Universidad Nacional de Rosario

Tesina de grado
para la obtención del grado de
Licenciado en Ciencias de la Computación

Enfoque basado en ontologías para el desarrollo de un asistente web nutricional para pacientes diabéticos.

Autor:
María Clementina Latanzi

Directora:
Dra. Pamela Viale
Co-Directora:
Dra. Marta Basualdo

Departamento de Ciencias de la Computación
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Av. Pellegrini 250, Rosario, Santa Fe, Argentina

16 de noviembre de 2015

II

Resumen

La Diabetes Mellitus es una enfermedad crónica, caracterizada por altos valores de glucosa en sangre debido a una deficiencia absoluta (diabetes Mellitus Tipo 1) o parcial (Tipo 2) en la secreción de insulina. El número de pacientes con Diabetes Mellitus (DM) ha crecido a nivel mundial de manera exponencial a lo largo de los años. Esto justifica el enorme esfuerzo que la comunidad internacional viene realizando para abordar multidisciplinariamente esta enfermedad mediante el desarrollo de tecnologías adecuadas. En este trabajo nos proponemos sintetizar una ontología que facilite el cálculo de una dosis oral de la glucosa ingerida, mediante la especificación de los alimentos comerciales que el paciente incorpora en su organismo. Para tal fin primero construiremos una ontología de Alimentos, y luego una ontología de la Ingesta.

El trabajo presentado en esta tesina constituye un aporte preliminar en vías del desarrollo de una plataforma de telemedicina o "e-health" para el tratamiento de pacientes diabéticos. Esta plataforma tendrá como objetivo asistir en el monitoreo a distancia de los valores de ciertas variables, como por ejemplo el azúcar o glucosa en sangre, de los pacientes. De esta forma se busca evitar episodios que puedan dañar seriamente su calidad de vida.

Durante esta tesina se desarrolló un primer prototipo centrado en la alimentación del paciente. El mismo utiliza las ontologías que mencionamos más arriba, a saber, la ontología de alimentos y la ontología de ingestas. Este prototipo permite el conteo automático de carbohidratos de cualquier ingesta que ingrese el paciente. Además, cada ingesta ingresada es evaluada por el prototipo y en caso de haber recomendaciones nutricionales al respecto, se mostrarán en la interfaz.

Agradecimientos

Quiero agradecer en primer lugar a Marta y Pamela, por confiar en mi para llevar a cabo este proyecto, por brindarme gran parte de su tiempo, incluyendo fin de semanas y feriados, por la gran dedicación y ayuda que me fue indispensable para realizar este trabajo.

Además mi agradecimiento al Fondo Fiduciario de Promoción de la Industria del Software (FONSOFT) por la adjudicación de la beca jóvenes profesionales TIC y a la Municipalidad de Rosario por la distinción en el concejo deliberante por la participación en el grupo de investigación sobre el manejo eficiente de tecnologías para diabetes.

Quiero agradecer también a todos mis compañeros ya convertidos en amigos, que me bancaron todos estos años (cosa no menor). En especial a mi compañero, hermano, amigo, con el que estudiamos desde análisis hasta Sistemas Operativos, gracias por tu mística Gon!.

Gracias también a todos los profesores y ayudantes que son el pilar fundamental de la carrera, que siempre tienen buena predisposición y además de eso son excelentes personas.

Gracias infinitas a mis padres que están siempre al pie del cañon, que son docentes y me enseñaros el valor de la educación, no como medio para obtener cosas sino por el valor del conocimiento en sí que nos hace ricos.

Gracias a mis hermanas que piensan que todo lo se y todo lo puedo y creen más en mí que yo misma.

Gracias a mis amigos y amigos que son incondicionales, que me alentaron en este camino y hacen de la vida algo hermoso.

Gracias a mi familia del corazón por el aguante, en especial a Nora, Juli y Marce que me incentivaron a concluir esta etapa.

Y por último gracias a la carrera, a las casualidades o a la vida por hacerme conocer a la persona más maravillosa con la que pude compartir este hermoso camino.

Índice general

Índice de figuras

Capítulo 1

Introducción

Este trabajo es parte de un proyecto más grande, interdisciplinario, que se basa en el manejo eficiente de tecnologías para diabetes. La diabetes mellitus es una enfermedad crónica que necesita un control exhaustivo. Por ello, es importante disponer de medios que faciliten ese control y su seguimiento por parte de los médicos y nutricionistas. Se considera que la disponibilidad de un servicio de telemedicina o "e-health" sería de gran utilidad para monitorear a distancia aquellos indicadores que podrían ayudar a identificar y prevenir episodios que puedan dañar seriamente su salud y su calidad de vida. E-health se refiere a la práctica de cuidados sanitarios apoyados en tecnologías de información y comunicaciones. Los servicios de tipo "e-health" se están convirtiendo en una alternativa atractiva para el cuidado de enfermos a distancia [CRM06; M+07]. Es importante destacar que la educación y asistencia nutricional es crucial para lograr un control glucémico de los pacientes diabéticos [Gag13]. Esta gestión de la nutrición, no sólo es la base del tratamiento sino de la prevención de la diabetes en personas adultas. En este contexto, el conteo de carbohidratos ayuda a fomentar una correcta planificación de comidas del paciente diabético, contribuyendo a un autocontrol en la cantidad de glucosa en sangre. El incremento de la glucosa en sangre luego de una comida depende del equilibrio entre la insulina del cuerpo y lo que se consume de carbohidratos. Si se tiene un equilibrio adecuado entre carbohidratos e insulina, es posible que el nivel de glucosa en la sangre se mantenga dentro de los niveles deseados.

Teniendo en cuenta este marco modelaremos una ontología del paciente enfocada en la nutrición. Una ontología es considerada como "una especificación explícita de una conceptualización" [Gru93], que permite representar el conocimiento consensuado sobre un dominio específico, convirtiendo a las ontologías en una herramienta valiosa para este proyecto. En este trabajo se desarrollarán dos ontologías. Una ontología que modelará los alimentos y otra ontología que describirá la ingesta del paciente, esta última pensada para asistir al paciente diabético en su control nutricional. Ambas ontologías serán integradas con otra, ya existente, que modela el sistema endocrino humano. El modelado de conocimiento utilizando ontologías permite la interoperabilidad que necesita la plataforma de tipo "e-health" para la asistencia de los pacientes diabéticos. Esto es de fundamental importancia debido a que los posibles futuros usuarios de la plataforma son

los médicos, enfermeras, ingenieros químicos, especialistas en ciencias de la computación y, por supuesto, también los pacientes. El objetivo es que la ontología permita superar el problema asociado al uso de diferentes terminologías por parte de los distintos usuarios de la futura plataforma. En el caso particular de la plataforma de "e-health", esta característica es muy ventajosa para un paciente que desee preguntar e interiorizarse con su tratamiento. Con esta herramienta de fácil acceso para el paciente se espera que aumente la participación del mismo en su tratamiento, lo que tendría un impacto directo en su calidad de vida.

1.1. Motivación

La diabetes es una de las enfermedades no transmisibles más comunes. Se estima que en el mundo existen 382 millones de personas que tienen diabetes y 316 millones sufren tolerancia anormal a la glucosa corriendo un riesgo muy elevado de contraer la enfermedad (ver Fig 1.1).

La diabetes está asociada a unos altos costes sanitarios para la sociedad, resultando en la pérdida de productividad laboral y el descenso de los índices de crecimiento económico. En el mundo, la cantidad de enfermos de diabetes se han incrementado y a finales de 2013, la diabetes causó 5,1 millones de muertes y un coste de 548.000 millones de USD en gasto sanitario. Las nuevas estimaciones muestran una tendencia creciente de diabetes en personas cada vez más jóvenes, y de seguir los patrones demográficos actuales, más de 592 millones de personas sufrirán de diabetes en la próxima generación (1 de cada 10)[IDF13]. En Argentina más de 1.6 millones de personas (que representan el 6 % de la población) sufren diabétes y se cree que casi medio millón de personas aún no fue diagnosticada. Además, las estimaciones indican que para el 2035 habrá 2.2 millones de personas con diabetes mellitus. Sólo en el 2013 hubo 15.328 muertes asociadas a esta enfermedad y se calcula que el gasto promedio del paciente con DM se elevó a 1174 USD.

En nuestro país se han implementado algunas estrategias para disminuir la carga que representa el tratamiento de la diabetes en el sistema de salud. Un ejemplo de esto es el Programa para la Prevención, Atención y Tratamiento de Personas con Diabetes (PROPAT) orientado a mejorar la calidad de atención de los pacientes y prevenir complicaciones agudas. Este programa ayudó a disminuir en un 30 por ciento los costos de atención. El estudio se basó en incluir innovaciones al sistema de atención de los pacientes, tales como prescripción de dieta por la nutricionista, interconsulta anual con otros especialistas, educación del paciente y otras estrategias de prevención [Gag+06]. En el proyecto que aquí se presenta, se desarrollarán herramientas computacionales y nuevas bases de conocimiento que podrían integrarse a programas como el PROPAT. El desarrollo de la plataforma e-health facilitaría notablemente el manejo de la información de pacientes dibéticos, mejoraría el nivel educativo de los mismos, guiándolos en una mejor alimentación, contribuyendo a construir su propio modelo de comportamiento para fortalecer actitudes de auto-control y así disminuiría la dependencia de los expertos involucrados.

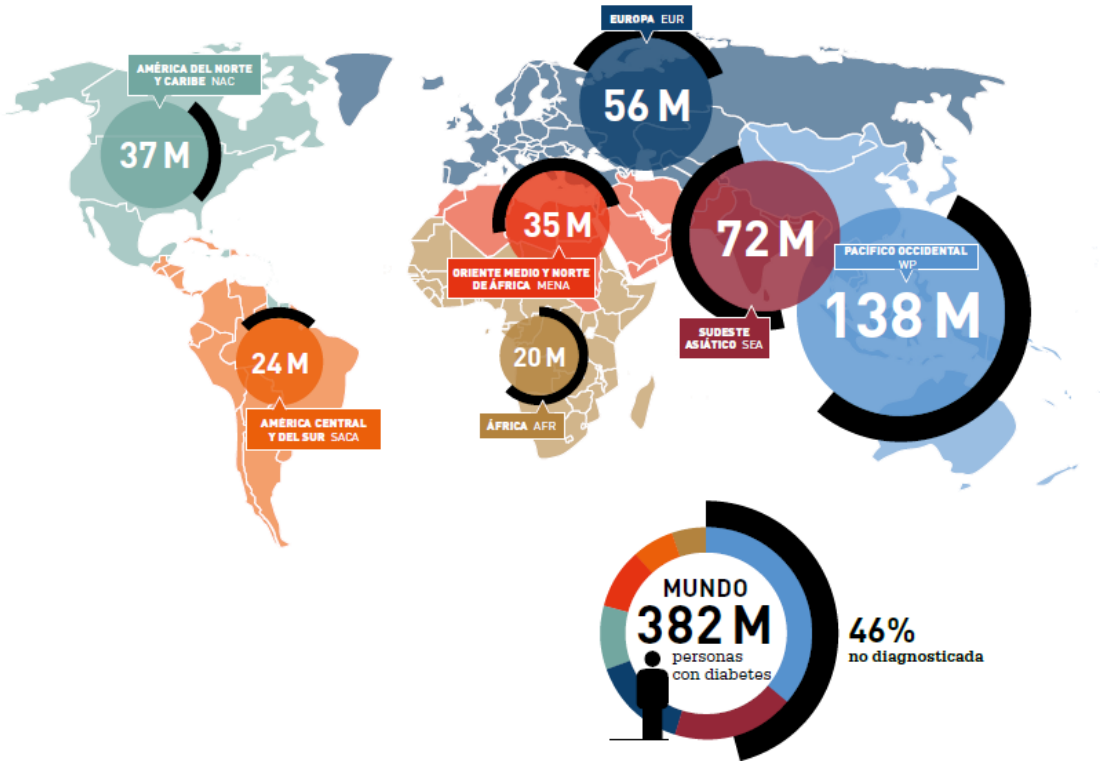

Figura 1.1: Número de personas con diabetes por región. Figura tomada de [IDF13].

1.2. ¿Qué es la Diabetes?

La diabetes es una enfermedad crónica que se produce cuando el cuerpo no puede producir suficiente insulina o no puede utilizar la insulina de manera eficaz. La insulina es una hormona producida en el páncreas que permite que la glucosa, proveniente de los alimentos ingeridos, se metabolice e ingrese a las células del cuerpo donde se convierte en la energía que necesitan los músculos y tejidos para su funcionamiento. De manera gráfica podríamos decir que la insulina actúa como una llave que permite a las células del cuerpo absorber la glucosa y utilizarla como energía (ver fig. 1.2).

Hay tres tipos principales de diabetes:

- diabetes mellitus tipo 1
- diabetes mellitus tipo 2
- diabetes mellitus gestacional

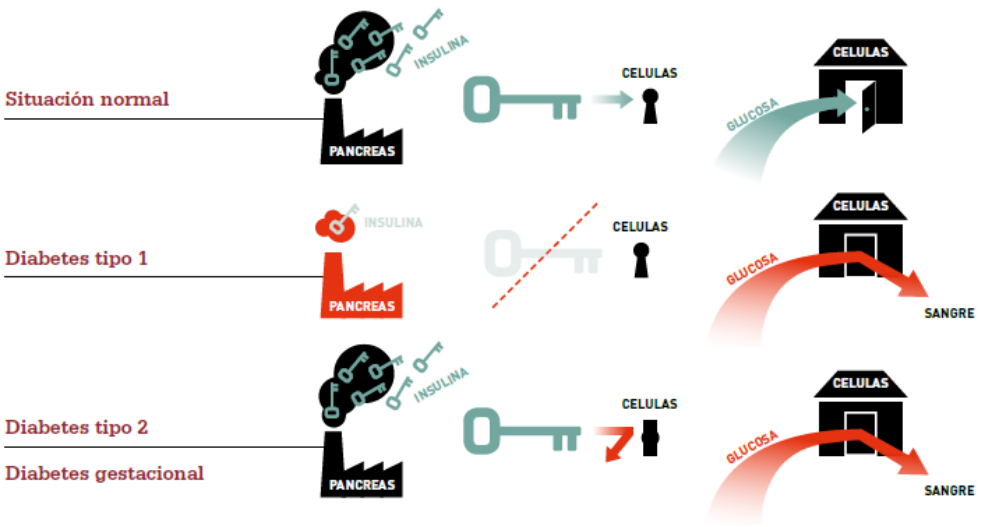

Figura 1.2: Representación de la función de la insulina. Figura tomada de [IDF13].

1.2.1. Diabetes mellitus tipo 1(DMTI)

La diabetes tipo 1 es causada por una reacción autoinmune (con inicio repentino) donde el sistema de defensa del cuerpo ataca las células beta productoras de insulina en el páncreas. Como resultado, el cuerpo ya no puede producir la insulina que necesita. Se desconoce aún la causa de la DMTI, y no se puede prevenir con el conocimiento actual. Sus síntomas consisten, entre otros, en la excreción excesiva de la orina (poliuria), hambre constante (polifagia), pérdida de peso, trastornos visuales y cansancio. Estos síntomas pueden aparecer de forma súbita. Las personas que tienen este tipo de diabetes necesitan insulina todos los días con el fin de controlar los niveles de glucosa en su sangre.

1.2.2. Diabetes mellitus tipo 2(DMTII)

La diabetes tipo 2 es el tipo más común de diabetes. Por lo general ocurre en adultos, pero se ve cada vez más en niños y adolescentes. En la diabetes tipo 2, el cuerpo es capaz de producir insulina pero o bien esta no es suficiente o el cuerpo es incapaz de responder a sus efectos (también conocidos como resistencia a la insulina). Esto conduce a una acumulación de glucosa en la sangre. Muchas personas con diabetes tipo 2 no son conscientes de su enfermedad durante mucho tiempo porque los síntomas pueden tardar años en aparecer o ser identificados. Durante ese tiempo el cuerpo está siendo dañado por el exceso de glucosa en la sangre. A menudo estos pacientes son diagnosticados sólo cuando las complicaciones de la diabetes ya se han desarrollado. Aunque las razones por las cuales se desarrolla la diabetes de tipo 2 aún no se conocen, hay varios factores de riesgo importantes. Éstos incluyen:

- la obesidad
- dieta pobre
- la inactividad física
- edad avanzada
- antecedentes familiares de diabetes
- etnicidad
- altos niveles de glucemia durante el embarazo que afectan al feto

A diferencia de las personas con diabetes tipo 1, la mayoría de las personas con diabetes tipo 2 por lo general no requieren dosis diarias de insulina para sobrevivir. Muchas personas son capaces de controlar su enfermedad a través de una dieta sana y una mayor actividad física o tomando medicación vía oral. Sin embargo, si no son capaces de regular sus niveles de glucosa en sangre, se puede prescribir insulina.

1.2.3. Diabetes mellitus gestacional(DMG)

Las mujeres que desarrollan una resistencia a la insulina y la consiguiente hiperglucemia durante el embarazo se dice que tienen diabetes gestacional. La diabetes gestacional tiende a ocurrir alrededor de la semana 24 de embarazo. La condición surge porque la acción de la insulina está bloqueada, probablemente por hormonas producidas por la placenta. Como la diabetes gestacional normalmente se desarrolla más tarde en el embarazo, el feto ya está bien formado, pero sigue creciendo. Por tanto, el riesgo inmediato para el bebé no es tan grave como para aquellos cuya madre tenía diabetes tipo 1 o diabetes tipo 2 antes del embarazo. De todas formas, la diabetes gestacional no controlada puede tener consecuencias graves tanto para la madre como para su bebé. Las mujeres con diabetes gestacional tienen que vigilar y controlar sus niveles de glucosa en sangre para reducir al mínimo los riesgos para el bebé. Normalmente, esto se puede lograr adoptando una dieta sana y haciendo ejercicio moderado, pero en algunos casos puede ser necesario también administrar insulina o medicación oral.

1.2.4. Complicaciones de la diabetes

Las personas con diabetes corren el riesgo de desarrollar una serie de problemas de salud que pueden provocar discapacidad o la muerte. Tener constantemente altos los niveles de glucosa en sangre puede conducir a enfermedades graves que afectan al corazón y a los vasos sanguíneos, ojos, riñones y nervios. Las personas con diabetes también tienen un mayor riesgo de desarrollar infecciones. En muchos países la diabetes es la principal causa de las enfermedades cardiovasculares, la ceguera, la insuficiencia renal y la amputación de miembros inferiores. La diabetes también acarrea una carga de complicaciones

agudas, como el nivel de glucosa en sangre excesivamente alto o excesivamente bajo (hipoglucemia). En el caso de la hipoglucemia, de no tratarse a tiempo, ésta puede producir convulsiones o hacer perder el conocimiento (desmayarse o entrar en coma).

1.3. Tratamientos

Desde un punto de vista global, la insulinoterapia puede dividirse en convencional e intensiva.

1.3.1. Insulinoterapia convencional

La insulinoterapia convencional comprende el uso de una o dos inyecciones de insulina (en ocasiones más) de acción intermedia/prolongada supliendo así las necesidades basales. La principal ventaja de esta terapia es la simplicidad de los regímenes (menos controles de la glucemia al día, variaciones de dosis menos frecuentes y menos administraciones de insulina). Por el contrario este sistema consigue un peor control glucémico y no logra muy buenos controles postprandiales [1].

1.3.2. Insulinoterapia intensificada

Esta forma intenta minimizar tanto el impacto de la enfermedad y sus complicaciones, como los efectos colaterales del tratamiento. Se puede realizar a través de inyecciones múltiples de insulina (IMI) o mediante la infusión continua de insulina subcutánea (ICIS).

Para el caso de tratamiento con múltiples inyecciones en el día se trabaja por lo general con tres aplicaciones diarias, utilizando insulina de acción corta, intermedia y rápida según sea necesario. Este tipo de tratamiento requiere especial atención del paciente, el cual debe controlar las dosis de insulina y los tipos de la misma de acuerdo a la situación del día y a la magnitud de cada comida. Existen técnicas de insulina de acción prolongada, la cual brinda cierto control para el paciente durante el período de un día, pero teniendo que realizar suplementos en los horarios que más se necesite.

Por otro lado, en el caso de tratamiento con infusión continua de insulina subcutánea con bomba, se utilizan, valga la redundancia, bombas de insulina; las cuales se han perfeccionado con los avances tecnológicos. Actualmente estos dispositivos funcionan administrando insulina mediante un tubo pequeño(catéter) y una cánula (equipo de infusión) que se implanta debajo de la piel. Dichos dispositivos se programan para administrar la insulina según las necesidades individuales (liberando una cantidad pre-

[1] El término postprandial se utiliza para referirse al estado fisiológico inducido por la ingesta de alimentos, y se caracteriza por presentar un alto nivel de azúcar en sangre (hiperglucemia), así como otros macronutrientes que hayan sido digeridos y absorbidos. Este estado se extiende aproximadamente desde 2 horas después del consumo de nutrientes hasta que se llega al nivel basal de glucosa en sangre (normoglicemia).

determinada de insulina de acción corta durante las 24 horas) y el paciente activa las dosis para cubrir las comidas y corregir la fluctuación de la glucemia.

1.4. El Páncreas Artificial (PA)

El PA es un dispositivo en desarrollo cuya finalidad es ayudar a las personas con diabetes a controlar automáticamente sus niveles de glucosa en la sangre proporcionándoles un substituto endócrino con las funciones de un páncreas saludable. El PA es un dispositivo exógeno con la capacidad de regular la concentración de glucosa en sangre mediante la liberación de insulina o glucagón en cantidades necesarias con el objetivo de conservar los niveles de glucosa en los parámetros deseables. [JO98]. Este dispositivo se compone de tres partes:

- Sensor continuo de glucosa
- Bombas de infusión continua
- Algoritmos de control

Hoy en día existen y están en fase de desarrollo dispositivos que cumplen con estas características. Estos dispositivos se componen de un monitor continuo de glucosa que va colocado sobre la piel con una pequeña aguja y que envía cada cinco minutos la información medida de la cantidad de glucosa en sangre a un teléfono inteligente (smartphone). Allí, mediante cálculos matemáticos (algoritmo de control) se determina la dosis de insulina necesaria con mayor exactitud y se envía la orden de liberarla a un administrador automático (bomba de infusión de insulina) que el paciente lleva conectado a su cuerpo a través de una pequeña aguja (ver en Fig. 1.3).

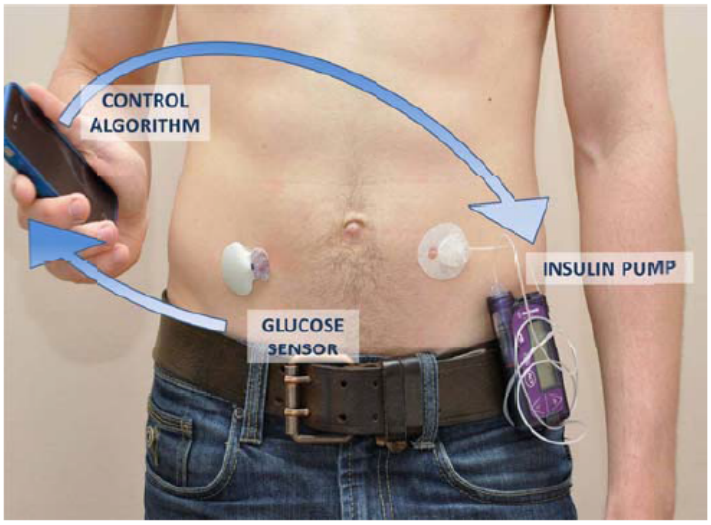

Figura 1.3: El páncreas artificial. Figura tomada de [DTU13].

1.4.1. Sensores Continuos de Glucosa

Con el paso del tiempo, las técnicas de monitorización de la glucosa han mejorado de forma considerable. Pese a esto siguen siendo el cuello de botella en el desarrollo del PA [HCD06] . Una serie de fabricantes han venido trabajando en el desarrollo de distintos sistemas mínimamente invasivos o no invasivos de monitorización continua de la glucosa:

- Sensores Mínimamente Invasivos. Son pequeños dispositivos que le suministra al paciente las lecturas de la glucosa en el líquido intersticial[2]. Los componentes de la monitorización son: un monitor (dispositivo pequeño del tamaño de un teléfono), y un sensor que mide la glucosa en el líquido intersticial y que se inserta en la zona del vientre. Posee una aguja o punta fina y flexible que es implantada en el tejido subcutáneo (ver fig. 1.4). Algunas limitaciones actuales asociadas a estos sensores son:
  - Tienen un periodo limitado de funcionamiento (entre tres y siete días).
  - Pueden producir infecciones cutáneas.
  - Necesitan recalibraciones frecuentes lo que implica pinchazos dolorosos.

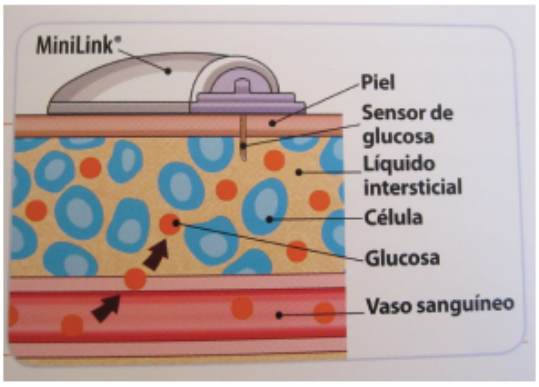

Figura 1.4: Sensor comercial de Medtronic. Figura tomada de [Med13]

- Los sensores no invasivos parecen ser los más prometedores a futuro, aunque aún están en fase de desarrollo [TMP07; OCJ09]. En este sistema la medición de glucosa se realiza mediante un reactivo (basado en la enzima glucosa oxidasa). Este reactivo entra en contacto con la piel y esa reacción enzimática es la que permite medir la concentración de glucosa. Hoy en día no existen ejemplos comerciales que hayan sido aprobados para su uso pero existen versiones con formato de reloj pulsera como el GlucoWatch (ver fig. 1.5).

[2]Es el líquido contenido en el intersticio, o espacio entre las células. Consiste en un solvente acuoso que contiene aminoácidos, azúcares, ácidos grasos, coenzimas, hormonas, neurotransmisores, sales minerales y productos de desecho de las células.

Figura 1.5: Monitor continuo de glucosa. Figura tomada de [Cad+06]

1.4.2. Bombas De Infusión Continua

La bomba de infusión continua es un dispositivo electromecánico pequeño (del tamaño aproximado de un teléfono móvil) capaz de suministrar, mediante su programación y de manera controlada, microbolos de insulina. Este dispositivo sustituye el proceso de múltiples inyecciones de insulina. La bomba de insulina se puede programar de modo que administre la insulina automáticamente las 24 horas del día para controlar la glucosa en sangre entre las comidas y durante el sueño. Además se puede determinar el tamaño del bolo usando una función de las bombas de insulina que ayuda a calcular el bolo adecuado basándose en la cantidad de carbohidratos ingerida, la insulina remanente en el paciente y la tasa de glucemia antes de la ingesta. En la actualidad podemos encontrar dos tipos de bombas de insulina:

- Bombas de infusión subcutánea. Administran la insulina a través de un tubo pequeño llamado catéter (que debe cambiarse cada 2 o 3 días) y una cánula (denominados equipo de infusión).

- Bombas implantables. Se colocan quirúrgicamente debajo de la piel (en la parte inferior del abdomen) se supervisa con un control externo que es exclusivo para cada bomba, lo que evita posibles interferencias. La bomba contiene un depósito para la insulina que también está debajo de la piel y debe ser repuesto después de 45 días.

1.4.3. Algoritmos de Control Empleados para el PA

La función del algoritmo de control es regular automáticamente la dosis exacta de insulina que debe administrar la bomba de insulina en cada momento, sin requerir la intervención del paciente. Este controlador también es el responsable de prevenir o minimizar los episodios de hipoglucemia e hiperglucemia y debe ser capaz de variar la administración de insulina en función de los cambios de los niveles de glucosa que se vayan registrando en el paciente a lo largo del día teniendo en cuenta las condiciones de ayuno, ingestas o actividad física diaria ([VPH10]). En los últimos tiempos se han desarrollado una gran cantidad de algoritmos de control que buscan cumplir con los requisitos antes citados. Pese a esto no existe aún uno que cumpla con todos los requerimientos de un modo efectivo.

Vamos a hacer foco en dos de estos algoritmos, ya que como mostraremos más adelante van a ser representados ontológicamente, en [M15].

- Control Proporcional Integrativo Derivativo (PID). Es un mecanismo de control por realimentación, ampliamente utilizado en la industria de procesos. Este calcula la desviación o error entre un valor medido y un valor deseado. El algoritmo del control PID consiste de tres parámetros distintos: el Proporcional, el Integral, y el Derivativo. El valor Proporcional depende del error actual. El Integral depende de los errores pasados y el Derivativo es una predicción de los errores futuros.

- Control Predictivo basado en Modelos (MPC). Este tipo de controladores están dentro de los controladores óptimos (aquellos en los cuales las actuaciones responden a la optimización de un criterio). Cuenta con modelos dinámicos del proceso de estudio en cuestión, obteniéndolo mediante técnicas de identificación [L99]. El criterio a optimizar en este controlador se relaciona con el comportamiento a futuro del sistema, que se predice gracias al modelo dinámico del mismo, denominado modelo predictivo. La principal virtud de los modelos de control predictivos es que permiten que el intervalo de tiempo actual sea optimizado, teniendo en cuenta intervalos de tiempo futuros y pasados. Este tipo de controladores tiene la capacidad de anticiparse a los acontecimientos futuros y actuar en consecuencia. En este contexto otro algoritmo de control a destacar es el denominado Control Predictivo Funcional (PFC) que es la tercera generación de una familia de algoritmos de control basados en modelos [JO09] y el mismo fue ampliamente aplicado en control de procesos [P+14].

Capítulo 2

Ontologías

2.1. Web semántica y ontologías

El objetivo de la Web Semántica es de proveer de estructura al contenido significativo de las páginas web, según Tim Beners-Lee (creador de la World Wide Web) y sus colaboradores [BHL14]. La Web Semántica no se considera por separado, sino que es más bien una extensión de la World Wide Web actual. En la Web Semántica se le da un significado bien definido a la información, lo que facilita un trabajo en conjunto entre las computadoras y los usuarios. Lo que se busca es que las máquinas se vuelvan más capaces de procesar y "entender" los datos que se limitan a mostrar en la actualidad. Para que la Web Semántica pueda funcionar, las computadoras deben tener acceso a colecciones estructuradas de información y disponer de un conjunto de reglas para realizar inferencias y llevar a cabo razonamientos automatizados. Una de las soluciones a este problema es proporcionada por un componente básico de la Web Semántica que son las llamadas ontologías.

2.1.1. ¿Qué son las ontologías?

En general en el campo de la ciencias de la computación hay un consenso para definir a las ontologías como una especificación explicita y formal de una conceptualización compartida [Gru93]. Expliquemos un poco mejor esta definición, convertida ya en estándar:

- Conceptualización hace referencia a un modelo abstracto de algún fenómeno del mundo del que se identifican los conceptos que son relevantes.

- El término explícito hace referencia a la necesidad de especificar de forma consciente las distintas partes que conforman una ontología como conceptos, propiedades, relaciones, funciones, axiomas y restricciones que la componen.

- Formal indica que la especificación debe representarse por medio de un lenguaje de representación formalizado, que sea legible e interpretable por máquinas.

- Compartida refiere a que tiene que haber un consenso previo sobre la información que ha sido acordado por un grupo de expertos.

En nuestro caso el modelo abstracto del mundo real que queremos modelar por una lado son los alimentos y sus composiciones nutricionales y por otro lado las ingestas de los pacientes diabéticos. El conocimiento debe ser aceptado por los diferentes actores del dominio como médicos, nutricionistas, ingenieros químicos, informáticos, enfermeras y pacientes.

En general, una ontología es una descripción explícita y formal de conceptos de un dominio de discurso. Las mismas se componen de:

- Conceptos: son las ideas básicas que se intentan formalizar. Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.
- Relaciones: representan la interacción y enlace entre los conceptos de un dominio. Permiten organizar las clases en una jerarquía taxonómica. Por ejemplo: *subclase-de*, *parte-de*, *parte-exhaustiva-de*, *conectado-a*, etc.
- Funciones: son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, pueden aparecer funciones como: *asignar-fecha*, *categorizar-clase*, etc.
- Instancias: se utilizan para representar objetos determinados de un concepto.
- Reglas de restricción o axiomas: son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Por ejemplo: *Si A y B son de la clase C entonces A no es subclase de B* o *Para todo A que cumpla la condición B1, A es C*, etc.

Los conceptos, también llamados clases, pueden ser descriptos mediantes relaciones, que suelen llamarse propiedades, o mediante restricciones sobre estas propiedades. Las clases son el centro de la mayoría de las ontologías, ya que describen los conceptos de un dominio. Por ejemplo, una clase *Alimentos* podría representar todos los alimentos consumidos frecuentemente en nuestro país. Alimentos específicos como *queso untable* son instancias de esta clase, también llamados individuos de clases. Una ontología junto con los individuos de clases constituyen una base de conocimientos.

2.1.2. Lenguajes de representación

Las ontologías permiten representar el conocimiento de manera intuitiva y extensible mediante el uso de jerarquías conceptuales. La comunidad de la web semántica ha invertido mucho esfuerzo para facilitar su creación y mantenimiento, desarrollando lenguajes de representación estándar, editores de ontologías, razonadores, repositorios, etc. En Inteligencia Artificial (IA) hay una larga tradición en el desarrollo y el uso de lenguajes de ontologías. En la actualidad, los lenguajes de ontologías más importantes para la Web son los siguientes:

- *XML*: proporciona una forma de escribir datos que es independiente de lenguajes, plataformas y herramientas, y da una sintaxis superficial para documentos estructurados que puede ser interpretada por computadores. No impone restricciones semánticas sobre el significado de estos documentos.

- *XML Schema(XMLS)*: es un lenguaje para la restricción de la estructura de los documentos XML. Permite definir qué elementos puede contener un documento XML, cómo están estructurados, qué atributos y de qué tipo pueden tener esos elementos. Es decir XMLS permite definir las etiquetas de marcado para estructurar los datos XML, por lo tanto es un metalenguaje. Permite definir nuevos lenguajes XML adecuados para su uso en distintos dominios de manera flexible y extensible. Sin embargo, XML o XMLS no son suficientes ya que aportan una estructura, pero no una semántica. La semántica es aparente para los humanos, pero no para las máquinas. La semántica estudia cómo los símbolos se refieren a los objetos. Es necesaria más expresividad para el procesamiento semántico. Así surge el lenguaje RDF como lenguaje para modelar datos.

- *RDF (Resource Description Framework)*: es un lenguaje para la representación de objetos (recursos) y las relaciones entre ellos. La información es un grafo dirigido etiquetado que modela las relaciones entre objetos. En el modelo se describen parejas de nombre y valores de una propiedad determinada, con tres tipos de objetos:

  Recurso. Es cualquier objeto: una página web, una base de datos, la dirección de la empresa, etc. Un recurso tiene un único identificador. Éstos, sean de la naturaleza que sean, se describen con un URI (Universal Resource Identifier) abstracto.

  Propiedad. Es la característica, atributo, relación o aspecto que describe un recurso. Cada propiedad tiene: significado, define sus valores posibles, define los tipos de recursos a los que es aplicable y también define la relación con otras propiedades.

  Sentencia. Agrupa un recurso determinado y una propiedad con un nombre y valor asociado al recurso.

  El modelo de datos RDF es similar a los enfoques de modelado conceptual clásicos como entidad-relación o diagramas de clases, ya que se basa en la idea de hacer declaraciones sobre los recursos en forma de expresiones ***sujeto-predicado-objeto***. Estas expresiones son conocidos como tripletas en terminología RDF (ver fig. 2.1). El sujeto indica el recurso y el predicado denota rasgos o aspectos del recurso y expresa una relación entre el sujeto y el objeto.

- *RDF schema (RDFS)*: es una extensión semántica de RDF. Un lenguaje primitivo de ontologías que proporciona los elementos básicos para la descripción de vocabularios. Describe propiedades y clases de recursos RDF, posee una semántica para dichas clases y propiedades para que puedan ser organizadas en forma jerárquica. Las clases definidas por RDFS son descritas por recursos *rdfs:Class* y *rdf:Resource*, y por propiedades como *rdf:type* y *rdfs:subClassOf*.
  Por lo tanto en RDFS una clase es cualquier recurso que tenga un tipo *(rdf:type)*

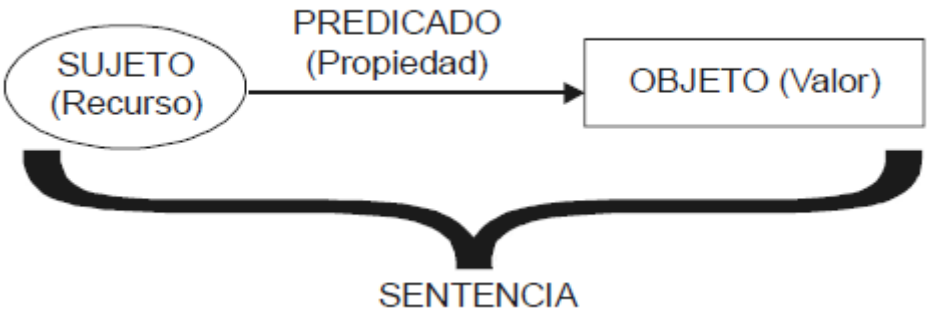

Figura 2.1: Representación gráfica de las declaraciones RDF.

como propiedad, cuyo valor sea un recurso *rdfs:Class*. Por ejemplo si queremos describir la clase Alimentos que contienen como principal nutriente las Proteínas con la URI como ex:AlimetosRicosEnProteinas
(usamos ex como http://www.example.org/schemas/alimentos), tendríamos la terna:

ex:AlimetosRicosEnProteinas rdf:type rdfs:Class

que representaría tal clase. La propiedad rdf:type indica que tal recurso es una instancia de una clase.
Para escribir propiedades específicas en RDFS usamos la clase rdfs:Property con las propiedades definidas rdfs:domain, rdfs:range y rdfs:subPropertyOf. Todas las propiedades en RDF se describen como instancias de la clase rdfs:Property. Así entonces si queremos decir que algo es una propiedad lo escribimos de la siguiente forma:

ex:tieneProteinas rdf:type rdfs:Property

Pero también necesitaremos describir cómo las propiedades se relacionan con los datos:

ex:tieneProteinas rdf:domain ex:AlimetosRicosEnProteinas
ex:tieneProteinas rdf:range xsd:integer

Esto indica que AlimentosRicosEnProteinas es una clase, tieneProteinas es una propiedad que se aplica para dicha clase y que el valor de tieneProteina debe ser un entero.
Pese a que RDFS resuelve algunas limitaciones de RDF, como por ejemplo la organización en jerarquías de clases de recursos, o las restricciones de rango de propiedades, no permite expresar por ejemplo clases disjuntas, o restricciones de car-

dinalidad o características especiales de las propiedades tales como la transitividad o la simetría.

- *OWL (Web Ontology Language)*: es un lenguaje de marcado semántico para publicar y compartir ontologías en la **World Wide Web** (ver fig. 2.2). Antes de la aparición de OWL, los lenguajes de representación de conocimiento no habían sido diseñados para ser compatibles con la arquitectura **WWW** ni con la Web Semántica. OWL se construye sobre RDF y RDFS, añadiendo constructores para aumentar expresividad. Añade más vocabulario para describir propiedades y clases tales como relaciones entre clases (por ejemplo, disjunción de clases), cardinalidad (por ejemplo, exactamente uno), igualdad, más tipos de propiedades, características de propiedades (por ejemplo, simetría), y clases enumeradas. OWL esta basado en lógica descriptiva (un fragmento decidible de lógica de primer orden), la cual es útil para realizar razonamiento computacional. Esta lógica permite computar jerarquías de clases y poder chequear inconsistencias en una ontología. Pese a que OWL es muy conveniente para optimizar el tiempo de procesamiento, tiene algunas carencias a la hora de modelar ontologías o construirlas a partir de un lenguaje natural. En 2009, se extiende OWL con un conjunto de características que permiten técnicas y semánticas mejor comprendidas y útiles para el desarrollo de nuevas herramientas, creando OWL2.

```
<owl:Class rdf:ID="Ingredientes"/>
        <rdfs:subClassOf rdf:resource=http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:Class rdf:ID="Menues">
        <rdfs:subClassOf rdf:resource=http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:ObjectProperty rdf:ID="menuTieneIngrediente">
        <rdfs:range rdf:resource="Ingredientes"/>
        <rdfs:domain rdf:resource="Menues"/>
</owl:ObjectProperty>
```

Figura 2.2: Ejemplo en OWL de la definición de una propiedad que relaciona un Menú con sus Ingredientes

2.1.3. Ontologías en OWL2

Como acabamos de mencionar, OWL2 es una extensión de OWL. Uno de los objetivos en el desarrollo de OWL2 fue el de mantener la compatibilidad con OWL, manteniendo válida en OWL2 cualquier ontología generada en OWL. OWL2 añade nueva funcionalidad

con respecto OWL. Algunas de estas nuevas funcionalidades son sintácticas, como por ejemplo la unión de clases disjuntas, mientras otras ofrecen una nueva expresividad, incluyendo:

- Facilidades sintácticas que hacen más fácil expresar algunas afirmaciones.
- Nuevos constructores que incrementan la expresividad.
- Soporte extendido para los tipos de datos.
- Capacidades de un modelado de metadatos simple.
- Anotaciones extendidas.

A continuación, se definirán diferentes conceptos presentes en OWL2, como son Clases, Propiedades, Restricciones, Individuos, Enumerados y Anotaciones.

Clases Las clases representan conceptos en el dominio y no los términos que denotan esos conceptos (ver fig. 2.3). Una clase define un grupo de individuos que pertenecen a la misma porque comparten algunas propiedades. Por ejemplo, *milanesa* y *tomate* son miembros de la clase *Alimentos*. Las clases pueden organizarse en una jerarquía de especialización usando *subClassOf*. Se puede encontrar una clase general llamada *Thing* que es una clase de todos los individuos y es una superclase de todas las clases de OWL. También se puede encontrar una clase general llamada *Nothing* que es la clase que no tiene instancias y es una subclase de todas las clases de OWL.

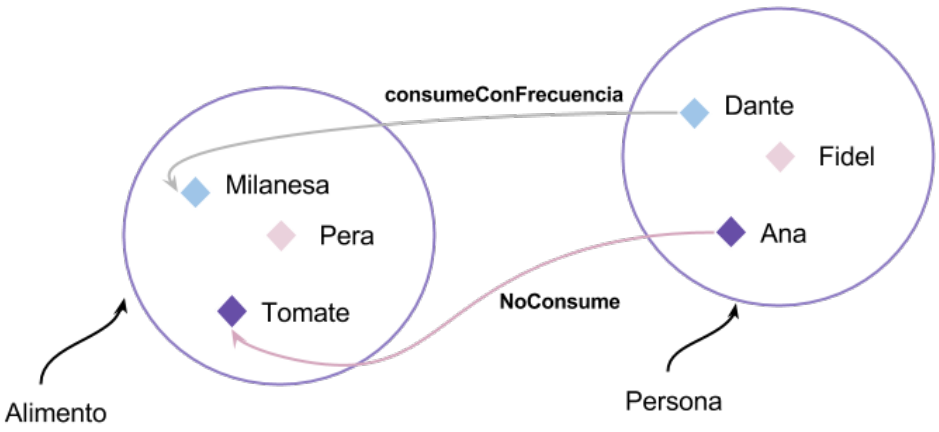

Figura 2.3: Representación de las clases Alimentos y Personas y sus relaciones.

Características de las Clases

- *Disjuntas*. Una Clase A es disjunta de otra B cuando ningún elemento de la Clase A sea elemento de la clase B y viceversa.

- *Jerarquías de clase.* Una subclase de una clase representa un concepto que es un *tipo de* concepto que la superclase representa. Si B es una subclase de A y C es una subclase de B, entonces C es una subclase de A. Por ejemplo, la clase *Frutas* podría estar definida como subclase de la clase *Alimento* y la clase *FrutasTipoA* como una subclase de la clase *Frutas*. De esto podemos deducir que si un individuo pertenece a la clase *FrutasTipoA*, entonces, también pertenece a la clase *Alimento*.

- *Condiciones Necesarias y suficientes (Clases Primitivas y Clases Definidas)* Condiciones necesarias pueden ser descritas como *si algo es miembro de esta clase entonces debe cumplir necesariamente estas condiciones*. Teniendo sólo condiciones necesarias, no podemos decir que *si algo cumple estas condiciones, entonces tiene que ser miembro de esta clase*. Una clase que sólo tiene condiciones necesarias es conocida como una Clase Primitiva. Una clase que tiene al menos un conjunto de condiciones necesarias y suficientes es conocida como Clase Definida y podría ser descripta como *si algo es miembro de esta clase entonces debe cumplir necesariamente estas condiciones y si algo cumple estas condiciones, entonces tiene que ser miembro de esta clase.*

Propiedades Las propiedades pueden utilizarse para establecer relaciones entre dos individuos llamadas *Object Property* (propiedades de objetos) o entre un individuo y un valor llamadas *Datatype Property* (propiedades de datos). Toda propiedad, a su vez, se compone de un dominio y un rango. El dominio es el conjunto de individuos a los que se pueden aplicar esta propiedad, mientras que el rango son el conjunto de individuos o valores de datos que puede tomar esa propiedad.

Object Property Las propiedades de objeto son relaciones que se establecen entre individuos de dos clases. En la fig. 2.3 tanto la propiedad *consumeConFrecuencia* y la propiedad *noConsume* son ejemplos de Object Property, ya que relacionan individuos de una clase, *Persona*, con individuos de otra clase, *Alimentos*.

OWL permite enriquecer el significado de las propiedades mediante el uso de las características de la propiedad. Entre las diferentes características que las propiedades pueden tener destacamos:

- Propiedad funcional. Es posible definir propiedades para que tengan un valor único. Dada una propiedad P, esta es funcional, si un individuo x se relaciona mediante P con a lo sumo un único indiviuo y. En la fig. 2.4, *tieneHistoriaClinica* es una propiedad funcional, ya que un paciente no puede corresponderse con mas de una historia clínica.

- Propiedad Inversa. Si una propiedad P es funcional inversa, la propiedad inversa I es una propiedad funcional. En la fig. 2.4 si consideramos la propiedad *pertenece* como inversa a la propiedad *tieneHistoriaClinica*. Como *tieneHistoriaClinica* es funcional, la propiedad *pertenece* es una propiedad funcional inversa.

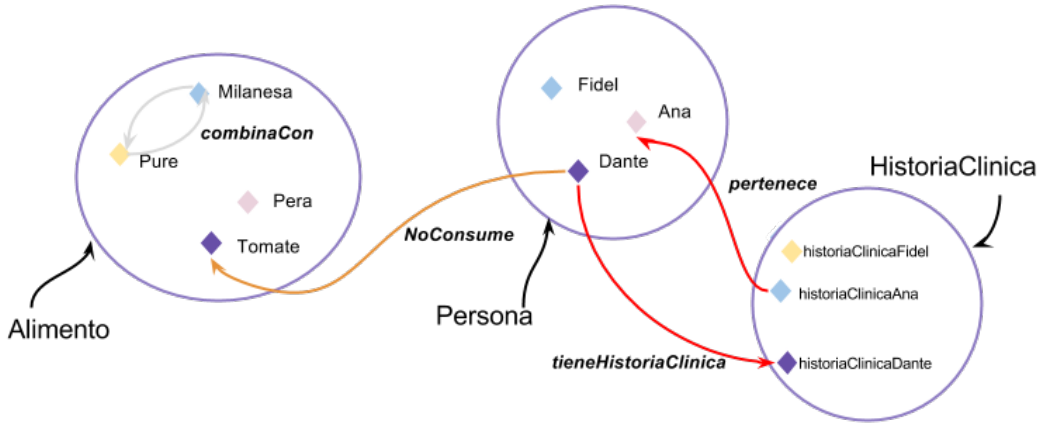

Figura 2.4: Ejemplos de Object Property

- Propiedad transitiva. Si una propiedad P es transitiva, y la propiedad relaciona un individuo A con otro individuo B y a su vez este individuo B con otro individuo C, entonces se puede inferir que el individuo A está relacionado con el individuo C a través de la propiedad P.
- Propiedad simétrica. Si una propiedad P es simétrica y la propiedad relaciona un individuo A con un individuo B, entonces el individuo B estará relacionado con el individuo A a través de la propiedad P. En la fig. 2.4 podemos ver que la propiedad *combinaCon* es una propiedad simétrica.
- Propiedad asimétrica. Si una propiedad P es asimétrica y la propiedad relaciona un individuo A con un individuo B entonces el individuo B no podría estar relacionado con el individuo A mediante la propiedad P. Ejemplos de propiedades asimétricas en la fig. 2.4 son *pertenece*, *NoConsume* y *tieneHistoriaClinica*.
- Propiedad reflexiva. Una propiedad P es reflexiva cuando la propiedad puede relacionar un individuo A consigo mismo.
- Propiedad irreflexiva. Si una propiedad P es irreflexiva, significa que relaciona un individuo A con un individuo B, donde el individuo A y el individuo B no pueden ser el mismo.

Datatype Property Las Datatype Property relacionan un individuo con un valor tipo de dato de XML o un literal RDF. En otras palabras, estas propiedades describen relaciones entre individuos y valores de datos (ver fig. 2.5).

Individuos Individuos son las instancias u objetos de las Clases o dominios definidos. Las instancias individuales son los conceptos más específicos representados en una base de conocimientos (ver fig. 2.6).

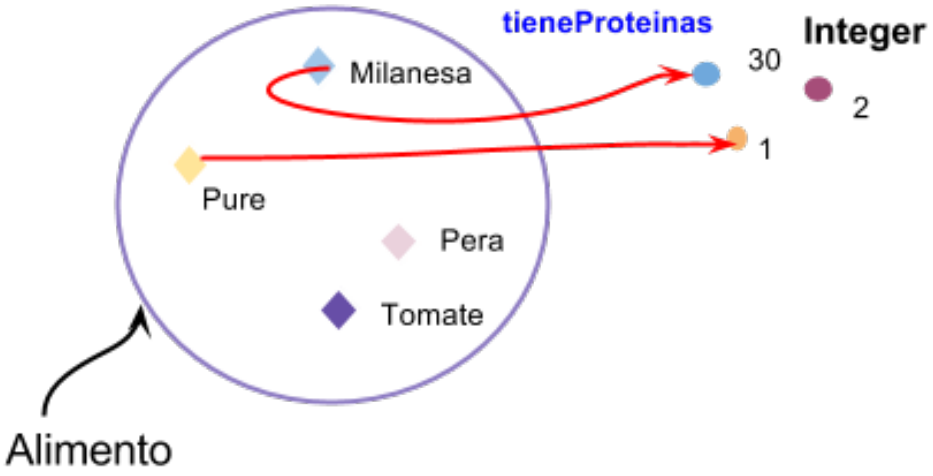

Figura 2.5: Ejemplo de Datatype Property.

Figura 2.6: Individuos.

Restricciones Una restricción define una clase de individuos basados en las relaciones entre miembros de una clase. En otras palabras, una restricción es un tipo de clase, de la misma manera que una clase con nombre es un tipo de clase. Una restricción describe una clase anónima (clase sin nombre). La clase anónima contiene todos los individuos que satisfagan la restricción. Por ejemplo, todos los individuos que tengan las relaciones requeridas para ser miembros de una clase. OWL permite definir todas las clases de individuos usando restricciones. Las restricciones OWL se pueden englobar en tres categorías:

- Restricciones de Cuantificadores (Existencial y Universal). Las restricciones de

cuantificadores, pueden ser categorizadas en restricciones existenciales y restricciones universales. La restricción Existencial define clases de individuos que participan al menos en una relación de una específica propiedad con individuos que son miembros de una clase específica. Las restricciones existenciales son las restricciones más comunes en las ontologías. Las restricciones existenciales también son conocidas como restricciones Some y representadas con el símbolo $\exists$.

Las restricciones Universales definen clases de individuos que dada una determinada propiedad sólo tiene relaciones mediante esta propiedad con miembros de una determinada clase. Las restricciones universales son dadas por el símbolo $\forall$. Las restricciones universales también son conocidas como restricciones AllValuesFrom. Para una propiedad dada, las restricciones universales no especifican la existencia de una relación. Estas restricciones indican que si una relación para una propiedad existe entonces los individuos de esta propiedad deben ser individuos miembros de una clase específica.

- Restricciones de cardinalidad. Las restricciones de cardinalidad definen la clase de individuos que tienen al menos, como máximo o exactamente un número específico de relaciones con otros individuos o valores de tipo de datos.

- Restricciones hasValue. Una restricción hasValue, denotada por el símbolo , define un conjunto de individuos que tienen al menos una relación con una propiedad específica con un individuo específico.

2.1.4. Reglas SWRL

Una ontología en OWL contiene una secuencia de hechos y axiomas. Los axiomas pueden ser de varias clases como axiomas de subclases, axiomas de equivalencias de clases, restricciones sobre propiedades. SWRL (Semantic Web Rule Language) propone extenderlos con axiomas de reglas ([Con04]). El Semantic Web Rule Language se basa en OWL DL y OWL Lite (sublenguajes de OWL) y utiliza el subconjunto de reglas RuleML del Rule Markup Language[1] las cuales son modeladas sobre cláusulas Horn. El propósito de SWRL es extender el conjunto de axiomas OWL al incluir cláusulas Horn que puedan ser combinadas con la base de conocimiento de OWL. Las cláusulas Horn representan condicionales if-then o más formalmente implicaciones. De esta manera, las reglas tienen la forma de una implicación entre un antecedente (referido como body en la sintaxis) y un consecuente (referido como head en la sintaxis).

$$antecedente \implies consecuente$$

Una regla puede ser interpretada como sigue: *si las condiciones especificadas en el antecedente son verdaderas entonces las condiciones especificadas en el consecuente también son verdaderas.* El antedecente y el consecuente ambos, están constituídos por cero o más

[1] http://ruleml.org

átomos. A su vez, tanto el antecedente y consecuente consisten en conjunciones positivas de los átomos.

$$atom \wedge atom \ldots \implies atom \wedge atom$$

Por lo tanto si se quiere representar una regla que refiere a un OR entre dos átomos, tendremos que representarla como dos reglas separadas

$$atom_1 \vee atom_2 \implies atom_3 \equiv (atom_1 \implies atom_3) \wedge (atom_2 \implies atom_3)$$

Los átomos pueden ser de la forma C(x), P(x, y), sameAs(x, y), differentFrom(x, y), donde C es una descripción OWL, P es una propiedad OWL y x e y pueden representar variables, individuos OWL o valores de datos OWL. Las variables son marcadas con un signo de "?" como prefijo. Veamos por ejemplo si quisiéramos dar una recomendación determinada dependiendo de la ingesta que realice el paciente. Para esto expresamos que si incluimos frutas en una comida, deberíamos evitar de consumirlas maduras. Podríamos escribirlo de la siguiente forma:

$$Ingesta(?x) \wedge tieneComida(?x, ?y) \wedge Frutas(?y) \implies tieneRecomendacion(?x, evitarFrutaMadura)$$

Donde *Ingesta* y *Frutas* son clases, *tieneComida* y *tieneRecomendacion* propiedades y *evitarFrutaMadura* un individuo de una clase *Recomendacion* y ?x e ?y variables.

Una de las características más potentes de SWRL es su capacidad para soportar predicados (built-in) definidos por el usuario. Un *built-in* es un predicado que toma uno o más argumentos y evalúa a true si los argumentos satisfacen el predicado. Un gran número de buil.ins para operaciones matemáticas y de String están definidos en el archivo *swrlb.owl* cuyo nameSpace es *http://www.w3.org/2003/11/swrlb*. Por convención los predicados SWRL deben estar precedidos por dicho nameSpace, por ejemplo:

$$swrlb : lessThan(?x, 10)$$

Indica que toma dos argumentos y retorna true si el primer argumento es menor que el segundo.

2.2. Ontologías relacionadas al trabajo

2.2.1. Ontologías nutricionales

Durante los últimos años, ha crecido el uso de las ontologías en el campo de las ciencias de la computación y la inteligencia artificial. Esto se debe a que proporcionan una forma de representar y compartir el conocimiento utilizando un vocabulario común, definiendo un formato de intercambio de conocimiento, proporcionando un protocolo específico de comunicación y permitiendo una reutilización más ágil del conocimiento. Existen diversos proyectos previos que trabajaron sobre el diseño de ontologías de alimentos y el control de la diabetes.

Hay una serie de sistemas de codificación existentes que se han ideado para clasificar alimentos y sus propiedades nutricionales, y varias bases de datos han sido desarrolladas con el mismo propósito. Sin embargo, existen muy pocos recursos ontológicos que describen los alimentos. Uno de ellos es Wine and Food Ontology [2]. Esta ontología, que fue diseñada para relacionar recetas de comidas con el vino más adecuado, no tiene en cuenta la información nutricional.

Por otro lado, la PIPS food Ontology [J+05] fue creada con el objetivo de representar un modelo abstracto de los diferentes tipos de alimentos disponibles para los usuarios de PIPS (Personalised Information Platform for Health and Life Services), junto con la información nutricional, incluyendo el tipo y la cantidad de nutrientes. PIPS es un proyecto de tipo e-health, que tiene como objetivo la creación de un servicio de multi-acceso para aumentar la calidad del cuidado de los enfermos, mediante la mejora de las comunicaciones entre los pacientes y los profesionales sanitarios [Dom+06]. Este sistema trabaja ante dos escenarios. El primero tiene su foco en la asistencia a pacientes crónicos, en su vida diaria ayudando a mejorar el cumplimiento de la terapia. El segundo se centra en el apoyo a los pacientes con sobrepeso asistiéndolos en su alimentación. En un escenario donde, por ejemplo, el paciente sufrió un infarto de miocardio unos años atrás y se encuentra actualmente en tratamiento por una miocardiopatía isquémica con insuficiencia cardíaca, el sistema le recuerda al paciente todas las acciones que debe realizar en forma oportuna. En este caso el sistema lo asiste recordándole el horario y la dosis de la medicación que tiene que tomar, rememorándole que debe realizar la medición de los signos vitales y que luego tiene que cargarlo en el sistema. Esta información es chequeada por profesionales en caso que no se encuentre en los parámetros normales. En otro escenario posible el paciente tiene sobrepeso e intolerancia a la lactosa. Además está bajo la supervisión de un nutricionista, siguiendo una dieta personalizada a sus gustos alimenticios, que se encuentra cargada en el sistema. El paciente puede consultar su dieta, y además tiene las recetas de las comidas. El sistema le informa los ingredientes que le hacen falta para cada menú, también puede consultar sobre si puede comer algún alimento particular ya que por su intolerancia a la lactosa no puede consumir cualquier alimento. Podemos observar la jerarquía de clases de la "PIPS Food Ontology" en la Fig. 2.7.

Food es el concepto raíz de esta ontología, todos los demás conceptos heredan las propiedades asociadas a tal concepto. Estas propiedades permiten describir un alimento en términos de sus nutrientes, y cuentan con 50 propiedades para describirlas. Asocian una propiedad de datos con cada nutriente, éstos tienen rango numérico y cardinalidad máxima que en la mayoría de los casos es 1, lo que significa que un nutriente puede estar presente en la descripción de alimentos. Además de los nutrientes, hay tres propiedades especiales, hasMaxAmount, hasMedAmount y hasMinAmount, que representan el máximo, medio y mínimo de la ingesta diaria recomendada por los nutricionistas.

En el estudio realizado en [LK07] se propone un mecanismo automático para contruir una ontología de comidas para utilizarla en servicios de educación para diabetes. Se basa en la idea de sustitución de alimentos. En general los pacientes aprenden a reemplazar

[2] http://www.w3.org/TR/2002/WD-owl-guide-20021104/food.owl

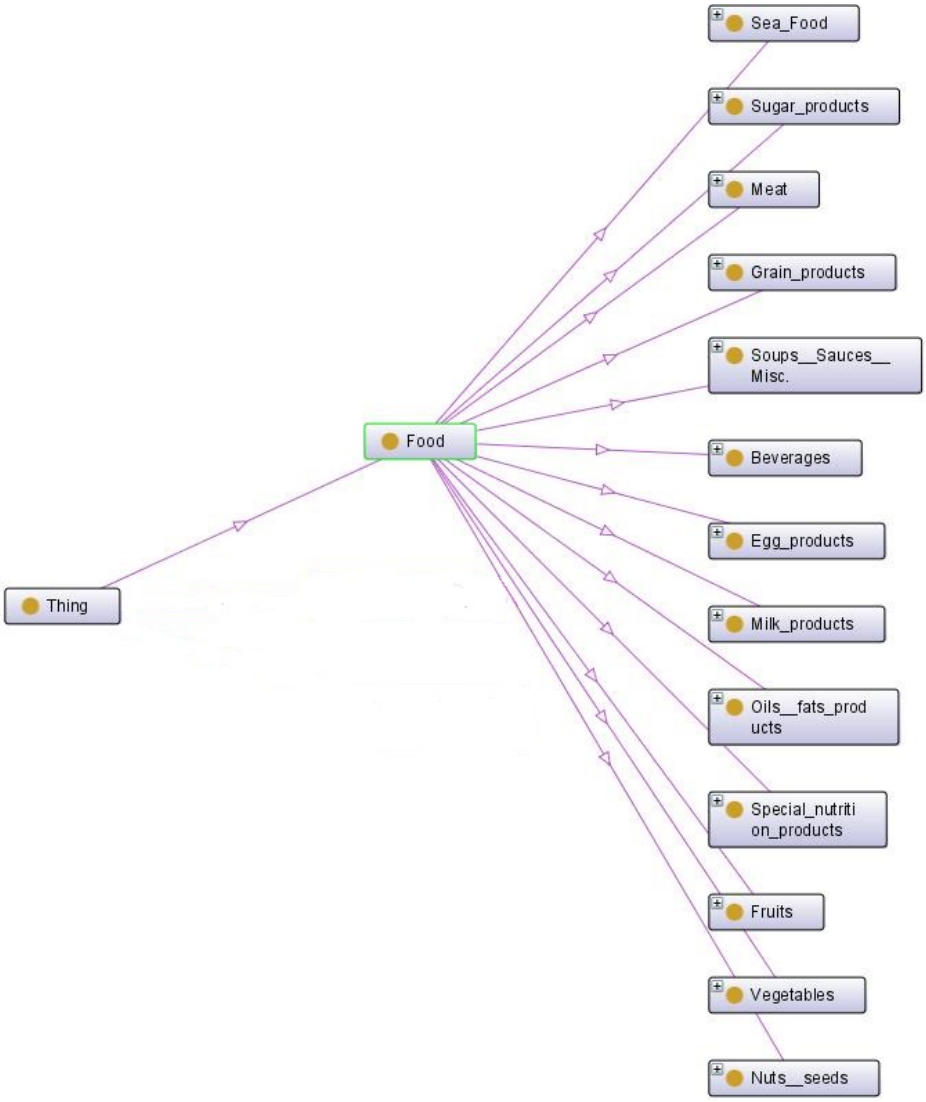

Figura 2.7: PIPS Food Ontology. Figura tomada de [J+05]

alimentos según a qué grupo de alimentos pertenecen, aunque esto no puede realizarse con exactitud porque además hay que tener en cuenta otros factores como las características nutricionales de cada alimento y su relación con los requerimientos nutricionales de cada paciente diabético. Por lo tanto, el sistema propone un mecanismo automático para ayudar a nutricionistas a crear nuevos grupos de alimentos y una clasificación jerárquica de remplazo y recomendación de alimentos. Para esto construyen dos ontologías, una ontología de alimentos y una ontología que facilita la sustitución de alimentos. Para llevar a cabo el desarrollo de la ontología de alimentos utilizan un conjunto de datos proporcionado por la base de datos de composición nutricional de los alimentos del Departamento

de Salud de taiwan(DOH- Deparment of Health, Executive Yuan, Taiwan). Los datos son agrupados en 18 categorías como granos, almidones, nueces y semillas, frutas y vegetales entre otros. En el cuadro 2.8 se puede ver un ejemplo de los datos con los que cuentan, para construir un esqueleto de ontología más significativa y útil. Además realizaron en-

Food-Group-Name	Food item code	Calory	Protein	Fat	Carbohydrate	dietary fiber
A-Grains	[A001200] Barley	367	9.3	3	74.2	15.3
	[A04740] Rice	183	3.1	0.3	41	0.6
B-Starch	[B001400]Sweet Potato	124	1	0.3	28.6	2.4
	[B006100]Potato	81	2.7	0.3	16.5	1.5
C-Nuts and seeds	[C002400]sesame seed	591	189	53.3	19.7	9.2
D-Fruits	[D014000]red apple	53	0.3	0.3	13.7	1.6
	[D014501]apple juice	42	0.1	0.4	10.9	0

Cuadro 2.8: Ejemplo de la base de datos de alimentos

cuestas a 28 educadores de diabetes calificados de Taiwan. En esas encuestas se les pidió a los expertos clasificar y darle relevancia a 24 nutrientes en términos de sus importancias para el cuidado de la dieta pacientes con diabetes. Se valorizó con un entero entre 0 a 100 para cada nutriente, siendo 0 el menos significativo y 100 el más significativo.

En [Lee+08] se propone un agente de recomendación de comida personal, basado en el modelo de la ontología de recomendación de alimentos para diabéticos. El agente puede crear un plan de alimentación de acuerdo a un estilo de vida personal y necesidades particulares de salud. El conocimiento requerido se almacena en el modelo de la ontología predefinido por expertos en el dominio. Hay dos tipos de ontologías aplicados a este trabajo: la ontología de comida taiwanesa y la ontología personal de alimentos. La pirámide nutricional de los alimentos divide los mismos en seis grupos principales, incluyendo granos y almidones, verduras, frutas, lácteos, carnes y proteínas, y grasas. En la fig. 2.9 se muestra un ejemplo ilustrativo de la ontología de comida taiwanesa. Cada concepto en la capa de instancia contiene un nombre con varios atributos, tales como tamaño de la porción, información nutricional por 100 gramos (calorías, proteínas, grasas, hidratos de carbono), y la porción correspondiente teniendo en cuanta la pirámide nutricional. Para la ontología personal de alimentos se tuvieron en cuenta los hábitos alimentarios (que varían según el individuo y pueden verse influenciados por la nacionalidad de origen), las preferencias personales, la condición social, la posición económica y la religión. Para la construcción de la ontología se tuvo en cuenta el proceso que realizan los expertos del dominio, como médicos nutricionistas, a la hora de planificar una dieta saludable. Estos en primer lugar utilizan la ecuación Harris-Benedict (ver cuadro 2.10) para averiguar los requerimientos calóricos diarios del paciente en función del peso, el sexo, la altura y la edad. Una vez que tienen esa información la multiplican por un factor entre 1.2 y 1.5 dependiendo de la actividad física que realice el paciente. Finalmente tienen en cuenta la proporción de hidratos de carbono, proteínas y grasas que deben consumir en el día. En la fig. 2.11 se observa un ejemplo ilustrativo de la ontología personal de alimentos.

Los hábitos alimenticios están íntimamente relacionados con la identidad cultural y son influenciados por la formación cultural y social de cada persona. Por esta razón, la clasificación de alimentos que utilizan los nutricionistas argentinos se rige por factores diferentes a aquellos considerados durante la construcción de las ontologías de alimentos

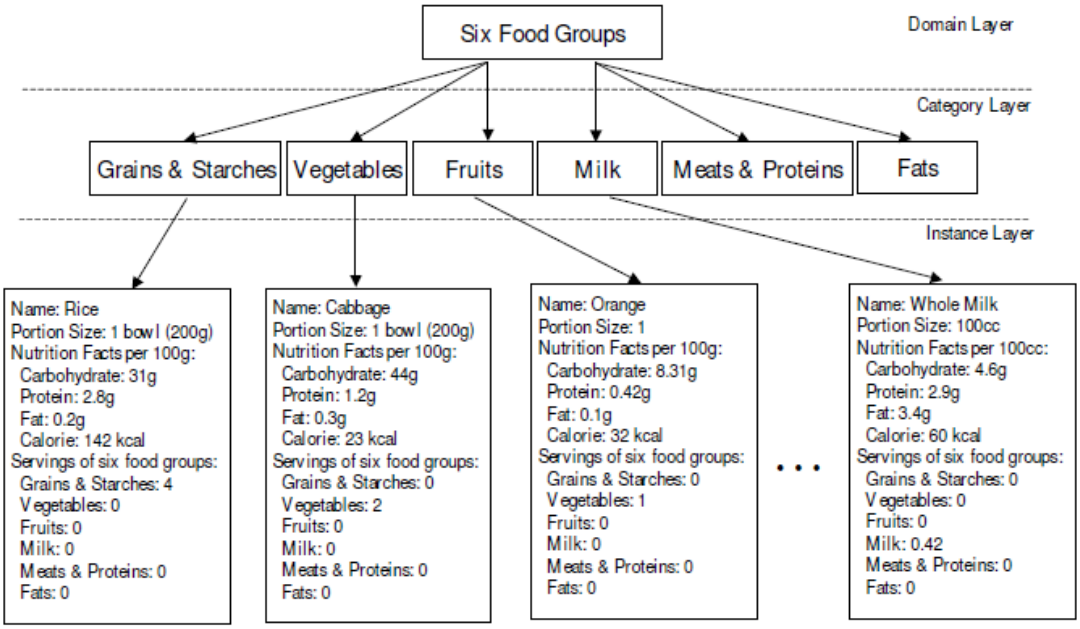

Figura 2.9: Ontologia de comida Taiwanesa. Figura tomada de [Lee+08].

Harris-Benedict ecuation

Sex	Basal Metabolic Rate(BMR)
Male	66 +(13.7 x Weight) + (5 x Height) - (6.8 x Age)
Female	655+ (9.6 x Weight) + (1.8 x Height) - (4.7 x Age)

Cuadro 2.10: Ecuación de Harris-Benedict

desarrolladas en los proyectos recién presentados, como el realizado en la unión europea y la ontología de comida taiwanesa. Por consiguiente, nos propusimos desarrollar una nueva ontología de alimentos, basándonos en las aquí descriptas, que intentará incluir, al menos, aquellos alimentos y productos que son de mayor consumo en nuestro país. Por otro lado, en estos artículos el desarrollo de la ontología está enfocado en la construcción de regímenes alimentarios, y si bien a nosotros nos interesa la calidad nutricional de la alimentación del paciente, no tendremos en cuenta los planes alimentarios. Para nuestra ontología nos enfocaremos en el análisis de las ingestas del paciente, teniendo en cuenta horario de las mismas y la cantidad de hidratos de carbono consumidos. Con el principal objetivo de poder integrar nuestra ontología con una ontología que modela el sistema endocrino de las personas, la cual necesita saber el conteo de carbohidratos de cada comida junto con el horario de la misma. Además, en nuestra ontología modelaremos algunas reglas obtenidas de las recomendaciones de la OMS [Eve+14b] y adaptadas previamente por los nutricionistas , que permitan orientar al paciente hacia una adecuada composición de la ingesta.

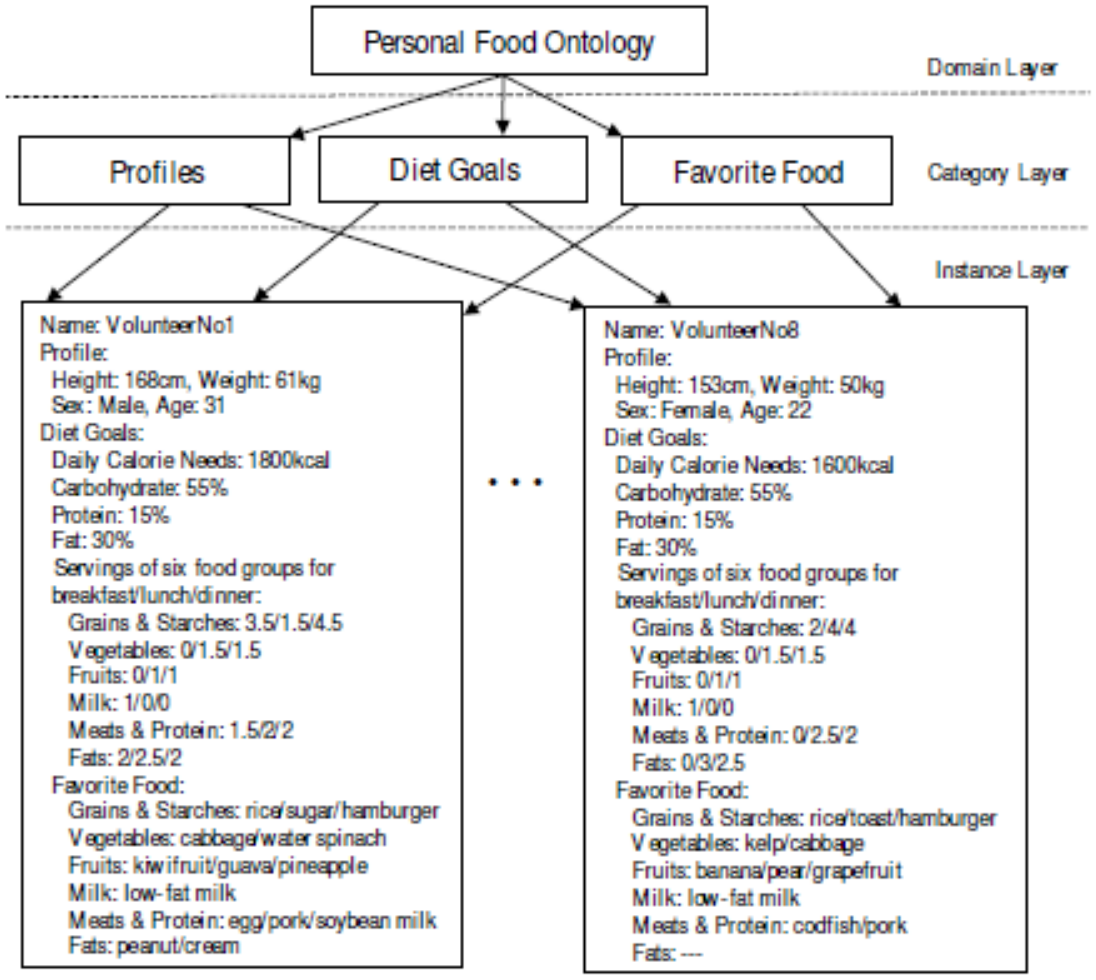

Figura 2.11: Ontología personal de alimentos. Figura tomada de [Lee+08].

2.2.2. El modelo simulador UVA/Padova y su representación ontológica

Este trabajo forma parte de un proyecto aún más grande que centra su atención en el tratamiento de pacientes diabéticos con Páncreas Artificial. En este marco, el objetivo del trabajo en [M15] es el de crear una ontología que permita modelar tanto al paciente diabético como el control necesario para mantener estables sus niveles de glucosa en sangre. Para esto se presenta una extensión de la ontología denominada OntoCAPE para capturar el conocimiento del modelado de tipo compartimental[3] del sistema endocrino de pacientes diabéticos. Entonces por un lado se realiza la implementación ontológica

[3]Se refiere a que está construido de un número finito de compartimentos con interconexiones específicas entre ellos. En términos matemáticos cada compartimento involucra a una ecuación diferencial, la cual representa un balance de masa por componente.

del Modelo Matemático del Sistema Endocrino Humano aprobado por la FDA (Food and Drugs Administration), que es el Simulador UVA/Padova y por otro lado en la representación, se asocia el Simulador a lazos de control, con los siguientes controladores: PID (Control Proporcional Integrativo Derivativo) y PFC (Control Predictivo Funcional), también en términos de ontologías.

Simulador UVA/Padova Equipos de trabajo de la Universidad de Virginia (en Estados Unidos) y de la Universidad de Padova (en Italia) han desarrollado un modelo/simulador matemático del Sistema Endocrino humano denominado UVA/Padova. La representación gráfica de este modelo puede observarse en la fig. 2.13. En dicho modelo se describe la relación entre la glucosa en plasma y la concentración de insulina, y los flujos que intervienen en el sistema endocrino, a saber, flujos de glucosa e insulina. Desde su desarrollo inicial fue evolucionando en nuevas versiones, cada versión busca mejorar sus cualidades de captura del comportamiento del sistema endocrino humano. La documentación referente a las distintas versiones de este modelo se encuentra disponible en [CRC07; B+08; B+10]. El modelo UVA/Padova puede ser utilizado para simular el comportamiento del sistema endocrino de humanos saludables, prediabéticos, diabéticos Tipo 2 y diabéticos Tipo 1, y ha sido validado con datos clínicos experimentales. La Administración de Alimentos y Drogas de Estados Unidos (conocida por sus siglas en inglés FDA – Food and Drugs Administration) ha aprobado la versión del paciente diabético tipo 1 como sustituto de las pruebas en animales para testeos pre-clínicos de algoritmos de control. El modelo matemático está compuesto por diferentes compartimentos que representan los Sub-Sistemas de Ingesta, Glucosa, Insulina y Espacio Subcutáneo del Sistema Endocrino Humando. Cada uno de estos compartimentos consta de un sistema de ecuaciones diferenciales que describen la dinámica del Sub-Sistema que representan. El Simulador UVA/Padova se ha implementado usando la plataforma MatLab[4]/Simulink[5].

OntoCAPE Se trata de una ontología definida dentro del dominio de la ingeniería de procesos asistida por computadora. Su nombre hace referencia a su utilidad: Onto, expresa que se trata de una ontología y CAPE es la sigla en inglés de Computer-Aided Process Engineering[6]. Esta ontología ha sido ideada por investigadores en la RWTH Aachen University en Alemania [W+10]. Es una ontología de gran dimensión, la misma se encuentra particionada en 62 sub-ontologías. Estas sub-ontologías pueden ser usadas por separado o bien como un producto completo. Las mismas se organizan en diferentes capas de abstracción, las capas más altas representan un conocimiento más general mientras que las capas inferiores describen aplicaciones en particular. OntoCAPE es una ontología desarrollada para la industria química. Como el Sistema Endocrino es un proceso químico, OntoCAPE (luego de efectuadas algunas extensiones) puede ser utilizada para su modelado.

[4] http://www.mathworks.com/products/matlab/
[5] http://www.mathworks.com/products/simulink/
[6] Ingeniería de Procesos Asistida por Computadora

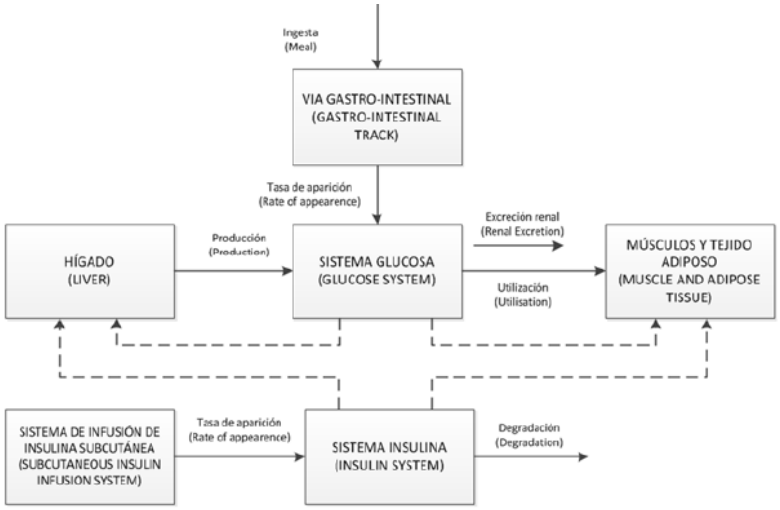

Figura 2.12: Esquema del sistema glucosa-insulina del Simulador UVA/Padova. Figura tomada de [M15].

Representación del Sistema Endocrino Humano en OntoCAPE El trabajo presentado en [M15] describe una representación ontológica del Sistema Endocrino Humano. El Sistema Endocrino Humano es un Sistema de Procesos Químicos Biológico, lo cual es considerado un caso particular de un Sistema de Procesos Químicos. Esta es la razón por la cual se ha elegido OntoCAPE como ontología base, para reutilizar sus definiciones y reglas y así modelarlo. Como dijimos antes, OntoCAPE está compuesta de sub-ontologías. La sub-ontología que se usa en [M15] para representar el sistema endocrino es la llamada **chemical_process_system**. Esta sub-ontología presenta las herramientas necesarias para realizar el modelado mereotopológico completo del Sistema Endocrino Humano, en los términos usados por el simulador UVA/Padova. Es decir, permite definir las clases, instancias y propiedades capaces de describir el modelo compartimental del simular UVA/Padova y, a su vez, asociar cada compartimento semánticamente con su respectivo sistema de ecuaciones diferenciales no-lineales. El conocimiento representado en la ontología puede ser descripto en distintos grados de detalle, al nivel que se desee. La representación propuesta en [M15] no solo modela la concepción mereotopológica sino que también se detallan flujos de masas, de señales y el concepto de sub-sistemas y super-sistemas.

La fig. 2.13 grafica un diagrama en bloques simplificado del modelo en MatLab/Simulink, donde puede observarse la re-alimentación entre la salida (glucosa) y la entrada (sensor de glucosa en sangre). El controlador analiza la diferencia entre la salida observada y la entrada deseada. Teniendo en cuenta esta diferencia genera una acción sobre la variable controlada, llamada pump (que representa la bomba de insulina), para corregir el estado de la glucosa en sangre, según los valores deseados. El bloque denomidado From File representa un anuncio de la ingesta de carbohidratos por parte del paciente que se informa al controlador. Dicho bloque es muy importante para los algoritmos de

control, ya sean controles PID como PFC, ya que teniendo conocimiento previo de la perturbación generan resultados más satisfactorios. La perturbación en este caso es la ingesta de carbohidratos. Otro de los bloques, llamado diabético, cuenta con un generador de comidas, meal generator, que representa la ingesta real del paciente. La ingesta está compuesta por tres ingestas dentro de un período de 30 horas (1800 minutos), para todos los pacientes de la base de datos. Para finalizar podemos presentar el bloque Subsystem1 donde se encuentra el bloque principal del modelo, el cual simula al sistema endocrino del paciente. Cabe destacar que en [M15] no se realizó modelado alguno respecto de la ingesta real del paciente o el anuncio de dicha ingesta. En la ontología se crearon los individuos correspondientes a ambos, pero tanto la ingesta real como el anuncio de la misma se realizan dentro de MatLab, en scripts y elementos de MatLab realizados a tal fin. En nuestro trabajo nos proponemos sintetizar una ontología que permita el cálculo de una dosis oral de glucosa ingerida, mediante la especificación de los alimentos comerciales que el paciente incorpora en su organismo. Con el objetivo final de incorporar nuestra ontología a la ontología recién descripta (que modela el sistema endocrino del pacientes diabéticos) a los efectos de capturar ese conocimiento faltante (ingesta real y anuncio de la ingesta), y adaptar el programa para que este mismo vincule esos modelos con la herramienta cálculo y simulación.

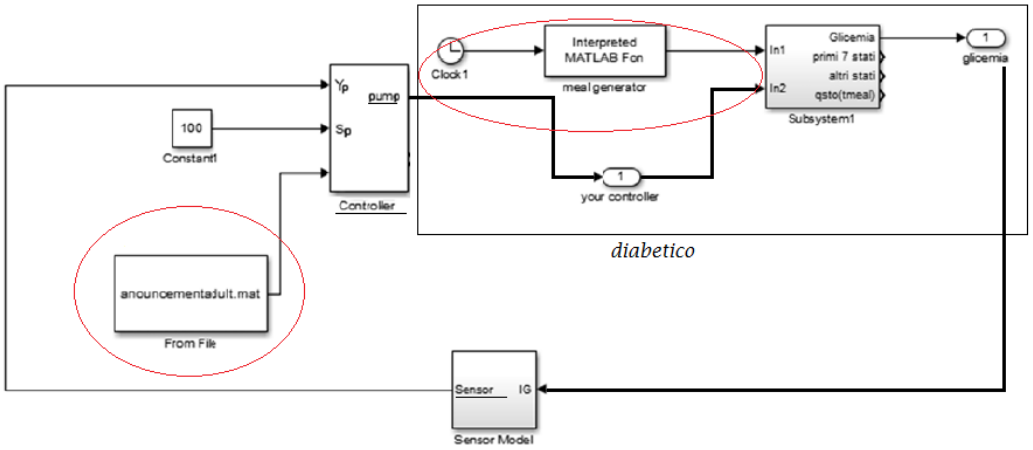

Figura 2.13: Modelo en MatLab/Simulink simplificado.

Capítulo 3

Creación de ontologías para la representación de Alimentos e Ingestas

En este capítulo, que constituye el nodo central de la tesis, explicaremos la construcción de una ontología que permita el cálculo de una dosis oral de carbohidratos ingeridos, mediante la especificación de alimentos comerciales que el paciente incorpora en su organismo. Para tal fin primero construiremos una ontología de Alimentos, y luego una ontología de la Ingesta. Finalmente ambas ontologías se integrarán a la ontología que modela el sistema endocrino de pacientes diabéticos, descripta en el capitulo anterior [M15]. Para la especificación formal de las ontologías, se ha elegido el lenguaje OWL (Ontology Web Languaje). El modelado fue logrado por medio del editor de ontologías Protege. Protege [Pro] es un editor de ontologías y bases de conocimiento gratuito y abierto, esta basado en Java y soporta Frames, XML Schema, RDF y OWL. Protege, al igual que otros sistemas basados en marcos, describen ontologías declarativas, estableciendo explícitamente cuál es la jerarquía de clases y a qué clases pertenecen los individuos.

3.1. Ontología de Alimentos

Nutrinfo es la comunidad virtual de profesionales de la nutrición más importante de habla hispana, con más de 32000 profesionales de la nutrición, que desde hace más de 14 años integra en su red el esfuerzo y experiencia de miles de profesionales, sociedades científicas, universidades y la industria alimenticia y farmacéutica. Han desarrollado lo que llamamos Vademecum de Alimentos, que es una tabla de composición química de los alimentos, que contienen la información nutricional de alimentos con foco en nuevos lanzamientos de la industria de alimentos en Argentina. En la fig. 3.1 se puede observar lo que sería un elemento perteneciente al Vademecum de alimentos. La información que se tiene que abstraer en la ontología debe incluir, la cantidad de hidratos de carbono, fibras, proteínas, grasas totales, sodio y calorías por porción. En términos prácticos la creación de la ontología incluyó:

Figura 3.1: Tabla de composición química de los alimentos

- Definición de clases en la ontología: la clase principal de nuestra ontología, es la clase Alimentos, que contiene los principales alimentos consumidos en Argentina. Además se crearon clases que agrupan diferentes alimentos según su composición o los determinados nutrientes que aportan al organismo como Vegetales, Frutas, Quesos, Carnes, Pescados y Mariscos, entre otros.

- Organización de las clases en una jerarquía taxonómica: durante la creación de la ontología de alimentos organizamos las clases en una taxonomia jeráquica siguiendo la misma utilizada en *Nutrinfo*. La clase **Alimentos** es la raíz de nuestra ontología y es la que representa el concepto fundamental en nuestro dominio de alimentos, en la misma se agrupan todos lo alimentos comerciales consumidos en Argentina. La clase Alimentos tiene 18 sub-clases, y algunas de estas a su vez tiene otras sub-clases como vemos en la fig. 3.2.

- Definición de las propiedades y descripción de los valores permitidos para las mismas: las relaciones que se definieron conforman la información nutricional relevante a un alimento en particular. Las mismas fueron modeladas con las siguientes *dataProperties*: **tieneSodio**, **tieneHidratos**, **tieneProteinas**, **tieneCalorias**, **tieneGrasasTotales**, **tieneFibras**,**tienePorcionGramos** (ver fig. 3.3). El dominio de las mismas es la clase *Alimentos*, mientras que el rango son los números Reales positivos (Float) expresados en gramos, miligramos y calorías según corresponde. Todas las propiedades recién descriptas son relaciones funcionales, ya que para un alimento sólo puede haber un valor para esas propiedades. Como se puede observar en la fig. 3.1 la información que se presenta corresponde para la porción de un alimento medido en gramos. Igualmente además de expresar a cuanto equivale la

porción en gramos, muestra su equivalencia en unidades, representado en nuestro caso por la *date property* **tienePorcionUnidades** cuyo dominio es la clase *Alimentos* y cuyo rango son los String. Esto es porque para los pacientes es más efectivo medir las porciones como un plato, una cucharada, una unidad, media taza, en lugar de las porciones expresadas en gramos. En este caso 100 gramos de ravioles son equivalentes a un plato, tal como muestra la figura.

Por lo tanto, la ontología de Alimentos se compone de todos aquellos elementos encontrados en la base de datos NutrInfo, ordenadas según la taxonomía presentada en la Fig. 3.2. A su vez, cada alimento está asociado a ciertas propiedades, ver ejemplo en Fig. 3.3, que brindan información nutricional muy importante.

3.2. Ontología que representa la Ingesta de un paciente diabético

Los hidratos de carbono son los únicos nutrientes que pueden elevar la glucemia. Conociendo la cantidad de carbohidratos consumidos se puede tener una aproximación de cómo se verá alterada la misma. Por esto necesitamos conocer en cada ingesta la cantidad de carbohidratos consumidos. Este dato representa la perturbación en los algoritmos de control de glucosa que se utilizan para mantener los niveles de glucosa dentro de los parámetros deseados. Volveremos sobre esto en la sección 4.4, cuando hablemos de la validación de esta tesina. Estos controles necesitan conocer con exactitud la cantidad de gramos de carbohidratos consumidos y el horario en que se realiza la ingesta. Para eso necesitamos saber los alimentos que consume el paciente, con la porción que este ingiriere en cada comida diaria junto con el horario que realiza dicha ingesta. A su vez, contar con un registro de la cantidad de carbohidratos ingeridos por un paciente y la hora de la ingesta es siempre de gran utilidad, sobre todo cuando se realizan estudios de Hemoglobina glucosilada o cuando se realizan monitoreos continuos de glucosa (Holter Glucémico). El primero es un examen que aporta una visión retrospectiva del control de la diabetes. Los glóbulos rojos que circulan por la sangre contienen una proteína llamada hemoglobina. La glucosa, que también circula por la sangre, se adhiere a la hemoglobina durante un periodo de entre 90 y 120 días. De esta manera, la prueba de la hemoglobina glucosilada se basa en la medición de la cantidad de glucosa adherida a los glóbulos rojos y su resultado se expresa en porcentaje, que determina el nivel medio de glucemia durante el trimestre anterior a la prueba. El segundo examen, (Holter glucémico) ofrece una imagen completa permitiendo conocer el perfil glucémico del paciente en su vida real y cotidiana. Al paciente se le proporciona un dispositivo para la monitorización continua de glucosa (MCG) que realiza hasta 288 mediciones que no se realiza directamente en sangre, sino que surge de los niveles de glucosa obtenidos en el líquido intersticial. Este líquido se encuentra localizado entre las células y su contenido de glucosa es similar al del plasma sanguíneo.

Para poder tener un visión completa del paciente al realizar estos estudios, pensamos que sería muy valioso contar con un registro de las diferentes ingestas y así poder analizar

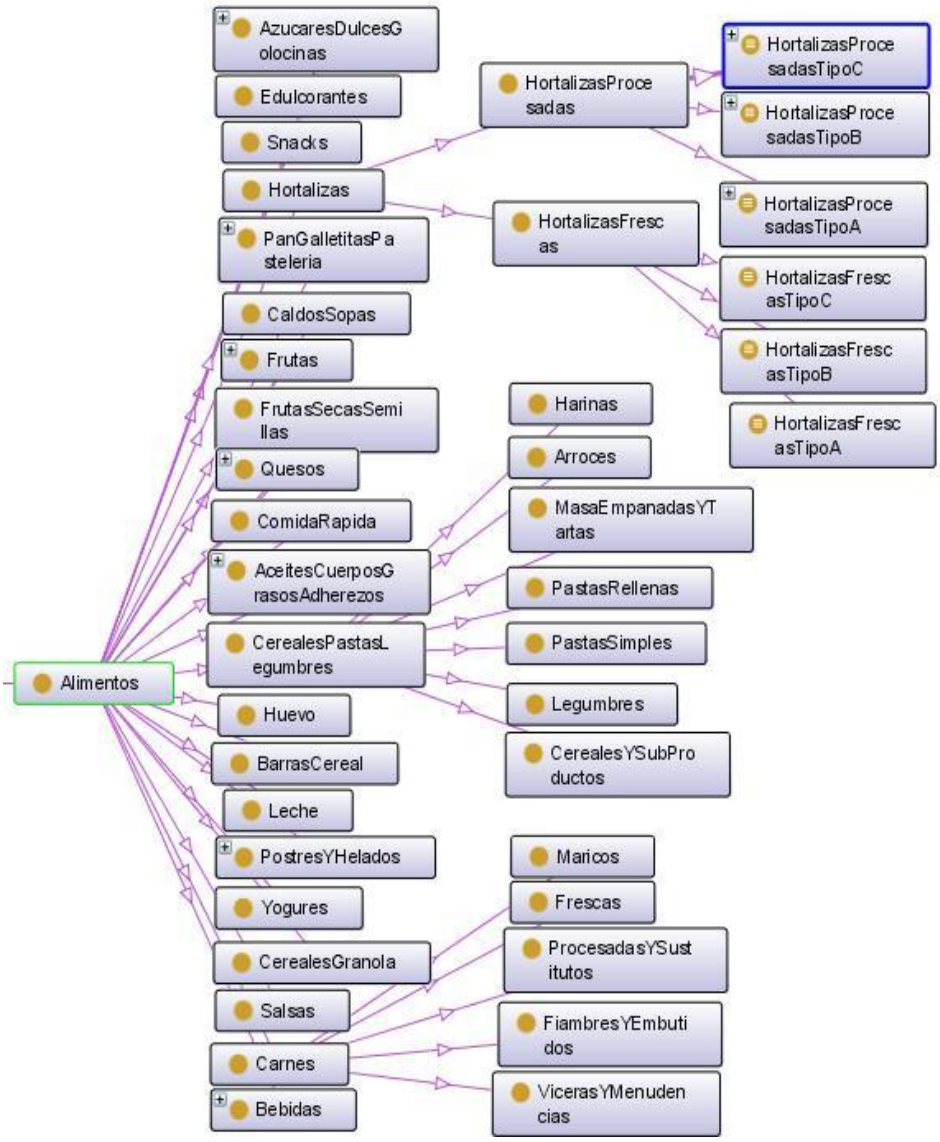

Figura 3.2: Taxonomia de la ontología alimentos

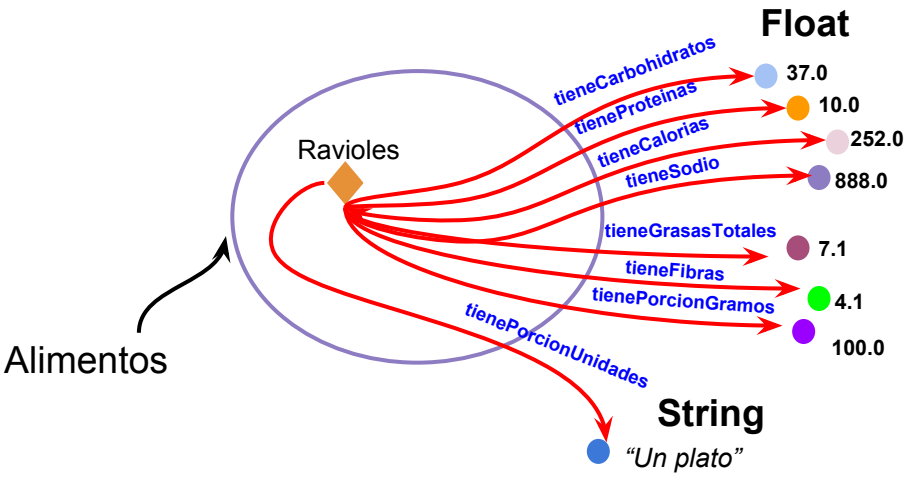

Figura 3.3: Propiedades que definen los Alimentos.

los resultados del estudio en relación a la ingesta de alimentos. Para llevar a cabo estos requerimientos necesitamos saber los alimentos que consume el paciente, con la porción de los mismos que éste ingiere en cada comida diaria, junto con el horario que realiza dicha ingesta. En base a esto, la ontología que diseñemos tendrá que ser capaz de responder, entre otras, las siguientes preguntas:

- **¿Qué alimentos consumió el paciente p1 en el desayuno?**
- **¿Qué alimentos consumió el paciente p1 en el día de hoy?**
- **¿Cuántas porciones del alimento a1 consumió el paciente p1?**
- **¿Cuántos carbohidratos consumió el paciente p1 en el almuerzo?¿Cuántos durante todo el día?**
- **¿Qué consumió el paciente en la última semana?**
- **¿En qué horario realizó la ingesta?**

Para poder responder dichas preguntas primero comencemos analizando dos aspectos:

1. La técnica que realiza el paciente denominada *conteo de carbohidratos*.
2. El Anuncio de la Ingesta, que se describe en el Modelo del sistema endócrino humano.

Con el propósito de que la ontología final pueda reflejar dichos conceptos.

Consideraciones sobre el Conteo de Carbohidratos Una persona durante el día realiza diferentes ingestas, en el desayuno, en el almuerzo, en la merienda, en la cena y en alguna colación. Sea cual fuera la terapia que sigue el paciente diabético, necesita conocer la cantidad de carbohidratos que consume para poder conocer cómo puede alterarse la glucemia. Generalmente a los paciente se los entrena para realizar lo que comúnmente se llama **conteo de carbohidratos**. Les enseñan cuáles son los alimentos que aportan este nutriente, cuáles no tienen que contabilizar, cómo leer las etiquetas de los alimentos que contienen la información nutricional y finalmente les hacen hacer una clasificación de alimentos en porciones de 15 gramos de carbohidratos para facilitar los cálculos. Dependiendo de esa cantidad de carbohidratos calculan la cantidad de insulina que deben inyectarse. La relación insulina/carbohidratos es diferente para cada persona y puede variar entre 5 gramos/ 1 unidad a 30 gramos/ 1 unidad dependiendo si son niños, adolescentes, adultos, embarazadas, persona con sobrepeso u obesidad. Los pacientes que utilizan bombas de infusión continua (medtronic es la marca comercial más conocida) no necesitan calcular la dosis de insulina, ya que estos dispositivos pueden calcular el tamaño del bolo[1] adecuado. Aunque si necesitan calcular los carbohidratos, dado que para determinar el tamaño del bolo se basan en la cantidad de carbohidratos ingeridos, la insulina remanente en el paciente y la tasa de glucemia antes de la ingesta.

Abstracción de la ingesta en el modelo endocrino del paciente diabético Hemos mencionado que las ontologías a desarrollar serán integradas con la ontología que describe el modelo endocrino del paciente. Dicha ontología describe términos importantes que tendremos que tener en cuenta y poder adaptarlos a nuestro modelo, entre ellos el anuncio de la ingesta. Como se observa en la fig. 2.13 el controlador implementado sobre el modelo endocrino del paciente diabético necesita como entradas conocer los anuncios de cada ingesta (desayuno, almuerzo, merienda, colación y cena) con la hora estipulada para la ingesta para poder realizar un mejor control y calcular con mayor exactitud la cantidad de insulina que debe colocarse el paciente, para mantenerse dentro de los rangos saludables.

3.2.1. Primera Versión. Construyendo la ontología.

El proceso de desarrollo de una ontología es un proceso iterativo compuesto de aproximaciones sucesivas hasta llegar a una versión final de la misma. En base a las consideraciones hasta recién expuestas mostraremos la ontología inicial, y exhibiremos su evolución a través de las sucesivas iteraciones hasta lograr una ontología que cumpla con las aspectos planteados con anterioridad.

En primer lugar presentaremos las clases principales de la primer versión de la ontología que hemos definido:

[1] El bolus o bolo consiste en la administración intravenosa de un medicamento a una velocidad rápida, pero controlada.

- **Paciente.** Es la clase que representa todos los pacientes. Está descripta mediantes los atributos que identifican al paciente como nombre y apellido, edad, peso, medidas(altura, contextura ósea, circunferencia de la cintura/cadera).

- **Ingesta.** Es la clase que agrupa las ingestas por día de los paciente. Cada individuo perteneciente a esta clase tiene una fecha y relaciones con individuos de la clase *ComidaDiaria*.

- **ComidasDiarias.** Es una superclase de las clases DESAYUNO, ALMUERZO, MERIENDA, COLACIÓN y CENA. En cada una de las subclase se agrupan las diferentes ingestas que se hicieron en un momento del día determinado. Cada individuo perteneciente a cada sub-clase se relaciona con los individuos de la clase *Alimentos*. La clase **Alimentos.** es la clase que agrupa los diferentes alimentos que puede ingerir el paciente. Esta clase pertenece a la ontología de alimentos que definimos anteriormente e importamos en esta ontología para poder utilizarla.

Una primera aproximación de nuestra ontología se puede observar en la fig. 3.4. Hemos incluido en esta figura y en las subsiguientes que describen a la ontología, la clase Alimentos pese a pertenecer a la ontología de Alimentos. Lo hicimos con el fin de tener una visión global.

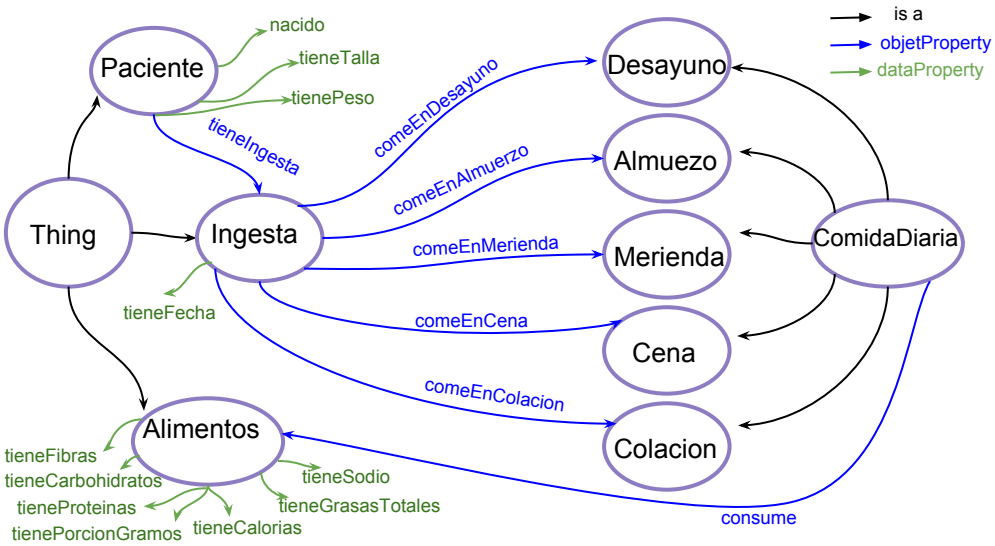

Figura 3.4: Ontología, 1era versión.

La primera aproximación de la ontología cuenta con **nacido**, **tieneTalla**, y **peso** como *dataProperty* de la clase *Paciente*. La relacion **tieneIngesta** tiene como *dominio* a *Paciente* y *rango* a *Ingesta*. La clase *Ingesta* a su vez tiene una *dataProperty* **tieneFecha** y 5 *objetProperties*, **comeEnColacion**, **comeEnDesayuno**, **comeEnAlmuerzo**, **comeEnMerienda**, **comeEnCena**, que tienen de *dominio* a *Ingesta* y *rango* a *Comi-*

daDiaria. Cada una de las comidas diarias tiene una relación **consume** con *dominio* a *ComidaDiaria* y *rango* a *Alimento*.

Cada paciente tiene diferentes ingestas, una por día que son elementos de la clase Ingesta, por ejemplo p_1 tieneIngesta $i1$. A su vez cada una de estas ingestas tiene diferentes ingestas diarias que son elementos de la clase ComidaDiaria por ejemplo i_1 $comeEnDesayuno$ $alimento_1$. Finalmente en cada una de estas ingestas diarias se consumen diferentes alimentos: d_1 consume $alimento_1$, d_1 consume $alimento_2$, etc. Cada uno de estos alimentos tiene una cantidad de hidratos de carbono(hcA_i), y la suma de todos los hidratos de carbono pertenecientes a todos los alimentos consumidos en cada comida diaria($hcDesayuno$, $hcAlmuerzo$, $hcMerienda$, $hcColacion$, $hcCena$) corresponde a la cantidad total consumida por día($hcIngesta$). Expresando formalmente de la siguiente manera:

$$hcDesayuno = hcA_1 + hcA_2 + ... + hcA_n$$

$$hcAlmuerzo = hcA_1 + hcA_2 + ... + hcA_n$$

$$hcMerienda = hcA_1 + hcA_2 + ... + hcA_n$$

$$hcColacion = hcA_1 + hcA_2 + ... + hcA_n$$

$$hcCena = hcA1 + hcA2 + ... + hcAn$$

$$hcIngesta = hcDesayuno + hcAlmuerzo + hcMerienda + hcColacion + hcCena$$

3.2.2. Segunda Versión. Resolviendo limitaciones de la ontología.

La versión preliminar, tienen una importante limitación, que tendremos que resolver, relacionada con las porciones de los alimentos que ingiere el paciente. Tratemos de responder la siguiente pregunta:

¿Cómo representar que el paciente consumió en el desayuno 2 porciones de un alimento a_1 y una porción de un alimento a_2?

En la ontología que se expone en la fig. 3.4, no esta representada la porción de cada alimento que consume el paciente. Como hemos mencionado con anterioridad una ingesta se compone de uno o varios alimentos con sus respectivas porciones, ya que la información que tiene cada alimento sobre la cantidad de nutrientes que posee esta dada por una porción del alimento. Por lo tanto en una ingesta de una comida diaria, por ejemplo un individuo de la clase desayuno, no solo deberá tener una relación con el alimento que consumió, sino que también tiene que tener una relación con la porción del mismo.

Como hemos vistos en el capítulo anterior en los lenguajes de la Web Semántica como Resource Description Framework (RDF) y el Lenguaje de Ontologías Web (OWL), las propiedades son relaciones binarias que se utilizan para unir dos individuos o un individuo y un valor. Para poder lograr que un individuo de la clase **ComidaDiaria** se una mediante dos propiedades (tieneAlimento y tienePorcion) a la clase Alimentos, vamos a realizar un proceso llamado reificación sobre la propiedad **consume**. Para esto creamos una nueva clase que servirá como nexo llamada **agrupaAlimentoPorción**. La

clase **ComidasDiarias** y la clase recién creada se unirán a través de la relación binaria **tieneAgrupaComidaPorcion**. Los individuos de la clase **agrupaAlimentoPorción** se relacionarán por un lado con los individuos de la clase **Alimentos** mediante la propiedad **tieneComida** y por otro lado se relacionan con valores enteros (que representa las porciones) mediante la propiedad **tieneComidaPorcion** (ver fig. 3.5).

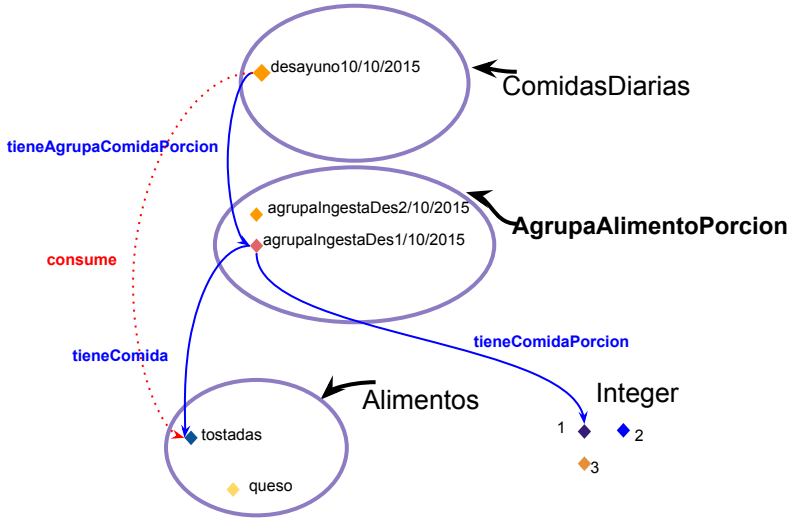

Figura 3.5: Reificación de la propiedad consume.

Otro de los cambios en la ontología fue agregar las diferentes propiedades nutricionales a cada paso de la ingesta del paciente, para poder saber la cantidad total de un nutriente determinado en una ingesta accediendo a la propiedad. Con lo cual por ejemplo, el valor de fibras que el paciente consumió en una día determinado, o la cantidad de hidratos de carbono que consumió en el desayuno, o la cantidad de Sodio que consumió de un alimento dadas las porciones, va a estar determinado por las propiedades *ingestaTieneFibras*, *comidaDiariaTieneCarbohidratos* y *tieneProteinasAgrupaAlimentoPorcion* respectivamente. Con el fin de poder tener la información de cual fue el horario de cada una de las ingestas realizadas se agrega la propiedad *ComidaDiariaTieneHora* en la clase *ComidaDiaria*.

Finalmente, veamos la correspondencia entre nuestra ingesta y el concepto de Anuncio de la Ingesta explicado en el principio de este capítulo. Cuando se realiza el anuncio de la ingesta la información importante a suministrarle al algoritmo de control es la hora y cantidad de carbohidratos, representados en nuestra ontología por las propiedades *ComidaDiariaTieneHora* y *comidaDiariaTieneCarbohidratos*. El anuncio de la ingesta estaría abstraído en la clase ComidaDiaria que es la que guarda esa información. En la fig. 3.6 se puede observar la ontología resultante, una vez solucionadas las limitaciones antes planteadas. Cabe destacar que por mayor simplicidad en el gráfico no hemos agregado todas las *dataProperties* referidas a determinar la cantidad de nutrientes.

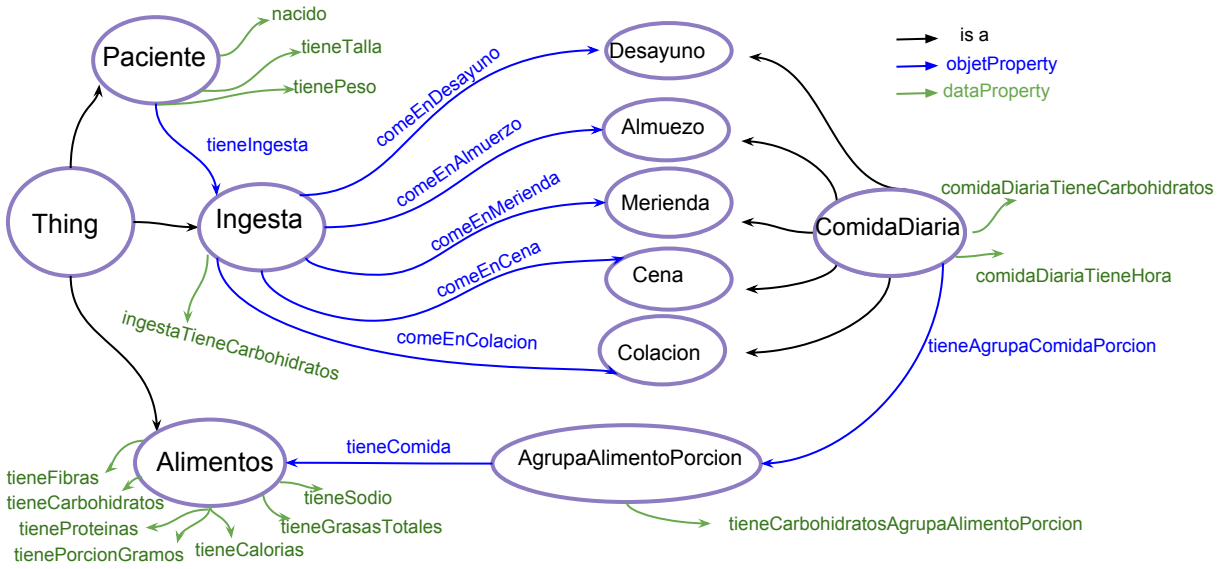

Figura 3.6: Ontología, 2da versión

3.2.3. Tercer Versión. Agregando recomendaciones nutricionales.

Como mencionamos en la introducción, nuestro objetivo no es solo asistir al paciente diabético en el conteo de carbohidratos sino además poder generar una educación nutricional del paciente brindándole recomendaciones alimenticias. Para la realización de las recomendaciones nutricionales nos basamos en las recomendaciones dadas por la ADA (American Diabetes Association) [Eve+14a] y las consultas con nutricionistas. Dividimos las recomendaciones en tres tipos:

1. Recomendaciones para la ingesta total del día.
2. Recomendaciones para la ingesta de alimentos particulares.
3. Recomendaciones para ingestas por comidas diarias como cena y almuerzo.

Esta información es abstraída en la ontología mediante una clase **RecomendacionesAlimenticias** con tres subclases **RecomendacionesIngestasDiarias**, **RecomendacionesComidasDiarias** y **RecomendacionesAlimentoPorcion** y tres propiedades: **tieneRecomendacionIngesta** con dominio *Ingestas* y rango *RecomendacionesIngestasDiarias*, **tieneRecomendacionComidaDiaria** con dominio *ComidaDiaria* y rango *RecomendacionesComidaDiaria* y **tieneRecomendacionAgrupaAlimentoPorcion** con dominio *AgrupaAlimentoPorcion* y rango *RecomendacionesAgrupaAlimentoPorcion*. Una representación de la ontología agregando las recomendaciones nutricionales puede verse en la fig. 3.7. Las recomendaciones tienen un nombre, por ejemplo *reducirSodio* y una descripción en lenguaje natural, en este caso “Reducir el consumo de sodio por día a

menos de 2300 mg". En términos de ontologías *reducirSodio* es un individuo de la clase *RecomendacionesIngestasDiarias* y la descripción es una objetProperty, *descripcionRegla*, con dominio *RecomendacionesIngestasDiarias* y rango el tipo *String*.

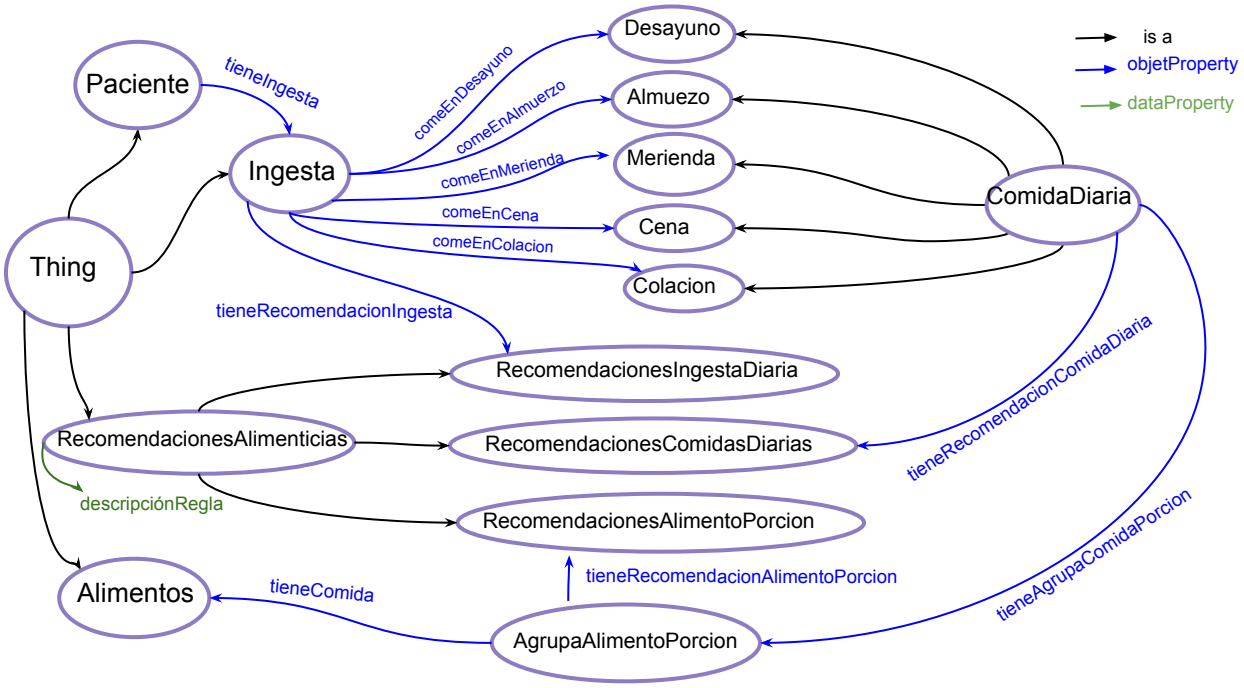

Figura 3.7: Ontología, 3er versión

Decidimos modelar la lógica sobre cuando ejecutar las recomendaciones, dependiendo si pasan ciertos hechos o no mediante reglas swrl. A continuación se enuncian las recomendaciones obtenidas en conjunto con su formulación en reglas swrl.

Recomendaciones sobre las ingestas individuales. Estas recomendaciones están relacionadas con el consumo de algunos alimentos particulares relacionadas con el índice glucémico (IG) de los alimentos. El índice glucémico se refiere a la velocidad con la que los alimentos con carbohidratos elevan el azúcar en sangre y los niveles de insulina. Algunas de las recomendaciones fueron:

- Elegir verduras frescas y no cocidas o en puré, pues de esta última forma su fibra desaparece y el IG se incrementa.

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge HortalizasAlgasHongos(?y) \implies tieneRecomendacionAlimentoPorcion(?x, elegirVerdurasFrescas)$$

- Consumir las frutas y verduras con piel, ya que esto significa más fibra y a mayor fibra, menor índice glucémico.

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge HortalizasAlgasHongos(?y) \implies tieneRecomendacionAlimentoPorcion(?x, consumoVerdurasPiel)$$

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge Frutas(?y) \implies tieneRecomendacionAlimentoPorcion(?x, consumoFrutasPiel)$$

- No pasarse con la cocción de los alimentos, sobre todo, cocinar el arroz y la pasta al dente. Si sólo nos limitamos a cocinar el alimento sin pasarlo de cocción, el alimento mantendrá parte de su estructura, tardará más en digerirse y transformarse en glucosa, por ende, tendrá un índice glucémico menor.

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge CerealesPastasLegumbres(?y) \implies tieneRecomendacionAlimentoPorcion(?x, coccionAlDenteLegumbresCereales)$$

- Evitar el consumo de licuados, exprimidos y jugos de frutas. Los alimentos procesados tienden a tener una clasificación más alta en el índice glucémico que los alimentos integrales como frutas frescas. Por lo tanto consumir preferentemente la fruta fresca en lugar de en licuados, exprimidos o jugos.

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge Frutas(?y) \implies tieneRecomendacionAlimentoPorcion(?x, evitarLicuadosExprimidosJugosFruta)$$

- Siempre que sea posible, refrigerar los cereales o vegetales tipo c. El calentamiento de ciertos almidones después de su almacenamiento en frío contribuye a bajar su índice glucémico.

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge CerealesPastasLegumbres(?y) \implies tieneRecomendacionAlimentoPorcion(?x, refrigerarCereales)$$

$$AgrupaComidaPorcion(?x) \wedge tieneComida(?x, ?y) \wedge HortalizasFrescasTipoC(?y) \implies tieneRecomendacionAlimentoPorcion(?x, refrigerarHortalizasFrescasTipoC)$$

Recomendaciones sobre las ingestas del día. Estas recomendaciones se realizan una vez que finalizó el día y tienen en cuenta todas las ingestas que fue realizando el paciente durante el mismo. Algunas de estas recomendaciones modeladas fueron:

- La recomendación para la población en general para reducir el sodio a menos de 2.300 mg / día es también apropiada para las personas con diabetes.

$$Ingesta(?x) \wedge ingestaTieneSodio(?x, ?y) \wedge swrlb : greaterThan(?y, 2300) \implies tieneRecomendacionIngesta(?x, reducirSodio)$$

- La Cantidad de Hidratos de Carbono no debe restringirse a menos de 130 g / día, para evitar cetosis [2].

$$Ingesta(?x) \wedge ingestaTieneCarbohidratos(?x, ?y) \wedge swrlb : lessThan(?y, 130)$$
$$\implies tieneRecomendacionIngesta(?x, aumentarConsumoCarbohidratos)$$

Recomendaciones sobre ingestas diarias (cena y almuerzo). Consiste en, dependiendo de los alimentos que eligió el paciente para consumir, recomendarle una ingesta similar basándose en lo que se conoce como método del plato. El mismo es un método utilizado por los nutricionistas para la educación de los pacientes diabéticos en la construcción de su alimentación. El método consiste en dividir un plato en dos mitades y presentarlo de la siguiente manera:

- En la primera mitad, introducir los vegetales, que se deben ir combinando en crudo (ensaladas, tomates, zanahorias, etc) y cocidos (acelgas, champiñones, espinacas, etc). Éstos, tienen pocas calorías, por lo que hay que ingerirlos en mayor cantidad.
- La segunda mitad del plato, hay que dividirla en dos cuartos. El primer 1/4 va a contener los alimentos que se destacan por su elevado aporte de proteínas, como son la carne, el pescado y los huevos. El segundo 1/4 estará compuesto por los carbohidratos: pasta, arroz, papa y legumbres principalmente. La cantidad recomendada de éste tipo de nutrientes, es de una taza o el equivalente de un puño cerrado.

Se realizaron algunos cambios en nuestra ontología para poder representar el conocimiento del método del plato. En primer lugar se agrego una clase **MetodoDelPlato** que tiene tres subclases:

1. **1/4Proteínas.** Agrupa las proteínas con su equivalente a un cuarto del plato. Por ejemplo 1/2 bife mediano o 1 filete de merluza chico (tamaño de la palma de la mano) o 1 filete de pollo chico (tamaño de la palma de la mano).
2. **1/2Vegetales.** Agrupa los vegetales con su equivalencia a la mitad de un plato. Por ejemplo medio plato de vegetales equivale a entre 8 y 10 hojas de lechuga + un tomate redondo o un tomate redondo + una cebolla chica.
3. **1/4HidratosDeCarbono.** Agrupa los hidratos de carbono con su respectiva equivalente a un cuarto del plato. Por ejemplo 2 rebanadas de pan integral, 1 pocillo de café de arroz, 1 pocillo de café de lentejas/porotos/garbanzos o 1 pocillo de café de fideos mostacholes.

Los platos que servirán para recomendarle a los pacientes se agrupan en la clase **ComidaDiariaMP**, y se relacionan con individuos pertenecientes a cada una de las tres clases recién mencionadas mediante la propiedad **tieneAgrupaComidaPorcion** como se puede ver en la fig. 3.8

[2]Situación metabólica del organismo originada por un déficit en el aporte de carbohidratos, lo que induce el catabolismo de las grasas a fin de obtener energía, generando unos compuestos denominados cuerpos cetónicos, los cuales descomponen las grasas en cadenas más cortas, generando acetoacetato que es usada como energía por el cerebro y el resto de órganos del cuerpo humano.

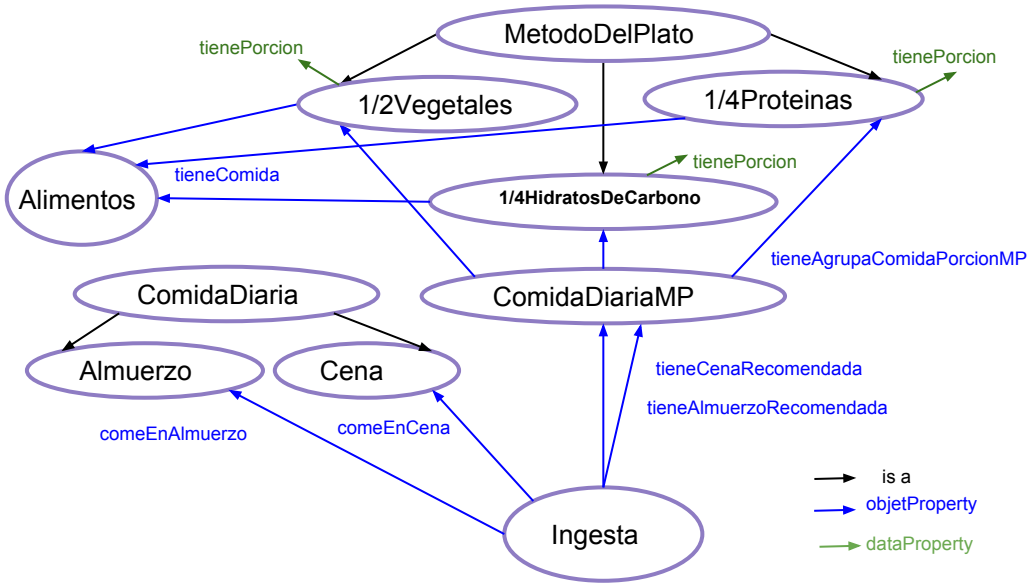

Figura 3.8: Modelado del Metodo del Plato en la Ontología

Veamos algunas de las reglas que utilizamos para las recomendaciones del método del plato : Si el paciente eligió para su almuerzo(o cena) solo hidratos de carbono, entonces se busca en los platos cargados en la ontología alguno que incluya esos hidratos y se le recomienda ese plato. Como ya hemos mencionado en el capítulo anterior, las reglas swrl no admiten el conector OR con lo que creamos dos reglas como vemos a continuación:

$$1/4HidratosDeCarbono(?hc) \wedge ComidaDiariaMP(?cMP) \wedge$$
$$tieneAgrupaComidaPorcionMP(?cMP, ?hc) \wedge Ingesta(?dia) \wedge tieneAlmuerzo(?dia, ?alm)$$
$$\wedge tieneAgrupaComidaPorcion(?alm, ?agrupaComida1) \wedge tieneComida(?agrupaComida1, ?w)$$
$$\wedge AlimentosRicosEnAlmidones(?w) \wedge tieneComidaMP(?hc, ?w)$$
$$\Longrightarrow tieneAlmuerzoRecomendado(?dia, ?cMP)$$

$$1/4HidratosDeCarbono(?hc) \wedge ComidaMP(?cMP) \wedge$$
$$tieneAgrupaComidaPorcionMP(?cMP, ?hc) \wedge Ingesta(?dia) \wedge tieneAlmuerzo(?dia, ?alm)$$
$$\wedge tieneAgrupaComidaPorcion(?alm, ?agrupaComida1) \wedge tieneComida(?agrupaComida1, ?w)$$
$$\wedge AlimentosRicosEnAlmidones(?w) \wedge tieneComidaMP(?hc, ?w)$$
$$\Longrightarrow tieneCenaRecomendada(?dia, ?cMP)$$

De la misma forma agregamos las reglas, para cuando sólo elige verduras o sólo elige proteínas. Cuando el paciente consume alimentos que se encuentran en dos de los tres

grupos por ejemplo proteínas e hidratos de carbono en la cena se aplicaría la siguiente regla:

$$1/4Proteinas(?p) \wedge 1/4HidratosDeCarbono(?hc) \wedge ComidaMP(?cMP) \wedge$$
$$tieneAgrupaComidaPorcionMP(?cMP, ?hc) \wedge tieneAgrupaComidaPorcionMP(?cMP, ?p)$$
$$\wedge Ingesta(?dia) \wedge tieneAlmuerzo(?dia, ?alm) \wedge tieneAgrupaComidaPorcion(?alm, ?agrupaComida1)$$
$$\wedge tieneComida(?agrupaComida1, ?w) \wedge tieneComida(?agrupaComida1, ?z)$$
$$\wedge AlimentosRicosEnAlmidones(?w) \wedge AlimentosRicosEnProteinas(?z)$$
$$\wedge tieneComidaMP(?hc, ?w) \wedge tieneComidaMP(?hc, ?z) \wedge$$
$$\Longrightarrow tieneCenaRecomendada(?dia, ?cMP)$$

Cuando consume los tres grupos de alimentos lo que nos interesa es que consuma las porciones adecuadas, por cual buscamos un individuo del método del plato que contenga los alimentos que consumió el paciente pero con las porciones adecuadas:

$$1/4Proteinas(?p) \wedge 1/4HidratosDeCarbono(?hc) \wedge 1/2Vegetales(?v) \wedge$$
$$ComidaMP(?cMP) \wedge tieneAgrupaComidaPorcionMP(?cMP, ?hc) \wedge$$
$$tieneAgrupaComidaPorcionMP(?cMP, ?p) \wedge tieneAgrupaComidaPorcionMP(?cMP, ?v)$$
$$\wedge Ingesta(?dia) \wedge tieneAlmuerzo(?dia, ?alm) \wedge tieneAgrupaComidaPorcion(?alm, ?agrupaComida1)$$
$$\wedge tieneComida(?agrupaComida1, ?w) \wedge tieneComida(?agrupaComida1, ?z)$$
$$\wedge tieneComida(?agrupaComida1, ?v) \wedge AlimentosRicosEnAlmidones(?w)$$
$$\wedge AlimentosRicosEnProteinas(?z) \wedge AlimentosRicosEnVegetales(?y)$$
$$\wedge tieneComidaMP(?hc, ?w) \wedge tieneComidaMP(?hc, ?z) \wedge$$
$$\wedge tieneComidaMP(?hc, ?v) \Longrightarrow tieneCenaRecomendada(?dia, ?cMP)$$

Para construir las reglas que recomiendan una ingesta apropiada según el método del plato hemos utilizado para su definición tres clases que no hemos presentado hasta el momento: **AlimentosRicosEnAlmidones**, **AlimentosRicosEnVegetales** y **AlimentosRicosEnProteínas**. Dichas clases pertenecen a la ontología de alimentos y representan una clasificación de la clase alimentos. Los individuos que pertenece a la clase *AlimentosRicosEnProteínas* son los que pertenecen al grupo de la carnes, pescados, huevos y legumbres. A su vez un individuo pertenece a la clase *AlimentosRicosEnAlmidones* si se encuentra en las hortalizas de tipo C como la papa y el camote, las pastas simples o rellenas, los arroces y las harinas. Finalmente un individuo pertenece a la clase *AlimentosRicosEnVegetales* si pertenece a la clase *VegetalesTipoA* o *VegetalesTipoB*. Necesitamos definir estas clases de tal forma que si un individuo cumple ciertas restricciones entonces pertenece a la clase y que si pertenece a dicha clase es porque cumple ciertas restricciones. Dado que se pretende que un razonador pueda inferir automáticamente los miembros de dichas clases definimos las clases como *CLASES DEFINIDAS*, esto significa definirlas como si fueran clases equivalentes. Por ejemplo podemos ver la definición de *AlimentosRicosEnProteinas*:

```
<owl:Class rdf:about="http://www.semanticweb.org/sig/ontologies/2015/5/
ontoAlimentos#AlimentosRicosEnProteinas">
 <owl:equivalentClass>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <rdf:Description rdf:about="http://www.semanticweb.org/
    sig/ontologies/2015/5/ontoALimentos#CarnesPescadosMariscos"/>
    <rdf:Description rdf:about="http://www.semanticweb.org/
    sig/ontologies/2015/5/ontoAlimentos#Huevos"/>
    <rdf:Description rdf:about="http://www.semanticweb.org/
    sig/ontologies/2015/5/ontoIngesta#Legrumbres"/>
   </owl:unionOf>
  </owl:Class>
  </owl:equivalentClass>
 <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/
 sig/ontologies/2015/5/ontoALimentos#Alimentos"/>
</owl:Class>
```

A lo largo de este capítulo pudimos describir el proceso iterativo que fuimos utilizando para la creación de nuestra ontología final, que contempla tanto la ingesta del paciente como las recomendaciones nutricionales del mismo cumpliendo así los objetivos propuestos relacionados con la creación de nuestra ontología.

Capítulo 4

Una interfaz para la ontología

Muchas veces la manipulación de la ontología (creación de clases, individuos, propiedades) puede resultar tedioso para las personas que no están ligadas a la informática. No se puede pretender que un paciente para poder contar con la información de los carbohidratos ingeridos o recibir alguna recomendación sobre su ingesta tenga que crear el individuo ingesta para cada día y a su vez tenga que crear cada una de las ingestas individuales y relacionarlas con los alimentos que consumió. Por lo que contar con una herramienta que abstraiga esta tarea y sea amigable para el paciente diabético es fundamental.

4.1. Tecnologías utilizadas

Para la realización de la interfaz de nuestra ontología hemos utilizado como lenguaje de programación **Java**, principalmente porque cuenta con librerías para el manejo de ontologías pero además porque es un lenguaje multiplataforma gratuito que cuenta con un gran soporte y documentación y una amplia variedad de librerías e interfaces de programación. En el presente proyecto se utilizó **Jena Ontology API**, que es una librería que permite escribir programas que interactúen con ontologías documentadas en diversos formatos, que van desde OWL (el más expresivo de los lenguajes) hasta RDFS (el más débil de los lenguajes disponibles para ontologías). A través de Jena Ontology API, se apunta a proveer una interfaz de programación consistente para el desarrollo de aplicaciones sobre ontologías, independientemente del lenguaje de ontologías que se use en los programas. En este sentido, la interfaz Jena es neutral, es decir que los nombres de las clases de Java no son específicos del lenguaje subyacente. Por ejemplo, la clase de Java OntClass puede representar una clase de OWL o de RDFS. Para representar las diferencias entre las representaciones, cada lenguaje de ontologías tiene un perfil, el cual lista los constructores permitidos y los nombres de las clases y propiedades. Así, en el perfil de OWL sería owl:ObjectProperty y en el perfil RDFS sería null ya que RDFS no define propiedades de objeto (Object properties).

4.2. Implementación

La implementación se dividió en dos partes. Primero se crearon todos los métodos necesarios para manipular la ontología utilizando las primitivas dadas por JENA creando una librería propia. Por otro lado se creó una interfaz de usuario que importa la librería para acceder a la ontología. La clase principal llamada **ontologiaPersona**, se encarga de cargar en memoria el fichero OWL donde se encuentra la ontología y además carga tres estructuras que contienen todas las clases, propiedades e individuos de la misma.

```
1  public class ontologiaPersona {
2   static  String Ontofile;
3   public OntModel m;
4
5  /*Estructura de datos para almacenar los nombre de las clases,
6   propiedades e individuos y su valor(uris). Podremos dado un
7   nombre saber cual uri tiene.*/
8   public Map<String, String> mapClases = new HashMap<String, String>();
9   public Map<String, String> mapPropiedad= new HashMap<String, String>();
10  public Map<String,String> mapIndividuo=new HashMap<String, String>();
11
12   //carga la ontologia desde el repositorio
13  public ontologiaPersona(String archivo){
14   Ontofile = archivo;
15   OntModelSpec oOntModelSpec=OntModelSpec.OWL_MEM;
16   m = ModelFactory.createOntologyModel(oOntModelSpec);
17   m.setStrictMode(false);
18   try {
19    InputStream in = FileManager.get().open(archivo);
20    if (in ==null) {
21     System.out.println("ERROR abriendo archivo" + Ontofile);
22     return;
23    }
24    else {
25     m.read(in, "OWL_MEM");
26     System.out.println("archivo" + Ontofile + " leido exitosamente" );
27     this.crearMaps();
28     System.out.println("ya creee los maps" );
29    }
30   }
31   catch (JenaException je) {
32    System.out.println("ERROR leyendo archivo" + je.getMessage());
33    je.printStackTrace();
34    System.exit(0);
35   }
36  }
37  OntModel getModelo(){
38  return m;
39  }
40  void crearMaps(){
41   ExtendedIterator<OntClass> clases = this.m.listClasses();
42   System.out.println("liste clases" );
43   long startMil = System.currentTimeMillis();
```

```
44   System.out.println("Start ListIndividuals: " + startMil);
45   ExtendedIterator<Individual> individuos = this.m.listIndividuals();
46   long endMil = System.currentTimeMillis();
47   System.out.println("Duration ListIndividuals: " + (endMil - startMil));
48   ExtendedIterator<OntProperty> propiedades = this.m.listAllOntProperties();
49   System.out.println("liste prop" );
50    if(clases !=null){
51    while(clases.hasNext()){
52     OntClass c  = clases.next();
53     if(c!=null){
54      String name =c.getLocalName();
55      String uri = c.getURI();
56      if(name != null){
57       this.mapClases.put(name, uri);
58      }
59     }
60    }}
61   if(individuos !=null){
62    while(individuos.hasNext()){
63     Individual i  = individuos.next();
64       if(i!=null){
65        String name =i.getLocalName();
66        String uri = i.getURI();
67         if(name != null){
68                  this.mapIndividuo.put(name, uri);
69                 }
70         }
71 }}
72   if(propiedades !=null){
73    while(propiedades.hasNext()){
74     Property p  = propiedades.next();
75     if(p!=null){
76      String name = p.getLocalName();
77      String uri = p.getURI();
78      if(name != null){
79       this.mapPropiedad.put(name, uri);
80      }
81     }
82    }}
83   return;
84  }
```

Listing 4.1: Clase OntoPersona

Algunos de los métodos creados fueron:

```
1 public String getValorPropiedadIndividuo(String ind, String prop)
```

El método getValorPropiedadIndividuo toma como entradas un individuo y una propiedad y devuelve como resultado el valor de dicha propiedad para ese individuo.

```
1  public List<RDFNode> getListValPropiedadIndividuo(String ind, String prop)
```

Dado un individuo I y una ObjectProperty P el método getListValPropiedadIndividuo devuelve una lista de los individuos que están relacionados con P a través de I.

```
public List<Individual> getIndividuosDeClases(String clase)
```

Dada una clase, el método getIndividuosDeClases devuelve todos los individuos que pertenecen a esa clase.

```
public int addPatient(String nombrePaciente)
```

Dado el nombre de un paciente addPatient agrega al paciente a la ontología y retorna 0 y si el paciente ya esta cargado en la ontología retorna 1.

```
public UrisAdd addIngestaPorComidaDiria(Ingesta ing )
```

addIngestaPorComidaDiria es el método principal ya que se encarga de crear todos los individuos y relaciones pertinentes al cargar la ingesta. Es decir crea el individuo ingesta, relaciona el individuo ingesta con el paciente y le agrega tieneFecha, agrega la ingesta según la comida diaria, creando el individuo adecuado, luego agrega la ingesta individual con sus respectivas porciones.

```
public void cargarValorPropIngesta(String ingesta)
```

La función cargarValorPropIngesta se encarga de calcular la cantidad de carbohidratos que tiene una ingesta y relaciona el individuo ingesta y el valor calculado con la propiedad **ingestaTieneCarbohidratos**.

```
public  List<IngestasDiarias> listarIngesta(String paciente, String date)
```

El método listarIngesta lista todas las ingestas de un paciente en una fecha dada. Si el campo date es null, retorna todas las ingestas del paciente.

Para modelar las reglas swrl e inferir resultados de la ontología utilizamos el subsistema de inferencia de Jena. El subsistema de inferencia Jena está diseñado para permitir una gama de motores de inferencia o razonadores para ser integrados en Jena. Estos motores se utilizan para derivar afirmaciones RDF adicionales que se implican de alguna base de conocimiento junto con cualquier información de la ontología y los axiomas y reglas asociadas con el razonador. El método que se encarga de cargar las reglas del sistema y llamar al razonador es el siguiente:

```
/* Dado un archivo dnd esten cargadaas las reglas (ej:file.rules),
 dado el modo(ej:forward) y el archivo donde esta el modelo
(ej: OntoPersona.owl) retorna el modelo luego de inferir las reglas.*/
public OntModel runRules(String rules, String mode, String model ){
 // create a default model (empty model)
 Model m1 = ModelFactory.createDefaultModel();

 Resource configuration = m1.createResource();
 configuration.addProperty(ReasonerVocabulary.PROPruleMode, mode);
 configuration.addProperty(ReasonerVocabulary.PROPruleSet, rules);

 // crear una instancia del razonador y cargar en m1 el modelo
 Reasoner  r=GenericRuleReasonerFactory.theInstance().create(configuration);
 m1 = FileManager.get().loadModel(model);
 // crear el modelo de inferencia
```

```
16  InfModel infModel = ModelFactory.createInfModel(r, m1);
17
18  // ejecutar las reglas y cargar en m la ontologia y guardarla.
19  infModel.prepare();
20  OntModelSpec oOntModelS=OntModelSpec.OWL_MEM_MICRO_RULE_INF;
21  OntModel ontModel=ModelFactory.createOntologyModel(oOntModelS, infModel);
22  m = ontModel;
23  saveOntology(Constantes.NOMBRE_ONTO);
24  return m;
25 }
```

4.3. Ejemplo de Uso

Veamos cómo sería la interacción entre el usuario y la aplicación y las tareas que se deben realizar para llevar acabo cada acción impuesta por el usuario:

1. **Login** Antes de comenzar a usar el sistema, en la ontología debe cargarse la información referida al paciente. Se carga el individuo paciente con sus respectivas propiedades como el nombre, la fecha de nacimiento, el peso, la talla, etc. El nombre de usuario guarda relación con el individuo de la clase PACIENTE, información que se carga al momento de hacer el login (fig. 3.8).

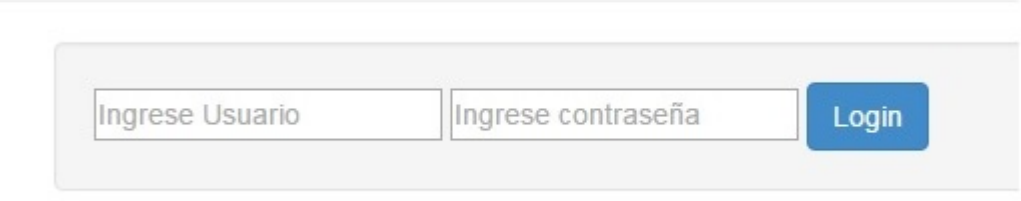

Figura 4.1: El paciente ingresa al sistema.

2. **Carga de ingesta**. El paciente comenzaría el día cargando lo que consume en el desayuno. En la pantalla se elige el tipo de comida diaria, en este caso Desayuno. En el campo *elige alimento* se listan todos los individuos que pertenecen a la clase alimentos de la ontología. Esta acción llama al método necesario pasándole como argumento la clase *ALIMENTOS*: **getIndividuosDeClases(ALIMENTOS)** (fig. 4.2). El paciente elige los alimentos y porciones correspondientes que va a ingerir y presiona guardar. En este caso se llama al método *addIngestaPorComidaDiria* que realiza los siguientes pasos:

   - utiliza el nombre del paciente para obtener el individuo: **Individual paciente = getIndividuo("nombrePaciente")**.
   - obtiene el URI de la ontología, por ejemplo: *http://www.semanticweb.org/sig/ontologies/2015/5/ingestaPaciente*. Construu-

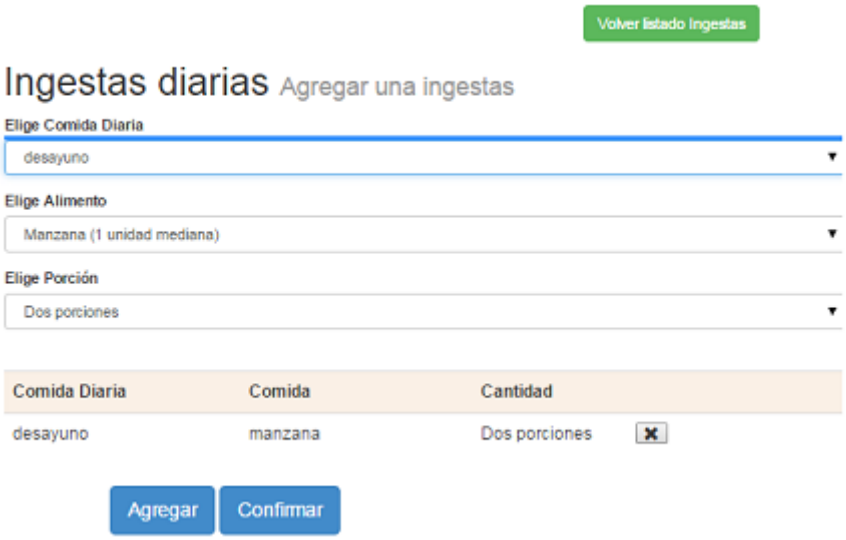

Figura 4.2: El paciente ingresa la ingesta.

ye el URI que le agregará a los individuos que debe crear. El mismo se compone por **URI base** + **nombrePaciente** que llamaremos URI base (uriB)

- Si no fue creado, crea el individuo ingesta. Para eso construye la URI utilizando la URI base y la fecha de la ingesta: **uriIngesta**= **uriB** + **ingesta** + **fecha**:

```
1 ingestaInd = m.createIndividual(uriIngesta, claseIngesta);
```

- relaciona la ingesta con el paciente:

```
1 Property tieneIngesta=getPropiedades(Constantes.P_TIENE_INGESTA);
2 paciente.addProperty(tieneIngesta, ingestaInd);
```

- agrega la propiedad *tieneFecha* a la Ingesta:

```
1 Property tieneFecha = getPropiedades(Constantes.P_TIENE_FECHA);
2 ingestaInd.addLiteral(tieneFecha,fecha);
```

- agrega la ingesta según la comida diaria, en este caso la ingesta del desayuno:

```
1 String uriIngD = bNS+ desayuno+ fecha;
2 OntClass ingD = getClases(desayuno);
3 Property propIngestaDiaria = getPropiedades(comeEnDesayuno);
4 Individual ingestaDiaria = m.createIndividual(uriIngD, ingD);
5 ingestaInd.addProperty(propIngestaDiaria, ingestaDiaria);
```

- finalmente agrupa para cada alimento consumido, el alimento con su respectiva porción, en la clase *AgrupaALimentoPorcion* y carga la propiedad *tieneCarbohidratosComida*:

```
float ct = 0;
List<UrisAgrupaCP> urisAgrupaCP=new ArrayList<UrisAgrupaCP>();
 for(AlimentoPorcion alimentoPorcion: ing.getAlimentoPorcion()){
 String uriComPor= bNS+ desayuno+alimentoPorcion.getAlimento()
 + fecha);
 OntClass claseComP=getClases(Constantes.C_COM_PORCION);

 Individual comidaPInd=m.createIndividual(uriComPor, claseComP);

 Property propComPor=getPropiedades(Constantes.P_T_AGRUPA_C_P);
 ingestaDiaria.addProperty(propComPor,comidaPInd);
 urisAgrupaCP.add(new UrisAgrupaCP(uriComidaPorcion));

 String a = alimentoPorcion.getAlimento();
 Individual indALimento = getIndividuo(a);
 Property propCom=getPropiedades(Constantes.PROP_TIENE_COMIDA);
 comidaPInd.addProperty(propCom, indALimento);

 Float porcion=alimentoPorcion.getPorcion();
 Property propPorc=getPropiedades(Constantes.PROP_COMIDA_T_PORC);
 comidaPInd.addLiteral(propPorc, porcion);
 saveOntology(Constantes.NOMBRE_ONTO);
 reloadOntology();
 String ind = comidaPorcionInd.getLocalName();

 float ch=cargarPropiedadesDerivadas(Constantes.PROP_TIENE_COMIDA,
 Constantes.PROP_COMIDA_TIENE_PORC,ind,
 Constantes.PROP_TIENE_CARBOHIDRATOS_COMIDA,
 Constantes.PROP_TIENE_CARBOHIDRATOS_COMIDA_A);
 if (ch >0){
  ct = ct + ch;
  }
 }
Property propCarbohidratoCD=getPropiedades(Constantes.PROP_CD_HC);
ingestaDiaria.addLiteral(propCarbohidratoCD, ct);
```

3. Listados de ingestas y Recomendaciones. Una vez ingresada la ingesta, se listan las ingestas del día, se muestra la cantidad de carbohidratos ingeridos por el paciente y si hay se agregan las recomendaciones asociadas a la ingesta. Si se trata de una cena o almuerzo se listan las ingestas recomendadas por el método del plato. (fig. 4.3). Se llaman a tres métodos:

   - se ejecutan las reglas cargadas en el sistema :

```
public OntModel runRules(String rules, String mode, String model)
```

   - se listan las ingestas del paciente:

```
public List<IngestasDiarias> listarIng(String paciente,String f)
```

   - se listan las ingestas recomendadas del paciente:

Figura 4.3: Lista de ingestas e ingestas recomendadas.

```
1 public List<IngestasDiarias> listIngRec(String paciente,String f)
```

Se pretende que todo el conocimiento modelado tanto de los alimentos, la ingesta del paciente o el sistema endocrino del paciente provea el marco necesario para el desarrollo de una plataforma de telemedicina para la asistencia del paciente. En este marco el prototipo que planteamos en esta sección pretende ser un pequeño paso hacia la concreción de esta plataforma que debería ser de gran utilidad para monitorear a distancia las variables más importantes de los pacientes diabéticos con el fin de prevenir episodios que puedan dañar seriamente la calidad de vida de los mismos.

4.4. Validaciones usando modelo simulador UVA/Padova

En esta sección mostraremos mediante algunas pruebas realizadas sobre el simulador UVA/Pandova la importancia de la realización del anuncio de la ingesta del paciente en la eficacia de los algoritmos de control. Como hemos mencionado en los capítulos previos el PA trata de emular el comportamiento de un páncreas humano saludable. El mismo se compone de: un sensor continuo de glucosa en sangre, un algoritmo de control o computador de las dosis de insulina y una bomba de infusión de insulina la cual es comandada por dicho algoritmo. Con el objetivo de diseñar un controlador automático que conecte un medidor continuo de glucosa y una bomba de insulina, es necesario, generalmente, un modelo del Sistema Endocrino humano para verificar la efectividad del controlador antes de las pruebas clínicas. Uno de los modelos mayormente utilizados, dado que está aprobado por la FDA, es el Modelo Matemático UVA/Padova. Este modelo también puede ser utilizado para predecir el comportamiento de nuestro sistema endocrino. Es decir el paciente puede ingresar lo que piensa comer en el desayuno y así el modelo le muestra como se va a comportar su glucosa en sangre y la cantidad de insulina que debe suministrarse. La integración del modelo con nuestros datos podemos verlo en la fig. 4.4. Los

carbohidratos que se calculen en la interfaz se pasan como entrada al simulador como el anuncio de la ingesta. Cabe aclarar que el anuncio de la ingesta debe realizarse de forma manual. Es decir, utilizamos la interfaz para calcular la cantidad de carbohidratos de la futura ingesta, y luego cargamos manualmente dicho resultado en el modelo antes de poder simular el impacto de dicha ingesta sobre la glucosa del paciente.

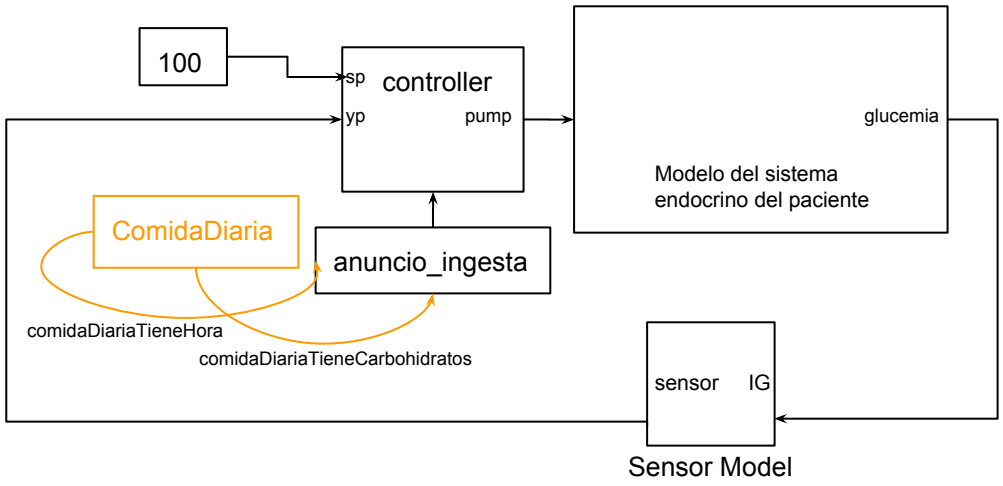

Figura 4.4: Modelo en MatLab/Simulink integrado para la verificación

Utilizaremos este modelo para mostrar que mientras más exactos son los datos del anuncio de la ingesta que se le suministran al algoritmo de control, mejor es su funcionamiento. Los experimentos realizados para probar el funcionamiento del controlador dependiendo del anuncio de la ingesta incluyen:

- No realizar el anuncio de la ingesta.
- Proporcionar anuncio de ingesta por adelantado.

El paciente a analizar en el presente caso de estudio es el paciente número 21 del Simulador UVA/Padova. Es un adulto con DMTI, tiene un peso corporal de 76,37 y se lo puede considerar un paciente controlable, o de riesgo bajo. La simulación se configura para operar de la siguiente manera: se somete al paciente a 30 horas de monitoreo de los niveles de glucosa en sangre de manera continua, teniendo en cuenta todo el tiempo la utilización de la bomba de insulina. La simulación sigue las siguientes características:

1. A las 7:30Hs del día 2 (hora 13,5 de simulación) el sujeto desayuna una ingesta particular. durante 2 minutos.
2. A las 12Hs del día 2 (hora 18 de simulación) el sujeto almuerza una ingesta con una duración de 15 minutos.
3. A las 18Hs del día 2 (hora 24 de simulación) el sujeto cena durante 15 minutos.

Este escenario se realiza con valores de carbohidratos ingeridos (CHO) acorde a un paciente adulto. Esto corresponde a aproximadamente 50 CHO en el desayuno, 65 CHO en el almuerzo y 80 CHO en la cena. Estos datos son cargados en el simulador como la ingesta real del paciente.

El paciente cargaría en la aplicación la ingesta que será anunciada al simulador.

- Ingresa lo que consumiría en el desayuno: Pan Lactal Lacteado, una porción (dos unidades) que contiene 23 CHO más Mermelada Arcor Sabor frutilla una porción (1 cuchara de postre) que contiene 8.3 CHO más Barra de cereal Cereal Fort naranja y avena tostada que contiene 18 CHO.

- Ingresa lo que consumiría en el almuerzo: Arroz Gallo carnaroli una porción (1 plato) que contiene 38 CHO más Medallón de pollo Granja de Sol con queso 1 porción (una unidad) que contiene 23 CHO.

- Ingresa lo que consumiría en la cena: Fideos Don Vicente Pasta larga al huevo una porción (un plato) que contiene 53 CHO más Salsa lista Knorr Cica pomarola tradicional 2 porciones (6 cucharadas soperas) que contiene 8.6 CHO más Grisines Fargo tradicionales una porción (10 unidades) que contiene 21 CHO.

Estos datos identifican al anuncio de la ingesta obtenidos de la aplicación y corresponden a 49.3 CHO en el desayuno, 61 CHO en el almuerzo y 82.6 CHO en la cena. Veamos mediante la simulación del modelo UVA/Padova como es la evolución de la glucosa en sangre y la acción de la bomba de insulina para este paciente.

El simulación UVA/Padova tiene la posibilidad de que le incluyamos al controlador un pre-aviso de lo que el paciente consume. Por lo tanto se mejora bastante la acción del controlador cuando existe ese pre-aviso. Si observamos la fig. 4.5 vemos que cuando hay pre-aviso el controlador puede mantener los niveles de insulina mas bajos, siendo en los minutos 900, 1180 y 1500 cuando realiza picos de los niveles de glucosa en sangre, aunque apenas sobrepasan los 160 mg/dl. Esos valores son bajados rápidamente para estabilizarse en valores cercanos a 100. Los niveles de glucosa en la gráfica donde no se realizó el anuncio tiene picos también en los mismos minutos y alcanzan hasta 200mg/dl. En este caso el controlador no logra estabilizar los valores tan rápido, como si lo hace al haber anuncio. Esto se debe a que cuando se realiza el anuncio de la ingesta se tiene la ventaja de que la dosis de insulina se anticipa a la cantidad de hidratos que debe metabolizar y por ello mejora notablemente el efecto en la glucemia. Técnicamente equivale a un esquema de control con acción feedforward o en avance. Esta metodología siempre reporta mejores resultados en particular cuando se conoce con exactitud la ingesta y los instantes de tiempo en que ingresa al organismo. En cambio cuando no tiene esa información tiene que esperar hasta que el valor de entrada del sensor de glucosa en sangre (valor **yp** en la fig. 4.4) indique que la glucemia esta por arriba del rango saludable o valor de referencia (el valor **sp** en la fig. 4.4). Este comportamiento puede observarse en la fig. 4.6. Se advierte un retardo en el tiempo que comienza a proporcionarse insulina cuando no se realiza el aviso. El sujeto ya comió los carbohidratos y la glucosa llega a la sangre, el sensor informa esta situación y recién ahí comienza a actuar el controlador indicando

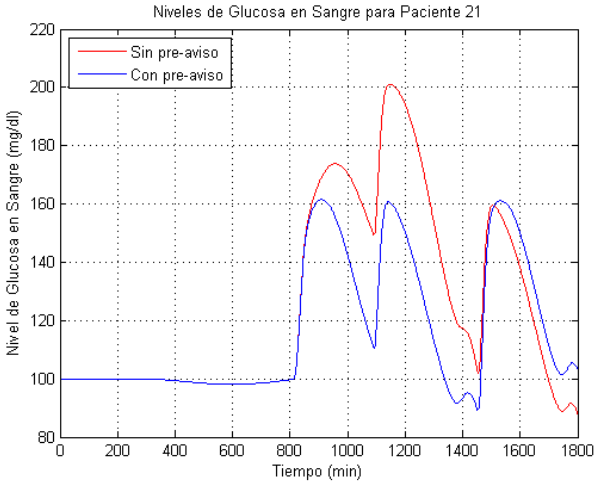

Figura 4.5: Niveles de glucosa para el paciente 21

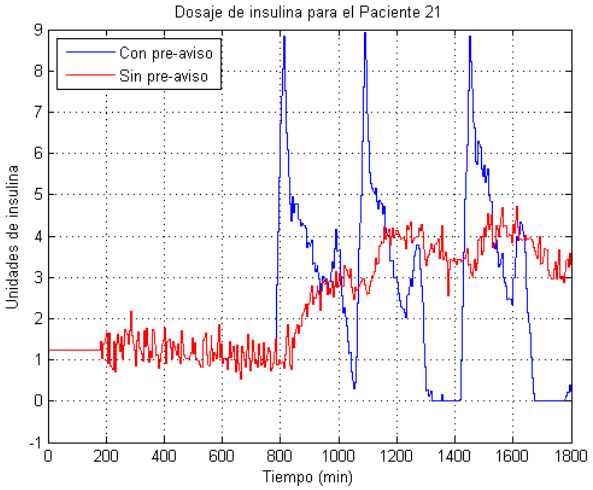

Figura 4.6: Dosaje de insulina para el paciente 21

que hay que suministrar una cantidad específica de insulina. En contraposición a cuando se realiza un anuncio donde el controlador comienza a suministrar insulina para que la ingesta del paciente pueda metabolizarse antes que la glucosa llegue a la sangre.

Capítulo 5

Conclusiones y trabajos futuros

5.1. Conclusiones

Al comenzar el desarrollo de este trabajo teníamos tres objetivos específicos. Por un lado poder modelar nuestra propia ontología de alimentos y base de conocimientos teniendo en cuenta los diferentes tipos de alimentos disponibles en Argentina. Por otro lado proponer una representación ontológica de los pacientes con especial enfoque en la ingesta de los pacientes diabéticos. Finalmente reunir información valiosa de los nutricionistas que ya trabajan con pacientes diabéticos a través de múltiples entrevistas con el objetivo de abstraer la terapia nutricional que aplican en sus pacientes en forma de reglas que luego sean utilizadas para formular las recomendaciones. En cuanto al primer objetivo, se ha puesto de manifiesto en este trabajo que era necesario la realización de una nueva ontología de alimentos, dado que la clasificación de alimentos que utilizan los nutricionistas argentinos se rige por factores diferentes a aquellos considerados durante la construcción de las ontologías de alimentos presentadas en la sección 2.2.1. Pese a esto se reutilizaron varias ideas de las ontologías existentes en función de la representación de ciertas propiedades de los alimentos y la taxonomía de la ontología, aunque principalmente se siguió el enfoque que usan en *Nutrinfo* dado que es una red virtual de prestigio en la comunidad de nutricionistas Argentinos.

El poder sintetizar una ontología que permita el cálculo de una dosis oral de carbohidratos ingeridos, mediante la especificación de alimentos comerciales que el paciente incorpora en su organismo representó un gran desafío. El desarrollo de la ontología en cuestión fue un proceso iterativo en conjunto con nutricionistas especializados en nutrición, e integrantes del grupo de investigación sobre el manejo eficiente de tecnologías para diabetes, entre ellos con el grupo que realizó el modelado del sistema endocrino del paciente. Durante el proceso del desarrollo de ambas ontologías hemos utilizado razonadores informáticos que han permitido no solo validar la consistencia del modelado de las ontologías sino también han sido utilizados para inferir todas las nociones completas acerca del conocimiento modelado.

La abstracción en reglas de la terapia nutricional aplicada por los médicos a sus pacientes fue uno de los puntos más dificultosos de elaborar. Proponer reglas sugiere algo

fijo, estático que se cumple bajo ciertas circuntancias en un grupo determinado de estudio. En este caso nosotros queríamos encontrar dichas reglas para pacientes diabéticos. Por ejemplo, queríamos saber si era posible determinar la cantidad ideal de carbohidratos que debía comer la persona en cada ingesta o qué alimentos o comidas debían evitar los pacientes diabéticos o cómo estos elevaban el azúcar en sangre. Las repuestas que encontrábamos muchas veces en los nutricionistas es que no todas las personas metabolizamos de la misma manera los alimentos, y que hay muchos factores que lo determinan como la edad del paciente, el sexo, el horario de la ingesta, las enfermedades que pueda tener el paciente, la genética, el estado de ánimo en el momento de la ingesta, el estrés del paciente, la actividad física realizada, entre otros. Por lo tanto nos inclinamos más por escribir recomendaciones que son las que generalmente utilizan los nutricionistas en la terapia con el paciente diabético utilizando para esto las recomendaciones de la OMS.

Como objetivo general del proyecto se consideró la disponibilidad de contar con un servicio de telemedicina o e-health que sería de gran utilidad para monitorear a distancia las variables más importantes de los pacientes diabéticos con el fin de prevenir episodios que puedan dañar seriamente su calidad de vida. En este contexto se realizó una plataforma computacional basada en las ontologías desarrolladas para asistir al paciente diabético en el conteo de carbohidratos y además proporcionarle recomendaciones alimentarias que se ajusten al paciente y sus necesidades particulares de salud. Constituyendo con esto la primera etapa para soportar la implementación de un sistema de telemedicina para profesionales de diferentes áreas abocadas al tratamiento de pacientes diabéticos. En el desarrollo de la plataforma computacional las mayores limitaciones se encontraron con la librería que se utilizó para acceder a la ontología. Las limitaciones están relacionadas con la performance del método que lista los individuos. Durante la realización de algunas pruebas se encontró que al llamar al método de la librería listIndividuals() tardaba más de 20 segundos para listar 50 individuos y el tiempo crecía a medida que se agregaban individuos en la ontología. Se reporto este error y obtuvimos una respuesta para poder mejorar la performance del método. Igualmente se debería analizar la posibilidad de utilizar otra API para OWL, como OWL API [API11] y realizar pruebas de performance para ver si obtenemos mejores resultados que con JENA.

Por otro lado podemos destacar que en las pruebas realizadas con el modelo de sistema endócrino humano, en términos del simulador UVA/Pandova se pudo observar la importancia de contar con una buena aproximación del anuncio de la ingesta lo que se resume en una buena respuesta del algoritmo de control, en base a mantener estables los niveles de glucosa en sangre.

5.2. Trabajos Futuros

La aplicación es sólo un prototipo. Todavía queda mucho trabajo por delante para que pueda ser utilizada por pacientes reales. En primer lugar la interfaz gráfica es bastante básica, habría que cambiarla para que sea más intuitiva y simple para el paciente. Además algo que se podría agregar para facilitar el trabajo del paciente es que tenga un lector de código de barra y entonces para cuando consume alimentos que tienen etiqueta puede

cargarlo con más rapidez. Más aún, podríamos pensar que sólo tomando una foto del plato con la futura ingesta, se calcule la cantidad de hidratos de carbono que la persona va a ingerir.

Otro aspecto a mejorar es la vinculación entre el simulador y la interfaz. Actualmente el anuncio de la ingesta debe realizarse de forma manual. Es decir, se utiliza la interfaz para calcular la cantidad de carbohidratos de la futura ingesta y luego se carga manualmente el resultado en el modelo antes de realizar la simulación que muestra el impacto de dicha ingesta sobre la glucosa del paciente. Por lo tanto sería interesante que esta vinculación pueda realizarse de manera automática a partir de los datos cargados por el usuario.

Por otro lado hemos mencionado que lo único que eleva la glucosa en sangre son los carbohidratos y por eso es fundamental poder realizar una buena aproximación del anuncio de la ingesta. No hemos mencionado que hay además otro aspecto que se debe tener en cuenta y tiene que ver con la absorción de carbohidratos. Un paciente diabético que consume **x** cantidad de carbohidratos necesita una cantidad **y** de insulina, el tema es como ir proporcionando esa insulina para que pueda el algoritmo de control representar más adecuadamente el trabajo que realiza el páncreas humano. No es lo mismo consumir un plato de fideos, que consumir un plato de fideos con vegetales. Es lo mismo en cantidad de carbohidratos pero es distinto en como el organismo realiza la absorción de los carbohidratos, ya que ciertas combinaciones de alimentos hacen que la absorción de los carbohidratos se ralentice. El proceso de absorción de carbohidratos es alterado por perturbaciones externas, y esto es un factor que tienen en cuenta los algoritmos de control. En el modelo de Uva/Pandola que hemos utilizado, la aparición de la glucosa no tiene en cuenta la composición de la comida. Para evaluar el efecto de las distintas combinaciones de las comidas el grupo MICElab de la universidad de Girona ha desarrollado una biblioteca de comidas mixtas la cual contiene diferentes perfiles de la tasa de aparición de la glucosa a partir de varias comidas compuestas, usando el método de identificación bayesiano [Her+12; Bar+12]. Sería interesante poder ampliar nuestra ontología para que en el anuncio de la ingesta además de proporcionar la cantidad de carbohidratos se proporcione algún tipo de función que vincule la ingesta que realiza el paciente con la relación en la tasa de aparición de la glucosa. De esta forma se brindaría información más exacta al algoritmo de control.

Bibliografía

[API11] OWL API. «OWL API». En: (2011). URL: http://owlapi.sourceforge.net/.

[B+08] Kovatchev B. y col. «Method, system and computer simulation environment for testing of monitoring and control strategies in diabetes.» En: (2008). URL: https://www.google.com/patents/US20100179768.

[B+10] Kovatchev B. y col. «System, method and computer simulation environment for in silico trials in prediabetes and type 2 diabetes». En: (2010).

[Bar+12] C. Barajas-Solano y col. «A review of absorption models for mixed meals». En: *In 5th International Conference on Advanced Technologies and Treatments for Diabetes* (2012).

[BHL14] Tim Berners-Lee, James Hendler y Ora Lassila. «The semantic web». En: *http://www.scientificamerican.com/article.cfm?id=the-semantic-web* (2014).

[Cad+06] A. Cadu y col. «Non-invasive glucose monitoring in patients with diabetes: a novel system based on impedance spectroscopy». En: *Biosensors and Bioelectronics* (2006).

[Con04] World Wide Web Consortium. «SWRL: A Semantic Web Rule Language Combining OWL and RuleML». En: (2004). URL: http://www.w3.org/Submission/SWRL/.

[CRC07] Dalla Man C., R. A. Rizza y C. Cobelli. «Meal simulation model of the glucose-insulin system». En: *IEEE Trans. Biomed. Eng* (2007).

[CRM06] Larizza C, Bellazzi R y Stefanelli M. «The M2DM Project–the experience of two Italian clinical sites with clinical evaluation of a multi-access service for the management of diabetes mellitus patients.» En: *Methods Inf Med* 45 (2006), págs. 79-84.

[Dom+06] David Dominguez y col. «PIPS: An Integrated Environment for Health Care Delivery and Healthy Lifestyle Support». En: *ECAI - 4th Workshop on Agents Applied in Health Care* (2006).

[DTU13] DTU. «The Artificial Pancreas. Diabetes and Control». En: (2013). URL: www.imm.dtu.dk/~jbjo/diabetescontrol.html.

[Eve+14a] Alison B. Evert y col. «Nutrition Therapy Recommendations for the Management of Adults With Diabetes». En: *Diabetes Care* (2014).

[Eve+14b] Alison B. Evert y col. «Nutrition Therapy Recommendations for the Management of Adults With Diabetes in Diabetes Care». En: *Diabetes Care* (2014).

[Gag+06] J.J. Gagliardino y col. «PROPAT: a study to improve the quality and reduce the cost of diabetes care.» En: *Diabetes Res Clin Pract.* 72 (2006), págs. 284-91.

[Gag13] Juan J Gagliardino. «Diabetes in Argentina: cost and management of diabetes and its complications and challenges for health policy». En: *Globalization and Health* (2013).

[Gru93] Thomas R. Gruber. «A Translation Approach to Portable Ontology Specifications». En: *Knowledge Acquisition Special issue: Current issues in knowledge modeling* (1993).

[HCD06] R. nda M. E. Wilinska Hovorka, L. J. Chassin y D. B. Dunger. «Roadmap to the artificial pancreas. Diabetes Research and Clinical Practice». En: *Diabetes Research and Clinical Practice* (2006).

[Her+12] P. Herrero y col. «Simple Robust Method for Estimating the Glucose Rate of Appearance from Mixed Meals.» En: *Journal of Diabetes Science and Technology* (2012).

[IDF13] IDF. «IDF Diabetes Atlas 6 th Edition». En: *Brussels, Belgium: International Diabetes Federation* (2013).

[J+05] Cantais J. y col. «An example of food ontology for diabetes control.» En: *ISWC - workshop on Ontology Patterns for the Semantic Web, Ireland.* (2005).

[JO09] Richalet J. y D. O'Donovan. «Predictive Functional Control - Principles and Industrial Applications». En: *Springer* (2009).

[JO98] Jaremko J. y Rorstad O. «Advances toward the implantable artificial pancreas for treatment of diabetes». En: *Diabetes Care* (1998).

[L99] Ljung L. «System Identification, Theory For The User». En: *Prentice Hall: New Jersey* (1999).

[Lee+08] Chang-Shing Lee y col. «Intelligent Ontological Agent for Diabetic Food Recommendation». En: *IEEE International Conference on Fuzzy Systems* (2008).

[LK07] Huan-Chung Li y Wei-Min Ko. «PIPS: An Integrated Environment for Health Care Delivery and Healthy Lifestyle Support». En: *International Conference on Machine Learning and Cybernetics, Hong Kong* (2007).

[M+07] Rigla M y col. «A telemedicine system that includes a personal assistant improves glycemic control in pump-treated patients with type 1 diabetes.» En: *J Diabetes Sci Technol* 1 (2007), págs. 505-10.

[M15] Bora J. Benegui M. «Utilización de Ontologías para Capturar el Conocimiento Experto en Modelado del Sistema Endocrino Humano y su Control». En: *Facultad de Ciencias Exactas, Ingeniería y Agrimensura, en colaboración con CIFASIS-CONICET. Rosario, Argentina* (2015).

[Med13] Medtronic. «Empezando con Monitoreo continuo de glucosa para el sistema MiniMed 530 G». En: (2013). URL: http://www.medtronicdiabetes.com/sites/default/files/library/download-library/workbooks/MP9501486-111_a.pdf.

[OCJ09] N. S. Oliver, C. Toumazouand A. E. G. Cass y D. G. Johnston. «Glucose sensors: a review of current and emerging technology». En: *Diabetic Medicine* (2009).

[P+14] Rullo P. y col. «Model predictive control to ensurehigh quality hydrogen production for fuel cells». En: *International Journal of Hydrogen Energy* (2014).

[Pro] Protege. «A free, open-source ontology editor and framework for building intelligent systems». En: (). URL: http://protege.stanford.edu/.

[TMP07] A. Tura, A. Maran y G. Pacini. «Non-invasive glucose monitoring: Assessment of technologies and devices according to quantitative criteria.» En: *Diabetes Research and Clinical Practice* (2007).

[VPH10] Bondia J.and J. Vehí, C. C. Palerm y P. Herrero. «El páncreas artificial: Control automático de infusión de insulina en diabetes mellitus tipo 1». En: *Rev. Iberoam. Autom. In.* (2010).

[W+10] Marquardt W. y col. «OntoCAPE A Re-Usable Ontology for Chemical Process Engineering». En: *Berlin: Springer RWTHedition* (2010).