. When suggested menu of the user has the same food item with the user's favorite item, the menu will be recommended to the person. In the if-clause sentence it will be:

If person x has favorite item y and menu z has food item y then person x will have recommended menu z.

x, y, and z is variables.

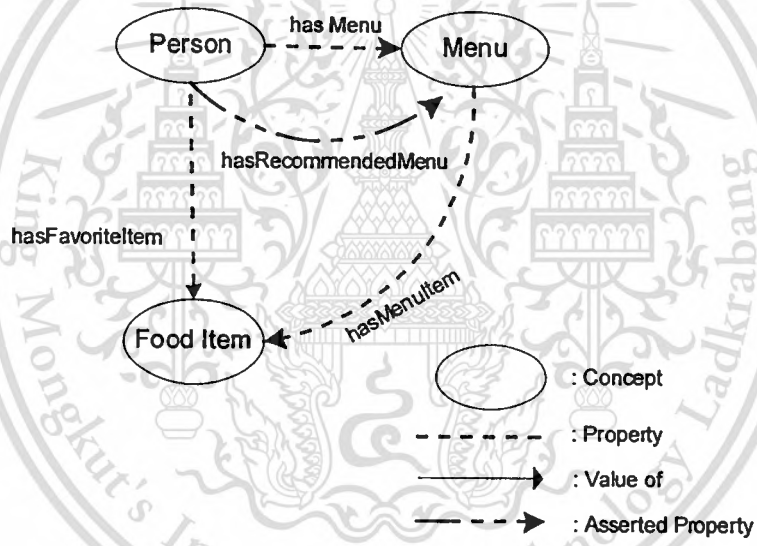


Figure 3.14 Ontology of menu recommendation based on favorite item.

3.5 Menu and Dish Creation Module

3.5.1 Module's Description

Menu and Dish Creation module has three main functions. The creation of menu, menu table and dish will be covered in this module.

The menu is composed from breakfast, lunch, dinner and additional snack and beverages for one day. Menu table is the collection of menu. Menu table can contains one to, seven days of daily menus. The dish is a particular item of prepared food. The creation of menu and dish are involving the food item. Food item is food ingredient or groceries. The kinds and taxonomy of food item will be explained in subchapter 3.5.2.

A menu can be created in two ways. The first is by collecting and composing dishes from the dish database in ONPAS. The second way is by filling all food information to construct one whole daily menu manually. The second way needs some exact literatures to make the right composition. In ONPAS, the second method can be done only by the user which has an administrator privileges, but for the first method, any registered user can use it. The detail of the menu creation process can be seen in subchapter 3.5.3. The information of menu which is stored in the database will be explained in subchapter 3.5.4.

A menu table can be created in two ways. The first is by taking user's suggested menu and the second one is by composing user's own daily menu in one to seven days by picking some dishes. All registered user can make a menu table. A menu table is limited to 7 days but user can have many more menu tables (depends on the database capacity). The detail of the menu table creation process can be seen in subchapter 3.5.5. The information of menu table which is stored in the database will be explained in subchapter 3.5.6.

The dishes can be only made by the administrator. A dish is made by filling complete information of dishes regarding to the scientific food composition table. The detail of the dish creation process can be seen in subchapter 3.5.7. The information of dish which is stored in the database will be explained in subchapter 3.5.8.

User has different privileges with Admin. Table 3.2 shows the User and Admin privileges in Menu and Dish Creation module. Admin has full access in every part of module. User has limitation in creating menu. User can create menu from dish only and can edit the menu that User created only. In menu table creation, Admin and User have the same right to create menu table but

they have different privileges in deleting the menu table. User can delete menu table that User created only. User has no right in creating, editing or deleting dish.

Table 3.2 User and Admin Privileges in Menu and Dish Creation Module

	Menu	Menu Table	Dish
User	C*, E*	C, D*	-
Admin	C, E, D	C, D	C, E, D

C : Create, E : Edit, D : Delete, * : limited

3.5.2 Food Item Ontology

The foods and the dishes that are used by ONPAS as the study cases are Thai foods and dishes. The database of Thai food and dish is taken from *ASEAN Food Composition Tables* [6] and *Thai Food Composition Tables* [7]. *ASEAN Food Composition Tables* and *Thai Food Composition Tables* are made by Institute of Nutrition, Mahidol University, Thailand. These references contain complete nutrition information of Thai food and dishes, including the macronutrient.

The Thai food and dish name in *ASEAN Food Composition Tables* and *Thai Food Composition Tables* are originally written in Thai. The Thai name will be translated to phonetic alphabet when stored in the database. The http://www.thai-language.com/dict website has a tool to convert Thai characters into phonetic alphabet.

The taxonomy of the food item refers to *ASEAN Food Composition Tables*. Food item is categorized into 13 classes as shown in Figure 3.15. The categorization refers to the type of the food item. In each class there are many kinds of food items which are declared as the subclasses. Figure 3.16 shows the complete taxonomy. ONPAS uses the taxonomy as the detail of food item categorization.

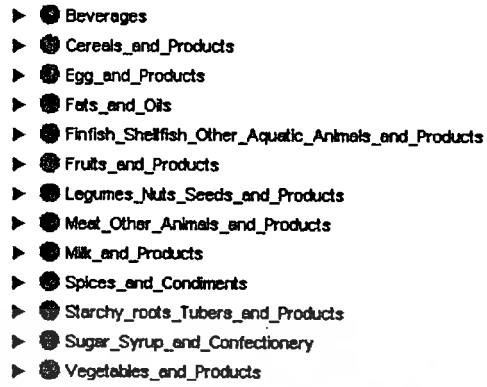


Figure 3.15 Classes of Food Item ontology.

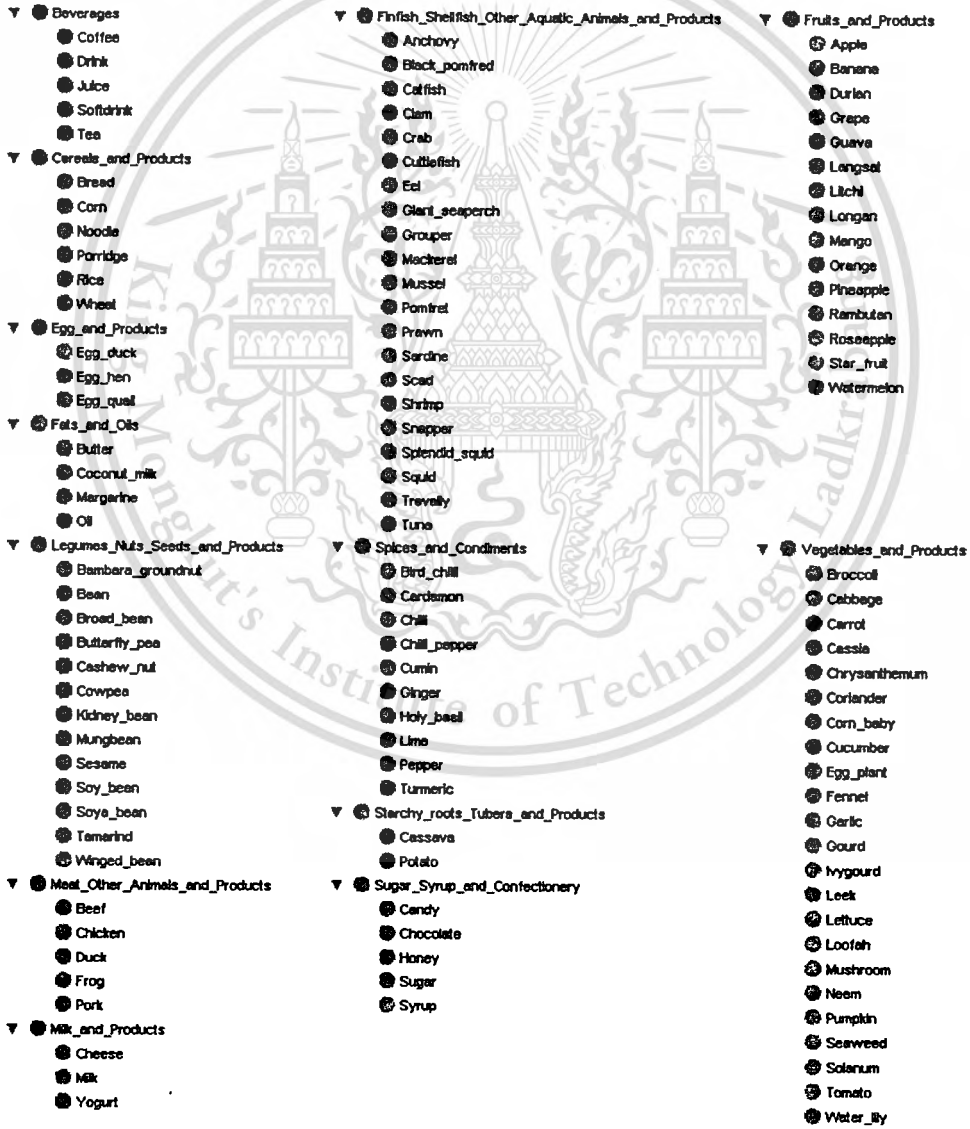


Figure 3.16 Subclasses of Food Item ontology.

3.5.3 Program Flow of Menu Creation

Creating a menu can be done in two ways as seen in Figure 3.17. User has the privileges only to create menu from dish. The program flow or the steps that User has to do to create menu can be seen in Figure 3.17 point number 1. Set nutrition composition of dish steps when creating menu means that the creator should set the weight composition of each dish so that the macronutrient composition is balanced. Regarding to the balanced macronutrient composition, Admin is assumed to know exactly the composition when fills all information to create menu by the second way that is shown Figure 3.17.

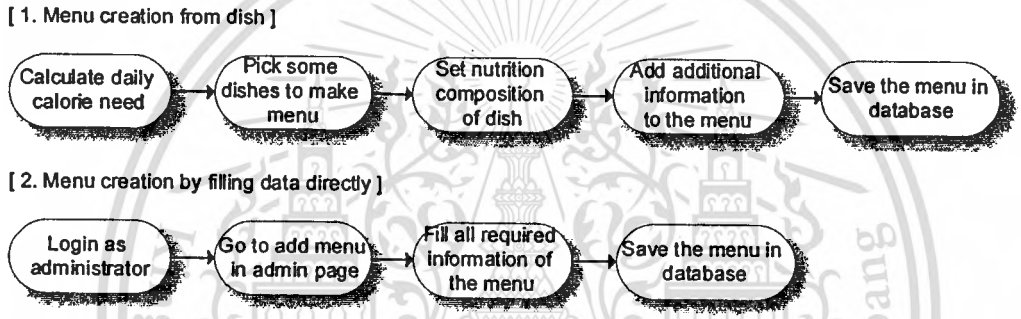


Figure 3.17 Program flow of menu creation.

3.5.4 Database Design of Menu

All of menu in ONPAS are stored in 'menu' table of ONPAS database. There are 14 fields in the table. Figure 3.18 shows the 'menu' table field and the data type in each field. 'menu_id' field contains the id of the menu. The id will be unique for each menu. 'menu_name' field stores the name of the menu. 'menu_calorie' field contains the total calorie of a menu. 'menu_main_item' field consists of food items in the menu and 'menu_item' contains all detail information of the menu including the weight composition of each dish. 'menu_rate', 'menu_vote' and 'menu_price' fields store menu's rate, amount of vote in a menu and the price of menu respectively. These three fields are used by menu recommendation module to give recommendation based on rate, vote and price. 'menu_deg_sweet', 'menu_deg_bitter', 'menu_deg_salty', 'menu_deg_sour', 'menu_deg_salty' fields contain of membership degree value for sweet, bitter, salty, sour, and spicy taste respectively. Finally, the 'menu_creator' field is a field that filled by the name of menu's creator.

Field	Type
menu_id	tinyint(10)
menu_name	varchar(256)
menu_calorie	float
menu_price	float
menu_main_item	varchar(1024)
menu_item	text
menu_deg_sweet	float
menu_deg_bitter	float
menu_deg_salty	float
menu_deg_sour	float
menu_deg_spicy	float
menu_rate	float
menu_vote	int(11)
menu_creator	varchar(50)

Figure 3.18 Database design of menu.

3.5.5 Program Flow of Menu Table Creation

Creating menu table or collection of daily menu for maximum 7 days is basically started with daily calorie needs calculation. There are two sources in making menu table as seen in Figure 3.19. The first source is from the suggested menu and the second one is from the dish database. When the source for making the menu table is from the suggested menu, user has to pick the suggest menu preference.

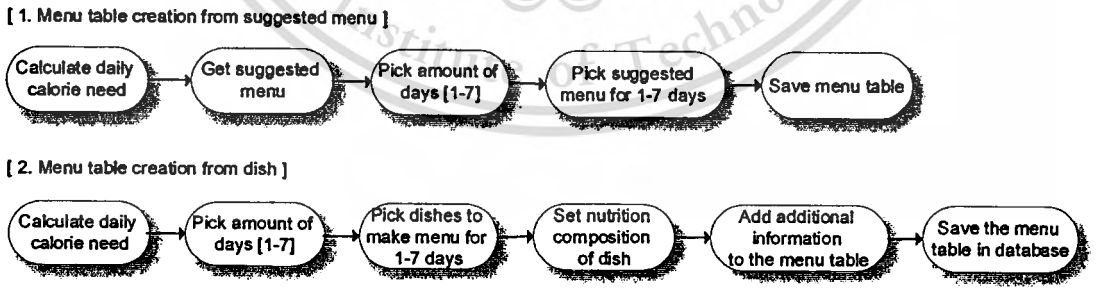


Figure 3.19 Program flow of menu table creation.

3.5.6 Database of Menu Table

Menu table database consists of 5 fields as shown in Figure 3.20. A menu table will have the unique id which is stored in 'menutable_id' field. 'menutable_name' field keeps the name of the menu table. 'menu_creator_id' becomes an importance field since it stores the creator id. The system needs this information to differentiate the menu tables based on its creator. 'menutable_detail' stores all detail information of the menu in 1-7 days. When the data is shown in the web page, it is divided in each day menu. 'menutable_date_created' is also an important data that stored the creation date of menu table.

Field	Type
menutable_id	tinyint(10)
menutable_name	varchar(256)
menutable_creator_id	tinyint(10)
menutable_detail	longtext
menutable_date_created	timestamp

Figure 3.20 Database design of menu table.

3.5.7 Program Flow of Dish Creation

The creation of dish can only be done by the Admin. The program flow will be start by login in ONPAS as an Admin. The next step is to go to add dish page, fill required information and save it. The graph of the flow is depicted in Figure 3.21.

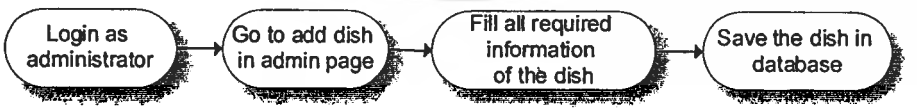


Figure 3.21 Program flow of dish creation.

3.5.8 Database Design of Dish

Dish information will be stored in the database with 10 kinds of data. Figure 3.22 shows the 10 fields in the dish table database. 'dish_id' keeps the unique id of every dish. 'dish_name' contains of dish name. 'dish_weight' field keeps the weight composition basis of the dish. The next four fields consist of the dish nutrient data. 'dish_calorie' keeps the total calorie of the dish at the weight composition basis. 'dish_carbohydrate', 'dish_fat' and 'dish_protein' keep the information of how many grams of carbohydrate, fat and protein respectively. 'dish_item' field consists of food items that are available in the dish. Finally, the 'dish_info' keeps the additional information of the dish. 'dish_category' contains of the category of dish. The category choice is already predefined. The categorization refers to [6]. The categories are:

- Cereal
- Egg
- Fish and Seafood
- Meat
- Soup
- Vegetable
- Desserts and Snacks
- Drink and Beverages

Figure 3.22 Database design of dish.

3.6 User and Admin Module

3.6.1 Module's Description

User and Admin module covers information and data management regarding to the privileges of three actors; User, non-member User and Admin.

Non-member User has the lowest privileges in the ONPAS. Non-member User has to register so that they can access Personal Data Calculation module, Menu Suggestion and Menu Recommendation module and few features in Menu and Dish Creation module. The detail of User registration is explained in subchapter 3.6.2.

A registered User has privileges in the Personal Data Calculation module, Menu Suggestion and Menu Recommendation module and few features in Menu and Dish Creation module. ONPAS provides the User with user page. The user page of a User can be accessed by log in into the system. The user page provides the last data calculation from Personal Data Calculation module and the list of menu and menu table that the user ever created.

Admin has the full privileges in the admin page. The admin page provides all lists of dish and menu. Admin has full privileges to add, edit or delete the dish and menu. Admin also has privileges in changing the basis value in fuzzy calculation for menu recommendation feature.

3.6.2 User Registration

User registration is a process which needed by non-member User to have the User's privileges. Non-member User can register by going to the registration page and filling all of the required information. All required information will be kept in the database. The database design of user is depicted in Figure 3.23.

Field	Type
user_id	tinyint(3)
user_name	varchar(25)
user_pass	varchar(25)
user_fullname	varchar(512)
user_email	varchar(256)
user_detail	text
user_degree	varchar(50)

Figure 3.23 Database design of user.

'user_id' field keeps the unique id of the user. 'user_name' is the username that is used to login into ONPAS Semantic Web. The username cannot be same as other users. The system automatically checks it if a new User doing the registration. The password of a user will be kept in the 'user_pass' field. 'user_fullname', 'user_email' and 'user_detail' keep the full name of the user, the email of the user and additional detail of the user respectively. 'user_degree' field keeps the priority of the user. Generally this field will give a 'user' priority to the new user. There is another priority which is 'admin'. This degree refers to the Admin. This priority can be only obtained by filling manually in the database. Only the super administrator which has the full access in the ONPAS database server can give this priority.

Chapter 4

System Implementation, Result and Discussion

4.1 Semantic Web Implementation

Chapter 4 explains about the Semantic Web application implementation of the ONPAS. Figure 4.1 shows the home page of the ONPAS. The implementation from the system design which is explained in chapter 3 also divided into four modules. They are Personal Data Calculation module, Menu Suggestion and Recommendation module, Menu and Dish creation module and User and Admin module. All of ontology designs from each module are designed and plotted by using Protégé and stored as OWL file called 'dailymenu.owl'. Each module will have a detail in the web programming implementation using JavaServer Pages and Jena Framework. It includes the script, the web pages (Graphical User Interface / GUI) and the result from every feature in each module.

The second part of chapter 4 is the system evaluation from the research objective. There is also a discussion about the barriers of the research and the possibility of system development.

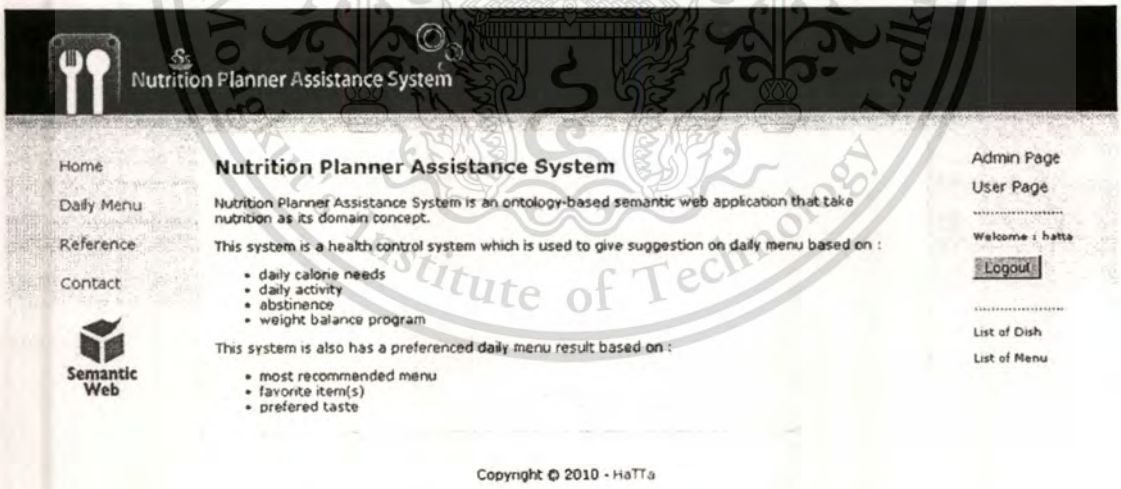


Figure 4.1 Ontology-based Nutrition Planner Assistance System (ONPAS) Semantic Web application.

4.1.1 Personal Data Calculation Module

Personal Data Calculation module's program flow is started by inputting personal data information. The GUI form is made using HTML, Java and JavaScript language. Figure 4.2 depicts the GUI form screenshot. The left part of the form is the personal data form and the right part is the daily menu suggestion preference form.

The form's field, from the name until the activity, is the usual form fields. The checkbox list field of the food item which acts as food abstinence of a user is created differently. The checkbox list of the food abstinence is made from the Food Item ontology using SPARQL query. Firstly, the ontology model of food item is read by using the script follows:

```
OntModel m = ModelFactory.createOntologyModel(); //creating ontology model
m.read("file:fooditem.owl"); //load Food Item ontology from .owl file

//create model in RDF to infer SPARQL
Model data = FileManager.get().loadModel("fooditem.owl");
```

The next step is taking the root classes of the ontology of the food item by the following script:

```
ExtendedIterator category = m.listHierarchyRootClasses();
```

The category name is taken one by one, and then SPARQL query is executed to get the subclasses of each category. The SPARQL query script is started with declaring the PREFIX of OWL, RDF and the URI of food item. The following script is shown the process of taking out each category and its subclasses excluding the script to print them out:

```
while(category.hasNext()) {
       OntResource mp = (OntResource)category.next(); //get category resource
       ..... //print out the category of food item
       //SPARQL query for taking the subclasses of each food item category
       String queryString = ("PREFIX owl: <http://www.w3.org/2002/07/owl#>"+
       "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>"+
        "PREFIX fooditem: <http://hatta.web.id/fooditem.owl#>"+
       "SELECT ?c WHERE { ?c rdfs:subClassOf
       fooditem:"+mp.getURI().replace("http://hatta.web.id/fooditem.owl#",
"")+" }
       ORDER BY ?c" );
Query query = QueryFactory.create(queryString);
QueryExecution qe = QueryExecutionFactory.create(query, data);
com.hp.hpl.jena.query.ResultSet results = qe.execSelect();
       while(results.hasNext()) {
       QuerySolution row = (QuerySolution)results.next();
       RDFNode object = row.get("c");
       ..... //print out the subclasses of each category
       } qe.close(); }
```

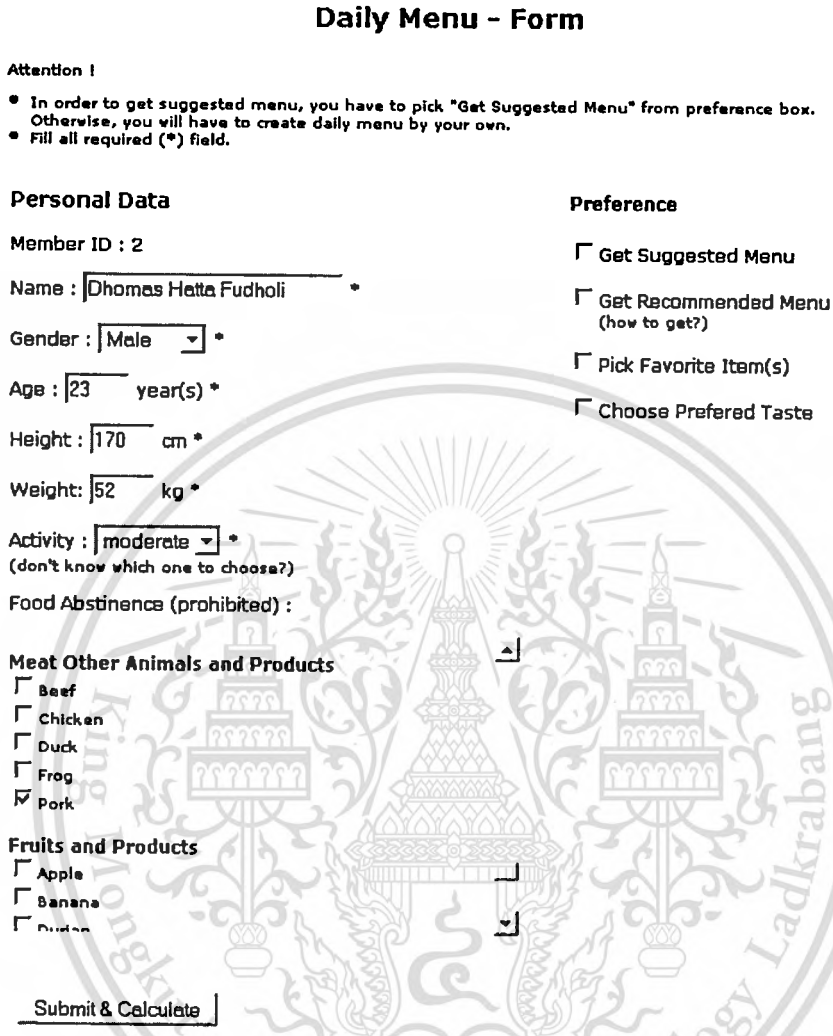


Figure 4.2 Personal data and daily menu form.

Calculating the daily calorie needs from the personal data that already inputted by the user will be explained next. Creating prefix and reading the ontology model is needed to know the ontology design. The following script is used to read the ONPAS ontology model which is stored in 'dailymenu.owl':

```
String dailymenuURI = "http://www.hatta.web.id/dailymenu.owl#";
PrintUtil.registerPrefix("dailymenu", dailymenuURI);

OntModel m = ModelFactory.createOntologyModel();
m.read("file:D:/NetBeansProjects/DailyMenu/web/data/dailymenu.owl");
```

The next step is creating an individual. The new individual is the user as a Person in the Personal Data Calculation ontology. The Person class in the Personal Data Calculation ontology is categorized into six subclasses. The way to make an individual in each category is the same. The following script is used to create a new individual that is categorized to a Person with has an age range 18 to 29:

```
if (personAge>=18 & personAge<30){
       OntClass person = m.getOntClass(dailymenuURI+"Age_18-29");
       Individual p = m.createIndividual(dailymenuURI+personName,person);}
```

The other personal data information that will be stored is the gender, age, height, weight and person's abstinence. Each of information has different data type. The gender will be stored as a string. Age, height, weight and abstinence will be stored as an integer, float, float and string respectively. The way to store those data is the same. The following script is Jena framework API representation to store the gender of a person:

```
DatatypeProperty hasGender = m.getDatatypeProperty(dailymenuURI+"hasGender");
Literal pg = m.createTypedLiteral(personGender, XSDDatatype.XSDstring);
Statement gender = m.createStatement(p, hasGender, pg);
m.add(gender);
```

Each user has his/her own OWL file to store his/her personal data and personal data calculation. The example of the OWL file name is 'dailymenu-2.owl'. The number 2 on the OWL file name is the user's ID. The following script is the way to store the data into the person's OWL file:

```
String output_filename = "dailymenu-"+session.getAttribute("user_id")+".owl";
m.write(new PrintWriter(new FileOutputStream(output_filename)));
```

User must choose the level of their activity. There are three kinds of level; light, moderate and heavy. For an elderly there will be one general activity factor. In each category level of activity and each gender contains different value of activity factor. The value in each category level and gender is already defined and stored in the main ontology model in 'dailymenu.owl'. To infer the activity factor, the following script is executed:

```
Resource configuration =  m.createResource();
configuration.addProperty(ReasonerVocabulary.PROPruleMode,"hybrid");
configuration.addProperty(ReasonerVocabulary.PROPruleSet,"dailymenu-
calculation.rules");
```

```
Reasoner reasoner =
GenericRuleReasonerFactory.theInstance().create(configuration);

Model data = FileManager.get().loadModel("dailymenu-
"+session.getAttribute("user_id")+".owl");
InfModel infmodel = ModelFactory.createInfModel(reasoner, data);
```

'dailymenu-calculation.rules' is a file that stores Jena rules to infer the activity factor value to the user. The following script is one of Jena rules that gives an activity factor value for a male person:

```
[hasMaleActivityFactor:
    (?p dailymenu:hasGender "Male")
    (?p dailymenu:hasActivity ?a)
    (?a dailymenu:hasMaleActivityFactor ?af)
->  (?p dailymenu:hasActivityFactor ?af)]
```

BMR, *TEE* and *BMI* (for adult only) are calculated using the ordinary calculation script.

All of the data and calculation result is kept in the person's OWL file. The following XML script is an example of OWL representation for the personal data:

```
<rdf:Description rdf:about="http://www.hatta.web.id/dailymenu.owl#Dhomas Hatta
Fudholi">
<hasHeight
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">170</hasHeight>
<has BMI rdf:datatype="http://www.w3.org/2001/XMLSchema#float">17.99</has BMI>
<hasActivity rdf:resource="http://www.hatta.web.id/dailymenu.owl#Moderate_A18-
A63"/>
<hasAbstinence
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Pork</hasAbstinence>
<hasTEE rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2600</hasTEE>
<hasWeight
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">52</hasWeight>
<rdf:type rdf:resource="http://www.hatta.web.id/dailymenu.owl#Age_18-29"/>
<hasGender
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Male</hasGender>
<has BMIStatus
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Underweight</has BMIStat
us>
<hasActivityFactor
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.78</hasActivityFactor>
<hasBasalMetabolicRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1475.16</hasBasalMetabol
icRate>
<hasAge rdf:datatype="http://www.w3.org/2001/XMLSchema#int">23</hasAge>
<hasTotalEnergyExpenditure
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">2625.78</hasTotalEnergyE
xpenditure>
</rdf:Description>
```

The result will be shown afterward. Figure 4.3 shows the example of calculation result which is located in the right side of the table. It also recapitulates person's personal data in the left side. When a person takes weight balance program, there will be recalculation process to add or subtract the daily calorie needs. Figure 4.4 demonstrates the example of recalculation result table.

Daily Menu - Result

Personal Data	Data Calculation
Name : Dhomas Hatta Fudholi Gender : Male Age : 23 year(s) Height : 170 cm Weight : 52 kg Activity : moderate Abstinence : Pork	Activity Factor : 1.78 Basal Metabolic Rate : 1475.16 kcal Total Energy Expenditure : 2625.78 kcal BMI :17.99 BMI Status : Underweight Weight Balance Program : Add0.5kg/week Submit & ReCalculate

Figure 4.3 Calculation result.

Daily Menu - Weight Balance Program - Result

Personal Data	Data Calculation
Name : Dhomas Hatta Fudholi Gender : Male Age : 23 year(s) Height : 170 cm Weight : 52 kg Activity : moderate Abstinence : Pork	Activity Factor : 1.78 Basal Metabolic Rate : 1475.16 kcal Total Energy Expenditure (balanced) : 3125.78 kcal BMI : 17.99 BMI Status : Underweight Weight Balance Program : Add0.5kg/week

Figure 4.4 Recalculation result of weight balance program.

4.1.2 Menu Suggestion and Recommendation Module

User can choose to get menu suggestion and menu recommendation by picking the preference in the personal data and daily menu form that is shown in Figure 4.2. There will be one option to get the suggested menu and three options of menu recommendation. Figure 4.5 shows the preference detail in the daily menu form. To get the menu recommendation, user needs to get the menu suggestion at first. Therefore, when the user picks the menu recommendation option, menu suggestion option is automatically picked.

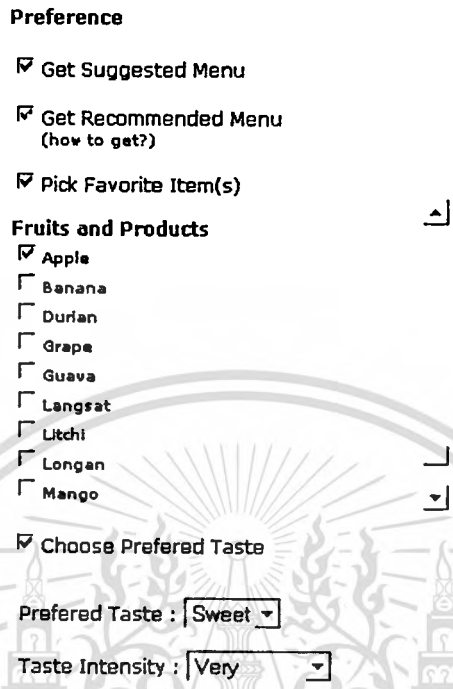


Figure 4.5 Daily menu form's preference for menu suggestion and recommendation.

The next explanations are the system implementation and result for the menu suggestion and followed by the system implementation and result for the menu recommendation.

a. Menu Suggestion

The main idea of giving suggestion of menu is to suggest a daily menu that has a total calorie value balance with one's daily calorie needs. In other case, a person has some abstinence of food. The ONPAS will separate the menu that is suggested to a person when the menu has the same items with the abstinence of a person. Some rules are needed to infer the suggested menu to a person. The rules will be stored as 'dailymenu-suggestion.rules' file in ONPAS. The Jena rules of menu suggestion that is generated will be dependent whether a person has food abstinence or not. The following Jena rule is a rule that is generated when a person does not have any food abstinence:

```
[hasMenu:
(?p dailymenu:hasTEE ?tee)
(?m dailymenu:hasCalorie ?tee)
->  (?p dailymenu:hasMenu ?m)]
```

If a person has some food abstinence, the rule will be more complicated since the system will separate and taking out the menu that is prohibited to a person. Besides inferring `hasMenu` property to a person, the rules will also inferring `hasProhibitedMenuforAbstinence` property. The following Jena rules are the rules to suggest menu and separate the prohibited menu:

```
[hasProhibitedMenuforAbstinence:
(?p dailymenu:hasTEE ?tee)
(?m dailymenu:hasCalorie ?tee)
(?p dailymenu:hasAbstinence ?abs)
(?m dailymenu:hasMenuItem ?abs)
->  (?p dailymenu:hasProhibitedMenuforAbstinence ?m)]
[hasMenu:
(?p dailymenu:hasTEE ?tee)
(?m dailymenu:hasCalorie ?tee)
(?p dailymenu:hasAbstinence ?abs)
noValue(?m dailymenu:hasMenuItem ?abs)
noValue(?p dailymenu:hasProhibitedMenuforAbstinence ?m)
->  (?p dailymenu:hasMenu ?m)]
```

`?p` represent a person and `?m` is the menu. `noValue` is a built-in function in Jena rules that act as a negation. `noValue` in the Jena rules above takes an important part to exclude the menu which is prohibited for a person.

The suggested menu and the prohibited menu are also kept in the user's OWL file. The following XML script is an example of an OWL from `hasMenu` and `hasProhibitedMenuforAbstinence` properties.

```
<hasMenu rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu-2600-50"/>
<hasMenu rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu-2600-38"/>
<hasMenu rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu-2600-48"/>
<hasProhibitedMenuforAbstinence
rdf:resource="http://www.hatta.web.id/dailymenu.owl#HaT Menu 2"/>
```

ONPAS uses the following two SPARQL query to take out the suggested menu and the menu that is prohibited:

```
PREFIX dailymenuURI: <http://www.hatta.web.id/dailymenu.owl#>
SELECT ?p ?m WHERE { ?p dailymenuURI:hasMenu ?m } ORDER BY ?m
PREFIX dailymenuURI: <http://www.hatta.web.id/dailymenu.owl#>
SELECT ?p ?m WHERE { ?p dailymenuURI:hasProhibitedMenuforAbstinence ?m } ORDER
BY ?m
```

Figure 4.6 shows the screenshot of giving menu suggestion to the user. The suggested menu is listed below the personal data calculation result. The prohibited menu is listed in the bottom of the list. To show that the prohibited menu algorithm works, on the right side of the figure, the

detail of a menu shows that the menu contains the user's food item abstinence. The detail of the menu can be shown by clicking a certain menu in the suggestion list.

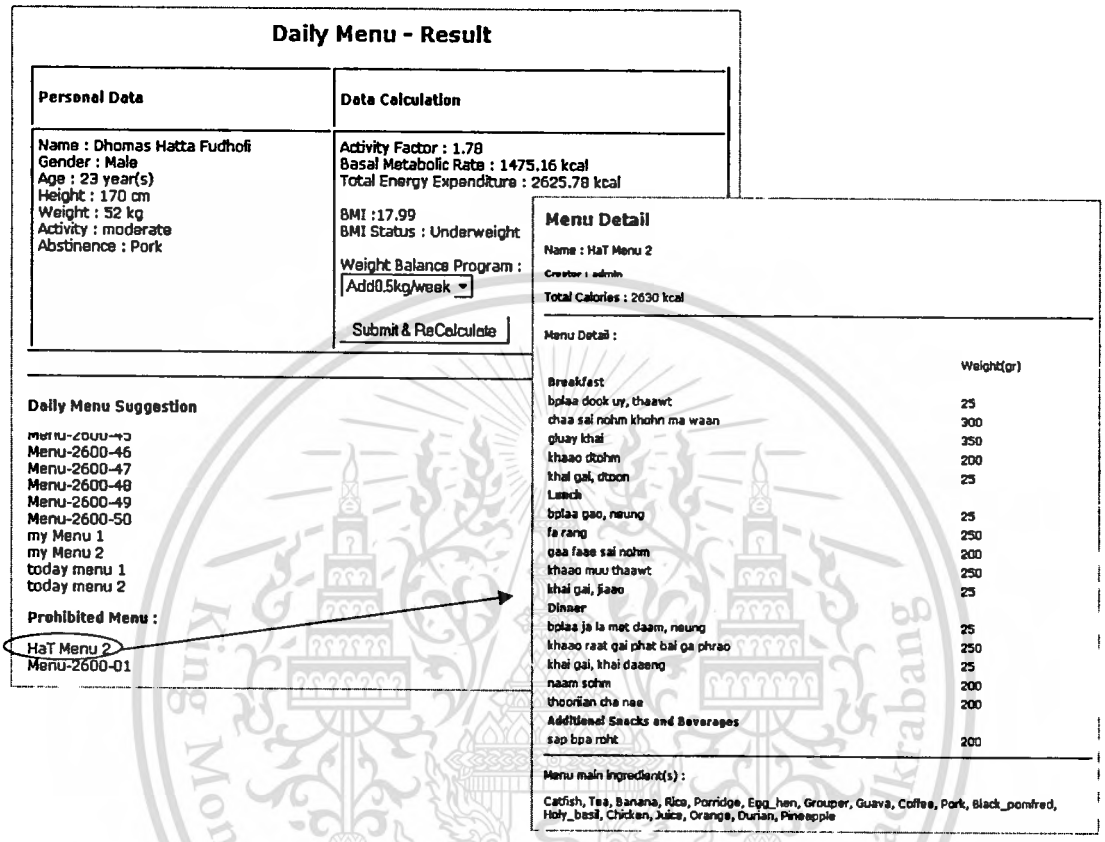


Figure 4.6 Menu suggestion result example.

b. Menu recommendation based on rate, vote and price.

Menu recommendation based on rate, vote and price is the first recommendation option in the daily menu form's preference (Figure 4.5). The way to get the value of rate, vote and price is explained in chapter 3. Figure 4.7 shows the screenshot of detail data of rate, vote and price in a menu that is located in the bottom of menu's detail. If a menu does not have the detail of these three values, the recommendation value of this menu will not be calculated and will not be recommended to a person.

The next step is to get the fuzzy value of rate, vote and price. The fuzzy value of high rate, low rate, high vote, low vote, cheap price and expensive price are achieved by evaluating the value of rate, vote and price on its membership degree function as shown in Figure 3.11. The membership degree function is a trapezium type function. There are two variable that is need to be filled in each function; a and b. The value of a and b can be assigned in the administration area, as shown in Figure 4.8.

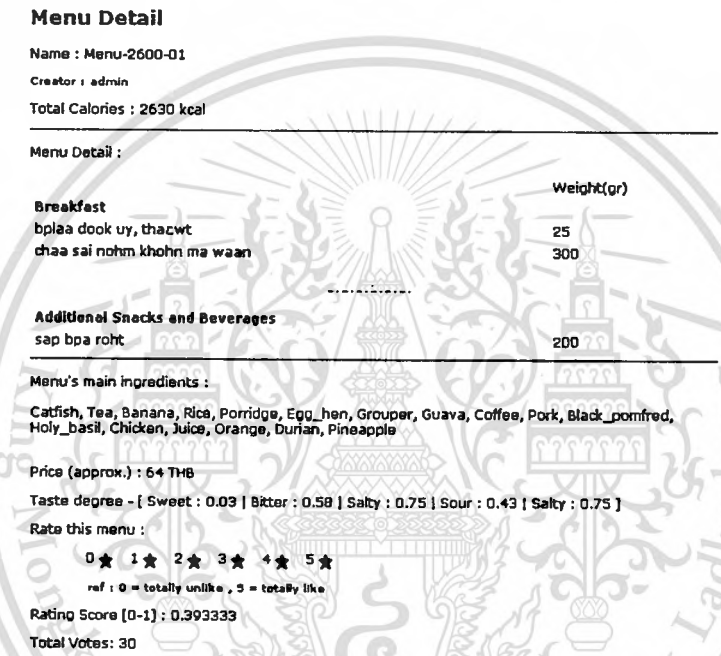


Figure 4.7 Detail of a menu that in the rate, vote, price and taste degree data.

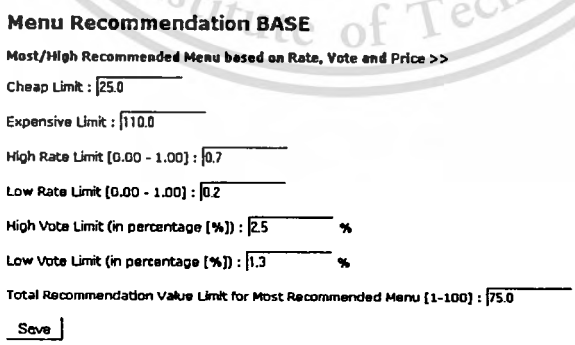


Figure 4.8 Administration area of base value in menu recommendation based on rate, vote and price.

After evaluating the real value of rate, vote and price, some new properties is assigned to the detail of the menu. `hasHighRate`, `hasLowRate`, `hasHighVote`, `hasLowVote`, `hasCheapDegree` and `hasExpensiveDegree` properties are already defined in the 'dailymenu.owl'. All of these properties will be assigned in the menu and filled with the *md* value. These properties have *md* of high rate, low rate, high vote, low rate, cheap price and expensive price respectively. The following script is the example of filling `hasCheapDegree` property in `menu` with the float data type value of variable `cheap_degree2` and add it to the ontology model which is declared as variable `m4`:

```
Individual menu = m3.getIndividual(dailymenuURI+menus);
DatatypeProperty hasCheapDegree =
m3.getDatatypeProperty(dailymenuURI+"hasCheapDegree");
Literal cheapDegree = m3.createTypedLiteral(cheap_degree2,
XSDDatatype.XSDfloat);
com.hp.hpl.jena.rdf.model.Statement menuCheapDegree = m3.createStatement(menu,
hasCheapDegree, cheapDegree);
m4.add(menuCheapDegree);
```

The other properties values are assigned using the similar way.

Fuzzy rules used in this recommendation process are shown in Table 3.1. There are eight fuzzy rules. The Jena rules to represent all of eight fuzzy rules will be generated if the first option of menu recommendation is picked. The Jena rules that are generated use `min` built-in function. `min` built-in function in Jena rule statement takes the minimum value from the second and the third variable and put the minimum value in the first variable. All of the rules use AND operator. The AND operator takes the minimum value of the statement in the rules.

The following Jena rules represent the eight fuzzy rules and results the value of low recommendation, medium recommendation and high recommendation that will be used in defuzzification process:

```
[hasLowRecommendationValue:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasCheapDegree ?c)
(?m dailymenu:hasExpensiveDegree ?e)
(?m dailymenu:hasLowRate ?lr)
(?m dailymenu:hasLowVote ?lv)
(?m dailymenu:hasHighVote ?hv)
min(?c,?e,?min)
min(?min,?lr,?min2)
min(?min2,?lv,?min3)
min(?min3,?hv,?min4)
->  (?m dailymenu:hasLowRecommendationValue ?min4)]
```

```
[hasMediumRecommendationValue:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasCheapDegree ?c)
(?m dailymenu:hasExpensiveDegree ?e)
(?m dailymenu:hasHighRate ?hr)
(?m dailymenu:hasLowVote ?lv)
(?m dailymenu:hasHighVote ?hv)
min(?c,?e,?min)
min(?min,?hr,?min2)
min(?min2,?lv,?min3)
min(?min3,?hv,?min4)
->  (?m dailymenu:hasMediumRecommendationValue ?min4)]

[hasHighRecommendationValue:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasCheapDegree ?c)
(?m dailymenu:hasHighRate ?hr)
(?m dailymenu:hasHighVote ?hv)
min(?c,?hr,?min)
min(?min,?hv,?min2)
->  (?m dailymenu:hasHighRecommendationValue ?min2)]
```

Low recommendation value, medium recommendation value and high recommendation value of a menu will be stored in `hasLowRecommendationValue`, `hasMediumRecommendationValue`, and `hasHighRecommendationValue` properties, respectively. These three values hold important role in deffuzification process. Figure 3.12 shows the defuzzification function for the menu recommendation based on rate, vote and price and equation *(3.3)* is the weight average calculation for defuzzification. This calculation is done by the general arithmetic algorithm in JSP.

The recommended menu is the menu that has *TRV* equal or more than the limit value that is assigned. The limit value can be assigned in the administration area. Person will have a new asserted property which is `hasRecommendedMenu` in the end of the process. The following Jena rule is generated to give menu recommendation:

```
[hasRecommendedMenu:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasTotalRecommendationValue ?trv)
greaterThan(?trv 75)
->  (?p dailymenu:hasRecommendedMenu ?m)]
```

`greatherThan` in Jena rules above is a Jena rules built-in function that is used to check whether variable `?trv` has the value greater than 75. `?p` represents the person and `?m` represents the menu. `?trv` is the total recommendation value of a menu. The final recommended menu result is listed below the daily menu suggestion as demonstrated in Figure 4.9. The following XML is the detail information of the menu in OWL when the menu recommendation based on rate, vote and price is executed:

```
<rdf:Description rdf:about="http://www.hatta.web.id/dailymenu.owl#Menu-2600-
02">
<rdf:type rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu"/>
<hasCalorie
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2600</hasCalorie>
<hasTotalRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">80.0</hasTotalRecommenda
tionValue>
<hasHighRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.45882352</hasHighRecom
mendationValue>
<hasHighVote
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</hasHighVote>
<hasMediumRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasMediumRecommenda
tionValue>
<hasLowRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.432558</hasLowRate>
<hasExpensiveDegree
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.54117647</hasExpensive
Degree>
<hasLowRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasLowRecommendatio
nValue>
<hasLowVote
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasLowVote>
<hasHighRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.56744197</hasHighRate>
<hasMenuItem
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Mushroom</hasMenuItem>
<hasCheapDegree
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.45882352</hasCheapDegr
ee>
</rdf:Description>
```

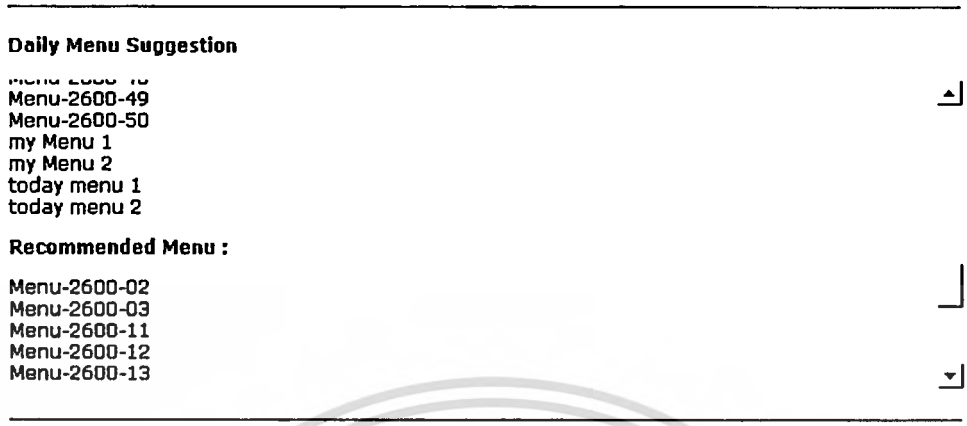


Figure 4.9 Menu recommendation result list.

c. Menu recommendation based on taste.

Taste will be the only one factor that is used in this recommendation process. There are five kinds of taste; sweet, bitter, salty, sour, and spicy. Menus that have the information in the membership degree of each taste will be included when a person preferred to get a recommendation based on taste. The detail of each taste degree can be seen in Figure 4.7.

In the menu recommendation preference that is shown in Figure 4.5, a person needs to select the taste that they prefer and the intensity of the taste. The intensity of the taste is the fuzzy linguistic value. The linguistic value can be administered in the administration area by the administrator. Admin not only can edit the linguistic value and its precise value but also can add new linguistic value and delete linguistic value. Figure 4.10 is the administration page of the menu recommendation based on taste. The other base value that can be administered is the tolerance value of the membership degree of taste.

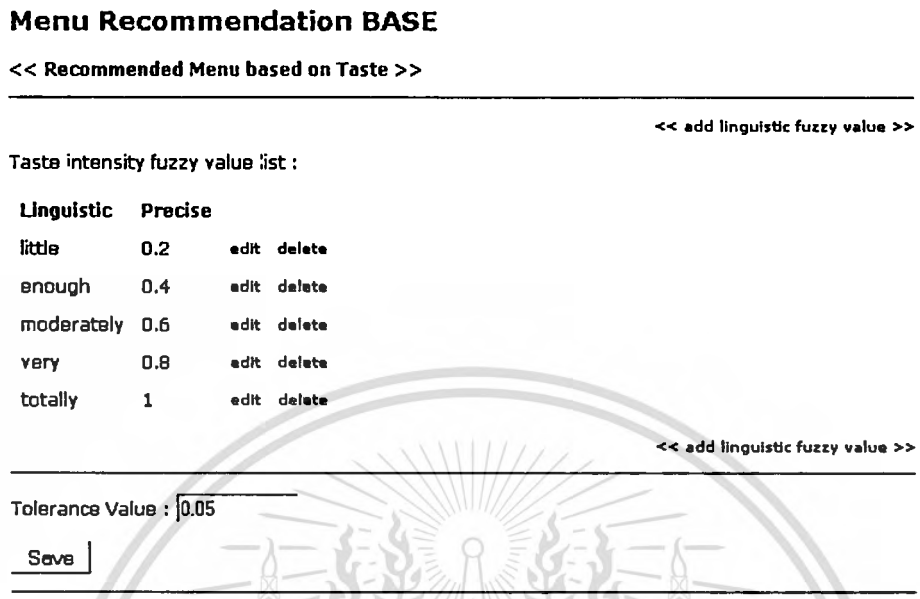


Figure 4.10 Menu recommendation based on taste administration area.

Each membership degree value of each taste in a menu will be assigned as properties value for five menu's properties in ontology. Those properties are `hasSweetTaste`, `hasBitterTaste`, `hasSaltyTaste`, `hasSourTaste`, `hasSpicyTaste`. The following XML script is an example of OWL that stores those five properties:

```
<rdf:Description rdf:about="http://www.hatta.web.id/dailymenu.owl#Menu-2600-
11">
<hasSpicyTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.59</hasSpicyTaste>
<hasSourTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.02</hasSourTaste>
<hasSaltyTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.59</hasSaltyTaste>
<hasBitterTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.34</hasBitterTaste>
<hasSweetTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.79</hasSweetTaste>
<rdf:type rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu"/>
<hasCalorie
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2600</hasCalorie>
</rdf:Description>
```

To give the recommendation there will be a Jena rule that is generated. The following Jena rule is the rule to give recommendation based on taste with the preferred taste that is very sweet with tolerance value equal to 0.05:

```
[hasRecommendedMenu:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasSweetTaste ?t)
greaterThan(?t 0.749)
lessThan(?t 0.851)
->  (?p dailymenu:hasRecommendedMenu ?m)]
```

?p is the person and ?m is the menu. ?t is the membership degree of sweet taste. As seen in Figure 4.9, the tolerance value is assigned in 0.05 and the linguistic value of very has the precise value 0.8 so the rules are generated to take the sweet taste menu with the degree greater than 0.749 and lower than 0.851.

d. Menu recommendation based on favorite item.

In contrast with the prohibited menu exclusion, when the user picks some favorite items for a recommendation, the ONPAS will recommend the menu which has the same food item. The following Jena rule is the rule that is generated when a person picks to get menu recommendation based on favorite item:

```
[hasRecommendedMenu:
(?p dailymenu:hasMenu ?m)
(?p dailymenu:hasFavoriteItem ?fi)
(?m dailymenu:hasMenuItem ?fi)
->  (?p dailymenu:hasRecommendedMenu ?m)]
```

?p represents the person. ?m represents the menu and ?fi represents the food item which is favourable by the person.

e. Menu recommendation based on rate, vote, price, taste and favorite item.

Menu recommendation based on rate, vote and price, menu recommendation based on taste and menu recommendation based on favorite item can be combined each other. The following explanation describes the possibilities of combining all recommendation option. Figure 4.10 shows all picked option in the preference of the menu suggestion and menu recommendation.

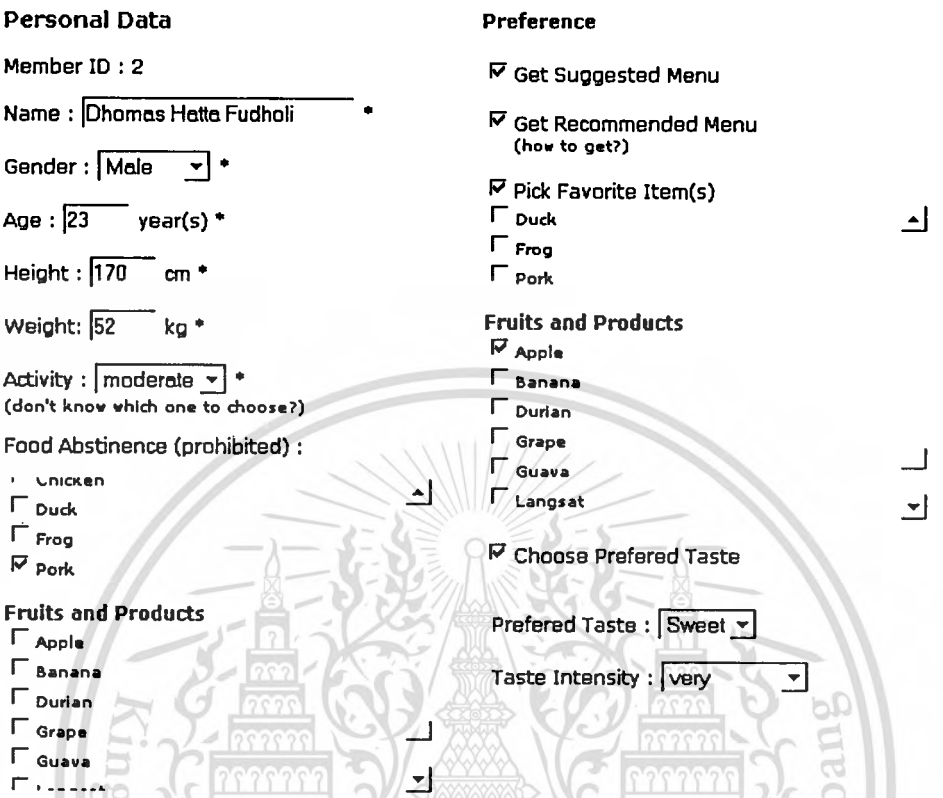


Figure 4.11 Personal data and daily menu form with all recommendation option is being picked.

All of properties value assignment that is done by each recommendation process is also applied to this recommendation process. The following OWL is an example of the data that is owned by 'Menu-2600-11' as the result of the menu's properties value assignment process for recommendation:

```
<rdf:Description rdf:about="http://www.hatta.web.id/dailymenu.owl#Menu-2600-
11">
<hasExpensiveDegree
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.2117647</hasExpensiveD
egree>
<hasLowRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasLowRecommendatio
nValue>
<hasHighRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.40888801</hasHighRecom
mendationValue>
<hasLowRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.59111195</hasLowRate>
<hasMenuItem
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Apple</hasMenuItem>
```

```
<hasCheapDegree
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.78823529</hasCheapDegr
ee>
<hasSaltyTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.59</hasSaltyTaste>
<hasCalorie
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2600</hasCalorie>
<hasLowVote
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasLowVote>
<hasSweetTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.79</hasSweetTaste>
<hasTotalRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">80.0</hasTotalRecommenda
tionValue>
<hasSourTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.02</hasSourTaste>
<hasHighRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.40888801</hasHighRate>
<hasBitterTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.34</hasBitterTaste>
<rdf:type rdf:resource="http://www.hatta.web.id/dailymenu.owl#Menu"/>
<hasHighVote
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</hasHighVote>
<hasSpicyTaste
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.59</hasSpicyTaste>
<hasMediumRecommendationValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.0</hasMediumRecommenda
tionValue>
</rdf:Description>
```

The Jena rule that is generated combines the rules from all recommendation processes. The following Jena rule is generated when the option of recommendation is chosen as same as in Figure 4.10:

```
[hasRecommendedMenu:
(?p dailymenu:hasMenu ?m)
(?m dailymenu:hasTotalRecommendationValue ?trv)
greaterThan(?trv 75.0)
(?p dailymenu:hasFavoriteItem ?fi)
(?m dailymenu:hasMenuItem ?fi)
(?m dailymenu:hasSweetTaste ?t)
greaterThan(?t 0.749)
lessThan(?t 0.851)
->  (?p dailymenu:hasRecommendedMenu ?m)]
```

Figure 4.11 shows the result of the calculation, menu suggestion and menu recommendation process from the input data in Figure 4.10. In Figure 4.11 there is one recommended menu which has name 'Menu-2600-11'. As mentioned before, 'Menu-2600-11' that is recommended to the user has all of requirements that the user request. It has *Total Recommendation Value* equal to 80.0, has sweet taste membership degree 0.79 and has an apple in the menu.

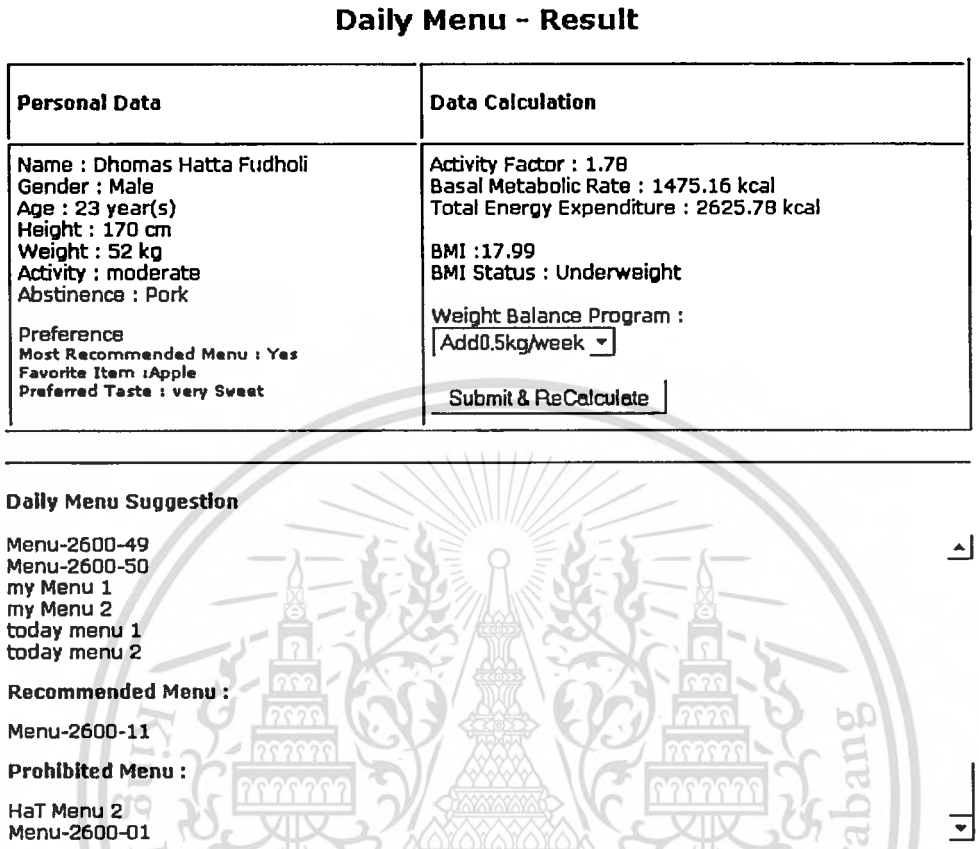


Figure 4.12 The result of the calculation, menu suggestion and menu recommendation process.

4.1.3 Menu and Dish Creation Module

Menu and Dish Creation module consists of menu creation process, menu table creation process, and dish creation process.

a. Menu creation

The creation of menu (daily menu) can be done in two ways. The first way is by filling all of information that is needed to make one daily menu. The information should be accurate or come from scientific literature. The first way can be only done by the administrator. The second one is by arranging the dish from the database so that it becomes one daily menu.

Menu creation from the literature needs all required information in the menu creation's form which is shown in Figure 4.13 to be filled. The information that must be filled is the name, total calorie, menu items and the detail of the menu. The detail of the menu shows the dishes that are inputted manually for the breakfast, lunch, dinner and additional snack for one day. The name of a menu cannot be the same to the existing menu in the database. Therefore, the system will check automatically whether the name is already used. When all required field is filled then the administrator can save it and the new menu will be shown in the menu list.

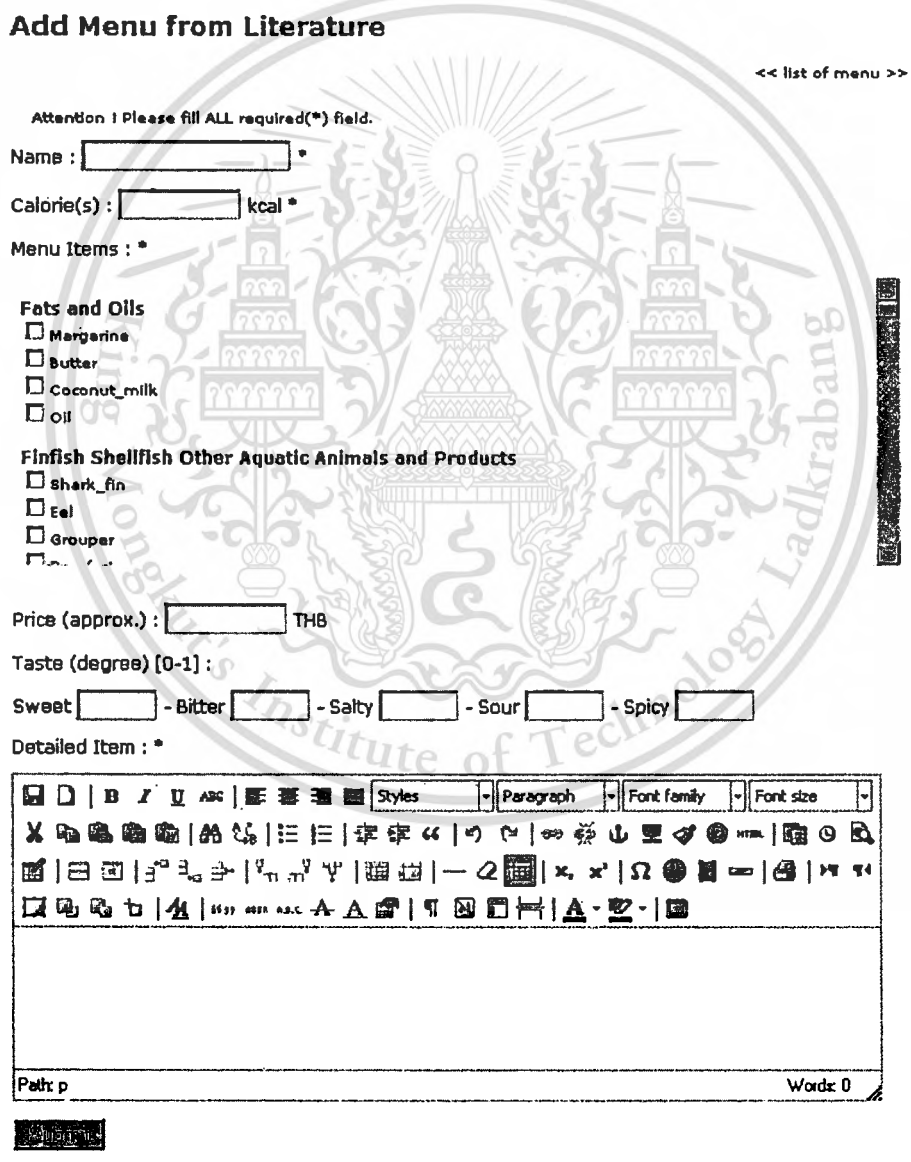
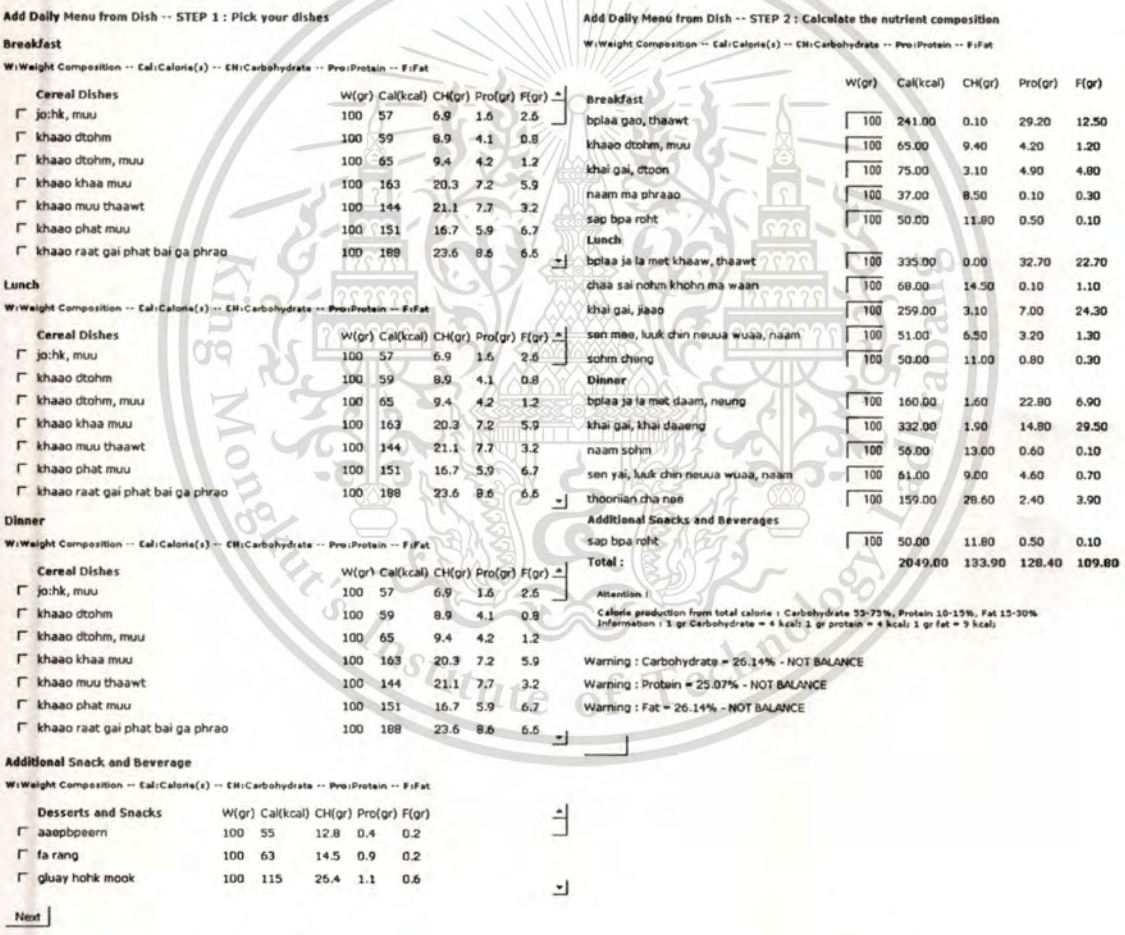


Figure 4.13 Menu creation from literature form.

The second way to create menu is by arranging the dishes. Firstly, the dishes for the breakfast, lunch, dinner and the additional snack are picked. In each meal time, the creator can pick more than one dish in all type of dish. Figure 4.14(a) shows the first step of menu creation from dish. It shows the complete list of dish in each meal time so that it can be picked. There is also information of the weight composition for a certain nutrient content. Composing the nutrient composition by arranging the weight of each dish will be the next step. Figure 4.14(b) shows the web page of this step. The composition of carbohydrate, protein and fat should be balanced. When the nutrient composition is balanced the warning will be disappear.



(a) (b)

Figure 4.14 (a) Picks the dish step of creating menu from dish.

(b) Arranges nutrition composition step of creating menu from dish.

The last step of making new menu from dish is to save the menu in the database. Figure 4.15 is the last form to fill to make menu from dish. After arranging the composition of the nutrient, the creator is asked to fill some information about the menu. The information is name, price, and the degree of taste. The price and the degree of taste can be filled blank. If those fields are left empty, the recommendation based on price and taste will not include this menu.

There is a role to the user when a user created menu from dish. They have to do the personal data calculation at first. When the user goes to the second step, which is arranging nutrition composition, the user has to make the total calorie of the new menu as same as the daily calorie needs of the user (*TEE*).

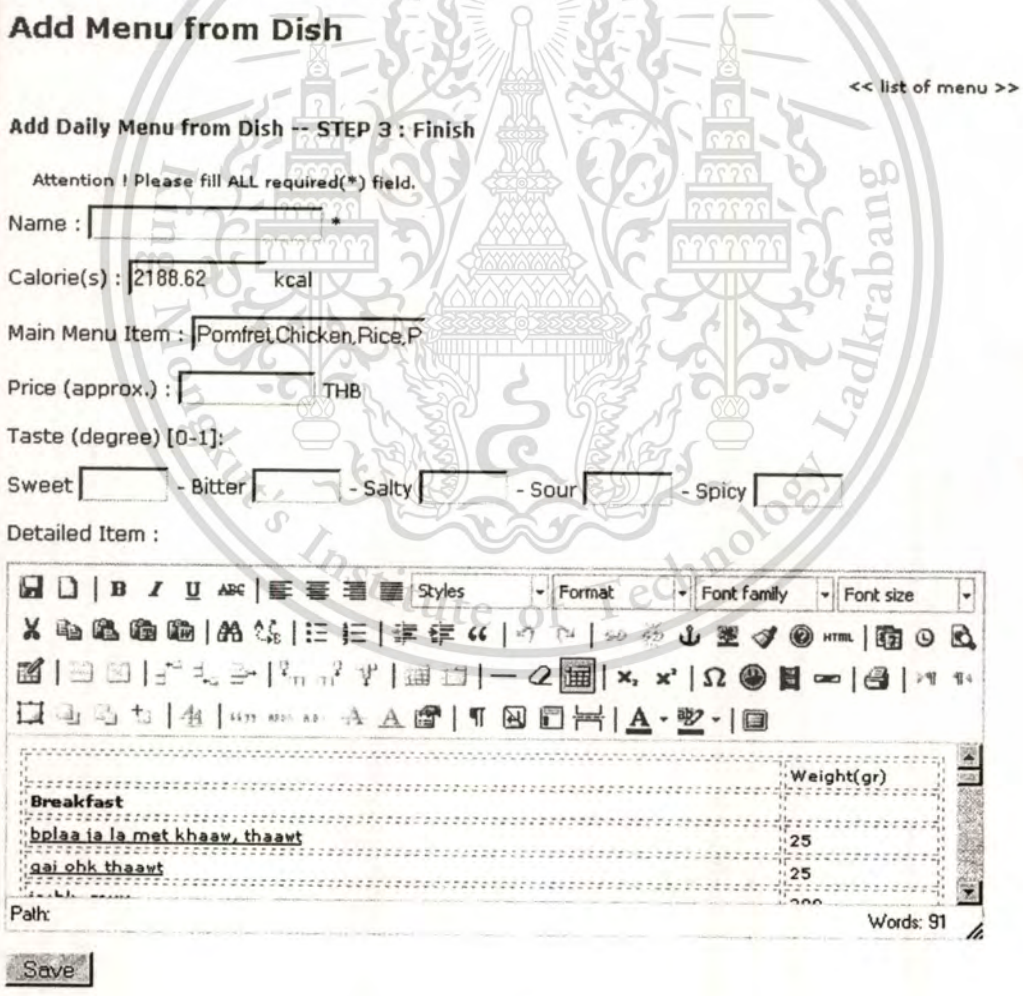


Figure 4.15 Form to add required information to the new menu.

The list of all menus in ONPAS is shown in Figure 4.16(a). The menu that is created by admin and user is different in color. Everyone can browse the list of menu, non-member user are included. When someone login to the system as an Admin, list of menu will also show the edit and delete links to edit or delete the menu. A menu that is created by a certain user is listed in his/her user page (Figure 4.16(b)). The user has a privilege to edit his/her own menu creation.

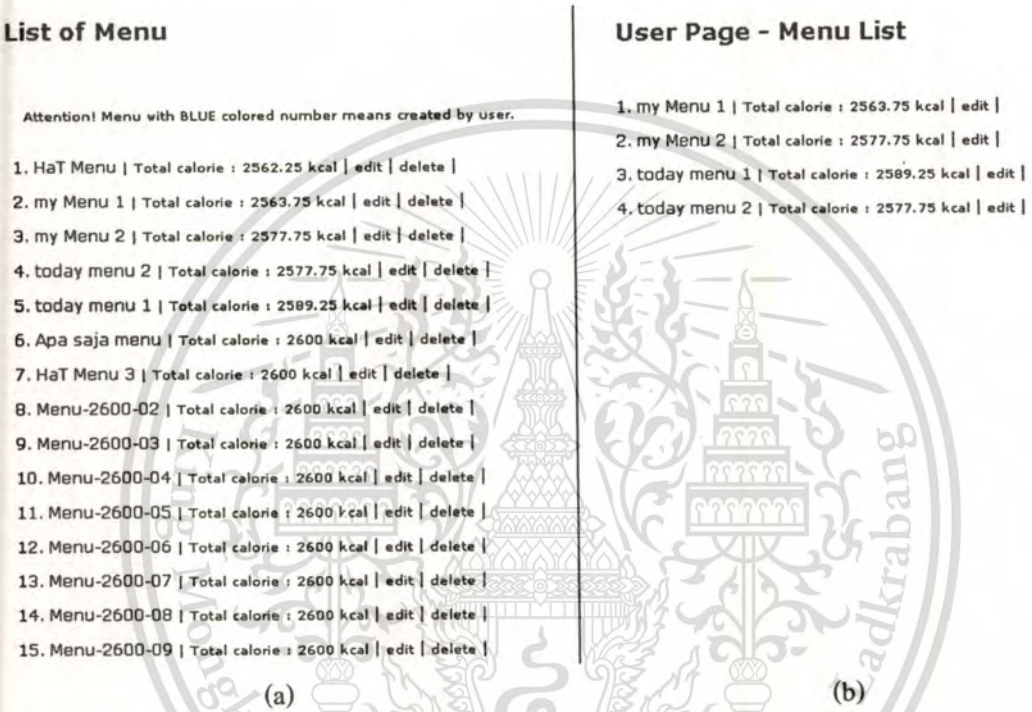


Figure 4.16 (a) List of all menus. (b) Example of menu's list that a certain user created.

b. Menu table creation

Menu table can be created after the user does the calculation of his/her daily calorie needs. Figure 4.17 shows the options that should be chosen when creating menu table. The first option is the source of the menu table. If the user picks to get suggestion in daily menu they can use the suggested menu as the source. The second source is from the dish database. The next option is the amount of day for the menu table. The user can make maximum seven days in one menu table.

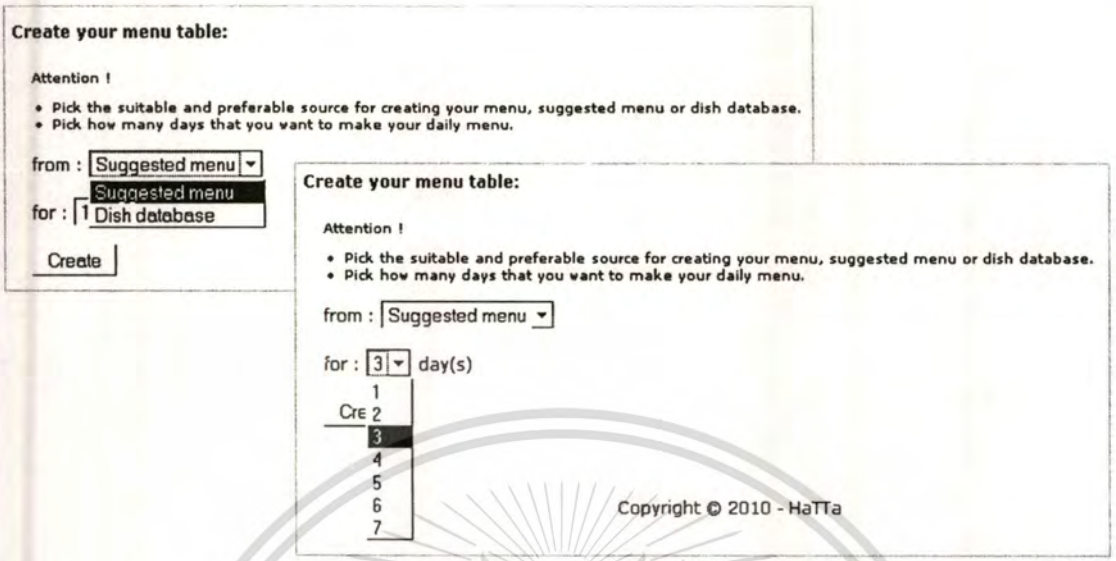


Figure 4.17 Menu table creation options.

When creating menu table from the suggested menu, there will be two more steps to make it done. Figure 4.18 shows the first step. User needs to pick which menu that is suitable to be the menu table for a certain days. If the user picks to get menu recommendation, the recommended menu will be listed in different color so that the user can easily differentiate it.

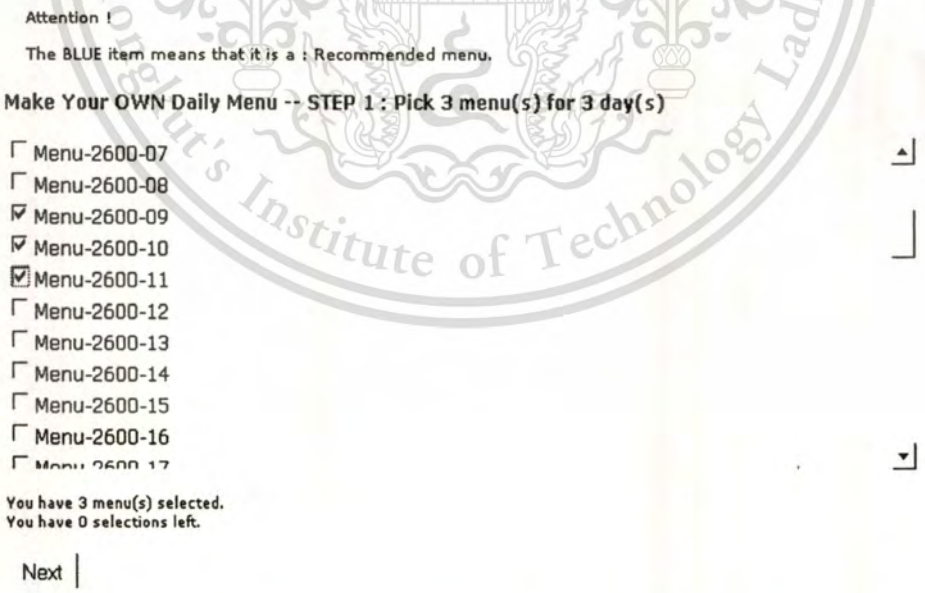


Figure 4.18 Picks menu from suggested menu for menu table creation process.

The suggested menu that is picked to be the element of the menu table is arranged by ONPAS as a table. Figure 4.19 is the screenshot of the menu table before is saved. The user needs to fill the detail of the menu table which is the name of the menu table before it is saved.

The process of creating menu table from the dish database is similar to creating menu from dish. User needs to pick some dishes for each meal time and needs to arrange the nutrient composition. Note that the total calorie of the menu should be balanced with the user's daily calorie needs. There is one different thing when creating menu and menu table from dish. The different is the multiplication of picking the dish and arranging the nutrient composition regarding to the amount of the day. Figure 4.20 shows the step of creating menu table from the dish. It is really similar except in the categorization of day. User needs to pick some dishes in all meal time in each day. User also needs to arrange the nutrient composition for all days. The last step would be saving the menu table that the user created.

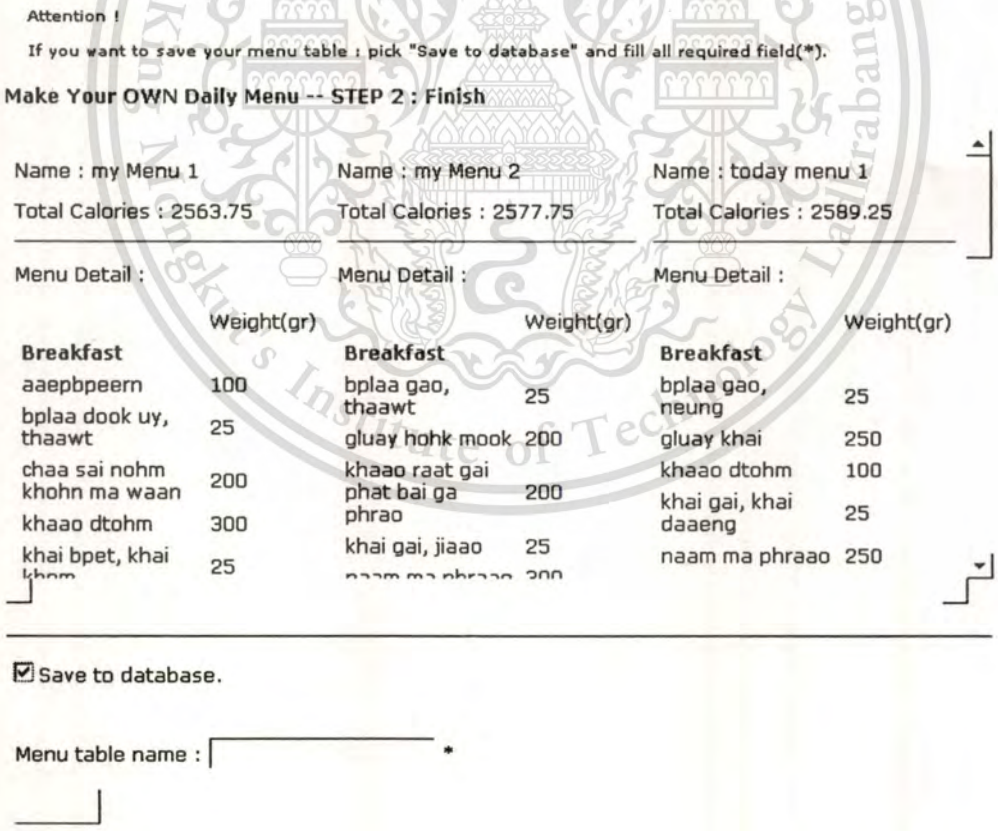


Figure 4.19 Menu table arrangement.

User can have more than one menu table. To have more than one menu table, the user needs to repeat the step of creating menu table. All of menu tables that a user has is listed in the user page. Figure 4.21 shows the example of menu table list that a user has.

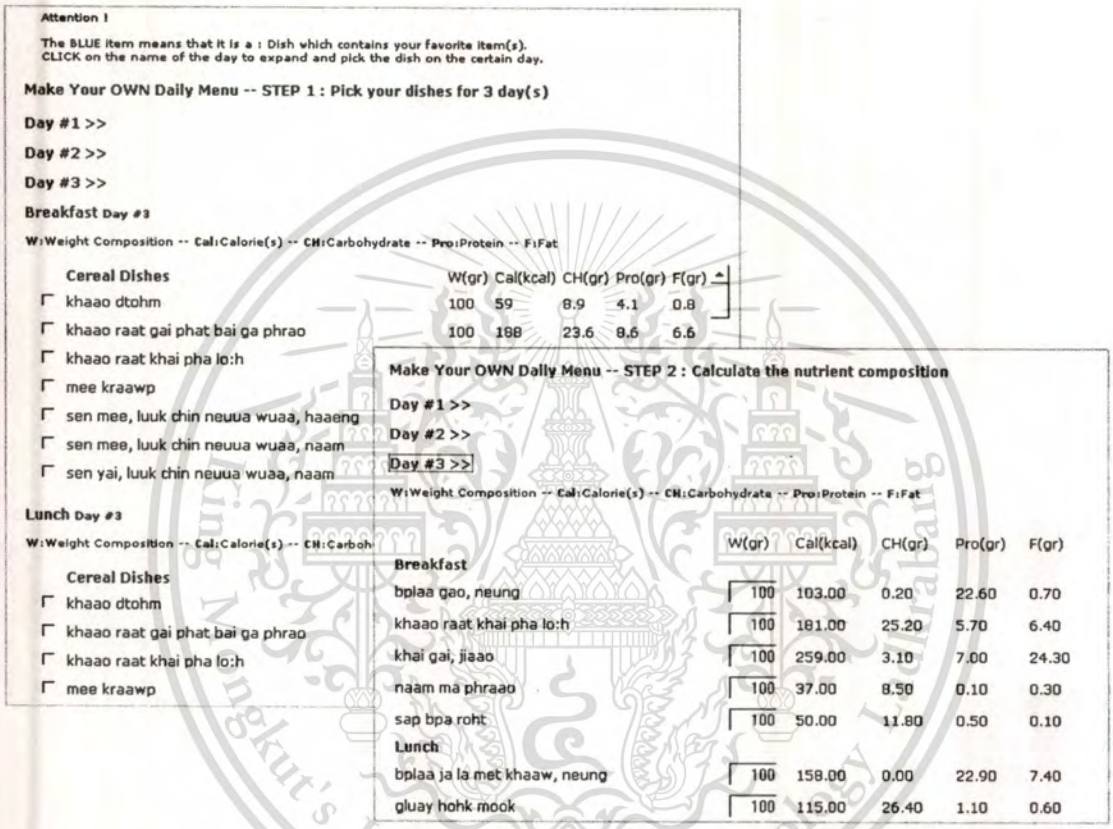


Figure 4.20 Creating menu table from dish database.

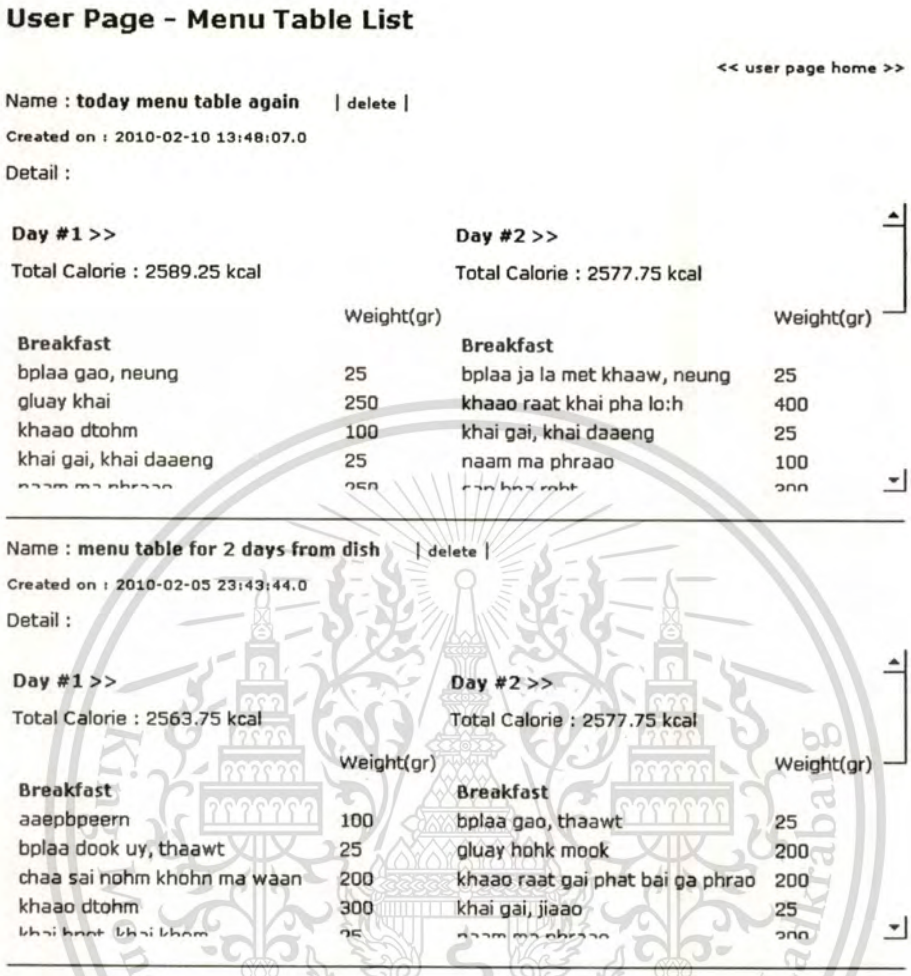


Figure 4.21 Menu table list in user page.

c. Dish creation

The creation of dish can only be done by the Admin. The administrator should fill all required information in the form which is shown in Figure 4.22. The dish that is used is Thai dishes. The name of the dish is in phonetic form. The weight composition, carbohydrate, protein and fat field hold the key to arrange the balanced nutrient when creating menu.

Figure 4.23 shows the dish list. The list of dish will be categorized by the type of dish. The categories of dish are 'Cereal', 'Egg', 'Fish and Seafood', 'Meat', 'Soup', 'Vegetable', 'Desserts and Snacks' and 'Drink and Beverages'. Any User includes a non-member User can browse the list of dish, but Admin is the only one who can edit or delete the dish.

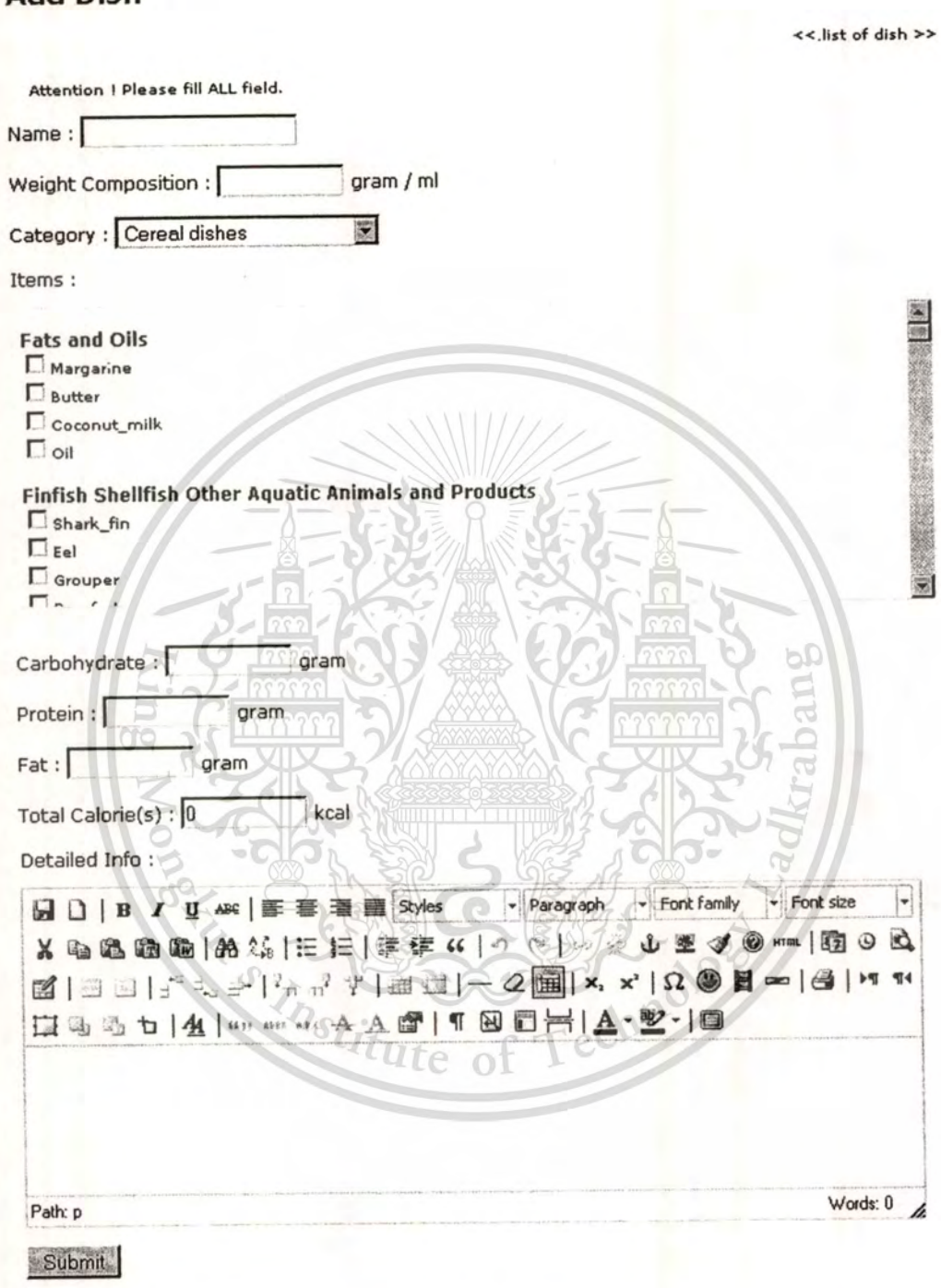


Figure 4.22 Dish creation form.

Figure 4.23 List of dish.

4.1.4 User and Admin Module

User and Admin module provides the user authentication process, registration for the new user, admin page as the Admin navigation page and the user page as the User navigation page.

A user has to be registered and login at first to access most of ONPAS functions. Figure 4.24 shows the login form that is used in ONPAS. If a user does not have any account in ONPAS, the user can register to get the new account.

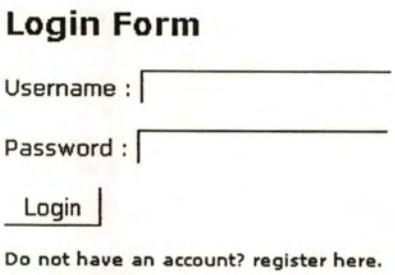


Figure 4.24 Login form.

The user registration form is depicted in Figure 4.25. A new user must fill the username, password, full name and email fields in the registration form. The username of the user cannot be same with other users. The ONPAS will check automatically whether the username is available.

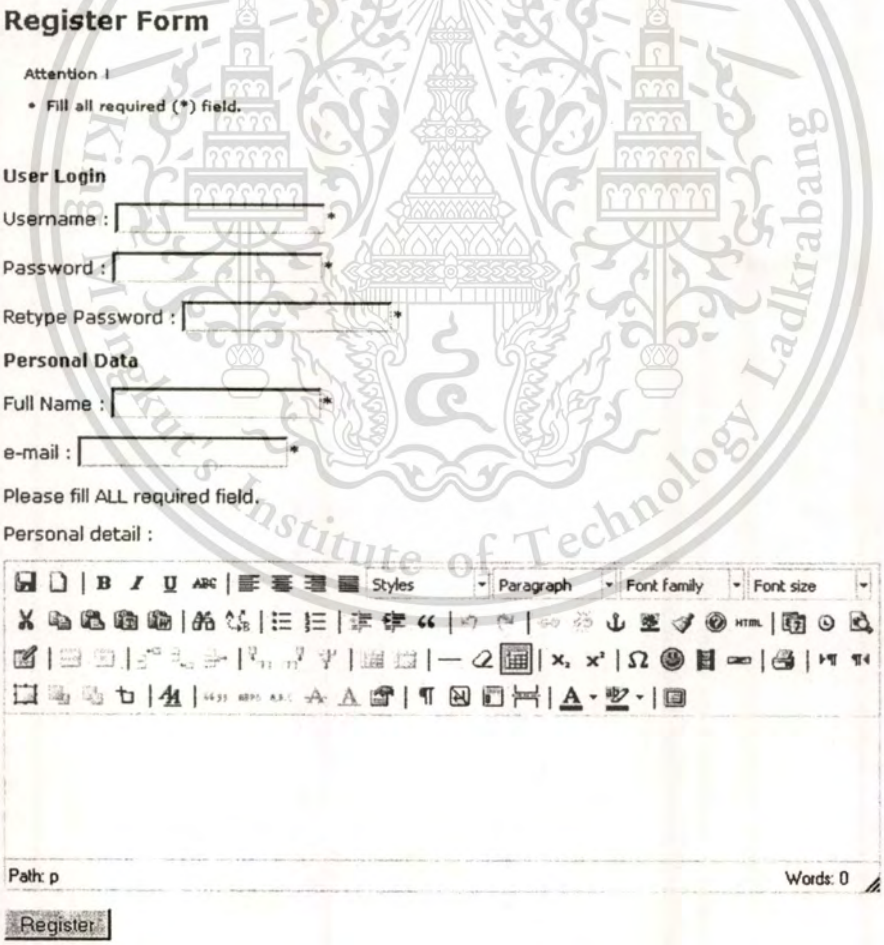


Figure 4.25 User registration form.

User page that is provided by this module is used as navigation for the user. As seen in Figure 4.26, user can see their personal detail and explore their menu and menu table creation.

Admin page provides the navigation to all administrative things. It provides all administration in menu, dish and the calculation bases.

Figure 4.26 User page.

Figure 4.27 Admin page.

4.2 System Evaluation

4.2.1 Evaluation of Research Objective

To measure the successfulness of the research, the final result of the research is then compared by the goal of the research. The goal of this research is to build an Ontology-based Nutrition Planning Assistance System (ONPAS) as a health control system. This system takes the nutrition domain and built using ontology method. The following items show some functions that are available in ONPAS:

a. Provides daily calorie needs information for a user.

   The ONPAS Semantic Web has a module which is called Personal Data Calculation module. This module provides the calculation of daily calorie needs. The user should fill some personal data information. This module will provide the Activity Factor, Basal Metabolic Rate, and finally the Total Energy Expenditure.

b. Provides menu suggestion which is daily menu to the user regarding to the user's daily calorie needs.

   ONPAS has proved successfully to give menu suggestion which has balanced calorie with the user's daily calorie needs. The suggested menu is daily menus that consist of the collection of dishes in each meal time in a day.

c. Provides menu recommendation to the user if the user prefers to get recommendation from their menu suggestion.

   The recommendation process is directed to give specific recommended menu from suggested menu based on some factors. There are three kinds of menu recommendation that are provided by ONPAS. The first is menu recommendation based on rate, vote and price. This recommendation is aimed to give menu which has high rate, high vote but in cheap price. The second one is recommendation based on taste. The user can pick his/her favorite taste of food and the intensity of the taste. The last recommendation is based on the favorite item. The user can pick some food items that is favorable to get recommended menu that consists of the favorable items.

d. Provides weight management program for adult by balancing the daily calorie needs.

Body Mass Index of adult user is also calculated when an adult user is getting their daily calorie needs. When the *BMI* of the user is abnormal, ONPAS offers the weight balance program to increase or decrease the weight. Personal Data Calculation module in ONPAS is then recalculating the daily calorie needs of the user.

e. Provides menu creation feature.

User can make his/her own menu from the dish database. The dish database that is used in this system is Thai dish database. The information of the food is taken from *ASEAN Food Composition Tables* [6] and *Thai Food Composition Tables* [7]. The menu that is made by the user must have total calorie of menu balance with user's daily calorie needs.

f. Provides a feature to make a menu table.

ONPAS gives a feature to make a collection of daily menu in maximum 7 days (1 week). User can have more than one menu table. This feature is created to maintain the balance of daily menu taken by the user for maximum one week.

4.2.2 Evaluation of ONPAS Semantic Web

A web portal is a web site that collects information for a group of users that have common interest [28]. Semantic Web portal is a web portal that developed using semantic web technologies. ONPAS Semantic Web is also can be called a portal since it is a web and has some collections of information in nutrition domain.

Holger Lausen et al (2005) presents evaluation scheme for Semantic Web portal in [29]. Figure 4.28 shows the scheme that is utilized for describing and evaluating Semantic Web portal. It consists of three layers: **Information Access** from the user's perspective, **Information Processing** features of the portal and the **Grounding Technologies**.

The Information Access layer comprises features for interaction between user and the system which are consolidated in a usability evaluation of the User Interface and in an assessment of the portal as a web-application. The Information Processing layer covers the information item

processing capabilities of a Semantic Web portal. For evaluation, the processing features and realization techniques are inspected for each phase of information item workflow depicted in Figure 4.28. Additionally, collaboration features are included to expand the information exchange and communication facilities of a Semantic Web portal. The lowest layer of the scheme encloses technologies enabling features on the upper levels and hence is referred to as the Grounding Technologies layer.

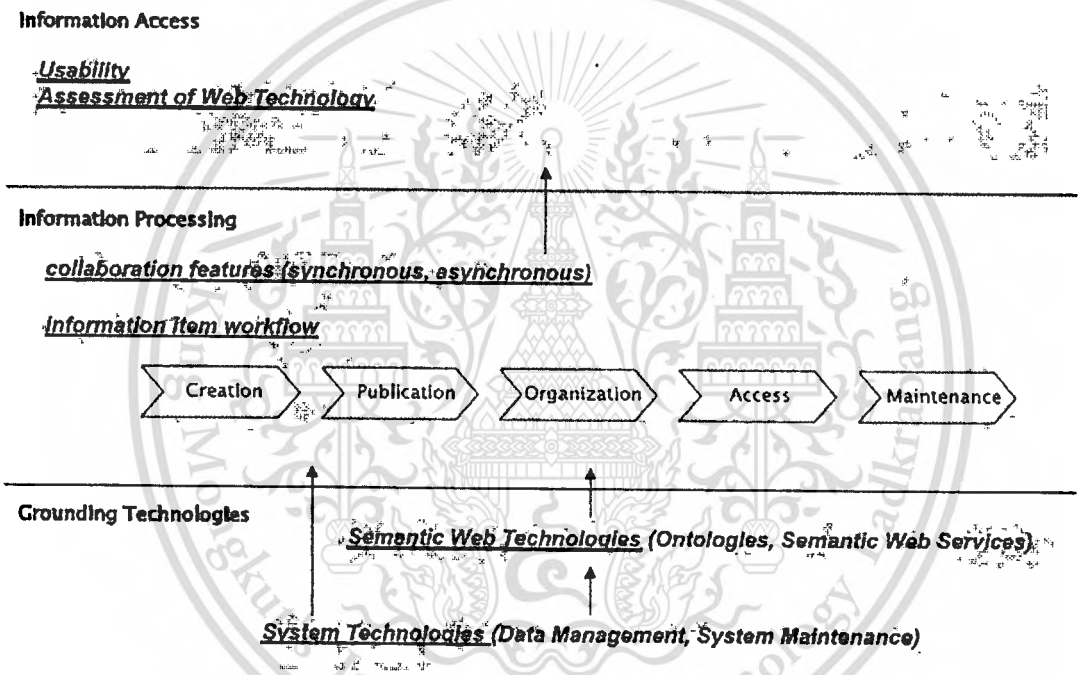


Figure 4.28 Semantic Web portal layer [29].

The following description is the evaluation of ONPAS Semantic Web using the evaluation scheme as mentioned before. The ONPAS Semantic Web evaluation is presented layer by layer in tabular arrangement.

Grounding Technologies

General		
	Operating System	Windows, possible in Unix/Linux
	Database	MySQL
	Document Repository	File System
	Web Server	Tomcat
	Applied Ontology	Four different ontology
	Ontology Language	RDF, RDFS, and OWL
	Inference Engine	Jena built-in inference engine
	Ontology Editor	Protégé
	UI Technology	HTML, CSS, JSP and JavaScript
	Browser Capabilities	Firefox 3.6, Flock 2.5, Opera 10, Safari 4, Google Chrome, Internet Explorer 8
System Technologies		
Data Management	Data Storage	- MySQL for raw data of menu, dish, user and some other value. - OWL files to store ontology and data in ontology which is accessible by using Jena Ontology API. - Jena rules files to store the knowledge base rules.
	Sorting & Indexing	- MySQL Query language can be used to sort the data from MySQL database. - SPARQL can be used to sort the information that is extracted from ontology.
	Data Transfer	- JDBC (Java Database Connectivity) for MySQL. - Jena Framework API for Ontology stored in OWL files.

System Maintenance	System Administration	ONPAS Semantic Web server
	Security Technology	Different user role, password authentication
Semantic Web Technologies		
Ontologies	Ontology	4 different ontologies are used in ONPAS (Personal Data Calculation ontology, Menu Suggestion ontology, Menu Recommendation ontology and Food Item Ontology)
	Ontology Structure	Ontologies are linked to several relations.
Inferencing & Reasoning		Jena built-in reasoner with Jena rules as the knowledge base rules.
Ontology Management	Editing	- User only creating new instance of Person class from the main ontology model. - Administrator creates the main model of all ontology using Protégé ontology editor.
	Maintenance/versioning	- User can update their own instance data in their ontology. - Administrator can change the main model of all ontology by using ontology editor.
	Ontology search for Administrator	Not available
	Standardization	RDF, RDFS, and OWL
Semantic Web Service		Not available

Information Processing

Creation	- Restricted to different user levels: administrator and registered user. - Form-based (editing support).
Publication	Two level publication: private (visible for the owner and administrator), published (to any visitor).
Organization	Each user owns their ontology document that is stored as OWL file.
Access	No searching capabilities
Maintenance	- Registered user can only update the information in their ontology. - Registered user can edit their menu and menu table creation. - Only administrator can change the ontology model. - Only administrator can change and delete the dish data.
Collaboration Features	Not available

Information Access

Usability		- The GUI of ONPAS Semantic Web is simple and easy to use. - Reference and additional instruction in each function is also provided.
General Assessment as Web Technology	Coverage	ONPAS covers the nutrition related information that includes the nutrient intake and nutrient supplement that is realized in balanced daily menu.
	Maturity of Implementation	- Complete for the certain objective in nutrition domain. - Not yet used in the real world.
	Personalization & Collaboration	No personalization and collaboration features.
	Consistency in Information Access	Information provided is consistent since the main ontology model used by every user is the same.
	Help & Documents	Reference and some additional instruction is provided.

The evaluation has pointed out some strength and weakness as well in ONPAS Semantic Web. The strong point is the development of ONPAS Semantic Web can cover all of objective that explained in subchapter 4.2.1. ONPAS Semantic Web covers the objectives with an easy interaction between the user and system. The easiness is provided from the features given in ONPAS Semantic Web and the User Interface design of the ONPAS. Delivering the information to the user is also becomes structured and clearly understandable. However, there are some other features that are lack in ONPAS Semantic Web. ONPAS does not give a searching feature to search the information inside the ontology and in the database. There is no personalization feature in the user area. ONPAS is not yet designed to collaborate with other systems. ONPAS still can be developed and expanded to be more interactive and useful as a health control system.

Chapter 5

Conclusion and Future Work

5.1 Conclusion and Discussion

In this thesis, a new ontology-based system is proposed and created to fulfill the motivation in making a health control system for the human society. A system called Ontology-based Nutrition Planning Assistance System (ONPAS) is made. ONPAS takes the nutrient as its domain. ONPAS has achievement to develop an optimal nutritional status for the daily life of a person. The optimal nutritional status is achieved when there is a balance making between nutrient intake and nutrient requirement of that person (reference to Figure 1.2 in chapter 1).

The way of ONPAS in supporting the achievement on optimal nutritional status is by comparing the daily calorie needs of person as the nutrient requirement with the nutrient intake that should be taken. ONPAS calculates the daily calorie needs based on the *TEE* of person. The *TEE* can be obtained by multiplying *BMR* and *AF* as seen in equation (2.1). To obtain the *BMR*, Schofield equation is used. All of the calculation variables come from the personal data information. ONPAS provides the form to be filled with personal data that is used to obtain the *TEE*. ONPAS will suggest daily menu to the user as the nutrient intake. The suggested menu will have the balance nutrient with the person need. The menu itself is designed as a daily menu with balanced macronutrient composition.

Ontology brings the ONPAS into a Semantic Web. Semantic Web is the most advanced web technology for now. It stays at the top of the web evolution era (reference to Figure 2.1 in chapter 2). Ontology is used as a methodology to bring out the semantic in Semantic Web technology. By using ontology, data is published using common vocabulary and grammar which is triples of subject predicate and object. The data is also ready for inference [3].

In addition of ONPAS feature, a recommendation support is provided to give more proactive function. To be more specific in giving recommendation fuzzy concept is used. A new concept in combining fuzzy and ontology is explained in [27]. By taking the concept of the fuzzy ontology, ONPAS perform the real implementation of fuzzy ontology in Semantic Web applications. The function is modified to fits triples concept of ontology. This recommendation is intended for the suggested menu.

Another feature is given to user that wants to make menu and menu table (collection of daily menu for maximum 7 days). This feature is supported by the database of Thai dish and food that is taken from [6] and [7]. The Thai dish and food data will be the boundaries of the food and dish data used in this system. The creation of menu table is also supports the weight balance program feature. Weight balance program feature from ONPAS is dedicated to an adult with unbalanced body weight. The measurement is taken with *BMI* calculation. A user can increase or decrease the weight by maximum 1 kilogram in a week regarding to the modification in food calorie intake. Therefore, the creation of menu table can be very useful. In other occasion, hopefully, the creation of menu table can support the nutritionist in the hospital or other health support company to create balanced daily menu.

ONPAS Semantic Web application is successfully implemented as an ontology-based and a web-based system that fulfill the objective of this thesis. Chapter 4 shows the implementation completeness of this system creation based of the objective mentioned in the chapter 1. ONPAS provides daily calorie needs information, gives menu recommendation to the user, offers menu recommendation for the user, offers a weight management program to adult user, provides menu creation and menu table creation to support the nutrient intake from a daily menu with Thai dish and food. Those following feature is the way of ONPAS in supporting the achievement of optimal nutritional status.

5.2 Future Work

The achievement of the objective mentioned in this thesis is fulfilled but there is nothing perfect. ONPAS Semantic Web application still can be developed to be a better system since there is still some feature is not available as evaluated.

The more concepts in nutrition domain that can be effects the nutrient intake suggestion can be considered as the most important to do in the future. The influence of diseases and training in the nutrient intake can be the addition in the concept for further development. It will need the collaboration with professionals of nutrition knowledge to make this happen.

The feature of the ONPAS Semantic Web can also be improved. The mobility of person in the real world is getting high. Therefore, the ONPAS can be developed as an agent based service that can be obtained in every part of the word and in every mobile device.

References

[1] T. R. Gruber, **"A translation approach to portable ontology specifications,"** *Knowl. Acquis.*, vol. 5, pp. 199-220, 1993.

[2] J. Cardoso, **"The Semantic Web Vision: Where Are We?,"** *Intelligent Systems, IEEE,* vol. 22, pp. 84-88, 2007.

[3] L. Ding, P. Kolari, Z. Ding, S. Avancha, and A. Joshi, **"Using Ontologies in the Semantic Web: A Survey,"** UMBC,July 2005.

[4] L. K. Mahan and S. Escott-Stump, **Krause's Food, Nutrition and Diet Therapy, 10th edition**: W.B. Saunders Company, 2000.

[5] J. N. K. Liu, **"Fuzzy Ontology Based System for Product Management and Recommendation,"** *INTERNATIONAL JOURNAL OF COMPUTERS,* vol. 1, 2007.

[6] P. Puwastien, B. Burlingame, M. Raroengwichit, and P. Sungpuag, **ASEAN Food Composition Tables**: Institute of Nutrition, Mahidol University, 2000.

[7] P. Puwastien, M. Raroengwichit, P. Sungpuag, and K. Judprasong, **Thai Food Composition Tables**: Institute of Nutrition, Mahidol University, 1999.

[8] N. S. Scrimshaw, J. C. Waterlow, and B. Schürch, **Energy and Protein requirements,** *Proceedings of an IDECG workshop*. London: International Dietary Energy Consultative Group, 1994.

[9] FAO, **"Human energy requirements, Report of a Joint FAO/WHO/UNU Expert Consultation,"** Rome,2001.

[10] WHO, **"Diet, Nutrition and The Prevention of Chronic Diseases,"** Geneva,2003.

[11] NIH, **"Clinical Guidelines on The Identification, Evaluation, and Treatment of Overweight and Obesity in Adults, The Evidence Report,"** National Institutes of Health,1998.

[12] L. M. Simmers, **Diversified Health Occupations, 6th ed.**: Delmar Cengage Learning, 2003.

[13] R. A. Roth and C. E. Townsend, **Nutrition & Diet Therapy, 8th ed.**: Delmar Cengage Learning, 2003.

[14] J. T. Pollock, **Semantic Web For Dummies**: Wiley Publishing, Inc., 2009.

[15] J. Hebeler, M. Fisher, R. Blace, and A. Perez-Lopez, **Semantic Web Programming**. Canada: Wiley Publishing, Inc., 2009.

[16] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, **Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web**: Springer, 2004.

[17] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. (1991) **Enabling Technology for Knowledge Sharing**. *AI Magazine*. 36-56.

[18] W. N. Borst, **"Construction of Engineering Ontologies for Knowledge Sharing and Reuse,"** Centre for Telematics and Information Technology, University of Tweenty, Enschede, The Netherlands, 1997.

[19] R. Studer, V. R. Benjamins, and D. Fensel, **"Knowledge engineering: Principles and methods,"** *Data & Knowledge Engineering,* vol. 25, pp. 161-197, 1998.

[20] W3Schools. **Introduction to XML**. [Online]. Available: http://www.w3schools.com/xml/xml_whatis.asp

[21] W3C. (2004). **RDF Vocabulary Description Language 1.0: RDF Schema**. [Online]. Available: http://www.w3.org/TR/rdf-schema/

[22] W3C. (2004). **OWL Web Ontology Language Guide.** [Online]. Available: http://www.w3.org/TR/owl-guide/

[23] W3C. (2008). **SPARQL Query Language for RDF.** [Online]. Available: http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/

[24] J. H. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubézy, H. Eriksson, N. F. Noy, and S. W. Tu, **"The evolution of Protégé: an environment for knowledge-based systems development,"** *International Journal of Human-Computer Studies,* vol. 58, pp. 89-123, 2003.

[25] B. Coppin, **Artificial Intelligence Illuminated**: Jones and Bartlett Publishers, 2004.

[26] L. A. Zadeh, **"Fuzzy sets,"** *Information and Control,* vol. 8, pp. 338-353, 1965.

[27] S. Calegari and D. Ciucci, **"Integrating Fuzzy Logic in Ontologies,"** in *ICEIS*, ed: INSTICC press, 2006, pp. 66-73.

[28] W3C. (2004). **OWL Web Ontology Language Use Cases and Requirements.** [Online]. Available: http://www.w3.org/TR/webont-req/

[29] H. Lausen, Y. Ding, M. Stollberg, D. Fensel, R. n. L. Herna´ndez, and S.-K. Han, **"Semantic web portals: state-of-the-art survey,"** *Journal of Knowledge Management,* vol. 9, pp. 40-49, 2005.

Biography

Personal Information

Name	DHOMAS HATTA FUDHOLI
Nationality	Indonesian
Date of birth	May 12, 1986
Place of birth	Yogyakarta, Indonesia

Education

Bachelor degree

Field	Electrical Engineering (Computer and Information System Concentration)
Duration	2003-2008
Department	Department of Electrical Engineering
Faculty	Engineering
University	Gadjah Mada University (UGM), Indonesia

Master degree

Field	Electronic Engineering
Duration	2008-2010
Department	Department of Electronics
Faculty	Engineering
University	King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand

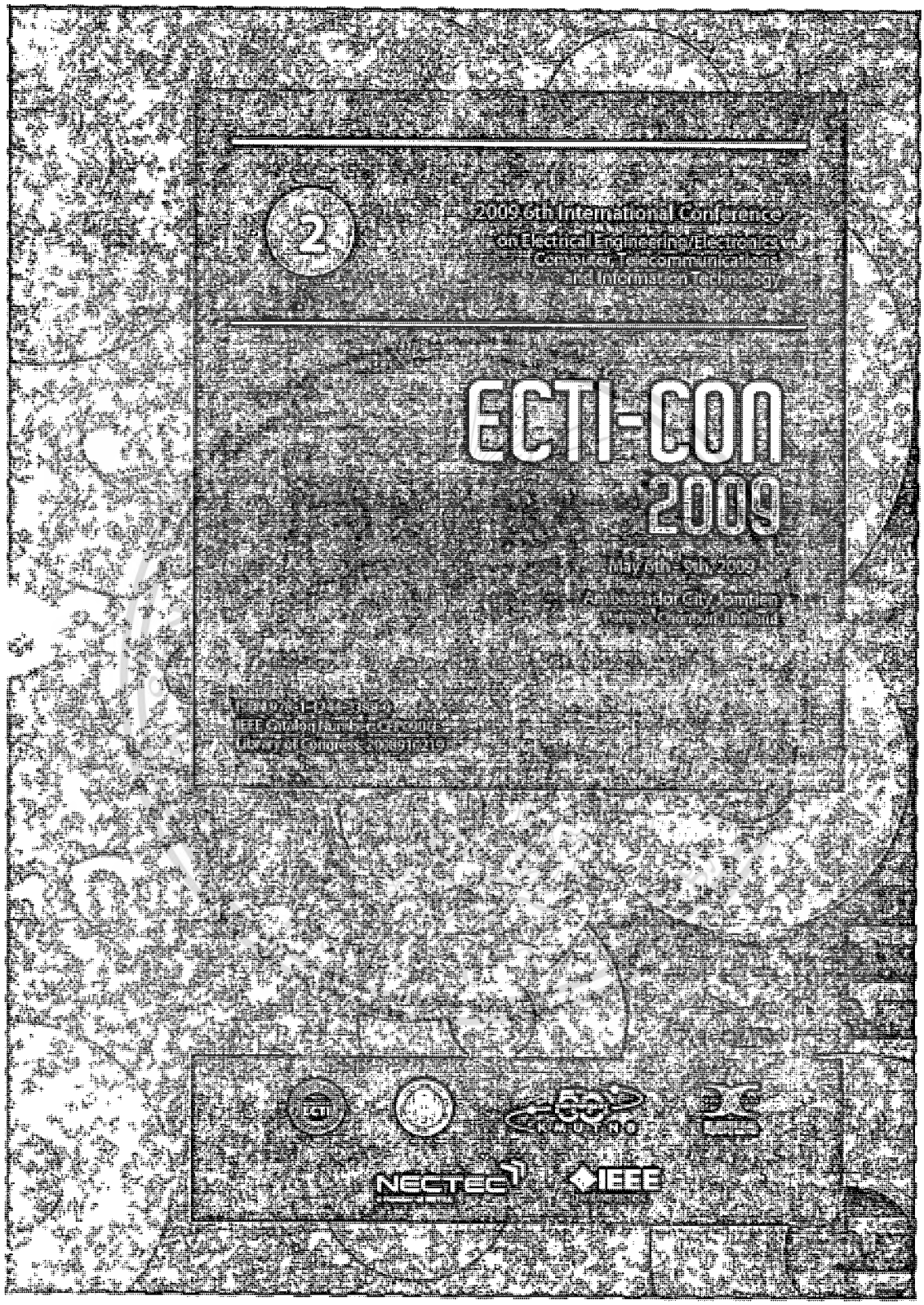
Research Interests

Information System, Software Engineering, Web Technology, Semantic Web, and Ontology.

List of International Conference Proceeding Papers

1. Dhomas Hatta Fudholi, Noppadol Maneerat, and Ruttikorn Varakulsiripunth, "Ontology-Based Daily Menu Assistance System," *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology (ECTI-CON 2009)*, May 6-9, 2009, Pattaya, Chonburi, Thailand, pp. 694-697.

2. Dhomas Hatta Fudholi, Noppadol Maneerat, Ruttikorn Varakulsiripunth, and Yasushi Kato, "Application of Protégé, SWRL and SQWRL in Fuzzy Ontology-Based Menu Recommendation," *2009 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2009)*, December 7-9, 2009, Kanazawa, Japan, pp. 631-634.



Ontology-Based Daily Menu Assistance System*

Dhomas Hatta Fudholi Noppadol Maneerat Ruttikorn Varakulsiripunth

Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang
Chalongkrung Road, Ladkrabang, Bangkok, 10520, Thailand

***Abstract*- The daily useful nutrition needs become important information to be given to the society. Occurrence of many nutrition cases, such as obesity, overweight or even underweight, shows that the society have not been given an appropriate information about their recommended daily values for key nutrients and also the suggestion of their daily menus. In this paper, we design ontology-based daily menu assistance system using the Indonesian food as a case study. The ontology is a simple methodology to describe the concept of an expert system. We have used it to model a nutrition needs domain in the information of health control system. This methodology is used to advise the appropriate Indonesian food menus to anyone based on their personal data, calories needed, activity factor, and abstinence.**

I. Introduction

States of nutritional deficiency or excess occur when the nutrient intake is not balanced with specific requirements for optimal health [1].

People are able to know what they have to eat in order to develop an optimal nutritional status by knowing how much calories they need and the suitable suggestion of their menus.

An individual's nutritional status reflects the degree to which physiologic needs for nutrients are being met. When adequate nutrients are consumed to support the body's daily needs and any increased metabolic demands, the person develops an optimal nutritional status. This status promotes growth and development, maintains general health, supports activities of daily living, and helps protect the body from disease and illness [1].

Semantic web technology is applied into daily menu assistance system. One of semantic web technology is using ontology to design and implement an expert system. Ontology is an explicit specification of a conceptualization [2]. It explains about a concept in some domains, properties and attributes, and roles (limitation) of the concept. Ontology has been proved to be very useful in sharing concepts across applications in an unambiguous way and also conceived as a good mechanism to describe the shareable and common understanding in a domain concept. There are many researches in healthcare and medical fields using a semantic web technology, one of them is [3].

II. Literature Review

A. Energy Requirement

The energy requirement of an individual, in a state of desirable equilibrium, is equal to the energy expenditure. In some clinical situations, where an improvement in nutritional status may be advisable, the energy requirement may be set at a higher level than the energy expenditure in order to produce, temporarily, a positive energy balance [4]. One of the methodologies to measure daily energy expenditure is by multiplying Basal Metabolic Rate (*BMR*) with Activity Factor (*AF*) which will be dependent on the degree and duration of physical activity or Physical Activity Level (PAL) [4]. The daily energy expenditure will be represented as Total Energy Expenditure (*TEE*) and (1) is the equation to calculate *TEE*.

$$TEE = BMR \times AF \quad (1)$$

TEE : Total Energy Expenditure (kcal/day),
BMR : Basal Metabolic Rate (kcal/day),
AF : Activity Factor.

The report from the 1985 Food and Agriculture Organization (FAO) / World Health Organization (WHO) / United Nations University (UNU) expert consultation used a set of equations derived mostly from studies in Western Europe and North America which is *Schofield* equation to calculate *BMR* based on person's age [5]. The set of equations can be seen in table I.

TABLE I
Equations for Estimating BMR from Body Weight (W^1)

	Age (Years)	BMR: kcal/day
Male	< 3	(59.512 × W) – 30.4
	3–10	(22.706 × W) + 504.3
	10–18	(17.686 × W) + 658.2
	18–30	(15.057 × W) + 692.2
	30–60	(11.472 × W) + 873.1
	≥ 60	(11.711 × W) + 587.7
Female	< 3	(58.317 × W) – 31.1
	3–10	(20.315 × W) + 485.9
	10–18	(13.384 × W) + 692.6
	18–30	(14.818 × W) + 486.6
	30–60	(8.126 × W) + 845.6
	≥ 60	(9.082 × W) + 658.5

W^1 is body weight in kilogram(s)

Activity Factor (*AF*) is a factor used in the equation to determine energy needs that takes into consideration. In every range of age activity factor would be categorized in different value.

*This research is funded by ASEAN University Network/Southeast Asia Engineering Education Development Network (AUN/SEED-Net) project through the Japan International Cooperation Agency (JICA).

978-1-4244-3388-9/09/$25.00 ©2009 IEEE

The activity factor for children and adolescents was studied with doubly-labeled water, heart rate monitoring and time allocation [4]. Table II shows different *AF* allocation for different age in children and adolescents. Children and adolescents with *light* physical lifestyles are they who every day spend several hours at school and spend most leisure time in activities that require little physical effort. Children and adolescents with *heavy* lifestyles are they who walk long distances every day and/or practice sports or exercises for several hours, several days of a week. Children and adolescents with habitual physical activity that is more strenuous than the *light* lifestyle, but not as demanding as the *heavy* lifestyle, would qualify in the category of *moderate* [5].

TABLE II
ACTIVITY FACTOR OF CHILDREN AND ADOLESCENTS

Age (years)	Sex	Light	Moderate	Heavy
1-5	Male, Female	1.44	1.61	-
6-13	Male	1.54	1.75	1.96
14-18	Male	1.60	1.82	2.04
6-13	Female	1.48	1.68	1.88
14-18	Female	1.46	1.66	1.86

The FAO/WHO/UNU Expert Consultation suggested the average daily physical activity of adults whose occupational work is classified as *light*, *moderate*, and *heavy*. It is shown in table III [4]. Adult people with *light* activity are people who have occupations that do not demand much physical effort and spend most of their leisure time sitting or standing, with little body displacement. People with *moderate* lifestyle are people that have occupations that are not strenuous in terms of energy demands, but involve more energy expenditure. People with *heavy* lifestyle are people that engage regularly in strenuous work or in strenuous leisure activities for several hours [5].

TABLE III
ACTIVITY FACTOR OF ADULT (18 – 64 YEARS)

	Light	Moderate	Heavy
Male	1.55	1.78	2.10
Female	1.56	1.64	1.82

The *AF* for older individuals was studied form older people in the mean ages of 64-74 years [4]. Goran&Poehlman (1992), Roberts et al (1993), and Pannemans&Westerterp (1995) study using the doubly labeled water method result the average *AF* for older individuals which is shown in Table IV.

TABLE IV
ACTIVITY FACTOR OF OLDER INDIVIDUALS

Male	Female
1.77	1.71

B. Daily Menu

Menu is defined by food composition that is consumed by someone for his/her meal in one meal time or one day. Balanced menu is a menu which consists of varied foods in appropriate quantity and proportion, to fulfill the nutritional needs of somebody [6].

To calculate a total calorie which is contained in every menu, we use a software named NutriSurvey. This software tool is the English version of the professional German nutrition software (EBISpro) which is developed by Dr. Juergen Erhardt from Hohenheim University, Germany. It contains all useful functions which are typical for this kind of software such as nutrient analysis and calculation of energy requirements, planning of diets, searching of nutrients in foods, handling of recipes, etc. This software can be used by every people in various countries who have different food, because it provides many food databases from all over the world including Indonesian food database that can be easily integrated. Some researches or surveys also use this software to analyze their consumption data, one of them is [7].

III. SYSTEM ARCHITECTURE

The daily menu assistance framework proposed in this paper is shown in Fig. 1. It consists of seven modules; they are user interface, tomcat server, information database, inference agent, inference engine, inference rules and the ontology itself. Firstly, users will input their personal data which is needed to calculate the energy expenditure. Next, the system will calculate the data to get the appropriate information such as *AF* and *TEE*. Then, inference rules which are ran by Jena inference engine will give an appropriate menu from the database. Jena is a free Java framework for building semantic web applications and a rule-based inference engine.

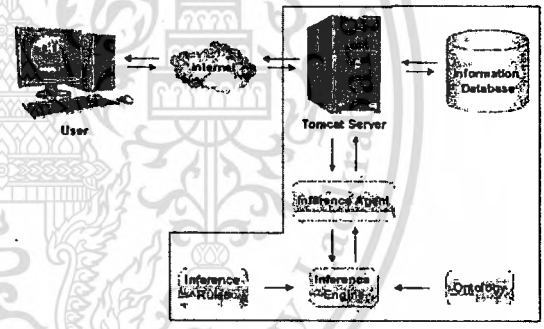


Figure 1. Daily menu assistance system architecture.

IV. ONTOLOGY AND APPLICATION BUILDING

The design of ontology is built in an Extensible Markup Language (XML) file called OWL. OWL (Web Ontology Language) is a web language which intends to interpret the web information into the machine-readable content with semantics.

A. Ontology Design

Daily nutrition ontology has four main classes as shown in Fig. 2. They are *Person, Activity, Indonesian Food Menu*, and *Weight Balance Program* class. Each class has its own property which is explained below. The property of each class is shown in Fig. 3.

Person class is divided into six subclasses regarding to the *Schofield* equation. This class has few elements, which are gender, weight, height, age, abstinence and activity.

Abstinence is food which is prohibited for a person to eat in case of allergy. Using these elements we can define the *AF* and *TEE* so that a person could get an appropriate menu based on his/her *TEE* and abstinence.

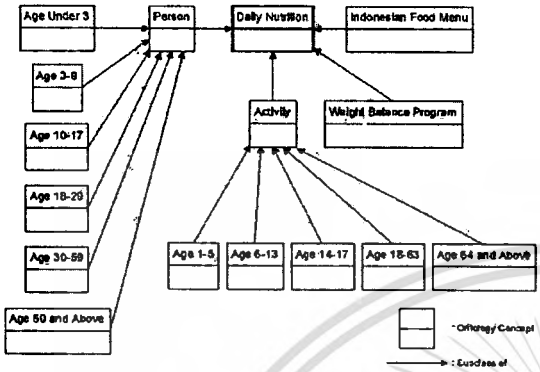


Figure 2. Ontology class for daily menu assistance.

Activity class has five subclasses. The value of AF is specific in every range of age and in different gender. It has two values that are stored in this class; they are male AF and female AF.

Indonesian Food Menu class has two properties which are calories and menu item. Daily menu assistance system will give an appropriate Indonesian food menu as a final result. It will show the menu item to the user.

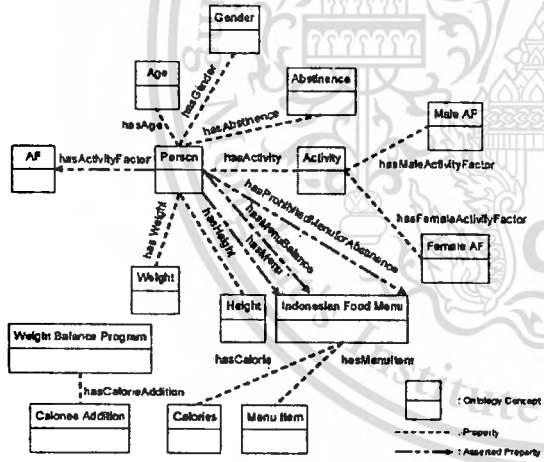


Figure 3. Class properties of daily menu assistance ontology.

Weight Balance Program class contains functions to manage body weight. BMI (Body Mass Index) will be calculated and the system will give a status whether a person has underweight, normal weight, or overweight. A person can select one of the programs after seeing the BMI status. The functions of this program is gaining or losing 0.5-1 kg of body weight in a week by increasing or decreasing 500-1000 kilocalories in a day from the total daily calories. The range of body weight balance is the safest but slow way to manage body weight from food consumption. Extreme diet can effect in one's health to nutrient deficiencies and psychological disturbance [8]. This class will be considered to give an appropriate balanced menu.

B. Inference Rules

Some inference rules are defined to give appropriate Indonesian food menus for the user. The rules are defined using Jena rules syntax since we use Jena as an inference engine. These rules and personal data will be used by inference engine to define an appropriate menu for the user.

We will explain few rules of this system. They are *ActivityFactorRule*, *MenuRule*, and *ProhibitedMenuRule*. We have to define the URI (Uniform Resource Identifier) in ontology model as an identifier. Daily menu assistance ontology uses 'nutrition' as variable for the URI. Getting an appropriate menu requires a matching method between person's calories needs and menu calories. Firstly we need to calculate the *TEE* of a person.

```
[ActivityFactorRuleforMale:                                          R1
  (?p nutrition:hasGender "Male")                                    R2
  (?p nutrition:hasActivity ?a)                                      R3
  (?a nutrition:hasMaleActivityFactor ?af)                           R4
 →(?p nutrition:hasActivityFactor ?af)]                              R5
[hasMenuRule:                                                        R6
  (?p nutrition:hasTEE ?tee)                                         R7
  (?m nutrition:hasCalories ?tee)                                    R8
 →(?p nutrition:hasMenu ?m)]                                         R9
[hasProhibitedMenuRule:                                              R10
  (?p nutrition:hasMenu ?m)                                          R11
  (?p nutrition:hasAbstinence ?abs)                                  R12
  (?m nutrition:hasMenuItem ?abs)                                    R13
 →(?p nutrition:hasProhibitedMenuforAbstinence ?m)]                  R14
```

Daily activities are defined with only the category of light, moderate and heavy. We use *ActifityFactorRule* to get the value of AF from the activity class based on the person's gender. *ActivityFactorRule* is divided into two because we have two genders which are male and female. We only give an example for male which is *ActivityFactorforMale* (R1). *?p* represents a person, *?a* represents the daily average activity (light, moderate, heavy), and *?af* represents the value for AF. Firstly, the rules make sure what gender that a person has (R2), then look into daily average activity they have (R3), and finally take out the value of AF (R4) and put it in the new asserted property of *Person* class (Fig. 3) which is *hasActivityFactor* (R5).

After calculating the *TEE* for a person, the system will execute *hasMenuRule* to get an appropriate menu for the user by matching the *TEE* and the total calories of menus. *hasMenuRule* rule (R6) has some declaration. First, (R7) declares that a person ('*?p*') has property *hasTEE* which is equals to some value and stored in '*?tee*' variable. Next, a menu ('*?m*') has value in *hasCalories* property which is the same as the value in '*?tee*' variable (R8). Finally, a new asserted property *hasMenu* as shown in Fig. 3 which is owned by Person class in (R9) will give an appropriate menu ('*?m*') for the user. If a weight balance program selected, the system

will recalculate *TEE* with the calories addition or calories reduction. It will give the person a new menu in *hasMenuBalance* asserted property which is shown in Fig. 3.

When a person has an abstinence to eat kinds of foods, the *hasProhibitedMenuRule* will take out the menu that has a food in menu items which is prohibited to eat. *hasProhibitedMenuRule* **(R10)** will take information what menus (*?m*) that a person (*?p*) has **(R11)**. Then, it will compare the prohibited food from a person **(R12)** and menu items in the menu **(R13)**. If it is match, the person will has a new asserted property *hasProhibitedMenuforAbstinence* (R14) of a certain menu (Fig. 3).

C. *Application and Result*

Daily menu assistance application is a web-based semantic application. Java Server Pages (JSP) technology used to build this application. The application has four main modules, they are home page, daily menu page, reference page, and contact page. Home page gives a brief knowledge about this application. Daily menu page is the core of this system because in this module we give the advice of an appropriate daily menu for the user. Reference page gives some detail knowledge and explanation about how to calculate *TEE*, *BMR*, BMI and also how to define *AF*.

To use the daily menu system, user needs to fill in some personal data such as name, gender, age, height, weight, activity and abstinence. Fig. 4 shows an example of user interface of personal data input and final result in giving appropriate menus. In the example, we put a person named 'Hatta'. He is 22 years old male. He has 170 cm of height and 50 kg of weight. He has moderate activity and shrimp is his abstinence in food. Daily menu assistance system then defines the *AF* and calculates the *BMR*, and *TEE* so that it can give an appropriate daily menu based users calories need and his abstinence. It also gives the BMI calculation result and status so that user can choose whether he/she want to follow the weight balance program. Weight balance program will recalculate daily calories and menu result for users if selected.

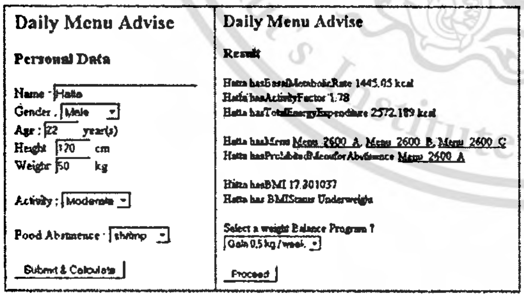


Figure 4. *Example of input data and final result in giving appropriate menus.*

When people want to know the detail of menu items from a certain menu, they can click the menu name to get the menu detail. Fig. 5 shows an example of detailed menu 'Menu 2600 A'.

Menu 2600 A

Eat Time	Food Items	Weight (gr)	Serving Size
Breakfast	Nasi (Rice)	250	1 ½ glass
	Tahu [illegible] kecap	75	[illegible]
	[illegible] kc panjang + tauge kedelai	100	[illegible] glass
	Minyak (Oil)	10	1 spoon
	Susu Bubuk (Milk Powder)	35	3 spoon
	Gula Pasir (Sugar)	10	1 spoon
Snack (10am)	Kue putu (Putu cake)	100	2
Lunch	Nasi (Rice)	250	1 ½ glass
	Rolade daging (Meat Rolade)	75	1 big slice
	Orak Arik Tahu (Tofu)	50	½ big slice
	Sayur bening bayam + wortel (spinach + carrot)	100	1 glass
	Minyak (Oil)	10	1 spoon
	Pepaya (Papaya)	100	1 medium slice
Snack (4pm)	Kolak pisang (banana kolak)	150	1 medium bowl
Dinner	Nasi (Rice)	250	1 ½ glass
	Udang (Shrimp)	75	1 big slice
	Tempe goreng tepung (fried tempe)	50	1 medium slice
	Ca labu siam (Ca pumpkin siam)	100	1 glass
	Minyak (Oil)	15	1 ½ spoon
	Apel (Apple)	100	1
Snack (9pm)	Susu bubuk (Milk Powder)	15	3 spoon
	Gula pasir (Sugar)	10	1 spoon

Figure 5.Example of detailed menu items of a menu.

V. CONCLUSION AND FUTURE WORKS

We have designed and developed the daily menu assistance using ontology concept including the knowledge of the domain concept and its relationships. We also have implemented it in the semantic web application. Daily menu assistance system can give an appropriate food menu based on personal daily calories needs. It can be one way to maintain daily healthy life. We can expand the foods menu not only Indonesian foods menu since Indonesian foods only the case study. Our future work would be extending the ontology and decision factors so that many advices can be provided. Soft computing technology can be also applied to get more precise in advice result.

ACKNOWLEDGMENT

This research is funded by ASEAN University Network/Southeast Asia Engineering Education Development Network (AUN/SEED-Net) project through the Japan International Cooperation Agency (JICA).

REFERENCES

[1] K. Mahan, S. Escott-Stump, *Krause's Food, Nutrition, and Diet Therapy*. 11th edition, Philadelphia: Saunders, 2004.
[2] T. Gruber, "Towards Principles for the Design of Ontologies Used for Knowledge Sharing", Int. Journal of Human-Computer Studies, 1995.
[3] D. Cerizza, I. Celino and E. Della Valle, "Semantic discovery of Medical Advice Services with Glue," ISWC 2006, Workshop for Semantic Web Health Care & Life Science, Georgia, 2006.
[4] N. S. Scrimshaw, J. C. Waterlow and B. Schürch, Editors, "Energy and Protein requirements", Proceedings of IDECG Workshop, International Dietary Energy Consultative Group, London, 1994.
[5] FAO, "Human energy requirements", Report of a Joint FAO/WHO/UNU Expert Consultation, Rome, 2001.
[6] S. Iff, M. Leuenberger, S. Ro¨sch, G. Knecht, B. Tanner, Z. Stanga, "Meeting the nutritional requirements of hospitalized patients: An interdisciplinary approach to hospital catering", Clinical Nutrition Journal, ScienceDirect, 2008.
[7] M. Andersson,et aL, "Dual fortification of salt with iodine and iron: a randomized, double-blind, controlled trial of micronized ferric pyrophosphate and encapsulated ferrous fumarate in southern India", American Journal of Clinical Nutrition, Vol 88, No. 5, 1378-1387, 2008.
[8] R. A. Roth, C. E. Townsend, *Nutrition & Diet Therapy*, Cengage Learning, 2002, p. 22.

2009 International Symposium on Intelligent Signal Processing and Communication Systems

Proceedings

December 7-9, 2009

Kanazawa Excel Hotel Tokyu, Kanazawa, Japan

Sponsored by IEEE Circuits and Systems Society

Co-sponsored by IEICE

2009 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2009) December 7-9, 2009

WP2-B-2

Application of Protégé, SWRL and SQWRL in Fuzzy Ontology-Based Menu Recommendation

Dhomas Hatta Fudholi[1] Noppadol Maneerat[1] Ruttikorn Varakulsiripunth[1] Yasushi Kato[2]

[1]Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang
Chalongkrung Road, Ladkrabang, Bangkok 10520, Thailand

[2]Department of Information Engineering, Advanced Course of Information Systems Engineering
Sendai National College of Technology, Sendai 989-3124, Japan

***Abstract*—We develop a daily menu assistance system which is built to give a suggested daily menu based on daily calories need. In this paper, we design a recommendation feature using fuzzy ontology to give menu recommendation by few factors, such as price, rate, vote and taste, since so many menus are suggested. Protégé, SWRL (Semantic Web Rule Language) and SQWRL (Semantic Query-Enhanced Web Rule Language) are used to show that this feature provides user the more specific, recommended and preferable menus. Protégé, an open-source ontology design software, is used to develop menu recommendation ontology. SWRL is used to build fuzzy ontology classes and properties mapping, and also calculations. SQWRL performs a query in menu recommendation results.**

I. Introduction

Ontology has been proved to be very useful in sharing concepts across applications in an unambiguous way and also conceived as a good mechanism to describe the shareable and common understanding in a domain concept. Ontology supports information exchange based on semantics rather than just syntax [1]. The view point in an individual's nutritional status reflects the degree to which physiologic needs for nutrients. This status promotes growth and helps to protect the body from diseases [3]. Daily menu assistance system has been created using ontology and takes issues in nutrients domain [2]. The system gives a suggestion of a balanced nutrient daily menu based on user's daily calories need.

A system that is integrated with recommendation support can provide a more personalized and proactive functions [4]. We design a recommendation feature to support the daily menu assistance system's decision processes. We add rate, vote, price and taste factors into the system. To get more precise and easy usage of the system, the factors would be in vague form, for example, "enough sweet food". Fuzzy concept is used to bring out the vague and imprecise value handling. Since the daily menu assistance system is an ontology-based system, the fuzzy concept will be integrated into the ontology.

Protégé is developed in Stanford Medical Informatics [5]. It supports OWL (Web Ontology Language) that intends to interpret the web information into machine-readable content with semantics. SWRL, a W3C member submission [6], is an effective representation for translating data between different ontologies. This includes mapping of classes, properties, and values. It also has structural transformations, unit conversions and other calculations. SQWRL is then used to perform a query in the menu recommendation result. SQWRL provides SQL (Structured Query Language)-like operations to format knowledge retrieved from OWL ontology. SQWRL is defined using a library of SWRL built-in that effectively turns SWRL into a query language [7].

This paper is organized as follows. Section II gives a brief review in fuzzy sets, fuzzy ontology and daily menu assistance system. Section III explains the concept design of the recommendation feature. Section IV shows experimental result from the menu recommendation feature.

II. Literature Review

A. Fuzzy Sets

Silvia C. et al gave some short and compact description in fuzzy sets [1]. Let us consider a nonempty set of objects U, called the universe. A *fuzzy set* is defined as a [0, 1]–valued function on U, $f : U \rightarrow [0, 1]$. Given an object $x \in U$, $f(x)$ represents the membership value of x to the set f . The definitions give a basic knowledge in how a fuzzy can handle the vague and imprecise things with giving a degree into the value. Defining fuzzy value can be done in two models, linguistic and precise. Table I shows an example of defining fuzzy value. Precise fuzzy modeling is used to get a good accuracy. The main objective using linguistic modeling is to obtain fuzzy value with a good interpretability.

TABLE I. Modeling of Fuzzy Value

Linguistic	little	enough	moderately	very	totally
Precise	0.2	0.4	0.6	0.8	1.0

B. Fuzzy Ontology

A fuzzy ontology that is described by Silvia C. et al [1] is an ontology extended with fuzzy values. We can assign fuzzy ontology through the following two functions:

$f : (Concepts \cup Instance) \times Properties \rightarrow Property\_Value \times [0,1]$, such as $f(MenuA, taste) \rightarrow (sweet, 0.8)$ and

$g : (Concepts \cup Instances) \times (Properties \cup Property\_Value) \rightarrow [0,1]$, such as $g(MenuA, cheap) \rightarrow 0.3$

Applications using fuzzy ontology can combine function *f* and *g* using logical operation such as OR and AND. OR fuzzy operator takes the maximum value and AND fuzzy operator will take the minimum value of the related sets [8].

C. Daily Menu Assistance System

We use ontology to model a nutrition needs domain in the information of health control system. This system is called the

978-1-4244-5016-9/09/$25.00 ©2009 IEEE

2009 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2009) December 7-9, 2009

daily menu assistance system [2]. It was created as a semantic web based application. It is used to suggest the appropriate daily menus to anyone based on his/her personal data, daily calories need, activity factor, and abstinence. Fig. 1 shows the ontology design of the daily menu assistance system. Menu recommendations and preferences will be very useful for the user to choose the preferable menu because there will be so many menus suggested by this system.

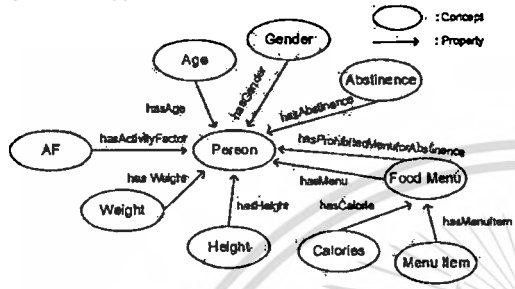


Figure 1. Ontology design for daily menu assistance system.

III. System and Concept Design

A. Ontology Design for Menu Recommendation

Adding recommendation can give a view point to the user in choosing menu. We make two recommendation processes MRP-1 and MRP-2. MRP-1 is a menu recommendation process based on menu's rate, vote and price. MRP-2 is a menu recommendation process based on the menu's taste. These two processes can be combined if preferred. All recommendations are calculated and compared between menus which are suggested to the user, not all menus stored in database.

Based on MRP-1 and MRP-2, we made the ontology design for menu recommendation as seen in Fig. 2. There are six main concepts, *Person*, *Menu*, *Taste*, *Price*, *Vote* and *Rate*. *Person* has three properties in *Menu*. *hasMenu* property contains all menus which are suggested to the user. *hasRecommendedMenu* property is an asserted property that is given by MRP-1 and *hasPreferredMenuofTaste* is an asserted property that is given by MRP-2. *Menu* has four properties, *hasTaste*, *hasRate*, *hasVote* and *hasPrice*. These four properties are used to infer the recommendation to *Person*. *Taste* is categorized into five tastes; sweet, bitter, salty, sour and spicy. *Price* is categorized into two; cheap and expensive. *Vote* and *Rate* is categorized into high and low. Since it uses a fuzzy ontology, each of categorization contains the *membership degree* (*md*) → [0,1].

B. Menu Recommendation Process: MRP-1

The rate of menu means the point that menus get from user appreciations. The vote of menu is the total amount of rate which is given by the user. The price of menu is the value of average amount of money to make a certain menu.

1. Value Calculation of Rate, Vote and Price in MRP-1

The way to get the value of *Rate* is by using the five stars rate, i.e. 0, 1, 2, 3, 4 and 5, will equal to assign *md*, 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0, respectively. *Vote* means how many user vote for a menu. The calculation of *Rate* and *Vote* is shown in (1) and (2) respectively as given below. Finally, the cost or price of the menu will be used as the value of *Price*.

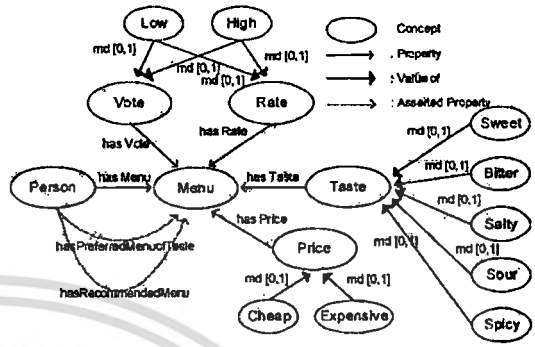


Figure 2. Ontology design for menu recommendation.

Few of build-in functions in SWRL that we use to simulate the calculation process in Protégé are; (i) *swrlb:add* (satisfied iff the first argument is equal to the arithmetic sum of the second argument through the last argument), (ii) *swrlb:subtract* (satisfied iff the first argument is equal to the arithmetic difference of the second argument minus the third argument), (iii) *swrlb:multiply* (satisfied iff the first argument is equal to the arithmetic product of the second argument through the last argument) and (iv) *swrlb:divide* (satisfied iff the first argument is equal to the arithmetic quotient of the second argument divided by the third argument) [6].

Rule-1 shows how SWRL with its built-in functions can be used to calculate the value of *Rate*. *?m* is menu, *?s0* to *?s5* represent the amount of rate in each star. Rule-1 returns a value *?tr* as a value of *Rate* in a menu.

$$Rate = \frac{((a1\times 0.2)+(a2\times 0.4)+(a3\times 0.6)+(a4\times 0.8)+(a5\times 1.0))}{(a0+a1+a2+a3+a4+a5)} \quad (1)$$

$$Votes = \frac{v}{V}, \quad v = a0 + a1 + a2 + a3 + a4 + a5 \quad (2)$$

a0:amount of 0 star rate, *a1*:amount of 1 star rate, *a2*:amount of 2 stars rate, *a3*:amount of 3 stars rate, *a4*:amount of 4 stars rate, *a5*:amount of 5 stars rate, v : amount of votes in a menu , V : amount of votes in suggested menus.

Rule-1 :
has0STARratings(?m,?s0) ∧ has1STARratings(?m,?s1) ∧ has2STARratings(?m,?s2) ∧ has3STARratings(?m,?s3) ∧ has4STARratings(?m,?s4) ∧ has5STARratings(?m,?s5) ∧ swrlb:multiply(?m1,?s1,0.2) ∧ swrlb:multiply(?m2,?s2,0.4) ∧ swrlb:multiply(?m3,?s3,0.6) ∧ swrlb:multiply(?m4,?s4,0.8) ∧ swrlb:multiply(?m5,?s5,1.0) ∧ swrlb:add(?add,?m1,?m2,?m3,?m4,?m5) ∧ swrlb:add(?sum,?s0,?s1,?s2,?s3,?s4,?s5) ∧ swrlb:divide(?tr,?add,?sum) → hasTotalRatingValue(?m, ?tr)

2. Fuzzy Ontology Sets in MRP-1

Fig. 3 shows the trapezium type membership function which is used to define the *md* of *Price*, *Rate* and *Vote*. The x-axis is the concept's real value of *Price*, *Rate* and *Vote*. The y-axis is the *md* of fuzzy set. f_{cheap}, $f_{expensive}$, f_{low}, and f_{high} are the membership functions for cheap price, expensive price, low rate or low vote and high rate or high vote, respectively. Variable a and b in f_{cheap} and $f_{expensive}$ is the limitation value of

cheap price and expensive price. Variable *a* and *b* in f_{low} and f_{high} is the limitation value of low and high in rate and vote.

Protégé does not have any feature to define *md* directly. It can only define the concept, the property and the property value. We then declare fuzzy ontology functions by putting the property value along with the property, and make the *md* as the value. For example, *f(Menu2000A,price)* = (*cheap*,0.7) means that the individual *Menu2000A* has *hasCheapPrice* property with property value equals to 0.7 and we use this value as the *md* of *cheap*.

All of membership functions are then built in SWRL. **Rule-2** is one example of SWRL representation for f_{cheap} to define the *md* of cheap from a certain price in a menu. *?m* represents menu and *?p* is the price of menu. *?c* represents variable *a* and *?e* represents variable *b*. The other three rules for $f_{expensive}$, f_{low}, and f_{high} are built in the similar way.

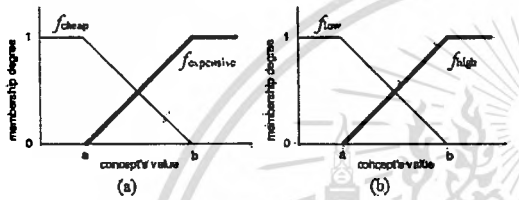


Figure 3. (a)Fuzzy set membership function for *Price*, (b)Fuzzy set membership function for *Rate* and *Vote*.

Rule-2 :
hasLimitValue(Cheap,?c) ∧ hasMenuPrice(?m,?p) ∧ swrlb:lessThan(?p,?c)
→ hasCheapPrice(?m,1.0)
hasLimitValue(Cheap,?c) ∧ hasLimitValue(Expensive,?e) ∧
hasMenuPrice(?m,?p) ∧ swrlb:greaterThanOrEqual(?p,?c) ∧
swrlb:lessThan(?p,?e) ∧ swrlb:subtract(?r,?e,?p) ∧
swrlb:subtract(?sub,?e,?c) ∧ swrlb:divide(?div,?r,?sub)
→ hasCheapPrice(?m,?div)
hasLimitValue(Expensive,?e) ∧ hasMenuPrice(?m,?p) ∧
swrlb:greaterThanOrEqual(?p,?e)
→ hasCheapPrice(?m,0.0)

3. *Fuzzy Rules in MRP-1*

MRP-1 needs some fuzzy rules to get the value of recommendation. Table II shows eight fuzzy rules which are used in MRP-1. We use AND operator in every rule. The way to specify the rules from Table II can be explained as follows. For example, **Fuzzy Rule No.8** declares that we give high recommendation of some menus if the menu has high vote, high rate and cheap price.

Rule-3 is SWRL rule language representation for **Fuzzy Rule No.8**. Variable *?m* is menu, *?c* is *md* of cheap, *?hv* is *md* of high vote, and *?hr* represents the *md* of high rate in a menu.

4. *Deffuzification of MRP-1*

Defuzzification process is needed to get the crisp value of recommendation. We use a singleton function for total recommendation value defuzzification as shown in Fig. 4. The degree of recommendation is divided into three degrees; low (20 points), medium (50 points) and high (80 points).

Calculating *Total Recommendation Value (TRV)* can be done by weighted average calculation in (3). The equation needs the result of low, medium and high recommendation value from evaluating fuzzy rules. **Rule-4** is the SWRL representation for (3). This rule produces total recommendation value (*?t*) for a menu (*?m*).

TABLE II. FUZZY RULES IN MRP-1

Fuzzy Rule No.	1	2	3	4	5	6	7	8
Rate	L	L	L	L	H	H	H	H
Vote	H	H	L	L	L	L	H	H
Price	E	C	E	C	E	C	E	C
Recommendation	*LOW*				*MEDIUM*		*HIGH*	

*L : Low, H : High, C : Cheap, E : Expensive

Fuzzy Rule No.8:
IF Vote is high AND Rate is high AND Price is cheap
THEN Recommendation is high

Rule-3 :
hasHighRate(?m,?hr) ∧ hasHighVote(?m,?hv) ∧ hasCheapPrice(?m,?c) ∧
swrlb:lessThanOrEqual(?hr,?hv) ∧ swrlb:lessThanOrEqual(?hr,?c)
→ hasHighRecommendationValue(?m,?hr)
hasHighRate(?m,?hr) ∧ hasHighVote(?m,?hv) ∧ hasCheapPrice(?m,?c) ∧
swrlb:lessThanOrEqual(?hv,?hr) ∧ swrlb:lessThanOrEqual(?hv,?c)
→ hasHighRecommendationValue(?m,?hv)
hasHighRate(?m,?hr) ∧ hasHighVote(?m,?hv) ∧ hasCheapPrice(?m,?c) ∧
swrlb:lessThanOrEqual(?c,?hr) ∧ swrlb:lessThanOrEqual(?c,?hv)
→ hasHighRecommendationValue(?m,?c)

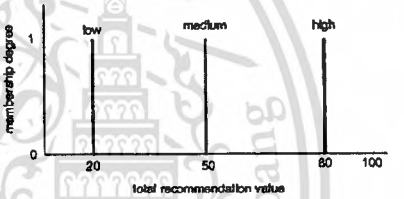


Figure 4. Defuzzification function of total menu recommendation points.

$$TRV = \frac{(lrv \times 20) + (mrv \times 50) + (hrv \times 80)}{lrv + mrv + hrv} \quad (3)$$

TRV : Total Recommendation Value, *lrv* : low recommendation value, *mrv* : medium recommendation value, *hrv* : high recommendation value,

Rule-4 :
hasLowRecommendationValue(?m, ?lrv) ∧
hasMediumRecommendationValue(?m, ?mrv) ∧
hasHighRecommendationValue(?m, ?hrv) ∧
swrlb:multiply(?mul1,?lrv,20.0) ∧ swrlb:multiply(?mul2,?mrv,50.0) ∧
swrlb:multiply(?mul3,?hrv,80.0) ∧ swrlb:add(?add1,?mul1,?mul2,?mul3) ∧
swrlb:add(?add2,?lrv,?mrv,?hrv) ∧ swrlb:divide(?t,?add1,?add2)
→ hasTotalRecommendationValue(?m,?t)

C. *Menu Recommendation Process: MRP-2*

MRP-2 will take out the preferred menu based on the taste. We also use the fuzzy concept. By defining *md* in taste, it will give more precise result since a very spicy menu is different from a little bit spicy menu. *Taste's md* can be assigned directly by putting some values. For example, *Menu2000A* has very sweet taste. Based on fuzzy value's modeling in Table I, we can write it as follows, *f(Menu2000A,taste)* = (*sweet*,0.8). MRP-2 process does not need a fuzzy rules and defuzzification because it only depends on one factor, taste. Tolerance value is added when querying menu with preferred taste. It extends the query. It also can give more choices even if there is no menu which has *Taste's md* exactly matches with user's request.

2009 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2009) December 7-9, 2009

IV. EXPERIMENTAL RESULT

A. Examples of Data Calculation

We take 50 sample menus data which is given to a person with random value of price, vote, rate and taste for simulation of MRP-1 and MRP-2. We calculated these sample data and resulting *TRV* and membership degree of taste. Fig. 5 shows the distribution of *TRV* in 50 sample menus. The x-axis is the range value of *TRV* and the y-axis is total count number or the amount of menus. Fig. 5 shows that the amount of menus which has *TRV* range from 80 to 82 points is 17 menus. Fig.6. shows the random membership degrees of sweet taste which are given to 50 sample menus.

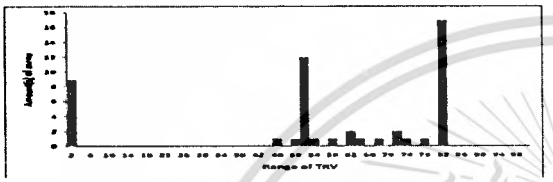


Figure 5. TRV distribution with 50 sample data.

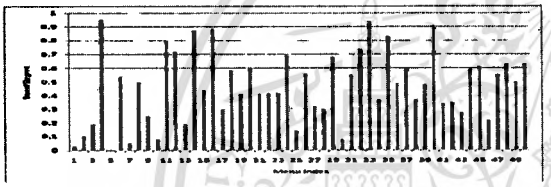


Figure 6. Sweet membership degree of 50 sample menus.

B. Query Example

We design query example using SQWRL in Protégé to be the final step to get recommended menus from MRP-1 and MRP-2.

Query-Rule 1 represents the query via process MRP-1. It takes out strongly recommended menus (*?m*) that is suggested to a person(*?p*). We assign "strongly recommended" to the menu which has *TRV*(*?trv*) equal or more than 80.0. Fig. 7 is the result of the query and shows information that person *Person_ONE* has 17 menus which has *TRV* point equals to 80.0 or more.

Query-Rule 2 represents the query via process MRP-2. We use a linguistic fuzzy value according to Table I. This query takes out the menu (*?m*) from a person (*?p*) which is in a preferred taste. In this case, we will take a very sweet taste (*?d*) menu with tolerance value (*?t*) equals to 0.05. The result shows very sweet menus which have *md* of sweet taste in the range of 0.75 – 0.85. Fig. 8(a) is the query result in Protégé's SQWRL Query Tab.

User can use these two processes, MRP-1 and MRP-2, at once and get the result as shown in Fig. 8(b), by combining **Query-Rule 1** and **Query-Rule 2**.

Query-Rule 1 :
hasMenu(?p,?m) ∧ hasTotalRecommendationValue(?m, ?trv) ∧ swrlb:greaterThanOrEqual(?trv, 80.0)
→ sqwrl:select(?p, ?m, ?trv)

Query-Rule 2 :
hasMenu(?p,?m) ∧ hasSweetTaste(?m,?s) ∧ hasValue(very,?d) ∧ hasValue(tolerance,?t) ∧ swrlb:subtract(?sub,?d,?t) ∧ swrlb:add(?add,?d,?t) ∧ swrlb:greaterThanOrEqual(?s,?sub) ∧ swrlb:lessThanOrEqual(?s,?add)
→ sqwrl:select(?p,?m,?s)

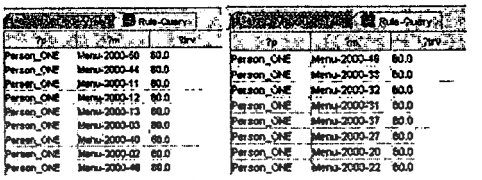


Figure 7. Query result for menu that has $TRV \geq 80$.

(a) (b)

Figure 8. (a) Query result for Query-Rule 2. (b) Query result from combining Query-Rule 1 and Query-Rule 2.

V. CONCLUSION AND FUTURE WORKS

Designing menu recommendation feature makes daily menu assistance system has more factors as consideration to suggest menus by giving the most recommended menus and the preferred menu based on taste. Since fuzzy ontology can use a vague concept, it is the best way to handle the recommendation feature factors. By using fuzzy ontology concept, the recommendation processes can be more precise and specific in giving the recommended and preferred menu. Protégé, SWRL and SQWRL can be used as tools to design a fuzzy ontology and perform recommendation query.

Our future work is to apply these menu recommendation feature into the daily menu assistance system as a semantic web application and expands the web services into agent-based daily menu assistance mobile services.

ACKNOWLEDGMENT

This research is funded by ASEAN University Network/Southeast Asia Engineering Education Development Network (AUN/SEED-Net) project through the Japan International Cooperation Agency (JICA).

REFERENCES

[1] S. Calegari, D. Ciucci, "Integrating Fuzzy Logic In Ontologies," ICEIS, 2006.

[2] D. Hatta Fudholi, N. Manecerat, R. Varakulsiripunth, "Ontology-Based Daily Menu Assistance System," ECTI-CON 2009.

[3] K. Mahan, S. Escott-Stump, "Krause's Food, Nutrition, and Diet Therapy," 11th edition, Philadelphia: Saunders, 2004.

[4] J. N. K. Liu, "Fuzzy Ontology Based System for Product Management and Recommendation," International Journal of Computers, Issue 3, Vol. 3, 2007.

[5] J. Gennari, et al. , "The evolution of Protégé 2000: An environment for knowledge-based systems development", International Journal of Human Computer Studies, 58(1):pp. 89–123, 2003.

[6] W3C, "SWRL: a Semantic Web Rule Language Combining OWL and RuleML," http://www.daml.org/2003/11/swrl/.

[7] Protégé Community Wiki , "SQWRL," http://protege.cim3.net/cgi-bin/wiki.pl?SQWRL, January 15, 2009.

[8] B. Coppin, "Artificial Intelligent Illuminated," Jones and Bartlett Publishers. London, US, pp. 529–551, 2004.