TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2012

81

Nelia Lasierra Beamonte

An ontology-driven architecture for data integration and management in home-based telemonitoring scenarios

Departamento

Instituto de Investigación en Ingeniería [I3A]

Director/es

Alesanco Iglesias, Álvaro
García Moros, José

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

Universidad Zaragoza
1542

Tesis Doctoral

AN ONTOLOGY-DRIVEN ARCHITECTURE FOR DATA INTEGRATION AND MANAGEMENT IN HOME-BASED TELEMONITORING SCENARIOS

Autor

Nelia Lasierra Beamonte

Director/es

Alesanco Iglesias, Álvaro
García Moros, José

UNIVERSIDAD DE ZARAGOZA
Instituto de Investigación en Ingeniería [I3A]

2012

University Zaragoza 1542

Aragón Institute of Engineering Research (i3A)

An ontology-driven architecture for data integration and management in home-based telemonitoring scenarios

Nelia Lasierra Beamonte

Supervisors:

Álvaro Alesanco Iglesias and José García Moros

Aragón Institute of Engineering Research (I3A)
Communications Technology Group (GTC)
University of Zaragoza

PhD Dissertation
Zaragoza, 2012
Biomedical Engineering Doctoral Program

Acknowledgments

Completing a PhD is a tough task that requires hard work and effort. This is often an overwhelming but also wonderful experience which I would not have been able to complete without the aid and support of countless people. Thus, it is my pleasure to thank the many people who made this thesis possible.

I would like to deeply thank **Álvaro**, one of my supervisors, for his guidance, encouragement and support over years. This dissertation would not have been possible without your advices, your ability to approach the research and your infinite patient to discuss ideas and difficulties. I am also very grateful to **Josechu**, my other supervisor, for providing me with the necessary support during this research. I would like to thank him also for his valuable advices, comments, great patience and support throughout these years.

I would like to express my gratitude to **Declan** and **Andre** for welcoming me so warmly in their respective research groups. I would like to thank them also the chance they gave me of learning and working in new research lines. It is a great pleasure to thank everyone from **KDEG** and the **UVic** who contributed to make my research stages such wonderful experiences.

A warm thank you to all my colleagues from the **University of Zaragoza** from whom I have learned so much and who with I have shared hours of fun and thesis work. A special thank to **Jesús**, **Carlos** and **Eva**. I am also indebted to the students I had the pleasure to work with and who trusted me to guide their master thesis. My sincere gratitude to **Marcos**, **Andreas**, **Verónica**, **Juan Carlos** and **Fernando**.

It is a great pleasure to thank **Rosa**, **Yolanda**, **Carmen**, **Eva**, **Sara** and the rest of clinical personnel that have contributed in this thesis and in our research.

I want to express my profound gratitude to **Melike** and **Petra** for accepting to be external reviewers of my PhD dissertation draft. I would like to thank also to all the committee members for accepting to evaluate my research.

I am truly indebted and thankful with my friends **Patricia** and **Laura** for so many years of fun, support and love. A warm thank you to my telecom friends **Leti**, **Blanca**, **Laura** and **Claudia** for their sharp conversations, friendship and valuable advices.

My greatest appreciation to all my **family** who have shown much love and support. In particular, a special thank to my sister, **Ana**, who has always supported me. Last but not least, I would like to thank my parents, **Marisa** and **José Manuel** for their unconditional support, for instilling in me confidence and a drive for pursuing my PhD. To them I dedicate this Thesis.

Agradecimientos

El desarrollo de una tesis doctoral require de trabajo y esfuerzo. Superar los retos y las dificultades es una tarea que aún se complica mucho más en lo que voy a llamar "momentos tesis". *Y es que el camino no siempre está iluminado.* Sin embargo, para lograr completar la tesis y superar así los obstáculos también hacen falta apoyo, confianza y ánimo. Y por supuesto, las alegrías de los logros conseguidos y los buenos momentos también son compañeros de viaje. Es claro por tanto que el mérito de la tesis no es solo mio. La culminación de este trabajo no hubiera sido posible sin mucha gente que me ha apoyado, animado, soportado y sufrido durante todos estos años. Es por ello un placer para mi aprovechar este espacio para agradecerles el haber hecho realidad esta tesis.

En primer lugar quiero agradecer a **Álvaro**, uno de mis directores, por darme la oportunidad de hacer este trabajo. No tengo la menor duda de que hoy no estaría escribiendo esto sino fuera por sus buenos consejos, su orientación, sus palabras de ánimo y la confianza que siempre me ha dado para seguir adelante. *Gracias por compartir conmigo tu trabajo y tus ideas, por darme siempre tu apoyo y por compensar la balanza cuando los momentos de agobio pesaban demasiado.* Debo agrader de manera muy especial también a mi otro director, **Josechu**, quien ha puesto a mi disposición siempre todos los medios necesarios para el desarrollo de esta tesis y me ha demostrado siempre su apoyo y confianza. *Gracias por tus comentarios, consejos, orientación y tu eficacia y disposición siempre para el trabajo*

Quiero aprovechar para agradecer a **Declan** y **Andre** el que me acogieran tan bien en sus respectivos grupos de investigación. Debo agradecerles además la oportunidad que me brindaron de aprender y trabajar en nuevas líneas de investigación. No puedo olvidarme de todas las personas que contribuyeron a que mi estancia en Dublín y en Victoria fuesen tan buenas experiencias. Gracias a mis compañeros de **KDEG** y de la **UVic**.

Muchos de los "momentos tesis" han sido compartidos con mis compañeros del grupo de investigación y con mis compañeros del laboratorio 2.05. Con ellos he vivido sin duda también gran parte de los buenos y tengo que agradecerles por sus consejos, su trabajo y sus ánimos. En especial a **Jesús**, a **Carlos** y a **Eva** con quienes he compartido muchas horas de tesis y conversaciones. Quiero extender este agradecimiento a los compañeros de la **Universidad de Zaragoza** con los que he compartido el laboratorio 2.03 y los desayunos de discusiones variadas durante todos estos años. Un agradecimiento especial a **Marcos**, **Andreas**, **Verónica**, **Juan Carlos** y **Fernando**, que confiaron en mi para que guiara el desarrollo de sus proyectos fin de carrera y trabajos de master. Espero que aprendieran conmigo tanto como yo hice con ellos.

No quiero olvidarme de todos los profesionales médicos que han contribuido en este trabajo. Muchas gracias a **Rosa**, a **Yolanda**, a **Carmen**, a **Eva**, a **Sara** y tantos otros que de forma voluntaria han participado en nuestras investigaciones.

Muchas gracias también a **Melike** y a **Petra** por revisar esta tesis. Y también a todas las personas que han formado parte del tribunal de evaluación.

A mis amigas **Patricia** y **Laura**, a las que conozco desde hace ya unos años y han estado apoyandome, animandome y a mi lado siempre que las he necesitado. También tengo que agradecer a mis amigas de "la fila" (**Leti**, **Blanca**, **Laura** y **Claudia**) que aunque dispersas por el mundo y nuevas tecnologías mediante, siempre tienen palabras de ánimo y buenos consejos. Muchas gracias también a **Edurne** y a todos los que durante estos años compartieron conmigo dias de conversación y diversión.

Y por último el mayor de los agradecimientos a mis primos, tios y **familia** en general. Gracias a todos (que son muchísimos y no tengo espacio para nombrarlos a todos) por el apoyo, el ánimo y la comprensión. Gracias en particular a mi hermana, **Ana**, por su apoyo no importa cual sea el pais al que vaya y su confianza. Aunque de pocas palabras, sin duda incondicional. Gracias a **mis abuelos**, otros incondicionales. Y por supuesto, gracias a mis padres, **Marisa** y **José Manuel** que nunca han dejado de confiar en mi y me han apoyado siempre. Gran parte de este trabajo es también suyo y a ellos se lo dedico. *Gracias por enseñarme a salir de la piscina pequeña.*

Abstract

The shift from traditional medical care to the use of new technology and engineering innovations is nowadays an interesting and growing research area mainly motivated by a growing population with chronic conditions and disabilities. By means of information and communications technologies (ICTs), telemedicine systems offer a good solution for providing medical care at a distance to any person in any place at any time. Although significant contributions have been made in this field in recent decades, telemedicine and in e-health scenarios in general still pose numerous challenges that need to be addressed by researchers in order to take maximum advantage of the benefits that these systems provide and to support their long-term implementation. The goal of this research thesis is to make contributions in the field of home-based telemonitoring scenarios. By periodically collecting patients' clinical data and transferring them to physicians located in remote sites, patient health status supervision and feedback provision is possible. This type of telemedicine system guarantees patient supervision while reducing costs (enabling more autonomous patient care and avoiding hospital overflows). Furthermore, patients' quality of life and empowerment are improved. Specifically, this research investigates how a new architecture based on ontologies can be successfully used to address the main challenges presented in home-based telemonitoring scenarios. The challenges include data integration, personalized care, multi-chronic conditions, clinical and technical management. These are the principal issues presented and discussed in this thesis.

The proposed new ontology-based architecture takes into account both practical and conceptual integration issues and the transference of data between the end points of the telemonitoring scenario (i.e, communication and message exchange). The architecture includes two layers: 1) a conceptual layer and 2) a data and communication layer. On the one hand, the conceptual layer based on ontologies is proposed to unify the management procedure and integrate incoming data from all the sources involved in the telemonitoring process. On the other hand, the data and communication layer based on web service technologies is proposed to provide practical back-up to the use of the ontology, to provide a real implementation of the tasks it describes and thus to provide a means of exchanging data.

This architecture takes advantage of the combination of ontologies, rules, web services and the autonomic computing paradigm. All are well-known technologies and popular solutions applied in the semantic web domain and network management field. A review of these technologies and related works that have made use of them is presented in this thesis in order to understand how they can be combined successfully to provide a solution for telemonitoring scenarios.

The design and development of the ontology used in the conceptual layer led to the study of the autonomic computing paradigm and its combination with ontologies. In addition, the OWL (Ontology Web Language) language was studied and selected to express the required knowledge in the ontology while the SPARQL language was examined for its effective use in defining rules. As an outcome of these research tasks, the HOTMES (Home Ontology for inTegrated Management in tElemonitoring Scenarios) ontology, presented in this thesis, was developed. The combination of the HOTMES ontology with SPARQL rules to provide a flexible solution for personalising management tasks and adapting the methodology for different management purposes is also discussed.

The use of Web Services (WSs) was investigated to support the exchange of information defined in the conceptual layer of the architecture. A generic ontology-based solution was designed to integrate data and management procedures in the data and communication layer of the architecture. This is an innovative REST-inspired architecture that allows information contained in an ontology to be exchanged in a generic manner. This layer structure and its communication method provide the approach with scalability and re-usability features.

The application of the HOTMES-based architecture has been studied for clinical purposes following three simple methodological stages described in this thesis. Data and management integration for context-aware and personalized monitoring services for patients with chronic conditions in the telemonitoring scenario are thus addressed. In particular, the extension of the HOTMES ontology defines a *patient profile*. These profiles in combination with individual rules provide clinical guidelines aiming to monitor and evaluate the evolution of the patient's health status evolution. This research implied a multi-disciplinary collaboration where clinicians had an essential role both in the ontology definition and in the validation of the proposed approach. *Patient profiles* were defined for 16 types of different diseases.

Finally, two solutions were explored and compared in this thesis to address the remote technical management of all devices that comprise the telemonitoring scenario. The first solution was based on the HOTMES ontology-based architecture. The second solution was based on the most popular TCP/IP management architecture, SNMP (Simple Network Management Protocol).

As a general conclusion, it has been demonstrated that the combination of ontologies, rules, WSs and the autonomic computing paradigm takes advantage of the main benefits that these technologies can offer in terms of knowledge representation, workflow organization, data transference, personalization of services and self-management capabilities. It has been proven that ontologies can be successfully used to provide clear descriptions of managed data (both clinical and technical) and ways of managing such information. This represents a further step towards the possibility of establishing more effective home-based telemonitoring systems and thus improving the remote care of patients with chronic diseases.

Resumen y Conclusiones

En los últimos años, el aumento de la población con enfermedades crónicas y su progresivo envejecimiento han provocado, entre otros, un incremento de la demanda de atención sanitaria y de los costes que esto conlleva. Este hecho ha impulsado un cambio en la prestación de servicios médicos mediante el diseño y desarrollo de nuevas alternativas tecnológicas. La innovación e investigación en Tecnologías de la Información y Comunicaciones (TICs) aplicadas a e-Salud juegan por tanto un papel muy destacado en el momento actual. En concreto, la telemonitorización domiciliaria es actualmente uno de los ámbitos de mayor interés en e-Salud. Este tipo de sistemas de telemedicina permite que los pacientes, desde sus casas, recojan de forma periódica datos clínicos e información de su estado de salud y los envíen al médico encargado de su supervisión situado en un centro hospitalario remoto. La utilización de estos sistemas reporta numerosas ventajas tanto a pacientes como a los centros sanitarios. Por un lado, ayudan a mejorar la calidad de vida de los pacientes y su independencia. Por otro, contribuyen a evitar el incremento de los costes sanitarios y reducir listas de espera.

Uno de los retos actuales y factor crítico de éxito es garantizar la interoperabilidad e integración de los diferentes sistemas y fuentes de información que participan de manera conjunta para dar soporte al proceso de telemonitorización del paciente. A pesar de las contribuciones y propuestas realizadas durante la última década en este campo quedan todavía problemas que deben solucionarse para favorecer el éxito e implantación a largo plazo de un sistema de telemonitorización. Por ello, el objetivo general de esta tesis ha sido investigar y realizar contribuciones en el ámbito de los escenarios de telemonitorización domiciliaria. En concreto, el principal objetivo ha sido estudiar la aplicación de tecnologías de representación de conocimiento como son las ontologías para solucionar los principales retos de gestión e integración que se plantean en dichos escenarios. La integración de datos, la personalización de servicios, la gestión de datos técnicos de los dispositivos que integran el escenario así como la gestión de los datos clínicos de pacientes con condiciones multi-crónicas son los temas más relevantes que se analizan y discuten en esta tesis.

En primer lugar, esta tesis presenta una arquitectura basada en ontologías para dar soporte de manera conjunta a servicios clínicos y técnicos en escenarios de telemonitorización domiciliaria. Esta arquitectura contempla tanto desde un punto de vista conceptual como desde un punto de vista práctico aspectos de integración y transferencia de datos entre los dos extremos que conforman la arquitectura del sistema de telemonitorización y se basa en la combinación de ontologías, reglas, servicios web y el paradigma de computación autonómica. Por un lado, la capa conceptual se basa en el diseño y desarrollo de una ontología para unificar el proceso de gestión e integrar los datos proporcionados por las diferentes fuentes que participan en el proceso de telemonitorización. Por otro lado, la capa de comunicación y datos está basada en tecnologías web. Esta capa da soporte a la implementación de las tareas que se describen en la ontología y extiende al campo físico los beneficios que se derivan de su utilización. Esta es sin duda una importante contribución de esta tesis.

Esta tesis doctoral propone el desarrollo de una ontología para dar soporte a la capa conceptual de la arquitectura cuyo diseño se basa en el paradigma de computación autonómica. Esta ontología ha sido denominada HOTMES (Home Ontology for inTegrated Management in tElemonitoring Scenarios) y proporciona una representación de las tareas de monitorización, análisis, planificación y ejecución. Estas son las tareas que marcan el flujo de gestión y contemplan además en su definición una descripción de la información gestionada. Además, se propone y discute la utilización del lenguaje SPARQL para la definición de reglas, que aplicadas sobre el modelo ontológico permitan individualizar los procesos definidos en ella. Por otro lado, esta tesis investiga la utilización de servicios web para el desarrollo de la capa de comunicación e intercambio de datos que de soporte físico a la información y procesos definidos en la capa conceptual. La solución que se propone está basada en el paradigma REST y constituye una innovadora arquitectura que permite intercambiar la información descrita en la capa conceptual de manera genérica. Obtenemos así por un lado, un modelo conceptual con una semántica clara, que permite tener una visión común para todas las entidades involucradas en el proceso de telemonitorización y que puede ser aplicado para definir perfiles de un amplio rango de casos de uso. Por otro lado, el servicio web que se incluye en la arquitectura describe de manera genérica los recursos intercambiados como son la ontología y las instancias de la misma. De esta forma se consigue una arquitectura flexible y escalable que puede ser utilizada para otras aplicaciones basadas en el uso de ontologías.

Las posibilidades y ventajas que ofrece esta arquitectura han sido demostradas estudiando su utilización para gestionar y transferir información tanto relativa a la monitorización de pacientes con diferentes morbilidades (información clínica) como relativa al funcionamiento de los dispositivos involucrados en el escenario (información técnica). Para estudiar la aplicación de la arquitectura propuesta a la gestión de datos clínicos mediante la extensión de la ontología desarrollada, esta tesis doctoral propone una metodología compuesta por 3 fases. Siguiendo esta metodología se han ido completando paso a paso las tareas necesarias para conseguir integrar la gestión de datos clínicos y la personalización de servicios para pacientes con condiciones crónicas. En concreto, la extensión de la ontología HOTMES describe lo que se ha denominado como *patient profile* o perfil de paciente. Un perfil de paciente puede verse como un contenedor con una estructura organizada donde se recoge toda la información de telemonitorización involucrada en el seguimiento de un paciente, es decir, tanto las pautas o guías clínicas para la monitorización del paciente como los resultados obtenidos de ella. Para su desarrollo se han tenido en cuenta tanto aspectos clínicos (a través de una colaboración multi-disciplicinar con personal médico) como aspectos relativos a la interoperabilidad de dispositivos médicos siendo la norma ISO/IEEE 11073 una de las fuentes utilizadas para la descripción del modelo. Esto facilita la posterior integración de datos procedentes de ambas fuentes. Gracias a dicha colaboración con personal médico especializado en atención primaria esta tesis reflexiona como la ontología propuesta puede utilizarse para describir perfiles de paciente reales. En concreto se han propuesto y validado 16 perfiles de paciente. La inclusión de reglas sobre el modelo permite individualizar los procesos de gestión y control del paciente en función de las necesidades que cada uno plantee.

Por último, esta tesis explora y compara dos alternativas para la gestión técnica de los dispositivos que integran la arquitectura física del sistema de telemonitorización, es decir, los dispositivos médicos y demás elementos de la red de comunicaciones. La primera de las soluciones se basa en la extensión de la ontología HOTMES (a través de la metodología de 3 fases previamente presentada) y adaptación de la arquitectura REST diseñada para el intercambio de mensajes e integración física de los datos. La segunda solución se basa en utilizar la arquitectura *de facto* para la gestión de redes TCP/IP, es decir, la arquitectura de gestión de redes SNMP. Por un lado, la primera de la soluciones proporciona mejores prestaciones en términos de mobilidad e independencia aportando funcionalidades de auto-gestión al dispositivo concetrador de datos situado en casa del paciente. Por otro lado, la solución basada en la arquitectura SNMP ofrece también buenas prestaciones para la gestión remota de los dispositivos mencionados. Esta solución basada en el desarrollo de un agente SNMPv3 (junto con su MIB) es también una solución escalable que ofrece buenas prestaciones en términos de simplicidad, seguridad y consumo de recursos.

Esta tesis profundiza y reflexiona por tanto, acerca de la utilización de ontologías y reglas para proporcionar una solución de gestión integrada en escenarios de telemonitorización domiciliaria teniendo en cuenta aspectos cruciales como son la interoperabilidad de dispositivos médicos, la información contextual proporcionada por los pacientes y que cada paciente presenta unas necesidades de monitorización individuales. La combinación de las ontologías con el paradigma de computación autonómica es una propuesta de gran interés que proporciona una manera simple, clara y fácil de entender el proceso de control y evaluación de los pacientes. Esta combinación consituye una innovación en el dominio de la e-Salud. Las contribuciones presentadas en esta tesis doctoral permiten dar un paso hacia delante en el desarrollo de sistemas de telemonitorización domiciliaria para la asistencia remota de personas con enfermedades crónicas. Un paso más por tanto para solucionar un problema que afecta a una sociedad cada vez más envejecida y que demanda cada día, más atención sanitaria.

Scientific Contributions

Next peer-reviewed scientific publications have been derived from this research. The methodology and results presented in this thesis are mainly based on them.

<u>Publications in International Journals (JCR indexed)</u>

N. Lasierra, A. Alesanco, Y. Gilaberte, R. Magallón, and J. García. Lessons learned after a three-year store and forward teledermatology experience using internet: Strengths and limitations. *Int J Med Informat*, 81:332-343, 2012.

N. Lasierra, A. Alesanco, and J. García. An SNMP based solution to enable remote ISO/IEEE 11073 technical management. *IEEE Trans Inform Tech Biomed*, 16(4):709-719, 2012

C. Campos, E. Caudevilla, A. Alesanco, **N. Lasierra**, O. Martinez, J. Fernández and J. García. Setting up a telemedicine service for remote real-time video-EEG consultation in la Rioja (Spain). *Int J Med Informat*, 81(6):404-414, 2012.

N. Lasierra, A. Alesanco, D. O'Sullivan and J. García. An autonomic ontology-based approach to manage information in home-based scenarios: from theory to practice. *Submitted (under Review)*

N. Lasierra, A. Alesanco, S. Guillén and J. García. A new approach to supervise remote patients with chronic conditions using ontologies and the autonomic computing paradigm. *Submitted (under Review)*

<u>Publications in International Conferences</u>

N. Lasierra, A. Alesanco J. García and D. O'Sullivan. Data Management in Home Scenarios Using an Autonomic Ontology-Based Approach. *9th IEEE Workshop on Managing Ubiquitous Communications and Services part of PerCom 2012*, Lugano, Switzerland, March 2012.

N. Lasierra, A. Alesanco and J. García. An Ontology approach to manage individual patient profiles in home-based telemonitoring scenarios. *10th IEEE Int. Conf. on Information Technology and Applications in Biomedicine (ITAB'10)*, Corfu (Greece), November 2010.

N. Lasierra, A. Alesanco and J. García. Home-Based Telemonitoring Architecture To Manage Health Information Based on Ontology Solutions. *10th IEEE Int. Conf. on Information Technology and Applications in Biomedicine (ITAB'10)*, Corfu (Greece), November 2010.

N. Lasierra, A. Alesanco, C. Campos, E. Caudevilla, J. Fernández and J. García. Experience of a Real-time Tele-EEG Service *Proc. of the 31st Int. Conf. of the IEEE Eng. in Med. and Biol. Soc, Minneapolis (US)*, Septiembre 2009, Proc. pp. 5211-5214.

E. Caudevilla, C. Campos, **N. Lasierra**, A. Alesanco, J. Fernández and J. García. Experiencia de tele-electroencefalografía en España: evaluación del impacto sobre el proceso organizativo. *III Congreso Latinoamericano en Informática Médica, Buenos Aires (Argentina)*, August 2008.

Publications in National Conferences

N. Lasierra, A. Alesanco, S.Guillén, J. García. Tres fases para el diseño y desarrollo de un sistema orientado a la monitorización remota de pacientes basado en ontologías *XXX Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*, San Sebastián (Spain), November 2012.

N. Lasierra, A.Muñoz, A. Alesanco, J.Escayola, J. García. Una solución SNMP para la gestión técnica de dispositivos médicos ISO/IEEE 11073. *XXIX Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*, Cáceres (Spain), November 2011.

N. Lasierra, A. Alesanco, J. García. Diseño de una ontología para la gestión de perfiles de pacientes en escenarios de telemonitorización domiciliaria. *XXVIII Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*. Madrid (Spain), November 2010.

N. Lasierra, A. Alesanco, J. García. Arquitectura de telemonitorización domiciliaria para la gestión de información clínica basada en ontologás. *XXVIII Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*, Madrid (Spain), November 2010.

N. Lasierra, M. López, J. García, A. Alesanco. Desarrollo de un agente SNMPv3 para modelado de usuario en entornos LAN. *VIII Jornadas de Ingeniería Telemática JITEL* Cartagena (Spain), September 2009, Proc. pp 397-404.

N. Lasierra, C. Campos, E. Caudevilla, A. Alesanco, J. Fernández, G. Palacios, J. García. Implantación y evaluación de un servicio de teleelectroencefalografía en tiempo real. *XXVI Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*, Valladolid (Spain), October 2008 Proc. pp 65-68.

N. Lasierra, A. Alesanco, Y. Gilaberte, N. González , M.A. Sanchez , R. Magallón , J. García. Diseño y evaluación de un sistema de Teledermatología basado en tecnologías Web. *XXVI Congreso Anual de la Sociedad Española de Ingeniería Biomédica (CASEIB)*, Valladolid (Spain). October 2008, Proc. pp 57-60.

Abbreviations

4KCARE	Knowledge-Based HomeCare eServices for an Aging Europe
ABox	Assertional box
AJAX	Asynchronous JavaScript And XML
AT	Analysis Task
BAN	Body Area Network
BMI	Body Mass Index
BP	Blood Pressure
BPEL4WS	Business Process Execution Language for Web Services
BPML	Business Process Modelling Language
BPMN	Business Process Modelling Notation
CA	Certificate Authority
CBC	Cipher Block Chaining
CDSS	Clinical decision support systems
CE	Compute Engine
CEN	European Commitee for Standardization
CHA	Continua Health Alliance
CHF	Congestive Heart Failure
CMIP	Common Management Information Protocol
COPD	Chronic Obstructive Pulmonary Disease
CORBA	Common Object Request Broker Architecture
DAML	DARPA Agent Markup Language

DES	Data Encryption Standard
DIM	Domain Information Model
DL	Description logic
DLP	Dyslipidemia
DM	Diabetes Mellitus
EEG	Electroencephalography
EHR	Electronic Health Record
EI	Environmental Information
EM	External Manager
ET	Execution Task
EU	European Union
$F-Logic$	Frame Logic
$FEV1$	Forced Expiratory Volume in one second
FHC	Calahorra Hospital Foundation
FOL	First-order logic
FSM	Finite-state machine
GP	General practitioner
GUI	Graphical User Interface
HCO	Health Care Organizations
HF	Heart Failure
HG	Home Gateway
$HL7$	Health Level Seven
$HMAC$	Hash-based Message Authentication Code
$HOTMES$	Home Ontology for inTegrated Management in homE-based Scenarios
$HR-MIB$	Host Resources MIB
HSP	San Pedro Hospital
HTA	Hypertension

$HTTP$	Hypertext Transfer Protocol
$HTTPS$	Hypertext Transfer Protocol Secure
IBM	International Business Machines Corporation
ICT	Information and Communications Technologies
IHD	Ischemic Heart Disease
IHE	Integrating the Healthcare Enterprise
IP	Internet Protocol
ISO	International Organization for Standarization
$JDBC$	Java Database Connectivity
$KBase$	Knowledge Base
KIF	Knowledge Interface Format
MAP	Mean Arterial Pressure
$MAPE$	Monitor Analyze Plan Execute
$MAST$	Method for Assessment of Telemedicine
MCD	Multi-Chronic Disease
MD	Medical Device
$MD - MIB$	Medical Device-MIB
$MD5$	Message Digest 5
MDS	Medical Device System
MIB	Management Information Base
MP	Management Profile
MT	Monitoring Task
$NETCONF$	Network Configuration Protocol
OCL	Object Constraint language
$OCML$	Operational Conceptual Modeling Language
OID	Object Identifier
OIL	Ontology Inference Layer

OO	Object Oriented
OWL	Ontology Web Language
$OWL-S$	Semantic Markup for Web Services
$PACS$	Picture archiving and communication system
PAN	Personal Area Network
PDU	Protocol Data Unit
$PHDWG$	PHD Working Group
PKI	Public-Key Infrastructure
PT	Planning Task
QLI	Qualitative Information
QNI	Quantitative Information
RDF	Resource Description Framework
$RDFS$	RDF Schema
$RDQL$	RDF Data Query Language
$REST$	Representational state transfer
$RMON$	Remote MONitoring
RPC	Remote Procedure Call
$RT-SA$	Real Time-Simple Array
$Rule-ML$	Rule Markup Language
$SeRQL$	Sesame RDF Query Language
$SHA-1$	Secure Hash Algorithm 1
$SHOE$	Simple HTML Ontology Extension
$SMIv2$	Structure of Management Information Version 2
$SMTP$	Simple Mail Transfer Protocol
$SNMP$	Simple Network Management Protocol
$SNOMED-CT$	Systematized Nomenclature of Medicine Clinical Terms
SOA	Service-Oriented Architecture

SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Data Encryption Standard
SSL	Secure Socket Layer
SWRL	Semantic Web Rule Language
TBox	Terminological box
TCP	Transmision Control Protocol
TLS	Transport Layer Security
TMN	Telecommunication Management Network
TS	Telemonitoring Server
UDDI	Universal Description, Discovery and Integration
UDP	User Datagram Protocol
UML	Unified Modeling Language
UMLS	Unified Medical Language System
UNA	Unique Name Assumption
URI	Uniform Resource Identifier
USM	User-Based Security Model
VBAC	View-Based Access Control
VC	Vital Constant
WHO	World Health Organization
WS	Web Service
$WS - Security$	Web Services Security
WSDL	Web Service Description Language
WSMO	Web Service Modeling Ontology
$X73$	ISO/IEEE 11073 family of standards
$X73$	ISO/IEEE 11073 Personal Health Device communication
$X73 - PHD$	ISO/IEEE 11073 Personal Health Device communication

$X73 - PoC$	CEN/ISO/IEEE 11073 Health Informatics Point-of-Care
XML	eXtensible Markup Languange
XOL	Ontology Exchange Language

Index

List of Figures

List of Tables

Chapter 1

Introduction

Advances in information and communication technologies have enabled the development of new systems to deliver medical services at a distance. These telemedicine systems are now a reality providing anyone with medical care at any place and at any time. However, while significant contributions have been made in this field in recent decades, telemedicine and e-health scenarios in general still present numerous challenges that should be addressed by researchers in order to completely exploit the advantages that these systems provide and thus support their long-term implementation. This chapter presents an overview of this thesis which aims to make a contribution to the e-health domain. Data integration, interoperability, personalized care, multi-chronic conditions, and clinical and technical management are the main topics and challenges presented and discussed.

1.1 Introduction and Motivation

1.1.1 Telemedicine and telemedicine scenarios

The shift from traditional care to the use of new technology and engineering innovations is nowadays an interesting and growing research area mainly motivated by trends in world demographics (a growing population with chronic conditions and disabilities). More than a third of the population of Europe is currently affected with a chronic disease [1] and, according to UN (United Nations) population forecasts, the proportion of the population aged 60 or over is expected to reach almost 35% of the total population by 2050 [2]. This dramatic process of population aging and the evolution of the population pyramid in Europe (currently static with low fertility and mortality, with increasingly lower percentages of younger and higher percentages of older people) will have a serious economic effect in terms of social security, health insurance and pensions systems. This is a big issue for governments to face. All the evidence shows that looking for alternative technological solutions such as telemedicine systems is necessary.

Telemedicine can be defined as the practice of clinical health care at a distance by means of information and communications technologies (ICT). These systems today are widely used for delivering medical services to remote areas, especially in countries where there is no easy access to medical care due to geographical barriers or a widely dispersed distribution of physicians. Using telemedicine systems provides benefits for patients in terms of accessibility and quality of life while also having a positive impact on hospital waiting lists and financial investment. For this reason, the utilization of these health services is being increased and they are frequently adopted as regular services available at any place at any time. Telemedicine applications play an important role under the e-Health umbrella by supporting healthcare practice using ICT in both clinical and non-clinical services such as health education. Furthermore, e-health can be described as a way of thinking or as an attitude towards improving health care [3].

These eHealth services can be applied in various different areas of daily medical practice. In addition, three main communication scenarios can be distinguished for their usage:

- **Scenario 1) Hospital - Hospital communication**: Telemedicine systems can be useful for obtaining second opinions or advice from other physicians and also for making easier the access to specific medical tests that can not be performed in the hospital where the corresponding physician or the patient are located. Tele-EEG (electroencephalography) or Tele-ictus are examples of systems applied in this scenario.

- **Scenario 2) Primary Care Centre - Hospital communication**: Telemedicine systems are valuable tools that can be used for supporting diagnoses in primary care centres. Tele-dermatology, tele-retinography or tele-echocardiography are the most popular types of systems implemented in this scenario [4, 5, 6].

- **Scenario 3) Home site - Health Care Site communication**: These systems, known as home-based telemonitoring systems, are oriented to make easier the supervision of chronic and elderly patients from their homes. Indeed, the development of these technology solutions for monitoring patients at home is nowadays one of the most popular research areas within the e-Health scope which is being expanded to the use of mobile software systems [7, 8].

Different challenges and problems arise during the setting-up of telemedicine systems in all these scenarios. Going one step further, it could be said that these challenges and problems arise during different phases and in different areas involved in the setting up process. As each scenario has its own particular conditions and demands, specific research needs to be done to solve the problems according to the scenario and the application. Nevertheless, in general the design, development and implementation of a telemedicine system can be carried out following a common methodology which should be executed according to the specific and particular conditions presented in each scenario. This methodology can be applied in general for the three telemedicine scenarios presented and in particular for telemonitoring to which major contributions presented in this thesis have been made.

1.1.2 Methodology for setting-up a telemedicine system

The great majority of telemedicine studies focus on showing these systems as valuable tools for medical practice but they do not give much consideration to the long-term implementation of these systems within the clinical routine of healthcare centres. The alignment of new information systems with the objectives of Health Care Organizations (HCO) is a key factor for their introduction and long-term success. Alignment theories should thus be taken into account when examining the long-term implementation of a telemedicine system [9, 10]. These theories suggest that as long as a telemedicine system contributes to the achievement of HCO objectives following defined strategic plans, it can be aligned with hospital strategy and its long-term set-up will be possible. Bush et. al. in [11] suggested five steps for aligning information systems with organizational objectives: Step 1) identify organization objectives; Step 2) identify organization strategy to cope with the objectives; Step 3) envision an information system to support organization strategies; Step 4) gain approval of health centres at a high-level to implement the system; Step 5) implement the system (see Figure 1.1).

In order to envision and implement a telemedicine system in accordance with Step 3, it should be taken into account that good telemedicine implementations are developed after a process where the dynamic interaction among a combination of socio-technical factors is optimized [12]. This point leads to 3 main ideas which should be considered within a methodology for setting-up a telemedicine system. The first idea is that challenges and problems involve not only technical but also clinical and social issues. Apart from developing good technology solutions to support telemedicine systems development, there are other factors that strongly condition the adoption of telemedicine systems and that consequently should also be investigated. The second idea is that the evaluation process is an important and dynamic factor as it provides real information about the way to maximize success in such a project and to guarantee its continuity. This should be considered from the outset of the project and should continue throughout the whole system set-up and subsequent long-term implementation. Finally, the third idea is that the interaction between the 3 disciplines involved in the telemedicine set-up should be considered. Clinical, technical, and social (including economic and organizational) issues should finally be considered all together through evaluation and e-health strategies for telemedicine system implementations. Figure 1.1 provides a graphical view of a methodology proposed for the set-up of a telemedicine system which combines the steps of alignment theories and the interaction of technical, clinical and social issues within the development and evaluation of the system.

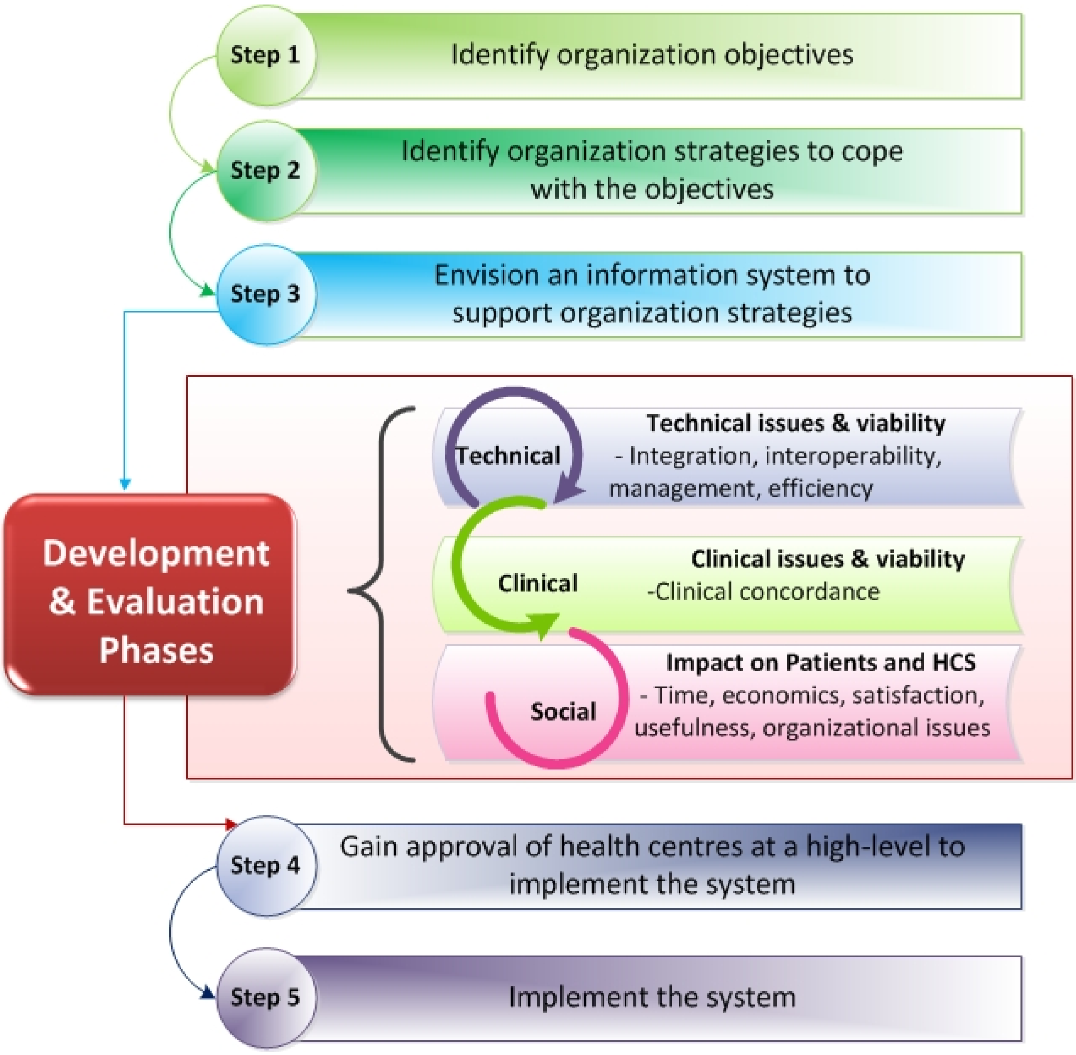

Figure 1.1: Methodology for setting-up a telemedicine system

Overall general agreement about how to evaluate telemedicine systems is still lacking [13]. Nevertheless, some initiatives such as the MethoTelemed project have emerged recently [14]. The team behind this initiative has proposed a new model to assess the effectiveness of telemedicine applications (Method for Assessment of Telemedicine, MAST [15]), to form a basis for decision-making in the EU (European Union) and other European countries on the use of telemedicine applications. This guidance is based mainly on a multidisciplinary assessment (thus applying evaluation in different areas such as clinical effectiveness and patient perspective) and a transferability assessment of the system. This proposed methodology is in line with other evaluation techniques published in the literature [16] and also with the evaluation process that we followed in the setting-up of the system in two telemedicine experiences [17, 18]. It is worth remarking that while interaction among all the factors should be considered to optimize the system, three phases can be distinguished during the system development and evaluation relating to each area. During the first phase technical issues should be addressed in order to develop the software prototype and then guarantee that there is no technical problem that could prevent or compromise the test performance. In the second phase, clinical viability studies should be performed in order to measure and demonstrate that both the technical and clinical quality achieved with the telemedicine system meet clinical quality standards. Finally, an impact study of the health care organization, patients and physicians should be carried out during the third stage. There is no doubt that these are relevant issues that contribute substantially to making the telemedicine system a success.

This evaluation methodology was applied in two telemedicine experiences performed between 2008 and 2011. These two experiences are used in the following subsections to explain and illustrate in detail how the three phases of the development and evaluation step can be performed for the setting-up of a real telemedicine system. The first experience consisted of the setting up of a tele-EEG system implemented between a primary hospital situated in Calahorra (Calahorra Hospital Foundation, FHC) and a secondary hospital in Logroño (San Pedro Hospital, HSP). Hence, this system was used in a type 1 telemedicine scenario for a hospital to hospital communication.

The EEG is a neurophysiological examination that consists of the recording and measuring of bioelectrical brain activity, commonly used in medical practice. Monitoring the EEG signal together with a video, known as videoelectroencephalography, enables the patients clinical (video) and electroencephalographic activity to be simultaneously monitored during a variable period of time. This is used for the evaluation of epileptic seizures and other paroxysmal disorders of brain function, in which a correlation between the clinical episode and the simultaneous electroencephalographic activity is required. Unfortunately, video-EEG as well as conventional electroencephalography services are now only available in Spain at tertiary or secondary hospitals. The introduction of telemedicine applied to EEG consultation, tele-EEG, provides patients with better access to specialized medical care, avoiding the need to travel to those hospitals unless absolutely necessary, and also reducing the costs and total time invested in carrying out tests. With the telemedicine system, patients are scheduled to undergo the video-EEG test at the FHC, where a nurse trained in EEG acquisition prepares the equipment and the patients, and follows any further indications given by the neurophysiologist located at the HSP. This physician visualizes the EEG signal together the video and provides a clinical assessment. The details of this experience can be found in [17].

The second experience consisted of the development and evaluation of a teledermatology system. This system was oriented to communicate primary care centres to secondary centres or hospitals where dermatologists were located. Hence, this system was used in a type 2 telemedicine scenario. The initiative arose when some GPs (general practitioners) hypothesized that by using a teledermatology system, many referrals to specialist hospitals could be avoided and others could be done much more quickly (in urgent cases). Hence, it was thought that a teledermatology system could help to improve patient care by speeding up the initial contact of patients with the dermatologist. In order to envision and implement the system, we took into account lessons learned from other experiences and determined 4 important factors to consider within the development and evaluation of the system: 1) the teledermatology software system had to be user-friendly and focused on physicians skills and requirements, 2) digital photographs of dermatological lesions had to be of high quality and the teledermatology system equipped with image analysis tools, 3) a clinical concordance study was needed to be carried out in order to determine what dermatological lesions the system could be useful for, 4) an impact study on the health system had to be undertaken in order to evaluate the long-term viability of the system. The details of this experience can be found in [18].

1.1.2.1 Technical phase

The technical phase includes two sub-phases: 1) the development of the prototype that will support the telemedicine system and 2) the viability study used to guarantee that the system can work correctly according to the environmental conditions in which it will be placed. Problems and errors occurring during this technical viability study and also during the subsequent study (clinical viability) could lead to modifications to the software prototype (see Figure 1.2).

One of the research questions governing the prototype development is how the system should be designed so that it can easily be used by clinical personnel and be adapted to the real clinical scenario. This requirement implies an interactive process with physicians being involved in the design from the outset. Hence, the software development should be planned as a collaborative process between engineers and physicians. However, underlying additional technical problems should first be addressed. At this technical phase, problems such as integration and interoperability should be taken into account in order to provide shareable healthcare. Since ICTs were introduced into the health environment, the development and inclusion of new elements into e-health infrastructures deal with this critical challenge known as interoperability. Making e-health elements interoperable means that they can communicate and exchange data seamlessly and accurately independently of the software and technology implementation. This problem is a particularly big issue in telemonitoring scenarios where different sources collaborate to provide a common service. In a similar manner, so it is integration. The need to define common information models or use common knowledge frameworks is a requirement extended to other telemedicine scenarios. As a matter of fact, the challenge of seamless integration with facility-based picture archiving and communication systems (PACS) in hospitals has been studied for years for teleradiology and teledermatology applications [19]. This technical phase was addressed in both the tele-EEG and teledermatology experiences.

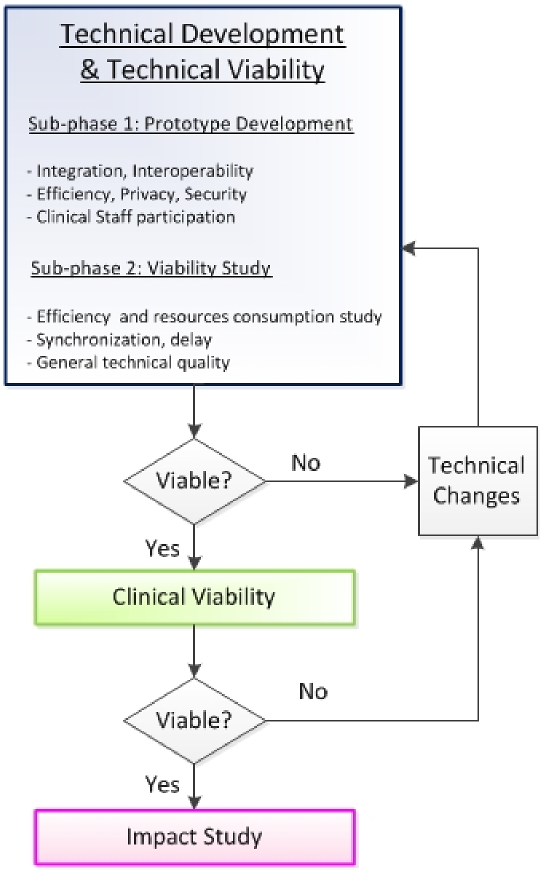

Figure 1.2: Development and evaluation phases

Regarding the tele-EEG experience, the main aspect to be controlled was the synchronization of ambient video and EEG display and the monitoring delay (to achieve real time functionality). Different types of data were collected within the technical viability study. During the first month, small technical changes were introduced to solve some problems that were reported regarding synchronization of ambient video and EEG display. Furthermore, a bandwidth measurement study was conducted. These measurements were made at different points of the telemedicine architecture in order to verify that the communication network capacity was sufficient for the tele-EEG system usage. In addition, both the nurse and the neurophysiologist had to complete a technical evaluation questionnaire for each tele-EEG session in order to report any technical errors. These data were collected during the twelve month evaluation period.

Regarding the teledermatology experience, a web application was developed to give support to the teledermatology system. The developed system was integrated in a secure environment where confidentiality, privacy and integrity of the exchanged medical data were always guaranteed. During 4 months, a small group of 4 physicians (1 dermatologist and 3 GPs) took part in the software validation, and a total of 17 teledermatology consultations were performed. Throughout this period, periodic meetings were planned where dermatologists and GPs pointed out weaknesses and usability issues of the system. The software system was improved in the light of these comments. In this way, all the necessary conditions in the clinical setting for the teledermatology process were identified through real practice.

1.1.2.2 Clinical phase

The clinical evaluation study is required to assess whether the introduction of telemedicine could affect the physician's diagnosis or could lead to an incorrect assessment of the patient's health. This study should be adapted to the individual application of each telemedicine system. See for example the clinical studies carried out in the teledermatology and tele-EEG experiences described below.

Regarding the teledermatology experience, in order to determine the reliability of the system, the degree of concordance (agreement) in diagnosis between teledermatology and face-to-face consultation was measured. The clinical concordance study addressed one basic research question: under our set conditions, how many consultations achieved concordance? In addition, it enabled two more research questions to be answered. How many times did the dermatologist offer a confident diagnosis and was it concordant? Was it related to the dermatological lesion type or the patients age?.

This clinical concordance study was conducted from April 2008 to July 2010. In primary care centres, patients with dermatological lesions that were clearly diagnosed by general practitioners (GPs) were not considered for a teledermatology consultation, thus avoiding bias in the concordance analysis. There were no more exclusion criteria for other patients with dermatological diseases; hence no pre-selection was involved (neither pathological nor in terms of demographic conditions) when performing teledermatology consultations. Figure 1.3 shows an example of the teledermatology procedure in a primary care centre. The PC screen shows the image viewer of the web application where all the clinical images uploaded for each consultation are available for display.

During this phase of the project, dermatologists were requested to issue an initial diagnosis evaluation using teledermatology. After a period of no longer than 1 week, patients were referred to a face-to-face consultation with the same dermatologist to obtain a definitive diagnosis. After the two consultations, the dermatologist had to complete two questionnaires for each patient. The first included questions about the diagnosis opinion, image quality level and quality of information provided by GPs. The dermatologists were also asked whether they found enough information in the teledermatology system to deliver a confident first diagnosis. They also selected the degree of possibility for offering a diagnosis with telemedicine on a scale ranging from 1 to 5. In the second questionnaire, the dermatologists categorized the concordance level of each pair of consultations as follows: agreement concordance (if both diagnoses provided under the different procedures were the same), partial concordance (if a differential diagnosis was proposed under any one of the procedures) or disagreement concordance (if a mistaken diagnosis was made using teledermatology).

A total of 120 complete concordance consultations were included in the study. The results showed that high rates of accuracy can be obtained with teledermatology both for adults and pediatrics with affordable technical requirements. Furthermore, high rates of agreement were found for most common dermatologic lesion referrals (inflammatory dermatoses) and it was demonstrated that if the dermatologist emitted a reliable diagnosis, total concordance was achieved in nearly 100% of the cases (without significant treatment differences for the wrong ones).

Figure 1.3: Example of a Teledermatology consultation

Regarding the Tele-EEG system (see Figure 1.4), in order to validate the remote monitoring procedure and to ensure the accuracy of the diagnosis provided using telemedicine, a diagnostic concordance evaluation test was conducted using intra-observer measurements. A total number of 20 video-EEG tests were included in the concordance study. During this evaluation period, the neurophysiologist was requested to give a diagnosis during the tele-EEG procedure for selected EEG parameters in the background rhythm in rest conditions, and under the different activation procedures such as intermittent photic stimulation and hyperpnea. Later, two weeks after the acquisition, the neurophysiologist saw the same tests again without using the remote control (meaning that she reviewed the recordings stored directly at the acquisition location) and recorded a new diagnosis for the selected EEG parameters, thus providing a comparison with the results obtained previously in the telemedicine process. The results showed that 100% concordance was obtained in all the evaluated parameters.

All the parameters evaluated both with and without telemedicine were given the same interpretation, and thus the neurophysiologist gave the same diagnosis.

Figure 1.4: Example of a Tele-EEG consultation

1.1.2.3 Social phase

The impact on hospital organization, patients and physicians should be measured at this stage of the setting-up process. This is an important evaluation stage that strongly conditions the set-up of a telemedicine system. Furthermore, apart from technical, clinical and social issues, other limiting factors such as the HCO participation from the outset of the project can determine its long-term establishment. Both qualitative and quantitative (e.g questionnaires and interviews) methods should be used for the data collection required for assessing each point included in the evaluation.

The impact on the health system was measured in our two telemedicine experiences. On the one hand, in the tele-EEG experience, the impact on hospital organization was analyzed in order to determine which changes would be required in both hospitals. Also, the impact evaluation exhaustively measured the patients perspective to determine the tele-EEG impact on their quality of life and also on their financial savings. As can be seen in the detail of the results [17], the system had a high level of acceptability amongst patients and also medical staff involved in the project, due to the reduction in social costs (patient transfers, working hours lost, and additional displacement costs). In addition, patients level of satisfaction was improved. No patient refused to undergo the EEG test using the telemedicine system, and patients, regardless of their age group, felt comfortable with the telemedicine system. Medical staff involved in the project stated they were comfortable using the telemedicine system and that it offered high quality for performing the EEG tests. During all the interviews they stressed the benefits the system had for patients in terms of reduction in social costs and especially the benefits for hospital inpatients (avoiding travel).

On the other hand, regarding the teledermatology experience, the impact on the health system was mainly measured in terms of personnel satisfaction and the time impact on clinical routine. Interviews and questionnaires were used to collect these data. The physicians' satisfaction and interest in teledermatology were also important strengths that supported its adoption and long-term set-up. In addition, the physicians involved described several other possible applications of teledermatology apart from its diagnosis function (filtering referrals, discarding urgencies, etc.). In contrast, the time invested in teleconsultation in primary care centres was as an important limitation. This was mainly due to a lack of integration with the EHR (electronic health record).

1.1.3 Home-based Telemonitoring Systems

Quality of care has been a growing area of interest and research in recent years, arising from the existing evidence that directly relates improving quality of care with a positive impact on the outcomes delivered by health systems. The WHO (World Health Organization) international organization states in its document about Quality of Care [20] that healthcare systems should make improvements in 6 dimensions of quality and consequently be **effective** (providing care adhering to clinical evidence and achieving positive outcomes for patients and the health system as a whole), **efficient** (using available resources properly and avoiding wasting money), **accessible** (thus solving time and geographical difficulties), **equitable** (they should not be reliant on personal characteristics), **acceptable and patient-centred** (patients' preferences should be taken into account for delivering services) and **safe** (risk and harm for patients should be minimized). In addition, two new emergent dimensions should be considered for quality of care management: 1) **security** (although this has always been considered in complex healthcare systems it has gradually gained importance) 2) **empowerment** (related to patients' participation in their disease management).

This new paradigm known as empowerment is a central idea in the EU health strategy supported by international health organizations including the WHO among others [21], and its effectiveness in yielding quality of care is an obvious and interesting area of research [1, 22]. Patient empowerment is considered as a philosophy of health care based on the perspective that better outcomes are achieved when patients become active participants in their own health management. The European experience of e-Health Patient Empowerment includes 1) providing patients with advice and trusted information, 2) supporting patients' education on health issues, 3) making the access to managing their own health data easy, 4) supporting the self-care paradigm and 5) supporting new models of chronic care. This new idea feeds into new ways of providing healthcare using ICT and new technologies. In this context, telemonitoring systems can be used as self-care management tools while collaborative processes among healthcare personnel and patients is maintained, thus the patient's safe control is guaranteed. Telemonitoring systems face the problem of delivering medicine to the current growing population with chronic conditions while at the same time covering the dimensions of quality of care (effectiveness, efficiency, patients' acceptability, safety, ...), and new paradigms such as empowerment can be supported. By periodically collecting patients' clinical data and transferring them to physicians located in remote sites, remote patient health status supervision and feedback provision is possible. This type of telemedicine system guarantees patient supervision while reducing costs (enabling more autonomous patient care and avoiding hospital overflows). Furthermore, patients' quality of life is improved. These are the main reasons that justify the increasing interest in research and in the setting-up of home-based telemonitoring systems.

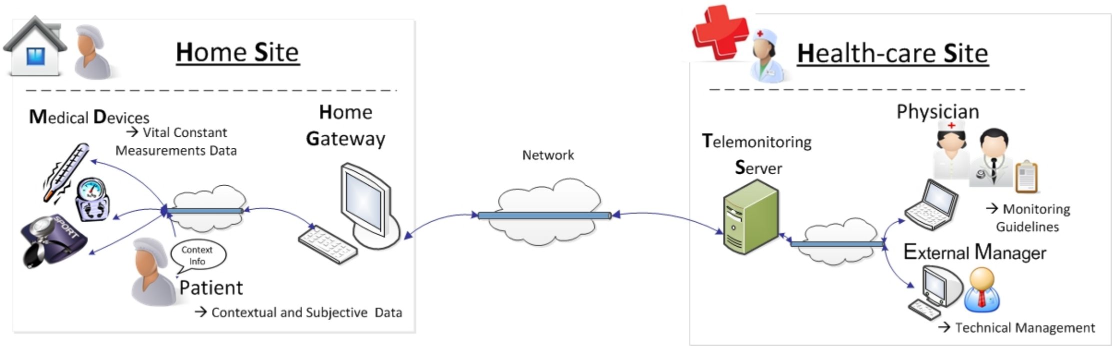

Figure 1.5: General Telemonitoring Architecture

In general, a typical home-based telemonitoring scenario includes two sites: the home site and the health-care site linked by a communication network (see Figure 1.5). At the home site, patients themselves periodically acquire vital signs measurements (e.g blood pressure) by using medical devices (MDs) to monitor their current health condition according to some indications provided by physicians (monitoring guidelines). This information acquired together with the patient's feedback is collected in a concentrator device (HG, home gateway) used to evaluate and/or transfer the acquired data outside the patient's home if necessary. Depending on the telemonitoring scenario, this evaluated data can alert patients to take some action or the transmitted data can make physicians in charge of the patient's supervision aware of an abnormal finding and consequently take action if required. At the health-care site, a server device is used to manage information from the home site as well as to manage and store the patient's monitoring guidelines defined by physicians (TS, Telemonitoring Server). In fact, this telemonitoring process, and consequently the evolution of the patient's health status, is managed through the indications or monitoring guidelines provided by physicians. Through these guidelines they indicate how many measurements the patient should perform, their schedule, whether or not they should be transferred to the health-care site, any additional subjective information that the patients should provide, and sometimes even a definition of alarms over acquired information in order to be alerted if an abnormal finding is detected.

According to the methodology presented in section 1.1.2, once a technology solution has been envisioned to reduce health-care costs and improve patient empowerment (hence in line the HCO strategies and objectives), technical issues and technical viability should be considered in the next development and evaluation stage. The introduction of these new elements in healthcare has driven researchers and industry to develop new standards and look for new technology solutions to address the challenges that this e-Health scenario presents in order to accurately and effciently deal with clinical data. In particular, at this technical stage **data integration and clinical management** are key challenges in this scenario where heterogeneous sources located at both end sites of the e-health infrastructure participate together in the telemonitoring process [23].

This integration should be addressed at both end sites of the scenario but also in the communication-link, thus integrating the way of **transferring and exchanging information efficiently** between them. In addition, providing **personalized care services** and taking into account the **patient's context** have been identified as additional requirements [24, 25]. Furthermore, apart from clinical data aspects, technical issues should be also addressed in this scenario. **Technical management** of all the devices that comprise the telemonitoring scenario is an important task that may or may not be integrated under the same architecture as clinical management. Hence, at this technical level, research is still required to address these challenges. Consequently there is a need for the development of new telemonitoring architectures. It should be pointed out that while clinical viability is dealt with in the next phase of the methodology (see section 1.1.2), clinical staff participation is vital for this technical stage. According to the proposed methodology, further investigation should be performed in the following phases in order to collect real evidence that the evolution of the patient's disease is controlled in the same manner as it would be if the patient attended the primary care centre (clinical viability). In the same manner, the impact on patients and physicians in terms of satisfaction and comfort should be measured.

1.1.3.1 Data Integration and Clinical Management

Data integration is required to guarantee interoperability and seamless access to data provided by all heterogeneous sources (MDs, physicians and patients) that participate together in the telemonitoring process (acquiring and transferring medical data, defining telemonitoring guidelines and providing contextual information). At the home site, a wide variety of MDs that use their own format to represent and transfer data to the HG are likely to be used together for the patient's supervision. To address this problem, some initiatives such as the CLSI-POCT1A standard or the ISO/IEEE 11073 (X73) family of standards have been proposed as a solution for integration and interoperability in MD data communication [26, 27]. Specifically, the CLSI-POCT1A establishes a set of specifications to allow multivendor interoperability between point-of care devices, data concentrators, and clinical information systems. On the other hand, the X73 family of standards (supported by some open initiatives such as Integrating the Healthcare Enterprise, IHE [28] or Continua Health Alliance [29]) describes how information can be exchanged between personal MDs (X73 agents) and a concentrator device or gateway (where the X73 manager is installed) used to gather all the measurements provided by the different agents. Nevertheless, although manufacturers have eventually started using this last standard, the present situation is that these standards are not implemented in the majority of cases and thus heterogeneous MDs from different manufacturers with different protocols are still being used at homesites in e-health scenarios [23]. Besides, apart from data acquired by MDs, additional information must be provided by patients and integrated together with the vital constant measurements acquired in order to accurately describe the patient's situation and thus perform a correct diagnostic supervision. On the other hand, at the healthcare site, physicians define and provide monitoring guidelines to supervise patients and collect transferred data from the HG. Although different information is managed at both end sites of the architecture, all these sources take part in the same telemonitoring process. Consequently, data provided by them (and later shared or exchanged) should be represented as common knowledge into which all sources may be mapped, thus making clear the interaction and relations among them. Apart from knowledge integration, data transference between the home site and the health-care site is an additional point to be addressed in this scenario. The development of architectures to support this knowledge exchange is required.

Popular solutions such as the standardized Health Level Seven7 (HL7) messages could be used to solve these clinical data transferences within telemonitoring scenarios [30]. HL7 refers to a family of standards used to transfer, share and exchange electronic health record information. It specifies a set of conceptual models, guidelines and messaging standards (which is in fact its most popular and important contribution) aiming to solve communication and interoperability issues between heterogeneous healthcare systems. However, this effective XML [31] based message transference solution lacks semantic representation which would ease data integration and semantic interoperability in contrast to other proposals that can be used such as OpenEHR or ontology solutions [32, 33]. On the one hand, ontologies (both for terminology and care process representation) have been proposed as an emerging and popular technology to address integration and knowledge representation issues in general healthcare practice. On the other hand, OpenEHR is a *de facto* standard that provides a solution based on the definition of archetypes to create re-usable clinical knowledge, adaptable computing systems and electronic health records. The OpenEHR strategy is based on the development of knowledge oriented frameworks including ontologies and formal terminology to address semantics representation and provide effective healthcare. Solutions specifically applied to the telemonitoring scenario are based on the development of archetypes to model vital constant measurements (eg. blood pressure or temperature) or intervention plans [34]. Both standards, HL7 and OpenEHR, could be good solutions for the transfer of data in the telemonitoring communication link. However, the application of any of these standards is not an easy process and developing specific guidelines for their usage makes their application more complex.

In summary, great efforts have been made in recent years in developing standards to deal with interoperability at different points of the e-health communication infrastructure (see Figure 1.6). Nevertheless, additional efforts are required to enable them to work together and ultimately provide a higher level of seamless integration. Furthermore, data collected in the HG should be evaluated in order to extract additional information that could even enable alarms to be triggered. Hence, a management procedure integration is required in order to provide a uniform and clear working process for acquiring, analyzing and making decisions about the patient's monitored data. That is to say, it is necessary to integrate clinical decision support tools or rule-based systems to automate and improve patient care supervision.

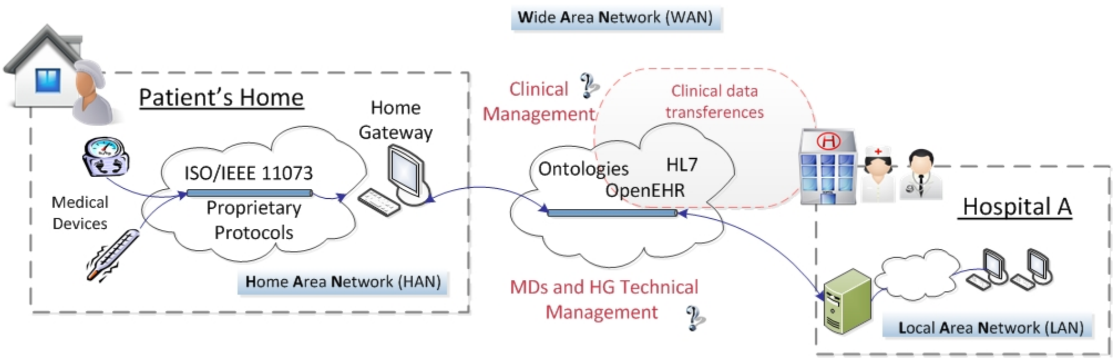

Figure 1.6: Standards in the Telemonitoring Scenario

This integration together with data transferences from the HG to remote hospitals should be addressed by the development of knowledge-oriented frameworks. A higher level of abstraction is required in order to provide a common understanding about data exchanged by all the entities, the way of dealing with such data and an integrated manner to exchange it. As mentioned above, this can be achieved by the use of ontologies and ontology-based architectures to support its physical implementation.

According to one of the most widely accepted definitions of ontologies in computer science, an ontology can be described as "an explicit and formal specification of a shared conceptualization" [35]. In practical terms, an ontology is a hierarchy of concepts with a set of properties and relations that provides a formalized representation of a common view of a determined domain. In contrast to other popular technologies such as XML schemes, ontologies provide knowledge representation and provide mechanisms to reason upon data defined in the model. By adding rules and logic it is possible to merge, extract and combine data in new ways to produce new information. Consequently, in this research the use of ontologies and their combination with the so-called autonomic computing paradigm has been investigated to address integration challenges in home based telemonitoring scenarios [36].

1.1.3.2 Context-aware and personalized medical care

According to its most frequently cited definition [37], context refers to any information that can be used to characterize the situation of an entity (person, place or computational object) and the interaction between them. Specifically, in home-based telemonitoring scenarios the patient's location, environmental conditions or simply the way of acquiring vital constant measurements can affect measurement results, and they should thus be taken into account when evaluating acquired clinical data. Furthermore, according to each patient's health conditions, the physician should define individual monitoring guidelines in order to supervise his/her health status evolution. All these individual factors can be termed as context and should be taken into account when monitoring and evaluating the patient's clinical data. In fact, a context-aware system provides personalized healthcare [24].

Providing personalized healthcare services is a challenging requirement in home-based telemonitoring scenarios [25]. No two patients are the same and consequently each one requires his or her own monitoring guidelines adapted to his or her own chronic conditions. Furthermore, patients with chronic diseases often suffer the occurrence of more than one chronic condition with an index-disease, that is to say co-morbidities [38]. These conditions affect each other, increasing the number of risk factors for patients and consequently having in general a significant link with mortality and prevalence rates, functional status and quality of life. For example, an adult patient with type 2 diabetes (DM) suffering also from hypertension and dyslipidemia (DLP) has an increased risk of vascular complications [39]. Hence, co-morbidities require more intensive health care services and further necessitate the provision of individual and personalized services at the home site. Consequently, the development of knowledge models applied in this scenario should be generic and not restricted to the monitoring of patients with one chronic condition. Such models should provide the flexibility to individualize the supervision according to each case and should be sufficiently scalable to be extended if additional chronic cases are to be covered with the telemonitoring system. In this thesis, the use of rules in combination with ontology knowledge is investigated in order to address the personalization problem. It should be noted that physicians' knowledge is a key consideration in the ontology development and in particular in the modelling context.

1.1.3.3 Technical management challenge

While considerable efforts have been made (and continue to be made) to deal with complexity in clinical data management, technical management issues have not been given much consideration by researchers. Hence, the remote technical management of the devices that comprise the telemonitoring scenario is not covered by any of the developed standards and neither are solutions proposed by researchers. It should be noted that technical troubles and technical management concerns are of great importance. Apart from ensuring that MDs are safe for patients (for which manufacturers are responsible and for which contributions have already been presented [40]) it is essential to simply ensure that any technical problem in the MD or in the HG (e.g abnormal resources consumption or measurement levels) that could prevent or interfere with patients' monitoring will be detected.

As explained above, the X73 standard has been proposed as a solution to transfer information in an interoperable manner from MDs to the HG. Although this standard does not deal with the remote (beyond the manager) technical management of MDs, the data flow from MDs to HG includes both technical and clinical information. By using this standard for MD transference, the MD technical management issue could be solved by proposing an extension of X73 functionalities. Nevertheless, MDs have extremely limited processing power, memory and battery life [41]. Therefore, by proposing this type of solution an additional overhead would be imposed on MDs to support technical management, the result being the likely shortening of their already limited battery life. Moreover, it would be more appropriate to provide a global solution for managing all the devices involved in the telemonitoring scenario. Apart from technical management, overall control of the use of HG resources such as memory and bandwidth as well as the number of connected MDs would also be desirable. Therefore, since the HG can be seen as a communication network element, using traditional management architectures for TCP/IP networks could be an interesting solution to provide the HG with its own management capabilities but also for the remote management of MDs as long as they are connected to the HG and transfer technical information to it.

Similar challenges in terms of interoperability and integration are also faced by network managers. They control the operational status of network elements and ensure, among other things, an effective utilization of network bandwidth in complex networks where heterogeneous devices from different manufacturers run together. Indeed, popular network protocols are used nowadays in e-health infrastructures such as for server and network management purposes in hospitals. As in healthcare scenarios, different standards have been proposed since the early 1990's to address these problems. Telecommunication Management Network (TMN) recommendations from the ITU-T (representative management solutions of this model are Common Management Information Protocol, CMIP, and Common Object Request Broker Architecture, CORBA) and SNMP (Simple Network Management Protocol) [42] architecture from

IETF (Internet Engineering Task Force) are good and popular solutions for traditional management tasks that rely on the agent-manager model. Nowadays, although new protocols such as NETCONF (Network Configuration Protocol) have been proposed to overcome certain limitations of SNMP in configuration tasks, the latter is still a *de facto* standard for managing TCP/IP networks and the most popular architectures supported by industry. Ontologies also play an important role in the network management field [43]. Although this technology is not as mature as the previously mentioned standards, important research and contributions have been made regarding its application in network management in the last decade. Ontologies are an interesting approach to solve this and other integration problems arising in growing telecommunications infrastructures where different information management models can be used.

This new challenge should be addressed taking into account the specific demands of e-Health infrastructures today in terms of interoperability and seamless integration. Furthermore, device heterogeneity, the limited processing power and battery life of MDs, and network costs and efficiency problems in the communication network should also be considered. Hence, in this research two management techniques have been investigated to address this new challenge: ontologies and SNMP.

1.2 Research questions and thesis proposal

The general approach of this thesis is to make a contribution to the field of home based telemonitoring scenarios given the relevance today of telemonitoring systems in e-health and the social benefits they provide for patients, physicians and the health system as a whole. Specifically, the research is focused on the application of knowledge-oriented frameworks based on ontologies to address the main challenges described in the introduction. As implied by the methodology presented for setting up a telemedicine system (see Figure 1.1), the main contribution lies in the technical phase. Taking into account MD interoperability, clinical knowledge and engineering concerns, this research investigates how an ontology-based architecture can be successfully used to address the main challenges presented in home-based telemonitoring scenarios to monitor patients with chronic conditions. That is to say, to integrate data provided by the different heterogeneous sources that participate together in the telemonitoring process, to provide a uniform means to automate and integrate its management procedure, to provide personalized services and care taking into account the patient's context and, furthermore, to support technical management under the same conceptual and physical architecture and thus address the challenges and requirements presented by this type of management. It should be noted that an SNMP solution has been investigated for this technical management and compared to the ontology-based solution. Furthermore, the combination of ontologies and the so-called autonomic computing paradigm is studied in this thesis and the use of rules to provide personalized care is also proposed.

1.3 Hypothesis, Objectives and Goals

This thesis makes a research contribution to the field of home-based telemonitoring systems within the application of ICT in the e-Health sphere. Specifically, this research investigates how a new architecture based on ontologies can be successfully used to address the main challenges presented in the technical phase (within the methodology proposed for the set-up of a telemedicine system) in home-based telemonitoring scenarios. The following underlying hypotheses are uppermost in this research approach:

- The usage of telemonitoring systems to monitor patients at home with chronic conditions has a positive impact on delivering quality of care. These systems improve patients' quality of life and empowerment, and also enable social costs to be reduced.

- The development of knowledge-oriented frameworks to support data management in telemonitoring scenarios enables a further step to be taken to unify data provided by the heterogeneous sources that participate together in the telemonitoring process. They provide a clear understanding of the managed data and can also be used to describe the management procedure. Hence, these frameworks can be used to address the main challenges involved in the technical development of the system.

- The incorporation of rules can be effectively used to define individual alarms within the monitoring guidelines, thus providing personalized medical care.

- Technical management also represents a significant challenge in home-based telemonitoring scenarios. The development of knowledge-oriented frameworks can be successfully applied to support technical management, thus enhancing the integration of both tasks (clinical and technical). However, other solutions based on traditional network management techniques can be applied to deal with the technical management of telemonitoring devices.

The following objectives have been identified to address the research approach set out in section 1.2 and taking the above-mentioned hypotheses into consideration:

1. To conduct reviews on the state of the art in general aspects of ontology design, languages and implementation. Specifically, to study the application of ontologies in the clinical sphere and to examine other solutions proposed in telemonitoring scenarios. To study the potential of ontologies for representing knowledge and addressing integration and interoperability challenges.

2. To design and develop an end-to-end telemonitoring architecture taking into account both conceptual issues (thus the domain modelling) and the transference of data between the end points (thus communication and messages exchange).

Hence the architecture should include two layers: a conceptual layer and a data and communication layer.

- To design and develop an ontology to support the conceptual layer of the proposed architecture and provide data and management integration in general home-based scenarios. In order to describe this conceptual model and exploit its features for the telemonitoring domain, the following sub-objectives have been set:
    - To study the combination of ontologies and the autonomic computing paradigm to automate management purposes in telemonitoring scenarios.
    - To propose a methodology to address the development and usage of the proposed ontology.
    - To study the combination of rules to provide personalized services.
- To investigate the use of Web Services (WSs) [44] to support the exchange of information defined in the conceptual layer of the architecture, that is to say, to provide a practical support for the developed ontology and an effective manner to transfer data.

3. To investigate and adapt the ontology-based architecture design to provide clinical management in the home-based telemonitoring scenario and address the main challenges identified in sections 1.1.3.1 and 1.1.3.2. In order to address this objective, the following tasks should be performed:

    - To conduct reviews of the X73 standard to take into account MD interoperability and other existing clinical standards in this scenario. To schedule interviews with clinical personnel for modelling clinical knowledge and the requirements of monitoring chronic patients for the design of the ontology.
    - To adjust the ontology design and evaluate its completeness and adaptability for monitoring patients with different or multiple chronic conditions. To evaluate the usage of rules applied over the ontology model to define personalized care.
    - To evaluate the applications of the model to define complex care and decision tasks.

4. To address the technical management challenge in home-based telemonitoring scenarios (identified in section 1.1.3.3) studying two different solutions: an ontology-based solution and an SNMP based solution. The following sub-objectives have been established:

- To review the X73 and the SNMP architecture to inspire the ontology model design covering MD technical features and generic network resource monitoring specifications.
- To investigate and adapt the ontology-based architecture design to provide solutions to technical management issues.
    - To adjust the ontology design and evaluate its completeness and adaptability to define simple and complex management tasks. To evaluate the usage of rules applied over the ontology model to define personalized management services.
- To propose and investigate an alternative for remote technical management based on the SNMP architecture. This research task implies the design of a complete SNMP based architecture including an agent entity and its MIB-module (Management Information Base).
- To compare the advantages and disadvantages of using one solution or another.

5. To develop a prototype and evaluate its efficiency for supporting clinical and technical management tasks in telemonitoring scenarios. Specifically, to implement its application to the X73 and SNMP standards.

1.4 Research Context

This thesis has largely been developed within the following funded projects of the Telemedicine and e-Health research line of the Communications Technologies Group (GTC) at the Aragón Institute of Engineering Research (I3A), University of Zaragoza:

1. MEC TSI2004-04940-C02-01: "Biomedical information transmission in health networks: coding/transmission methods development and QoS analysis".
2. MEC - PETRI PET2006-0579PET 2006-0579: "New e-Health service introduction related tele-cardiography and tele-encephalography".
3. MCINN - TIN2008-00933/TSI: "New telemonitoring systems for e-Health services".
4. MCINN - TIN-2011-23792/TSI: "Ontology-based interoperable architecture for patients telemonitoring and clinical decision support".

Additionally, three government scholarships have been granted to the PhD candidate:

1. A four year doctoral grant awarded by Diputación General de Aragón (DGA), Res. 7-09- 2009/n.185 from BOA, Boletín Oficial de Aragón.

2. A research fellowship awarded by European Program from Caja de Ahorros de la Inmaculada (CAI), Ref. IT10/11.

3. A research fellowship awarded by University of Zaragoza from the Mobility Program for Researches to non-European countries. Ref: Vicerrectorado de investigación MBR/mjs.

The research group collaborates with other entities, institutions, universities and hospitals both nationally and internationally. Specific mention can be made of the following international cooperation partners relevant to this thesis.

- The Knowledge and Data Engineering Group (KDEG) at Trinity College Dublin (TCD) in the School of Computer of Science and Statistics of University of Dublin, through Dr. Declan O'Sullivan, director of the KDEG.

- The School of Health Information Science from University of Victoria (Canada) through Dr. Andre Kushniruk head of the health information science research team.

Two research stages have been done in collaboration with these research groups.

At a national level, specific mention can be made of the following clinical collaboration partners:

- The "Instituto Aragonés de Ciencias de la Salud" through Sara Guillén and Dr. Rosa Magallón, head of Primary Care in Aragón.

- The Dermatology Service of the Hospital San Jorge (Huesca) through Dr. Yolanda Gilaberte and primary care centres in this area (Pirineos, Santo Grial, Sabiñanigo and Perpetuo Socorro).

- Clinical Personnel from the medical specialist center San José and Grande Covián and the nearby primary care centres (Canal Imperial, TorreRamona, Arrabal and Ruiseñores).

- The "Calahorra Hospital Foundation" and San Pedro Hospital from Logroño (La Rioja) where Carmen Campos is head of the neurophysiology department.

1.5 Thesis Overview

The remainder of this thesis is organized as follows. The state of the art relating to the main topics and concepts involved in this thesis research is presented in chapter 2. Specifically, the chapter contains a review of telemonitoring systems, background information about related standards, a description of ontologies and their application to the telemonitoring scenario, a discussion of rules and the autonomic computing paradigm and also the main reasons for selecting the technologies used in the solutions proposed in this thesis. Information about WS technologies and the SNMP architecture is provided as well in this chapter. Furthermore, related work is also discussed. The general proposed architecture and the layers in which it can be divided are described in chapter 3. The main modules that comprise the physical architecture, the communication structure and workflow are presented and also the ontology model is described in detail. Furthermore, a 3-stage methodology proposal for developing and working with the ontology is presented. The application study of the ontology and its extension to clinical management in home-based telemonitoring scenarios is presented in chapter 4. The clinical knowledge used to model concepts included in the ontology are presented and the application-based study defining supervision profiles for patients with chronic and multi-chronic conditions is described. Ontology evaluation is also discussed. Chapter 5 describes the extension and application of the model to the management of technical data in the e-health scenario. Technical data used to model concepts in the ontology are described and its relation to the X73 and SNMP standards is presented. Furthermore, an additional solution to address technical management based on the development of a complete SNMP architecture is described. A comparison between both solutions is also presented. Both chapters 4 and 5 also deal with practical issues and describe the prototype implementation used to run both clinical and technical management tasks at the home-site. Implementation details, the tests performed and efficiency results are also described in these chapters. Finally, the discussion, conclusions and future lines of research are presented in chapter 6.

Chapter 2

State of the Art

Ontologies, rules, web services, the autonomic computing paradigm and the SNMP architecture are well-known technologies and popular solutions applied in the semantic web domain and the network management field, respectively. All of them have been used in this work to provide an answer to the challenges presented in a telemonitoring scenario. As these technologies come from different worlds, the question is: how can they be used to solve problems presented in e-Health scenarios? With the aim of understanding how the combination of these technologies can be successfully used to provide a solution in telemonitoring scenarios, this chapter describes these popular technologies and related works that have used them. First, a review of the state of the art regarding telemonitoring systems development and also popular standards in e-health is presented in order to put into context the demands and features of this scenario. Then, details are given about ontologies, rules, web service technologies, the autonomic computing paradigm and the SNMP architecture.

2.1 Home-based Telemonitoring Systems: a review of the state of the art

The advantages that home-based telemonitoring systems provide in terms of cost reductions for hospitals and improvements in quality of life of patients are demonstrated by the plethora of experiences that have been reported over the years about tele-health systems implementations for monitoring patients at home [45]. Some of these systems are focused on monitoring patients with one chronic condition, such as diabetes [46] or heart failure (HF) [47] while others aim to monitor, treat and manage chronically ill patients in general [7]. Apart from patient supervision, these telemedicine systems can be developed for many other services such as for medication control management and adverse drug events control [45]. Furthermore, systems that transfer data to the health-care site may include clinical decision support tools to improve health care reviews of acquired data and clinical decision making [7, 48]. These systems are based on different technologies for delivering monitoring services and experiences include both mobile-based solutions for ubiquitous health delivery [49, 45] and telemedicine systems placed at home based on web systems or other ICT technology support [7].

To deliver personalized health care services is an important issue within the development of home-based telemonitoring systems [24, 25]. Interoperability and integration are critical challenges that also need to be addressed when developing monitoring systems in order to provide effective healthcare and to make possible seamless communication among the different heterogeneous health entities that participate in the monitoring process [50, 51]. For these reasons, substantial efforts have been made during recent years in standards and knowledge-based solutions research.

2.2 Standards solutions for e-Health scenarios

As stated in chapter 1, making e-health entities interoperable means that they can communicate and exchange data seamlessly and accurately independently of software technology implementations. Hence, in order to address this challenge, some standards have been proposed to be implemented at different points of the telemonitoring communication infrastructure (already presented in chapter 1) such as the X73 for MDs communication, HL7 for transferring messages and the OpenEHR initiative. These last standards have been become popular because of their usefulness for exchanging EHR (electronic health record) information, as is the case with the EN13606 standard. All the standards mentioned may be used for clinical purposes in e-Health scenarios.

In particular, the X73 standard was chosen to inspire the structure of the clinical measurements (acquired by an MD) description included in the ontology. Furthermore, its integration with the proposed ontology-based solution has also been studied. This family of standards has been popularized and recommended by the IHE and CHA organizations. It provides a complete solution for MD connectivity and addresses the consistency problem in information caused when MDs from different manufacturers are used together within the same health service. The main goal in using this standard for the design of the ontology (clinical extension) was to take advantage of efforts made in terms of nomenclature, terminology and representation of information provided by each MD described in the standard. Other standard alternatives can be used in the clinical domain for medical device transferences such as the CANOpen [52]. CANOpen provides a communication protocol and device profile specification aimed to be used initially in automation. Consequently, its features are focused on automated control and its usage seems to be more suitable in complex environments. In contrast, the X73 usage has always been focused on MD communications. Improvements and evolutions made over the years to the standard have increased its international popularity as a standard addressing MD interoperability [53].

Other standardized nomenclature in the e-Health domain has been used in this thesis. Specifically, concepts described in the ontology were related to standardized terms in medicine by using the SNOMED-CT nomenclature [54]. SNOMED-CT is nowadays a leading ontology of biomedical terminologies that covers a wide range of clinical concepts including not only diseases but also procedures and clinical findings. This is an important resource for healthcare and contribution to standardization in health information technologies. By using these codes, accuracy in EHR information is improved and its communication made easier, thus leading to improvements in patient care.

2.2.1 ISO/IEEE 11073 (X73)

The X73 family of standards is a reference frame for MD interoperability. Although early drafts relating to the standard date back to the mid 1980's, the first version created in common by the ISO (International Organization for Standardization) and CEN (European Committee for Standardization) organizations, the CEN/ISO/IEEE 11073 Health Informatics-Point-of-Care Medical Device Communication (X73-PoC), was released in 2004. This X73-PoC family of standards leveraged previous works published in the mid 1980's to the different layers of the communication process. The latest version of the standard released in 2008 (11073-20601 Personal Health Device communication: Application profile-Optimized Exchange Protocol, X73-PHD or X73) is adapted to the new paradigm of personal health care and well-being [55]. Improvements made in this new version take into account patient mobility and new concepts such as Personal Area Network (PAN) and Body Area Network (BAN). It is worth noting that advances in the latest version were proposed and approved by the PHD Working Group (PHDWG).

The standard can be explained through its abstract model and its layer structure. The overall X73PHD standard is basically divided into three main models (see Figure 2.1):

1. **Domain Information Model (DIM)**: the DIM defined in X73 describes a hierarchical set of objects with attributes and methods which characterizes the information that can be sent from an agent to a manager (e.g. data measurements and generic device information). The DIM is the most relevant component regarding the description of technical features of MDs and the description of clinical measurements acquired by the MD. Specifically, within the DIM, the MDS (Medical Device System) class represents the identification and status of each agent, and its attributes contain most of the technical information that can be retrieved from MDs. Clinical measurements and related context information are represented by using Numeric, Enumeration and Real Time-Simple Array (RT-SA) objects. In particular, Numeric object is used to represent numerical measurements, RT-SA is for waveform measurements and Enumeration objects are

used to report status or contextual information. The DIM also includes objects for data storage (Persistent Metric (PM)) and the Scanner class for data reporting. Objects classes and attributes within the DIM are referenced using nomenclature codes described in the ISO/IEEE 11073-10101 document. Attributes of each class are defined by using the name of the attribute, its nomenclature reference identifier (ID), its type, its qualifier and a description of the attribute. It should be noted that each attribute data type is defined using an Abstract Syntax Notation One (ASN.1). The DIM was the most relevant part of the standard used for the extension design of the ontology (for clinical and technical purposes).

2. **Service Model**: the Service Model provides primitives to explain how the data defined in the DIM can be exchanged between the agent and the manager. Thus, this model provides a means of interaction between agents and managers. These primitives include messages used by the manager to retrieve information (Get), modify some specific attributes (Set), execute some actions over the agent features (Action) and Event Reporting commands (messages sent by the agent to the manager).

3. **Communication Model**: finally, the communication model establishes a finite-state machine (FSM) for both agents and managers that controls its communication state. This FSM defines the dynamic behaviour of both entities by describing the states and substates while they are passing through a communication process. The main states defined in the FSM are: DISCONNECTED, CONNECTED, UNASSOCIATED, ASSOCIATED, CONFIGURING and OPERATING. Initially both entities are in the disconnected state until the devices are turned on and go to the connected state. Then, once the agent sends an association request to the manager and it is accepted, they go to the associated state. In this state, if the manager does not know the agent configuration, it will be asked and consequently both will be moved to the configuring state. It if does, they will go directly to the operating state. In this last state, the agent begins sending measurements to the manager. The communication model supports one or more agent communications with a single manager over point-to-point connections.

From these models, the X73 PHD standard relies on a layer structure to describe its communication protocol. This architecture protocol is divided into three main levels:

1. **Transport layer**: the X73 protocol allows support for various transports; it is not specific to any particular one.

2. **Optimized Exchange Protocol**: the Optimized Exchange Protocol (reported in the ISO/IEEE 11073-20601 document) describes the data exchange protocol for

the agents and managers communication. Moreover, it describes the application layer services.

3. **Device specializations**: these documents (11073-104zz) indicate the details relative to how a specific type of agent works (e.g., blood pressure monitor or weighing scale). Thanks to the abstract information model used in the standard to define MDs specializations, the incorporation of new MDs is easier. Some device specializations have already been developed and others are currently under development. In particular for this thesis research a total of 8 device specialization documents were reviewed, referring to the weighing scale (-10415), thermometer (-10408), blood-pressure monitor (-10407), pulse-oximeter (-10404), glucose meter (-10417), body composition analyzer (-10420), medication monitor (-10472) and peak expiratory flow monitor (-10421).

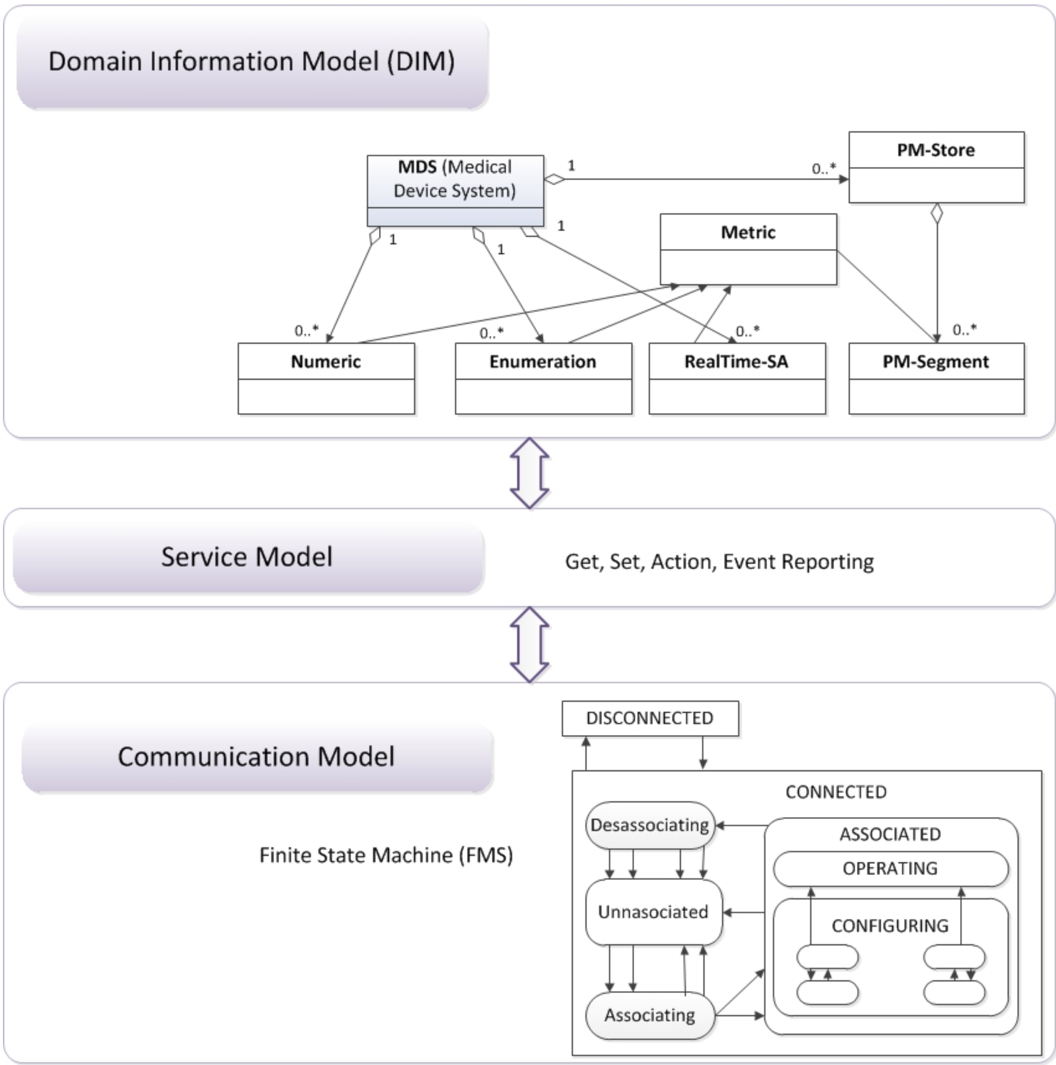

Figure 2.1: X73-PHD Abstract Model

2.2.2 X73 experiences

The X73 has been investigated for a wide variety of telemonitoring scenarios including bedside monitoring in hospitals, wearable implementations and home-based telemonitoring systems. Remarkable works have been reported in recent years regarding X73 research in terms of experiences of implementation of the standard in wearable devices [56, 57], contributions to new device specializations [53, 58], its application to new cases of use [59] and also transport technology (e.g Bluetooth) and optimization research [41, 60]. An interesting work can be found in [50]. This work provides a complete e-health platform based on standards aiming to solve integration and interoperability challenges in an end-to-end telemonitoring architecture. Specifically, the use of the X73 is proposed to be used at the home site, and the rest of the technologies used are oriented to transfer and integrate data acquired by using this standard. Although this is a promising standard, it has not yet been widely adopted by industry and consequently challenges at the home-site remain uncovered.

2.3 Ontologies

2.3.1 Introduction

Ontologies have been successfully used in recent decades in the world of computer science and particularly in the semantic web domain for the purpose of achieving a comprehensive and readable machine common understanding [61, 62]. According to the Studer definition [35] ("an explicit and formal specification of a shared conceptualization"), an ontology is an abstract model (conceptualization) that provides a controlled vocabulary for the description of concepts, each with an explicitly defined semantics (explicit) and machine processable language (formal). Furthermore, ontologies aim to capture consensual knowledge by a group of people and they may be reused across different applications (shared) [63].

In simple words, ontologies represent concepts and basic relationships for the purpose of comprehension of a common knowledge area. Hence, to develop an ontology means to formalize a common view of a certain domain. The key of their success is mainly based on two relevant characteristics: first, ontologies provide a knowledge framework with clear semantics and syntax representation and second, it is possible to infer knowledge from them. This second characteristic constitutes an important advantage of ontologies compared to other data structure technologies such as XML schemes. Ontologies provide mechanisms to reason upon data defined in the model and to execute more complex and efficient searches. By adding rules and logic it is possible to merge, extract and combine data in new ways to produce new information.

Ontologies have been used in this thesis for two purposes: clinical and technical management. On the one hand, as explained in the introduction, ontologies offer a powerful solution to address the data integration challenges presented in the telemonitoring scenario. Furthermore, they can be used to formally describe and thus integrate clinical management tasks. In the same way as for clinical management, they can be successfully used for technical management purposes. This is in line with solutions reported in recent years for the network management domain to handle multi-language [64]. management problems. Both clinical and technical management services can be implemented under the same architecture.

In general, ontology knowledge can be specified by the following components: concepts, relations, functions, axioms and instances [65].

- *Concepts* refer to concrete or abstract entities and are used to describe the domain. Concepts can be organized into a hierarchy of relationships, termed as taxonomy.
- *Relations and functions* refer to interaction between concepts of the domain and attributes.
- *Axioms* are used to model sentences that are always true. They are used to associate classes and properties with some specifications of their characteristics. Hence, they can be used for constraints purposes, verifying correctness or deducting new information.
- Finally, *instances* refer to specific elements within the domain of a concept.

It should be noted that two types of ontologies can be distinguished: lightweight ontologies and heavyweight ontologies. Lightweight ontologies are mainly taxonomies that include a set of concepts and hierarchical relationships among the concepts. They are easier to operate but some agreements are required for interoperability purposes among all the parts. Heavyweight ontologies provide greater detail and add axioms and constrains to lightweight ontologies. They are easier to deploy but difficult to develop [63, 66].

2.3.2 Ontology Languages

2.3.2.1 Ontology Languages Review

In computer science, there are plenty of formal languages that can be used to define and construct ontologies. These languages allow encoding knowledge contained in an ontology in a simple, formal and human-readable way. They are usually declarative languages and the main differences among them lie in the degree of expressiveness (the elements they include and how they can be used) and the inference engine of the language. These ontology languages can be compared in terms of the features and elements that they provide for specifying the ontology knowledge [65]. Usually, they are generalizations of frame languages and are commonly based on first-order logic

(FOL) or description logic (DL). Frames are data structures for representing entities (classes or individuals) used for knowledge representation in a domain. FOL refers to a symbolized reasoning in which each statement comprises a subject and a predicative and the latter defines or modifies the properties of the subject. Finally, DL is used for formal reasoning on the concepts of an application domain. Knowledge systems based on DL consist of two components: TBox (terminological box) and ABox (assertional box). TBox contains concepts and relations between concepts while ABox contains assertions about individuals [67].

Ontologies languages can be classified in two groups [68]:

1. **Traditional Ontology Languages**: these languages were created before the semantic web boom. They were defined during the 1990's and were based on first-order logic and on its combination with frames. Some of them can be highlighted:

    - *KIF (Knowledge Interface Format)*: a language based on first-order logic defined to exchange knowledge between different systems.
    - *Ontolingua*: this language was based on KIF and the combination of frame ontology forms. This is a powerful language in terms of expressiveness but due to this high degree of expressiveness, no reasoning support is provided with this language.
    - *OCML (Operational Conceptual Modeling Language)*: a frame-based language that can be considered similar to Ontolingua but with some extra components.
    - *F-Logic (Frame Logic)*: this language combines frames and first-order logic. Its features include, among others, concept representation, binary relations, instances, axioms and deductive rules.

2. **Web-based semantic ontology languages**: These languages were created within the Semantic Web context in order to exploit Web characteristics. Most of them are based on XML syntax and, according to their structure (frame, FOL or DL), they can be grouped in different sub-groups. The following are some good examples of these ontology markup languages.

    - *XOL (Ontology Exchange Language)* : initially designed for exchange of bio-informatics ontologies, this XML based-language can be used for ontologies in any domain.
    - *SHOE (Simple HTML Ontology Extension)*: this was proposed as an extension of HTML used to embed knowledge in web pages. It was subsequently modified to base its structure on XML.
    - *RDF (Resource Description Framework)* [69]: this language was created by the W3C (the World Wide Web Consortium) for describing web resources. The structure of any RDF expression is a collection of triples, each consisting

of a subject, a predicative and an object. A set of these triples is named as an RDF graph. All these elements are identified by a URI (Uniform Resource Identifier).

- *RDF Schema*: this extends RDF and its combination with RDF is known as RDF(S). This RDF(S) language incorporates among others, mechanisms for describing relationships between classes (for hierarchical definitions) and properties.
- *DAML-OIL*: semantic language that is based on RDF triples and was developed as a combination of features from both DAML (DARPA Agent Markup Language) and OIL (Ontology Inference Layer).
- *OWL (Ontology Web Language)* [70]: OWL language is a vocabulary extension of RDF. It describes the structure of a domain in terms of classes and properties, and provides a set of axioms to assert assumptions or equivalence with respect to classes or properties. OWL language, also known as OWL 1, is explained in detail in the next subsection.
- *OWL 2* [71]: this language was proposed by the W3C in order to replace the OWL language. Detailed information of this language is provided in the following subsections.

Apart from these groups, in [64] the combination of UML (Unified Modeling Language) with OCL (Object Constraint language) is proposed as an additional ontology language. UML is a standardized modeling language that includes a set of graphic modules to specify, visualize and modify artifacts of an object-oriented software system. This is a mature and widely used modelling language that is also very popular for defining ontologies. By using the OCL language, it is possible to specify restrictions over the UML model.

Figure 2.2 depicts the position of the main languages described within the hierarchy illustrated in the Semantic Web Stack. Query and rule languages depicted in Figure 2.2 are described in the following subsections.

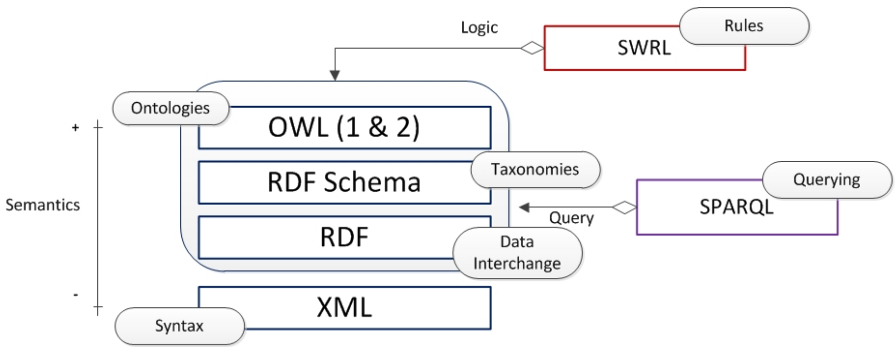

Figure 2.2: Semantic Web Structure

2.3.2.2 OWL language

While there are various languages for expressing ontologies, the standardized RDF schema and OWL have been gaining popularity in the semantic web world. In fact, OWL is the current standard for ontology definition. The OWL language is a vocabulary extension of RDF and offers an enriched expressiveness in the descriptions of classes and properties. OWL provides a set of vocabularies to define concepts, relations and axioms and brings the reasoning power of description logic to the semantic web. The OWL language specification includes the definition of three sub-languages that provide different levels of expressiveness ordered by increasing expressiveness as follows: OWL Lite, OWL DL and OWL Full. In most cases, OWL DL (based on the *DL SHOIN* language) is chosen by ontology developers because it provides the maximum expressiveness capacity that can be offered while total computational capacity is guaranteed. Indeed, it was chosen for this thesis research. As OWL is an extension from RDF, any OWL ontology can be seen as an RDF graph which is in turn a set of RDF triples. In general, an RDF graph forms an OWL Full ontology. As OWL DL and OWL Lite place some expressiveness restrictions in the vocabularies, any RDF document will generally be in OWL Full, unless specifically constructed to be in OWL DL or Lite.

OWL is an object-oriented ontology language. An ontology can be formally described in OWL using following basic elements:

1. *Classes* : these represent entities or concepts about the described domain. They constitute the base for the domain description.
2. *Individuals* : they are instances or members of a class. Hence, a class is simply a name and collection of properties that describe a set of individuals. Classes are used to classify individuals into groups which share common characteristics. A set of individuals associated to a class is designated as *class extension.*
3. *Properties* : there are two types of properties in OWL:
   - *Object Properties* : these express relationships between individuals of two classes.
   - *Data Properties* : these are used to link individuals with datatype values.

Finally, OWL provides annotations (*owl:AnnotationsProperties*) which can be used in OWL DL under certain conditions to define annotations on classes, properties, individuals and ontology headers. Furthermore, there are five predefined annotations in OWL, namely: *owl:versionInfo*, *rdfs:label*, *rdfs:comment*, *rdfs:seeAlso*, *rdfs:isDefinedBy.*

An OWL ontology document begins with a *namespace* declaration. A namespace is a collection of element types and attribute names that helps to make the rest of the ontology much more readable. These collections of attributes and names are uniquely identified by the name of the namespace of which they are part. A standard initial component of an ontology includes a set of XML namespace declarations associated with the ontology itself, the URI of the ontology and RDF, RDFS and XML scheme datatypes (since OWL depends on the construct defined by them). After the namespace declaration, an OWL document optionally includes a set of ontology headers (used to provide information about the version and the inclusion of other ontologies) and then a set of classes, properties and individual declarations.

2.3.2.3 OWL language elements

As already mentioned, the main elements of an OWL ontology concern classes, properties and individuals. This section presents more detailed information about the OWL vocabulary used to define relationships and constraints regarding the terms mentioned above.

Classes (I) : Class Descriptors

Class descriptors are used to precisely describe OWL classes. The following types of descriptors can be distinguished in OWL :

- ***owl:Class*(URI)**: a class can be simply described by its name, the URI. There are two special classes: *owl:Thing* and *owl:NoThing*. The first refers to a set of individuals. All OWL classes are sub-classes of this class. The second refers to an empty set of individuals. This latter class is a sub-class of every OWL class

- **Enumeration *owl:OneOf***: this construction enables a class to be identified by an exact list of individuals contained in this class.

- **Intersection, Union and Complement**: the *owl:intersectionOf* can be seen as the AND logic operator. This statement indicates that the new class extension contains members contained in class extensions of all classes included in the list. The *owl:unionOf* can be seen as the OR logic operator. This statement indicates that the new class extensions contains individuals contained in one or in another class extensions of all classes included in the list. The *owl:complementOf* can be seen as the NOT operator. This statement indicates that the new class extensions contains individuals that do not belong to the specified class.

- **Property Restrictions → Value Constraints**: these restrictions put constraints on the range of the property. The *owl:allValuesfrom* is used to specify that the values of the property are all instances of the class indicated by this clause. The *owl:someValuesfrom* indicates that at least one individual belongs to that class and *owl:hasValue* links the restriction class to a value that can be either an individual or data value.

- **Property Restrictions → Cardinality Constraints**: these are used to specify that any instance of a class may have an exact (*owl:cardinality*), or a least (*owl:minCardinality*) or a greatest (*owl:maxCardinality*) number of values for a particular property. In contrast with OWL-Lite, positive integer values other than 0 and 1 are permitted in OWL DL for cardinality restrictions.

Classes (II): Class Axioms

Class descriptions form the building blocks for defining classes through class axioms. The simplest form of a class axiom is a class description that just states the existence of a class, using *owl:Class* with a class identifier. There are other class axioms such as *rdfs:subClassOf* to define sub-class relationships, *owl:equivalentClass* used to express that the class extension of two classes descriptions are the same and *owl:disjointWith* which is used to express that two classes have no members in common.

Properties : Properties Axioms

As stated before, there are two types of properties: object and data properties. Additionally in OWL, property axioms define additional characteristics of properties. The following types of properties can be distinguished:

- **Axioms based on RDF Schema constructs**: these constructs can be used to specify the domain of a property, that is to say indicate that the subjects of such property statements must belong to the class extension of the indicated class description through *rdfs:domain* and the range of the property. *rdfs:range* is used to indicate that the values of that property must belong to the class description specified by the data range. Finally, *rdfs:subPropertyOf* is used to indicate that the property is a sub-property of the indicated.

- **Relations to other properties**: on the one hand, *owl:equivalentProperty* is used to indicate that both properties have the same domain and range but not meaning. On the other hand, *owl:inverseOf* is used to indicate that the relation to the domain and range class descriptors is the inverse.

- **Global Cardinality restrictions**: OWL provides tags to describe both functional properties (*owl:FunctionalProperty*) which indicate that the property can have only one unique value and for each instance x, and also inverse functional properties *owl:InverseFunctionalProperty.* These characteristics together with transitivity and symmetry characteristics define particular relationships between the subjects and the objects of the properties, and enable inference of particular facts.

- **Logical property characteristics**: finally, OWL provides transitivity and symmetry characteristics. Symmetry (*owl:SymmetricProperty*) specifies that an attribute relationship is valid in two directions. If a property P is Transivity (*owl:TransitiveProperty*) it specifies that P(x,y) and P(y,z) implies P(x,z).

Individuals

In contrast to OWL-FULL, in OWL-DL an individual can never be at the same time a class. It could be said that individuals are defined with facts, that is to say individual axioms. On the one hand, individuals can be described as members of a class and have property values. On the other hand, there are facts about individual identity. The OWL language does not use "unique identifiers" that is to say it does not make UNA (Unique Name Assumption). Consequently, OWL provides explicit tags to express whether URIs refer to the same individuals or not. It is possible to indicate that two individuals are the same by using *owl:sameAs*, or different by using *owl:differentFrom* and *allDifferent*.

Figure 2.3 provides a graphical representation of an OWL ontology (main elements) by means of UML blocks. This structure will be used latter for describing the ontology and thus facilitate comprehension of the model. UML classes represent OWL classes, attributes represent OWL DataProperties and the relations between classes represent OWL ObjectProperties. Cardinality restrictions are represented as strings such as 0..1 (zero or one), 1..* (one or more), 0..* (zero or more) and 1..1 (exactly one).

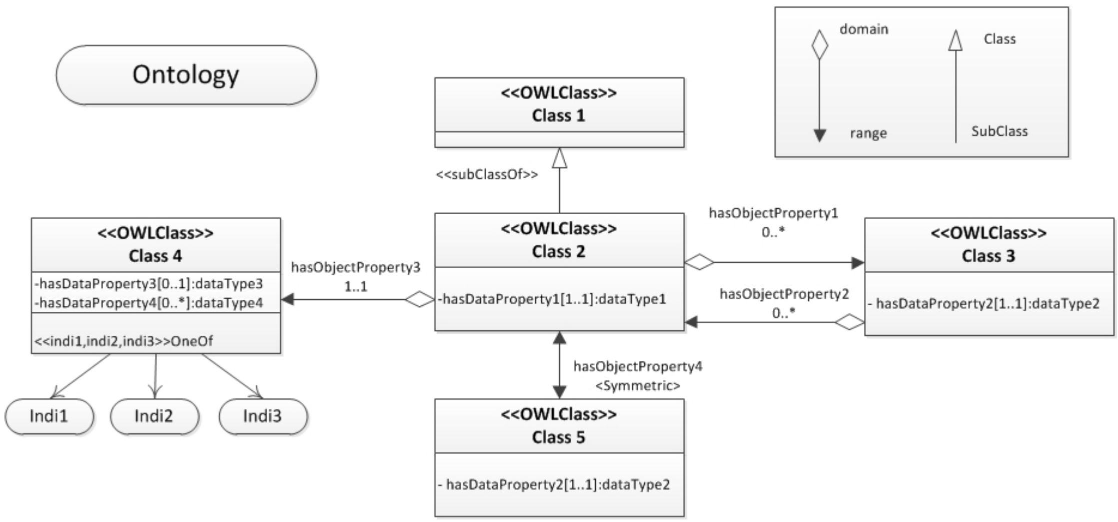

Figure 2.3: Ontology graphical representation

2.3.2.4 OWL 2 language

OWL 2 was proposed by the W3C in [71] as an extension of OWL 1. OWL 2 adds new functionality with respect to OWL 1. Some of its features are: keys, property chains, richer datatypes and data ranges, qualified cardinality restrictions, asymmetric, reflexive and disjoint properties and enhanced annotations capability.

- **New construct for Properties**: OWL 2 provides more constructs for giving information about properties such as restrictions on properties, new characteristics of properties, incompatibility of properties, property chains and keys.

  1) Self Restrictions *ObjectHasSelf*: this restriction is used to express relations between subclasses of a given class.

  2) Qualified Cardinality Restrictions *qualified*: these cardinality restrictions permit specification, apart from the number of instances, of the class or data range of the instances to be counted.

  3) Disjoint Properties: in the same way that classes can be disjoint in OWL, so can the properties in OWL 2.

  4) Property Chain Inclusion *ObjectPropertyChain*: OWL 2 permits to define properties as a composition of other properties.

  5) Keys: This key term refer to the same functionality that the "Primary Key" has for DDBB implementations. Keys are very important for applications in order to uniquely identify an individual by values that are key properties. By using key properties we can assume that, if K is a key and X and Y two individuals from a given class if (X,K) and (Y,K) then X and Y would be the same individual. OWL 2 allows a key to be defined by its construct *owl:hasKey*. Although OWL 1 does not provide a means to define keys, in OWL-DL it is possible by using Annotations Properties to define some properties as primary keys.

- **Extended datatype capabilities**: OWL 2 provides richer dataypes. For instance, it is possible to restrict the range of a datatype for an integer value and it is also possible to represent relationships between values for some datatype values, datatype range combinations and also new datatype definitions.

- **Syntactic sugar**: these new features are added to make some common patterns easier to write such as Disjoint Classes and Disjoint Unions. Furthermore, NegativePropertyAssertion is added to express that an individual does not have the values of a certain property.

- **Additional features**: extended annotations or metamodelling capabilities (allowing different uses of the same term such as classes and individuals). IRIs (Internationalized Resource Identifiers) are used for identifying ontologies and their elements and Declarations are introduced for explicitly declaring each new entity as part of the vocabulary.

OWL 2 provides 3 different profiles or sublanguages with useful computational properties or implementation possibilities: OWL 2 EL, OWL 2 QL and OWL RL. Developers will choose one or another to construct the ontology according to the expressiveness required by the application, the importance of the reasoning required and the size of the datasets.

- **OWL 2 EL**: this profile is adequate for developing ontologies that contain very large number of properties or classes but simple ontologies.

- **OWL 2 QL**: this profile and OWL 2 RL are suitable for lightweight ontologies in applications that use very large volumes of instance data. It will be chosen if reasoning is mostly done by queries.

- **OWL 2 RL**: this profile is adequate in applications that require some reasoning but also expressive power.

2.3.3 Reasoning

One of the main advantages of ontologies in contrast to other popular technologies to represent data is that it is possible to infer knowledge from them. By executing reasoning tasks, it is possible to extract knowledge that has not been explicitly declared in the ontology from what is already known. Hence, reasoning could be defined as a set of processes that enable the finding of implicit facts in the ontology given through explicitly stated facts. Nowadays reasoners can be classified in two groups: DL reasoners and logic program reasoners. The semantics reasoners are computing applications that allow generating knowledge and making inferences by using an inference engine and a set of rules expressed in semantic languages. These reasoners are based on different algorithms such as rule-based algorithms, tableaux (deductive method), datalog, resolution methods or first-order prover [72].

As stated in the introduction to ontology languages, DL allows the knowledge base (KBase) to be described through concepts and axioms. Specifically, this KBase is completed through the combination of the TBox and the ABox. A DL reasoner should be able to perform the following tasks [67, 72, 73, 74]:

- **Ontology Consistency**: ensures that the ontology does not contain any contradictory facts. This is the operation to check the consistency of an ABox with respect to a TBox, that is to say, to determine whether individuals in the ABox do not violate descriptions and axioms described by the TBox.

- **Satisfiability of a concept**: determines whether or not a description of a concept is contradictory. If a class is satisfiable, it will be possible to define an instance of the class and the ontology will continue being consistent.

- **Subsumption of concepts**: given an ontology O and two class descriptions A and B, this verifies whether the description of A is more general than B, meaning that B is a subset of A.

- **Instance checking**: checks whether the individual is an instance of a concept in the ontology given.

- **Classification**: according to axioms in the TBox, the reasoner computes the subclasses relations to create a complete class hierarchy.

- **Retrieval and Realization of individuals**: finds all instances of a concept and determines the most specific classes that an individual belongs to.

- **Conjunctive Query Answering**: returns the answers of a conjunctive query q with respect to the ontology.

Pellet [75], Racer [76], FaCT++ [77] or HermiT [78] are examples of good and popular reasoners. According to the ontology features and the reasoning application requirements, it will be more appropriate (for efficiency) to choose one or another.

2.3.4 Ontology Evaluation

Ontologies are engineering artifacts and, as any other, they should be evaluated [79]. The evaluation of the ontology is an important issue to be considered in order to determine whether or not the ontology is suitable for a certain application. Furthermore, ontology evaluation is the task of measuring the quality of the ontology. By evaluating ontologies, mistakes and omissions can be detected and the potential of the ontology can be fully exploited. It helps to develop good ontologies which have a higher degree of reuse and cooperation with other applications. Ontology evaluation also plays an important role during the development of the ontology. It helps developers to measure their results and guides them in the construct process. Evaluation also leads to an increase in the availability of the ontology and lower maintenance costs of the created knowledge.

In order to measure the quality of an ontology, as suggested in Brank et. al [80], it is often more practical to evaluate different levels of the ontology separately rather than to evaluate the ontology as a whole. According to Brank et. al, the following levels should be considered:

1. *Lexical, vocabulary, or data layer*: this level concerns concepts included in the ontology and the vocabulary used to represent these concepts. Precision and recall are commonly used to evaluate lexical content.

2. *Hierarchy or taxonomy*: this relates to the evaluation of the relationships between concepts included in the ontology.

3. *Other semantic relations*: to evaluate relations different to “is-a” relations.

4. *Context or application level*: this level should be carefully considered and it could be said that it is the most important level of evaluation. As stated in [81], there is not only one correct way to model a domain and the best solution depends on the application where it is to be used. As long as the ontology helps to provide good results for a given task, it could be said that the ontology is good for that

application [80]. Ontologies may therefore be evaluated simply by plugging them into an application and measuring the achievements obtained.

5. *Syntactic level*: this should be considered when the ontology is manually developed in order to verify that the language is formal and that it matches the syntactic requirements of that language.

6. *Structure, architecture and design*: this level assesses whether the ontology fits with the pre-defined principles for the design.

However, there are no standard criteria to evaluate ontologies and different frameworks have been presented. See for example [79], where an evaluation framework based on eight ontology quality criteria is proposed:

1. *Accuracy*: this criterion states whether the axioms of the ontology are good enough to describe the knowledge of the domain. The correct definition of classes, properties and individuals leads to higher accuracy.

2. *Adaptability*: this criterion measures how far the ontology anticipates its uses.

3. *Clarity*: this measures how good the ontology is in terms of understandability of the defined elements.

4. *Completeness*: this criterion is used to determine if the domain is well covered. Competency questions (used to design the ontology) should be answered with the ontology.

5. *Computational efficiency*: this measures the efficiency of semantic tools (speed of reasoners or own applications) to deal with the ontology. The size of the ontology, axioms and complex reasoning tasks strongly influence this measurement.

6. *Conciseness*: this criterion is used to determine if the ontology includes irrelevant concepts or redundant representations.

7. *Consistency*: checks if the ontology includes contradictions.

8. *Organizational fitness*: this criterion is used to decide whether an ontology is easily deployed within an organization.

As will be explained in the next chapter, a combination of both approaches was taken into account for the ontology evaluation. These ontology quality criteria were considered when evaluating different levels of the ontology. Specifically, application level evaluation was considered in our methodology approach. It should be noted that the aim of this ontology evaluation is not to obtain a metric to quantify how good or bad the ontology is. The aim is to consider a set of factors that help to improve the ontology and to determine if it is well formed and appropriate to be applied to a certain domain and purpose.

2.3.5 Ontologies in HealthCare and Telemonitoring Scenarios

Ontology definition has gained in importance in recent times for knowledge representation and semantic integration in many areas such as home-smart environments [82], network management [43, 83] and also in health care. The development of new ontologies has increased and consequently their relations and combined use motivates researchers today to investigate mapping and automatic mapping process techniques in order to provide a common layer from which several ontologies may be used together [84].

The use of ontologies in the medical field is extensive not only to define controlled medical vocabularies such as SNOMED-CT [54] or UMLS (Unified Medical Language System), but also to facilitate communication and solve interoperability problems when different entities deal with the same data [85, 86, 87, 88, 89]. Ontologies provide a high level of abstraction and expressiveness to define concepts and the knowledge involved with precision, thus making data integration easier in scenarios such as telemonitoring where diverse heterogeneous sources take part in the same process.

There is a wide range of health care applications where ontologies have been proposed as a solution to achieve integration, interoperability and, ultimately, common shareable knowledge. On the one hand, see for example the AGT-MO ontology [90] used to represent the domain of cancer research and management. This ontology is the main part of an infrastructure used to address the efficient execution of discovery-driven scientific workflows in the context of multi-centric, postgenomic clinical trials. On the other hand, a different application of ontologies in healthcare can be seen in [91] where the use of ontologies is proposed to describe the organizational knowledge of a real healthcare system, thus describing its management and the interaction among the actors. Finally, the use of ontologies for the implementation of clinical decision support systems (CDSS) is widespread [92, 93]. See, for example, [92] where the design and implementation of a clinical decision support tool for risk management in hemodialysis is proposed. Another good example can be seen in [93] where the combination of an ontology together a preoperative risk assessment tool is used to provide recommendations (test and clinical precaution protocols) and support functionalities including risk assessment. Later publications have even addressed EHR interoperability and standards heterogeneity (OpenEHR [32], EN13606 [94] and HL7 [30]) used to transfer EHR information through an ontology based-system [95]. There is no doubt that ontologies are key technologies used in healthcare for seamless integration purposes.

Ontologies have already been successfully applied specifically in the telemonitoring domain and other solutions have been reported regarding remote control of patients with chronic conditions [85, 89, 88, 86, 96]. On the one hand, see for example the 4KCARE (Knowledge-Based HomeCare eServices for an Aging Europe) project [89], which includes the use of ontologies for representing actors [97], diseases and symptoms and also formal intervention plans [98]. Another example can be seen in [86] where a context management middleware to support health information exchange and alerts handling for patient supervision in telemonitoring scenarios is presented. An interesting ontology-based solution is reported in [96] where a context-aware framework executed on embedded wearable systems is proposed. Context-aware and personalized care are two significant challenges addressed in this work. Good examples of ontology usage in the telemonitoring domain for specific chronic patients can also be found in [85] and [88]. An ontology-based system to monitor patients suffering from a degenerative disease such Alzheimer's is proposed in [85] and for dementia supervision in [88]. On the other hand, [99] provides a detailed methodology to develop an ontology in the patient profile domain of complex chronic health conditions. It contributes with a rich description of symptoms, standardized vocabularies and knowledge profiles.

Some of the described solutions propose the combination of rules with the ontology model to provide personalized care, such as [86] or [96]. Although both are based on ontologies to achieve data integration, these approaches lack the provision of a clear methodology to describe and explain the sequence of tasks for processing acquired data, that is to say the workflow that dictates the management service (e.g telemonitoring) process. Complexity is usually an obvious handicap for the widespread adoption of these types of proposed solutions.

2.4 Rules

Ontologies are used to express knowledge in a static and declarative way as a set of things that are true. Furthermore, the expressivity of the languages as RDF and OWL has limited the potential of semantics reasoners. Hence, generally, developed solutions combine knowledge presented in ontologies with dynamic knowledge presented by the use of rules [86, 100, 101].

2.4.1 Language rules

Most solutions that include rules are based on the use of SWRL (Semantic Web Rule Language) [102] in combination with OWL. Nevertheless, there are other popular languages such as Rule-ML (Rule Markup Language) language [103] or DL Safe Rules [104], and choosing between them is conditioned by the characteristics of the ontology and the practical application.

- **SWRL (Semantic Web Rule Language)**: this language is based on the combination of OWL DL and Lite with Rule-ML . The main purpose of this language is to extend the set of OWL axioms by including *Horn-Like* clauses (*If..Then..Else*) that can be combined with the OWL knowledge base. In its abstract syntax, a SWRL rule is composed of an antecedent (body) and consequent (head) pair. That is to say, a SWRL rule can be expressed as: antecedent →

consequent. In this way, according with the conditions specified in the antecedent, the conditions specified in the consequent should be held. It should be noted, that the antecedent for each rule is a conjunction of one or several literals [102]. Protégé editor can be used to define these rules and there are tools that can be used to handle them such as the Bossam Java inference engine [105].

- **Rule-ML language**: this is a markup language developed to express rules in XML for deduction and inference tasks [103].

- **DL Safe Rules**: DL-safe rules are SWRL rules that are restricted to known individuals. They can be seen like a sub-set of SWRL used to solve decidable problems of SWRL. They are supported by the KAON2 reasoner.

2.4.2 Rules for personalized services

A system based on the use of rules usually contains a set of if-then rules (which indicate what should be done according to a situation) and a rule engine used to apply them. By using rules, the behaviour of individuals can be expressed inside a domain. Hence, they can be used to generate new knowledge and can also be used to provide personalized services. A good experience of the combination of ontologies and rules can be found in [101]. Reasoning based on SWRL rules is proposed in this work to infer useful services and thus propose and recommend personalized services to each customer. These rules are described in general terms. The combination of ontologies and rules has also been applied in the healthcare domain as part of a decision support system [106, 107, 86]. For example, the work presented in [86] defines a set of pre-defined rules to make inferences over the ontology and determine if an alarm has been triggered according to some threshold values. These rules can be changed according to each patient's situation, hence they can be used to provide personalized care services.

The use of rules in combination with ontology instances has been studied in this thesis to provide personalized care services and personalized management tasks in general. However, in contrast to most solutions, SPARQL (SPARQL Protocol and RDF Query Language) has been proposed to express the rules [108]. Furthermore, few approaches have studied the use of SPARQL to express rules. These include [109] and [110] which provide a framework for users to exploit the rich expressiveness that this language offers (enabling a wide variety of expressing rules). In this work, suggested theory about the expressiveness SPARQL offers has been put into practice by using it to define rules to express conditions over the ontology instances. Furthermore, the inclusion of some SPARQL rules in the ontology model has been proposed in this work to express functions.

2.4.3 SPARQL as a rule language

SPARQL is established as the standard query language for obtaining information from RDF graphs. This query answering mechanism is very important as it provides a manner for users and applications to interact with ontologies and data [111]. Although other query languages have been proposed such as RDQL (RDF Data Query Language) or SeRQL (Sesame RDF Query Language), this language proposed by the W3C in 2008 has been adopted by most RDF based applications. Nowadays, a second version of this query language, SPARQL 1.1, is currently under development [108] [112].

The language is based on the RDF triple patterns idea and the main mechanism for computing query results in SPARQL is based on subgraph matching by using the RDF triples in both the queried RDF data and the query pattern. It should be noted that SPARQL does not deal with nested queries or recursion and that there are a lot of tools and frameworks to deal with them. Many RDF APIs and databases come with SPARQL support (such as Jena ARQ [113]). From a practical point of view, this doubtless constitutes an advantage over other languages proposed in the semantic web.

There exists no standard OWL query language. Nevertheless, it is possible to use SPARQL queries successfully against the structure of the OWL ontology. Since SPARQL is a query language for RDF, it should be considered how OWL is represented in RDF. This means that OWL should be translated to an RDF graph. This conversion process is done by using entailments relations. It is thus said that an ontology O entails S written $O| = S$ when all interpretations that satisfy O also satisfy S. So far there has always been a straightforward correspondence between one triple and one OWL axiom. However, this is not a direct process and there are also different ways of defining a graph pattern matching extension for an entailment relation [114]. Related problems and challenges are nowadays faced by researchers and developers of reasoner applications. By using an OWL DL SPARQL engine (such as ARQ Jena engine), SPARQL is useful for querying the structure of the ontology. Although OWL-DL axioms do not always result in a unique deductive closure, SPARQL has been used successfully in this research to query triples entailed from subclass, range, domain, and other relations and for filtering individuals with specific features by using the ARQ Jena. This engine is responsible for the OWL entailment, providing the impression that certain triples exist and thus applying the SPARQL queries and returning a result.

A SPARQL query is a quadruple Q = (V, P, DS, SM), where V is a result form, P is a graph pattern, DS is a dataset, and SM is a set of solution modifiers [109]. The simple way of using a SPARQL query is by a structure of `SELECT-FROM-WHERE` clauses. In this way, the result form represents a set of variables appearing in the `SELECT`. SPARQL queries are executed against the DS which represents a collection of RDF graphs. An RDF DS comprises one graph that is called the default graph, and zero or more named graphs identified by an IRI. These IRIs correspond to the concept of URI identifiers from RDF documents (URIs provide a common syntax for naming a resource regardless of the protocol used to access a resource). The DS forms the `FROM` part. By using the FROM and FROM NAMED clauses, the dataset used for the query will consist of the default graph and the set of (IRI, graph) pairs. Finally, the graph pattern forms the `WHERE` part which is constituted by a set of RDF triples.

An example of a simple SPARQL query is depicted in Figure 2.4. It should be noted that the PREFIX keyword associates a prefix label with an IRI. Prefixes rdf, rdf, owl and xsd refer to RDF, RDFS, OWL, XML Schema Datatypes. Although they are not involved in the execution of the depicted query, by using SPARQL it is possible to query their native restrictions.

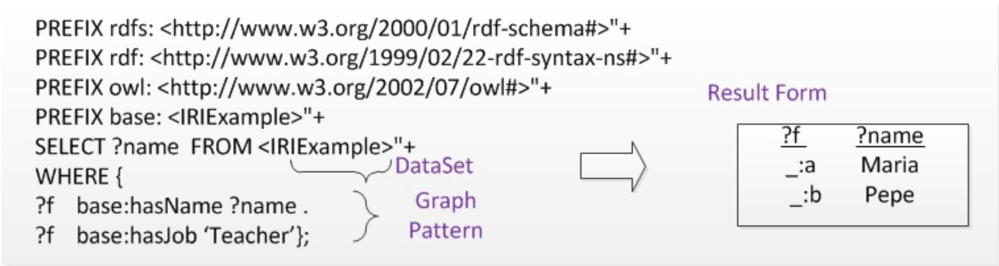

Figure 2.4: Example of a SPARQL Query

Although SPARQL is a query language with a `SELECT` keyword, it can be regarded as more than this due to the fact that it provides more query forms that enable its functionalities to be extended. This `SELECT` keyword returns all, or a subset of, the variables bound in a query pattern match. SPARQL also provides the means to check whether certain conditions currently hold in an RDF model (`ASK`), to derive new triples from existing triples (`CONSTRUCT`), and to perform `DELETE` or `DESCRIBE` operations. Although the simplest form of `WHERE` consists of a basic graph pattern, it is possible to construct complex patterns by using `UNION`, `OPTIONAL` and `FILTER` clauses.

In fact, SPARQL can be used as a rule language by combining `CONSTRUCT` clause and `FILTER` restrictions [109]. On the one hand, the `CONSTRUCT` query form returns a single RDF graph. This graph is built based on the results of matching with the graph pattern of the query and by taking the specified graph template. On the other hand, the `FILTER` clause can be used to restrict solutions to those which the filter expression considers as TRUE. Only if the filter function evaluates to true is the solution to be included in the solution sequence. A SPARQL query `FILTER` can even be used to restrict arithmetic expressions. As it can be seen in Figure 2.5, an If-Then SWRL rule expressed over an instance could be easily converted into a SPARQL query rule. Consequently, SPARQL language can be used as a rule language because it provides great power to filter individuals with specific characteristics. SPARQL rules can be defined in order to personalize services according to specific individual behaviours. This expressiveness capacity and the tools available to deal with it were the main reasons for choosing this language to express rules in this research.

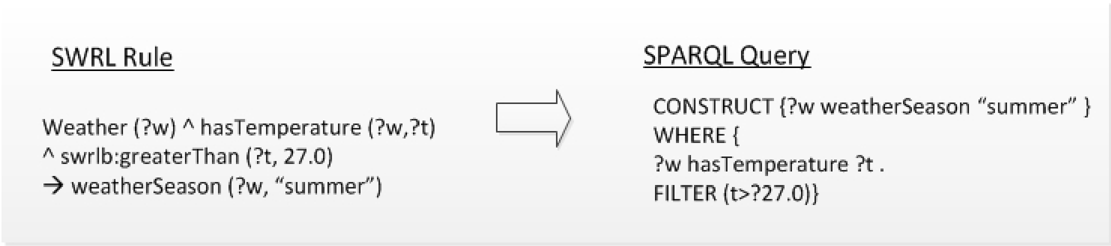

Figure 2.5: SWRL rule to SPARQL query

2.5 Autonomic Computing Paradigm

Advances in technology and networking have resulted in the growing dynamic and heterogeneous information services present in many aspects of our lives. As a consequence of this growing communication infrastructure, the complexity of required management tasks has been significantly increased [115]. As a solution, IBM (International Business Machines Corporation) started in 2001 an alternative paradigm for system application design and management called the autonomic computing paradigm. This paradigm was based on the idea that systems had to develop self-management capabilities avoiding administrators for maintaining computer systems. The autonomic computing paradigm was inspired by the human autonomic nervous system. This human systems reacts to every change in the environment in order to maintain internal balance. Based on this system, an autonomic computing system will modify its behaviour according to changes in its environment. Consequently, the goal of this paradigm is to achieve that computing systems can manage themselves in accordance with high-level guidance from humans [36] [116].

This IBM solution is based on the division of actions that are taken when trying to provide autonomy to a process. These autonomic systems are able to self-control their internal functions and operations thanks to the autonomic element which constitutes the main core of the autonomic system. The autonomic element implements a closed-control loop that dictates the work flow of the system behaviour [36]. Implementing this classic control loop, also known as the MAPE process (Monitor Analyze Plan Execute), the state of the resource domain is first monitored, second analyzed, then a plan to react to that event and correct it is formed and finally it is executed. Since the automatic behaviour may be considered as a closed-control loop, automatic means that no external intervention is needed for the autonomic element to close the loop. Consequently, a typical architecture of an autonomic element would include, as can be seen in Figure 2.6, a monitoring module, an analysis module, a planning module, an execution module and a knowledge module containing the necessary description of the domain that the autonomic system requires. The control loop only handles known environmental conditions and is based on the knowledge module embedded in the element. This knowledge engine contains the mapping between the environmental states or conditions and the correct behaviour. An autonomic element will typically consist of one or more managed elements and a single autonomic manager that runs the closed control loop in order to control managed elements. By using sensor modules, managed data is acquired and then processed, and by using the effector modules, required actions are executed. As can be seen in Figure 2.6, these latter modules are also related with external users (administrators) and policies used to guide their performance.

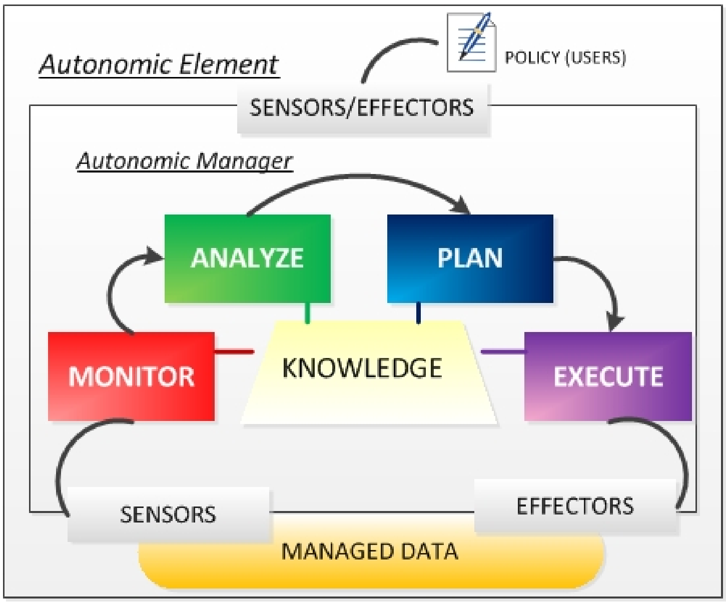

Figure 2.6: Autonomic Computing System Architecture

The behaviour of an autonomic application is characterized by 3 properties: autonomic, adaptive and awareness. Autonomic means that these systems self-manage their internal behaviour according to policies that human or other elements have previously determined. This kind of system generally monitors its own detecting and reacting to anomalous findings in its performance [36]. Moreover, adaptive means that these systems check for upgrades in order to cope with temporal and spatial changes. Finally, aware indicates that their operational behaviour will be adapted in response to context or state changes. The essence of autonomic systems is self-management. According to the IBM definition, this feature is supported by four primary properties: self-configuring (autonomic systems should be able to configure and reconfigure themselves under different conditions), self-optimizing (autonomic systems should be able to detect optimal behaviours and adapt themselves), self-healing (autonomic systems should be able to detect problems and also to recover from them) and self-protecting (autonomic systems should be able to detect and protect themselves against external attacks) [117]. An autonomic system implements self-management capabilities by using the closed control loop described above to collect information, make decisions, and adapt and react if necessary [115].

Implementing autonomic systems implies addressing challenges at different levels. First, at a conceptual level the main challenges include appropriate definition of abstract models to describe the behaviour of the autonomic system. Then, at the software and middleware levels, implementing autonomic element architectures requires addressing challenges in terms of security, messaging and self-management property implementation.

As will be explained in the next chapter, this paradigm has been used to inspire the structure of the ontology developed in this thesis and is consequently the base of the architecture proposed.

2.5.1 Using the MAPE model: a review of the state of the art

Motivated by the large scale of computing and heterogeneous systems with the complex connectivity and interactions that comprise communications networks, today's researchers and the IT-industry have proposed substantial innovations in network management. A wide range of autonomic computing-based solutions have already been developed. Most of these solutions aim to decentralize management tasks, thus offering a flexible and agile response based on self-management ideas [118, 119].

Ontologies have also been studied for autonomic computing-based solutions [120, 121]. In particular, the use of ontologies has been studied as high-level models for describing the knowledge upon which the processing engine is based [122]. Regarding this application, ontologies have been used to provide knowledge formalization in combination with the above-mentioned paradigm to organize processes in other works such as [101, 123] for different application fields. See, for example, [101] where an autonomic system has been developed for managing a home area network and providing personalized services to customers. A different approach is presented in [121] where a knowledge delivery service is proposed to deliver knowledge to nodes from an autonomic network interested in that knowledge.

Autonomic systems may require knowledge from different sources and this needs to be represented in a common and formal way. Thus, ontologies have been used in autonomic applications as core knowledge used to represent manipulated data. This paradigm that has been widely used in the engineering and management domain could also provide advantages and constitute a good innovative solution for the healthcare domain.

2.6 Web Services Technologies

The architecture proposed in this thesis is based on the use of web services technologies to transfer information between both end sites. According to the 3WC, a Web Service (WS) is a "software system designed to support interoperable machine-to-machine interaction over a communication network". To achieve this goal, each web service has an interface written in a machine-processable format (as a rule, the Web Service Description Language (WSDL)) where the functionalities of the web service are described. Systems may interact with the web services by exchanging SOAP (Simple Object Access Protocol) messages serialized in XML for its message format and sent over other application layer protocols, usually Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP) [44]. Although SOAP-based web services are the most popular types of WSs, as described in the following subsections there are other styles of programming a WS.

2.6.1 Web Services platform elements

A SOAP-based web service platform can be described by its main elements: WSDL, SOAP and UDDI (Universal Description, Discovery and Integration).

- **WSDL**: this is an XML based language used for defining the functionality of web services. A WSDL document contains a set of definitions, operations and messages that are described abstractly and will be used to define the interaction with the web service [124]. Specifically, a WSDL document contains types (datatype definitions used in the messages exchanged), messages (in WSDL 1.1, abstract definition that contains the information needed to perform the operation), portType (defines the abstract operations to be executed), binding (specifies the SOAP binding style transport protocol), port (defined as endpoint in WDSL 2.0, specifies an address for a connection to the Web service, typically a HTTP URL string) and service (used to aggregate related ports).
- **SOAP**: this is a protocol that relies on XML for exchanging information in Web service implementations. It is extensible, can be used over any transport protocol and is independent of programming models. A SOAP node can be the initial SOAP sender, an ultimate SOAP receiver, or a SOAP intermediary. SOAP can be considerably slower than other middleware technologies such as CORBA. A SOAP message contains the following elements: an envelope that identifies the XML document as a SOAP message, a header, a body element that contains the information and a fault element containing errors information.
- **UDDI**: this is an XML-based registry for storing and locating web service applications. It attends to SOAP messages and provides access for WSDL documents describing web services.

2.6.2 Styles of use

Web services can be used in 3 common styles of architecture: Remote Procedure Call (RPC), Service-Oriented Architectures (SOAs) and Representational state transfer (REST).

- **RPC**: a remote procedure call allows programs on different machines to interact using a simple call/return procedure. Hence, it allows a computer to cause a subroutine in a remote machine and then obtain a result. RPC uses the client/server model. The requesting program is the client and the server is the service-providing program where functions are to be executed. Hence, an interface with a set of named operations with designed types should be specified in the server. In this way, when the client requires, it initiates an RCP by sending a request message to the server in order to execute a procedure. Then, the server sends the response to the client. When the software uses object-oriented principles, RPC is called the remote method invocation. This method is widely accepted, used and understood.

  Similarly to web services, many other different technologies can be used to implement this technique (eg. CORBA). In particular, for a WS implementation the interface is described by WSDL which constitutes the basic unit of RPC web services. The workflow of a RCP request in the web service architecture is depicted in Figure 2.7. This workflow is as follows: 1) when the client program needs to invoke the WS, it first calls the client stub. The stub is a piece of code used for converting parameters passed during a RCP. The stub generates a SOAP request and interprets SOAP responses, thus it turns the local call into the proper SOAP message. Then, this request is sent over the network usually by means of HTTP. Once the request is handled in the web service, the server stub known as the skeleton interprets the SOAP request and invokes the service implementation. Then, the result will be handled again by the skeleton, wrapped in a SOAP message and sent over the network using HTTP.

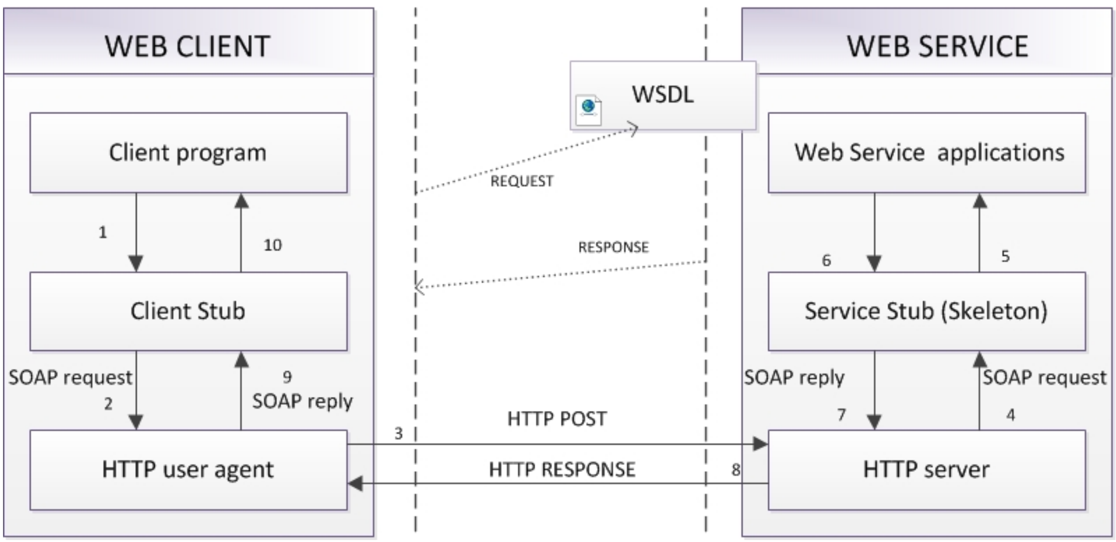

Figure 2.7: RCP Web service architecture

- **SOA**: web services can be used to implement an architecture based on service oriented architecture (SOA). SOA is an architectural style for building software applications in the form of interoperable services. The main idea behind SOA is to define service functionalities and register them into service repositories so as to be shared, queried and reused by a client. The most important principle of SOA is interoperability. By adopting SOAs using Web service technologies, enterprises can flexibly solve enterprise-wide and cross-enterprise integration challenges. Regarding web service technologies, a SOA is essentially a collection of services which communicate with each other based on the RPC-style with the difference that the basic unit of communication is a message rather than an operation. Services are described using WSDL and published in a UDDI registry so that they can be found and used by web service clients. This searching process is done by looking in the registry for services that match certain criteria. Each SOA entity can play one or two roles: Service provider (usually web services) and Service Customer (usually web client).

- **REST**: this is a style of software architecture for distributed hypermedia systems such as the World Wide Web first defined in 2000 by Roy T. Fielding. REST architectures consist of clients and servers. This style is based on the idea of transferring representations of resources, a resource being any item of interest. Hence, requests made from the client and responses from the web service are used to transfer resource information. Although REST was initially described in the context of HTTP, this paradigm can be applied to other protocols or implementations. Web services can also be described using this style. The key advantages of the REST architecture are scalability of components, generality of interfaces and independent deployment of blocks [125]. Although WSDL (version 2.0) can be used to describe the web service, it is not the best choice for documenting a REST service. The alternative is WADL, the Web Application Description Language [126]. This language is lightweight and easier to understand than WSDL, although not as flexible. REST web services include the following characteristics:

  1. Client-server interaction.
  2. Named resources: each resource is identified through an URI (uniform resource identifier).
  3. Stateless: each request made is independently and thus each client's request contains all the information needed to understand that request.
  4. Uniform interface: all resources are accessed (by using their URI) with a generic interface by using generic methods. For instance, an HTTP request and response would be performed through four operations: GET, POST, PUT and DELETE.

5. Layered system: clients do not know if they are connected to the end server or to an intermediate server.
6. Cache: Responses must be labelled as cacheable or non-cacheable in order to eliminate unnecessary information to improve the performance.

One of the advantages of REST architectures is that SOAP packages are avoided so the message payload is decreased. REST architectures can use the HTTP protocol directly in data exchange providing fast interactions. This new abstraction makes information available using the same rules and provides improvements in terms of scalability, coupling and performance. By defining uniform interfaces, the complexity of semantics is reduced. Messages used in a REST architecture are self-describing and can be easily understood. On the other hand, the advantages of SOAP-based architectures are the definition of complex methods and the use of other transport layers apart from HTTP such as SMTP. In contrast to a REST architecture, an RCP-style is composed of an interface defined by users for the particular service it provides. Each interface defines its own semantics and operating parameters. Both approaches have advantages and disadvantages and it is up to developers to make a decision. The choice between both styles will be determined by the service and the scenario conditions. Appropriate areas for REST architectures are those with limited bandwidth resources, stateless operations and caching situations. Appropriate areas for SOAP-based architectures are those with asynchronous processing, high security demands, formal contracts between both sides and stateful operations.

2.6.3 Network management with web services

Since 2002, web services have been investigated as an alternative network management technology. This solution, which takes advantage of XML technology, has been proposed in order to overcome limitations of traditional centralized management approaches in terms of flexibility, scalability and general deployment concerns. Web service technologies provide a standard means of inter-operating between different software applications and constitute an interesting solution given that the technology is generic, supported by many vendors and available in a variety of platforms and/or frameworks. Furthermore, one key for the success of this technology is that there are many tools for its implementation, and integration with other applications is simple for developers [127].

In order to evaluate this new functionality, several investigations have been carried out in order to compare the performance of web services and SNMP for network monitoring. For example, in [128], the performance of web services was compared to CORBA and SNMP, concluding that web service technology was promising although it had more overheads than SNMP. Some other studies have specified that web services consume less bandwidth than SNMP when a large number of management objects needs to be retrieved, thus concluding that SNMP was a better option for retrieving few objects [129]. Other investigations have covered a wide range of evaluation points (not only bandwidth). For example, in [127], apart from bandwidth usage, CPU time, memory and round trip delay were investigated. In terms of bandwidth consumption, it was concluded that SNMP was better for both large and single numbers of objects requested. Nevertheless, this conclusion differed when compression was used. In this case, web services provided better results for retrieving large number of objects. It was also concluded that although SNMP message coding was less expensive, it was not a main factor that would determine performance, and web services could therefore not be discarded for network monitoring. Furthermore, more advanced data structures can be designed with web services and XML than with SNMP.

2.6.4 Ontologies and web services

Ontologies have been successfully used to provide clear descriptions of web service definitions and thus facilitate their automated discovery and composition. These research efforts have led to the development of so-called semantic web services. Ontologies that have become popularized include OWL-S and WSMO [130, 131]. On the one hand, OWL-S is an ontology for describing Semantic Web Services. It has three main parts that describe what the service does (service profile), how a client can interact with the service (process model) and the details that the client needs to interact with it (service grounding). On the other hand, WSMO provides a conceptual framework for describing relevant aspects of web services to make the process of automatically discovering web services easier. It has 4 main components: goals, ontologies, mediators and web services.

In this thesis, ontologies and web services are used in a different way. Web services are used as the software technology to access and exchange information described in an ontology. Although there are many works that describe the details of the ontology approach, they do not devote very much attention to the architecture implementation used to exchange the information described. Web services have been used in other works to exchange data contained in an ontology, such as [86, 88]. However, our proposal (described in chapter 3), inspired on the REST style, provides a different design approach that may be reusable for other systems based on ontologies.

2.6.5 Security in Web Services

Nowadays there are two possibilities for providing security to a web service implementation: WS-Security (Web Services Security) and TLS (Transport Layer Security) or the previous SSL (Secure Socket Layer). It should be noted that WS-Security can only be used in SOAP-based architectures, thus security in REST architectures relies on HTTPS.

- **TLS or SSL**: the most common schema of web service security is using TLS over HTTP, thus sending messages over HTTPS. Nevertheless, this option has some limitations for web service implementations, and XML-based solutions have been proposed to address these problems and ensure end-to-end security. TLS provides point-to-point security that is sometimes not enough for web services if several intermediary nodes exist between two endpoints.

- **WS-Security**: this set of specifications provides a rich variety of enhancements to SOAP messaging for applying security to web services. These mechanisms provide messaging integrity, confidentiality and authentication [132]. WS-Security uses security tokens to provide authentication. These tokens can be signed (using, for example, X.509 certificates [133]) and unsigned (such as in login/password). Message integrity is addressed by using XML signatures. Finally, confidentiality is achieved by adding XML encryption. The main problem of this security alternative is the overhead introduced to encode keys and message signatures. TLS can reduce this overhead significantly and that is why it is still being used.

2.7 SNMP for technical management

In addition to clinical issues, technical issues are also important and should be considered within a telemonitoring scenario. Two solutions have been investigated in this thesis to address technical management: one based on ontologies and the other on the SNMP architecture. This latter architecture was chosen because it provides an external manager (EM) with a standard interface that eases the communication between network devices and with structured data through MIB (Management Information Base) definitions. In contrast to other common technologies for network management such as CORBA [134] or NETCONF [135], SNMP is a widely extended management architecture implemented by most manufacturers of devices. Moreover, on the one hand, CORBA architecture seems to be more suitable in applications where there is a large amount of data to be exchanged and memory consumption is not a major problem [136]. On the other hand, NETCONF is an attractive approach based on XML that aims to overcome SNMP limitations such as configuration tasks. Nevertheless, there is a lack of support from the industry for these architectures and in this scenario they do not have major advantages over SNMP architecture [135] .

2.7.1 Simple Network Management Protocol

Since SNMPv1 was proposed in 1990 in [137] as a simple application layer protocol that implements communication between a management console and the managed agents, it has been the most popular standard for managing TCP/IP networks. This original version of the protocol rapidly gained popularity and was widely deployed.

However, its deficiencies led to the development of a second version of the protocol, SNMPv2. This second version, issued initially in 1993, incorporated functional enhancements for transferring bulk data but still lacked a security facility. Finally, after many proposals, in 1998 SNMPv3 was defined as a security capability to be used in conjunction with SNMPv2 (preferred) or SNMPv1. This latest version of the architecture, SNMPv3, includes a rich array of security functions such as network security and access control [138].

One of the main reasons for the popularity of SNMP is that it was designed to be easy to implement and consumes minimal computational and network resources. This application protocol makes use of TCP/IP communications facilities and typically operates over the user datagram protocol (UDP), although it may also operate over TCP. But SNMP is much more than a protocol for the exchange of management information. It provides a complete management architecture defining also a format for representing management information and a framework for organizing systems into managing systems and managed agents [42]. It should be noted that SNMP has also been successfully applied outside the network management field domain and, for example, some applications based on SNMP have been developed to deal with clinical data transferences from the home site to the health-care site [139, 140].

The SNMP architecture can be described by the following key elements (see Figure 2.8): 1) a managed entity (agent) and a management station (manager) which are SNMP entities, 2) a set of SNMP messages within the network management protocol function and 3) the MIB.

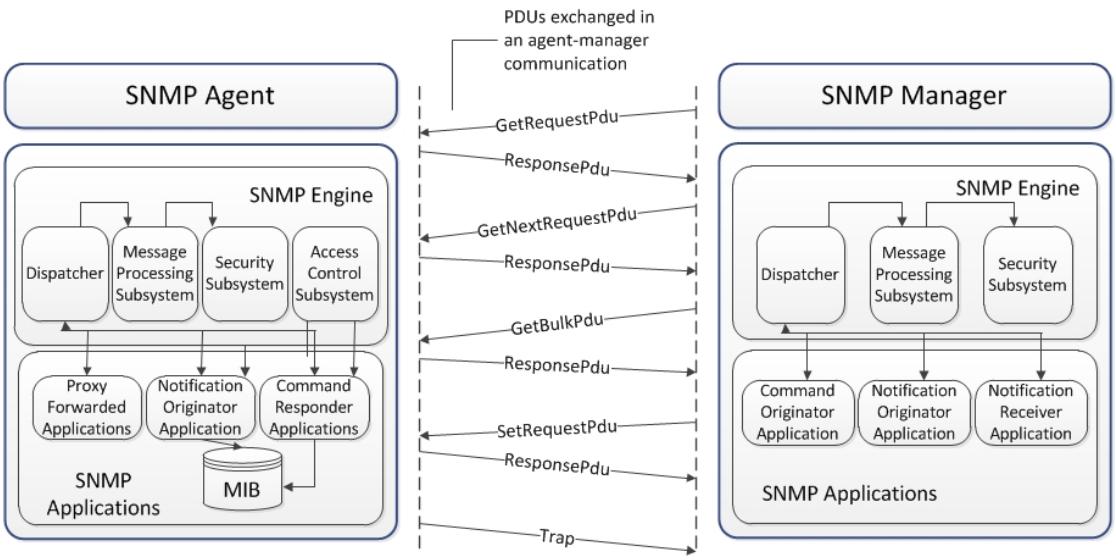

Figure 2.8: SNMP agent/manager architecture

2.7.1.1 SNMP entities

The SNMP architecture is based on the interaction of a collection of SNMP entities. Each SNMP entity contains a set of modules that interacts to provide services. Specifically, each entity contains an SNMP engine which is capable of sending and receiving SNMP messages and a set of SNMP applications. The role of each SNMP entity will be determined by the modules implemented in that entity. Each entity may act as an agent, a manager, or a combination of the two (thanks to this modular implementation an entity can perform both management tasks). SNMP allows for multiple agents to talk to multiple managers and it thus allows having multiple managers polling the same agent for required information.

- **Agent**: this is an SNMP entity used to collect information about its local environment, that is to say, the features and conditions around the device where it is running. All this information that can be managed in a device, is stored in an extensible data base design called MIB. An SNMP agent is capable of responding to manager commands and providing management information accessed through the MIB. It is also capable of sending unsolicited information to the manager if an abnormal finding is detected.

  In terms of SNMPv3 terminology, a traditional agent may include 3 applications: 1) Command Responder Applications used to respond to incoming requests from the manager; it will thus access the MIB to retrieve managed data; 2) a Notification Originator Application used to send asynchronous messages to the manager and 3) a Proxy Forwarder Application to exchange messages between the entities.

- **Manager**: this is an SNMP entity used to manage and retrieve information from an SNMP agent. By polling the SNMP agent will access information stored in the MIBs. The manager station is used as an interface for the human manager to manage the network system. In terms of SNMPv3 terminology, a traditional manager may include 3 applications: 1) Command Generator Applications used to generate messages for retrieving information to be sent to the remote agent, 2) a Notification Originator Application to generate asynchronous messages to be sent to another manager and 3) Notification Receiver Application used to process asynchronous messages such as traps or inform messages.

The SNMP engine provides services between the transport layer and the SNMP applications. For instance, it encapsulates the PDUs (Protocol Data Units) into messages for transmission and also in the inverse manner to process information. This SNMP engine for an agent is the same as the engine used for the manager plus an Access Control Subsystem in the case of an SNMPv3 agent.

2.7.1.2 MIB

One of the most powerful features of SNMP is the MIB structures definition. MIB is a virtual database that organizes hierarchically a collection of information used for managing the entities in a communication network. MIBs describe in a tree-like structure the management data of a device system, identifying each managed object with an OID (Object Identifier). This identifier consists of a sequence of integers (non-negatives) and points that uniquely identifies a managed object inside a MIB. When a manager needs to access a specific object from the agent it will ask for its OID. In fact, when referring to a collection of management information available in an entity it is more correctly described as an MIB-module. MIB-modules contain different types of managed objects. A managed object is one of any number of specific characteristics of a managed device. In SNMP there are two types of managed objects: scalar objects and tabular objects. Scalar objects define a single object instance whereas tabular objects define multiple related object instances grouped in MIB tables [42]. Objects in the MIB are defined using a subset of ASN.1 called the Structure of Management Information Version 2 (SMIv2) [141]. This structure defines the general framework within which a MIB can be defined and constructed.

An SMI document can be divided into three parts: module definitions, object definitions and notification definitions:

- **Module Definitions**: the first part of the document is used for describing information about the MIB-module. The MODULE-IDENTITY macro should appear just once in the document and is used to provide the following information by using specific clauses: date and time of the last time the MIB-module was edited (LAST-UPDATED), the organization under which it was developed (ORGANIZATION), information about the person who is in charge of its technical support (CONTACTINFO), textual description about its contents (DESCRIPTION) and the revision version (REVISION).

- **Object definitions**: the second part of the document describes the managed objects using the OBJECT-TYPE macro. Objects can be described within groups named using an OBJECT-IDENTITY macro and are referred to as “leaf nodes”. This macro will provide information about the description and status clause. According to SMIv2 notation, each object is described using five clauses: *object-type*, *syntax*, *max-access*, *status* and *description*. *Object-type* indicates the string name of the object. *Syntax* indicates the abstract data structure corresponding to that object. The data structure must be a simple base (integer32, octet string, counter32, gauge32, timeticks, etc), the BITS construct, or a textual convention (newly-defined type). The SEQUENCE OF and SEQUENCE are also possible for tabular object definitions. *Max-access* indicates if the object can be non-accessible (NA), accessible-for-notify (AN), read-only (RO), read and write (R/W) or read and create (RC) by an SNMP manager. The *Status* field indicates if the object is current, obsolete or deprecated. Finally, the *description* field contains a textual description of the semantic of the object. Other clauses such as Defval (for defining default values), index (for defining conceptual tables), augment clauses (similar to index) and reference clauses are not needed in the object definition. As stated before, apart from scalar objects, tabular objects can be defined within an MIB-module. These tabular structures will be used to order a collection of objects within an MIB. Each conceptual table contains zero or more rows and each row contains one or more scalar objects. Hence, tabular objects can contain several instances of each object, which can be seen like a table in which objects are the columns and their instances are the rows. These tabular objects are formalized by using the SEQUENCE OF <EntryType> within the OBJECT-TYPE macro where <EntryType> is a SEQUENCE type defined as follows: <EntryType>::= SEQUENCE { <type1>, ... , <typeN> } . These clauses are used to describe an object which is a table and a list of objects which correspond to rows of the table. Each object in a table is identified using the table index. A table can have a single index or multiple indexes.

- **Notification definitions** : the last part of an SMI document is used for describing the information contained within the traps and inform messages using the NOTIFICATION-TYPE macro. Each notification object will be described by an OBJECTS clause that will indicate the list of objects of the MIB included in the notification and the status, description and reference clause.

MIB II is the most important standardized MIB specification (originally defined in [142]). It covers a wide range of network and resource managed objects and is the common data-base MIB for device management in Internet. An extensive set of standardized MIB structures have been defined as part of the SNMP suite. Apart from standard MIBs, SNMP allows vendors to define private extensions that can be added in a private sub-tree in order to manage specific features of their products. This MIB-tree structure can be seen in Figure 2.9. The most important addition to the standard MIB II is the RMON (Remote MONitoring) specification [143]. Within each group of RMON there may be one or more control tables and one or more data tables. A control table, which is typically read-write, contains parameters that describe the data in the data table which is typically read-only [42]. In this way, the remote station configures in the control table the information that is desired to be presented in the data table. Adding or deleting parameters in an existing row is controlled by a row of the table designated as the Control Row. This control row will indicate a value as follows: 2) is used to create a new row, 3) indicates that the row is under creation, 1) is used to validate the row and 4) is used to delete it. DataType syntax of this row is defined as EntryStatus. It should be noted that SMIv2 defines a new type of syntax defined as RowStatus to manage the creation, activation, and maintenance of rows of a table. This dataype syntax defines the following substates: 1) active (the entrance is valid), 2) notinService (the row is present but not created), 3) notReady (if there is missing information), 4) createAndGo (create a new row available for use), 5) createAndWait (create a new row but not available for use) and 6) destroy (to remove the row).

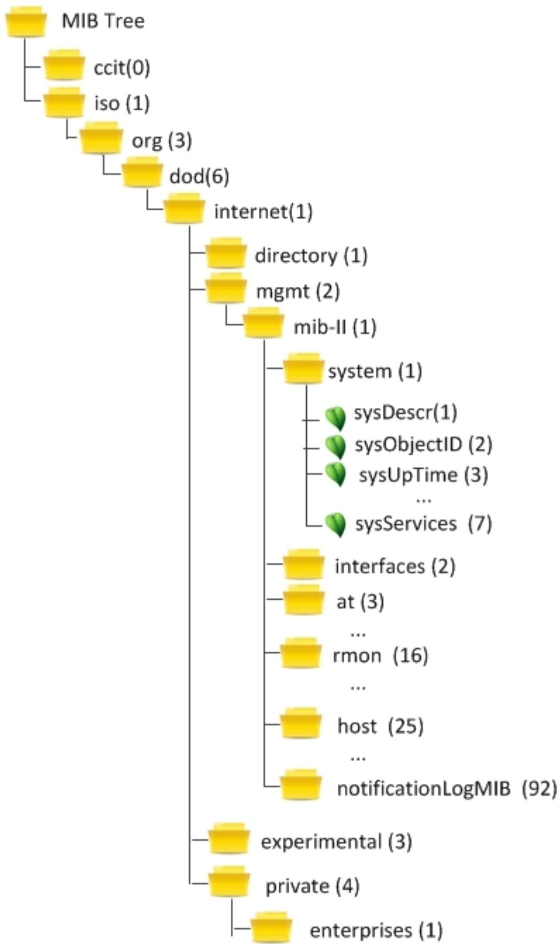

Figure 2.9: MIB-tree structure

2.7.1.3 SNMP messages

SNMP is a simple protocol and in essence provides 4 key capabilities: 1) **Get**, which enables the manager to retrieve information from the agent (data stored in the MIB), 2) **Set**, which is used by the manager to modify some values in the agent's MIB, 3) **Trap**, enables the agent to send asynchronous messages to the manager and 4) **Inform**, enables the manager to send alert messages to another manager (in SNMPv2 and SNMPv3).

To perform these functionalities, SNMP specifies a set of PDUs (7 different ones for version 3) for the agent-manager communication. These PDUs are encapsulated in SNMP messages with the suitable header according to the version of the protocol implemented. These SNMP facilities enable access to scalar objects. However, by using one single SNMP message it is possible to group a limited number of requests.

Consequently, a PDU contains a list of *variable-bindings* which gathers a pair of OID and scalar values. The burden of the network can be considerably reduced by using this technique. The specified PDUs are as follows:

- *GetRequestPdu*: used by the manager to retrieve one single item.
- *GetNextRequestPdu*: used by the manager to retrieve the next object of the specified OID. This command is used for traversing a tree-structured MIB.
- *GetBulkRequestPdu*: this command was introduced in the SNMPv2. It is used to request large amounts of data and is used for the efficient and rapid retrieval of tables. The GetBulkRequest includes a list of L variable names (OIDs) in the variable-bindings list and the values of the *non-repeaters* and the *max-repetitions* fields. This command performs continuous GetNext operations based on the maxrepetitions value, and the non-repeaters determines the number of variables from the variable-bindings list for which a simple GetNext should be done. In simple words, this command executes a simple GetNextRequest for the first N variablebindings and then an M continuous GetRequest for the R rest variables (R=L-N). Consequently, the total number of requested variable bindings in one message is given by N + (M * R) where N = MAX [MIN (non-repeaters, L), 0] and M = MAX [max-repetitions, 0].
- *SetRequestPdu*: used by the manager to write information and thus modify an item value of the agent's MIB.
- *GetResponsePdu*: used by the agent to return requested variables. If it is the response to a Set command it will include an acknowledgment in the response. This PDU includes an error status field to indicate if any error occurred relating to the requested variables. SNMPv2 includes 18 error-codes such as "noSuchName" to indicate that the name of a requested object was not found or "wrongValue" to indicate that the given value is not possible for the associated object. It can also be used by the manager to provide a response to an InformRequest.
- *TrapPdu*: used by the agent to notify information to the manager in an event-driven manner. It can report a list of variable-bindings related with the event that has occurred.
- *InformRequestPdu*: used by the manager (in the SNMPv2 and SNMPv3) for asynchronous notification to another manager.

2.7.1.4 Adding security to SNMP: SNMPv3

SNMPv3 was proposed as a security enhancement to SNMP [144]. Specifically, SNMMPv3 addresses three problems posed by previous versions by means of (by the incorporation of the USM module, User-Based Security Model [145]), access control (by the incorporation of the VBAC module, View-Based Access Control [146]) and strong privacy through data encryption. The USM provides authentication and privacy functions at message level. It is used to authenticate entities and provides encryption services to secure communications between agents. The VACM verifies that a given user is allowed to access a particular MIB object and to perform particular tasks.

The USM was designed to ensure: 1) data integrity, that is to say, it is not altered during the transmission, 2) data origin authentication, which means ensuring that the identity of the user and where the data came from are known, 3) data confidentiality, for avoiding a third entity observing exchanges between a manager-agent pair, and 4) message timeliness, that is to say, ensuring that messages are received in a timely manner and that delays, duplication or re-ordering are avoided. However, USM does not provide security against denial of service threats and traffic pattern analysis.

The USM is able to protect SNMPv3 packets by using two cryptographic functions: authentication and encryption. To perform these tasks, each SNMP engine requires a privacy and an authentication key. Hence, each SNMP entity (agent and manager) will keep its keys and also the remote keys for which SNMP communication is desired. Note that these keys are not exchanged via SNMP and that they are not stored in the MIBs. The USM specifies the use of a hash function and a secret key to produce the authentication code by using Message Digest 5 (MD5) and Secure Hash Algorithm 1 (SHA-1) algorithms. For encryption, the USM specifies the cipher block chaining (CBC) mode of the Data Encryption Standard (DES) protocol. The private key is used as input to the encryption block.

In order to implement the security model, and thus the USM, the structure for SNMP packets version 3 was changed and included the following fields (see Figure 2.10): *msgVersion*, *msgID*, *msgMaxSize*, *msgFlags*, *msgSecurityModel*, *msgSecurityParameters*, will contain the PDU together with the *contextEngineID* and the *contextEngineName* which are used for uniquely identifying an SNMP entity and a context within it.

The USM security model uses the following security parameters:

1. *msgAuthoritativeEngineID*: refers to the SNMP authoritative engine ID. The authoritative engine is the agent in a manager-agent communication.

2. *msgAuthoritativeEngineBoots*: refers to the *snmpEngineBoots* of the authoritative engine, that is to say the number of times the SNMP engine has been re-booted since the *snmpEngineID* was last set.

3. *msgAuthoritativeEngineTime*: refers to the number of seconds since the *snmpEngineBoots* counter of the agent was last incremented.

4. *msgUserName*: indicates the user whose secret keys are used for the encryption and privacy functions.

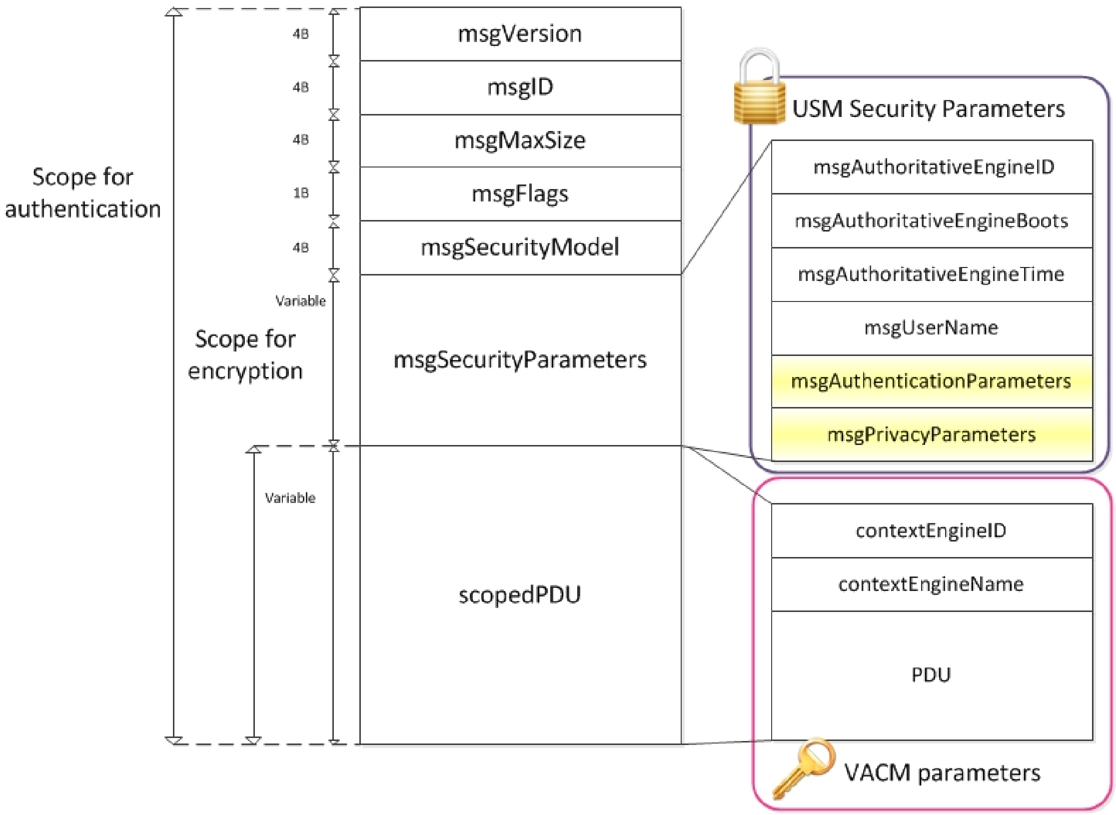

Figure 2.10: SNMPv3 packet structure

5. *msgAuthenticationParameters*: contains the HMAC (Hash-based Message Authentication Code) code of the packet (if it has been authenticated).

6. *msgPrivacyParameters*: if the message has been encrypted then this field contains the initial value (or salt) that has been used as input in the DES algorithm.

If the security level comprises both authentication and privacy tasks (authentication (authNoPriv), both (authPriv) or none (noAuthNoPriv) can be selected), then the workflow related to the message coding will be as follows for transmission: first, the scoped PDU is encrypted by using the DES algorithm. Then, the entire message is input to the HMAC. For incoming messages the process will be as follows: first, the USM computes a message digest using the secret authentication key for the user specified in *msgUserName*. Once this authentication is performed, the USM checks that the message is valid within a time window. This timeliness process is done by using the *snmpEngineBoots* and *snmpEngineTime* of the agent to avoid message delay and message replay troubles. At the beginning of the communication the manager will learn these agent parameters and then, as part of the authentication process, these parameters will be evaluated against the *msgAuthoritativeEngineBoots* and *msgAuthoritativeEngineTime*. If the packet is timely, then the USM performs a decryption of the scopedPDU and returns the plain text.

As can be seen in Figure 2.8, both agent and manager entities implement a Security Subsystem module (for SNMPv3) used to perform authentication and encryption functions. According to the security model used (USM or none), this subsystem may encrypt the PDU, introduce the required parameters in the message header and may include an authentication code in the message header. This module will also be used to process incoming messages to check for the authentication code and perform decryption. However, only the agent entity implements the Access Control Subsystem.

This mechanism determines whether a manager has access to a managed object. The VACM uses the following parameters from the SNMPv3 packet for determining message access:

1. *msgFlags*: used to indicate the security level of the message.
2. *msgSecurityModel*: security model used to perform security functions (USM).
3. *scopedPDU*: this field contains the *contextEngineID* and the *contextName* which correspond to the identification of the manager in request commands and the PDU which contains the list of requested variable-bindings.

Each variable-binding within the PDU is individually examined by the VACM module by using 4 tables. These elements are responsible of different access mechanisms:

- **Context Table**: a context is a name given to a collection of managed objects within a MIB. According to the contextName of the scopedPDU, access will be denied or not if such context exists.
- **Security To Group Table**: a group is a set of zero or more *securityModel* and *securityName* tuples on whose behalf managed objects can be accessed, and is used to define an access control policy for a group of individuals operating in an SNMP entity. According to the *msgSecurityModel* extracted when the message is processed, access will be provided to the corresponding group.
- **Access Table**: Is used to determine for each defined group the rights of access. Each MIB can be authorized to be read, written or used for notifying access according to each group and context name.
- **MIBs tree Table**: this table stores the MIB view. An MIB view defines a set of managed objects within an MIB. The MIB view is defined in terms of subtrees defined through a pair of an OID subtree value together with a bit string mask value. Each OID found in the scopedPDU is checked against the MIB view table. If it is part of this MIB value then access is provided.

Chapter 3

System Architecture

Theory and practice are disjointed disciplines that should go hand in hand when designing any software solution. This relationship between the two has been taken into account in designing the architecture proposed in this thesis to address integration challenges in home-based telemonitoring scenarios. From a theoretical point of view, a conceptual layer based on ontologies is proposed to unify the management procedure and integrate incoming data from all the sources involved in the telemonitoring process. From a practical point of view, a data and communication layer based on web service technologies is proposed to support in practice the benefits of the ontology and provide a real implementation of the tasks described in it. The main structure that supports the combination of these technologies is presented in this chapter.

3.1 General Description

New tendencies in the development of information systems are oriented to providing them with more autonomy and thus with some type of self-management capabilities in order to avoid continuous human intervention in the management of their applications. This is a consequence of growing communication infrastructures and increasing complexity in the management tasks they require, and also of the dynamic environments where these systems are used. These problems are also faced in general e-health infrastructures and in particular in the case of home-based telemonitoring in terms of mobility and dynamic environment where various elements are eventually added, removed or moved. Hence, the application of some ideas based on the so-called autonomic computing paradigm seems to be appropriate in this scenario. The autonomic computing paradigm is one of the key elements inspiring the developed ontology and consequently in establishing the modules required for the architecture as a whole. The telemedicine architecture proposed in this thesis was designed to provide both clinical and technical services in a telemonitoring scenario aiming to cover the main challenges described in chapter 1.

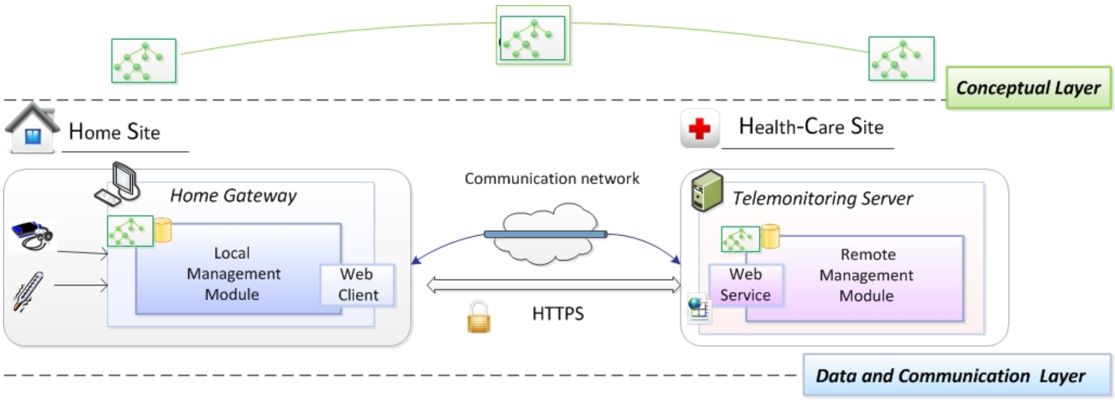

Figure 3.1: General Architecture (layers)

This architecture, depicted in Figure 3.1, is divided into two layers: the conceptual and the data and communication layers. As shown in the figure, in the conceptual layer the communication between both end sites is driven by the ontology used in the architecture. Both end sites of the telemonitoring scenario have to deal with heterogeneous information coming from different data sources. Although the same information is not managed at both end sites, data provided by the different sources is related given that these sources take part in the same monitoring process (for technical or clinical purposes). The idea proposed and implemented was to develop an ontology to clearly describe the managed data and explicitly describe in the data within the ontology the representation of the management procedures, ie the tasks that comprise the control loop of the autonomic computing paradigm. In this way, the developed ontology represents shared knowledge in order to achieve integration at both end sites of the architecture and also in their communication link. It should be noted that the managed data described in the ontology depends on the management service supported with the architecture. Consequently, first a generic ontology for describing general management was designed and then two extensions of the ontology were proposed to support each clinical and technical application. From a physical point of view, the telemonitoring system is composed of the Telemonitoring Server, TS, placed in the health-care site, and several HGs linked to it. The TS could be any device used to manage the information provided by the different home sites. Then, the communication between these devices is established using web service technologies. Consequently, at the data and communication layer, a web service together with a remote management module are implemented in the TS and a web client together with a local management module are implemented in each HG.

This architecture has been used to provide two types of services: (1) monitoring patients with different types of chronic conditions, thus clinical management services and (2) providing remote management of MDs and the HG, that is to say, technical management. It has been applied to these services following the three-stage approach depicted in Figure 3.2. The first stage deals with the ontology engineering. This phase comprises two sub-stages addressing first the specification, conceptualization and formalization of the ontology and then its implementation [87]. This is the most important stage in terms of data integration purposes. The second stage deals with the ontology application-based study. As stated in [81], there is no single correct way to model a domain. The best solution depends on the application where it is going to be used. As long as the ontology helps to provide good results for a given task, it could be said that the ontology is good for that application [80]. The main goal of the second stage was to demonstrate by defining different cases of use that the proposed ontology could be used to define any type of required profile. These first and second stages include the design, development and evaluation of the ontology. Finally, a third stage deals with the prototype development. Apart from the design of a common understandable knowledge for all the sources that participate in the telemonitoring process (ontology), the implementation of a prototype to hold the ontology and implement specified tasks was required. Hence, the last stage of the three-stage approach was practical. It consisted of the development of the modules of the data and communication layer of the architecture. Following this three stage approach, chapters 4 and 5 explain how the architecture presented in this chapter can be used to provide management services.

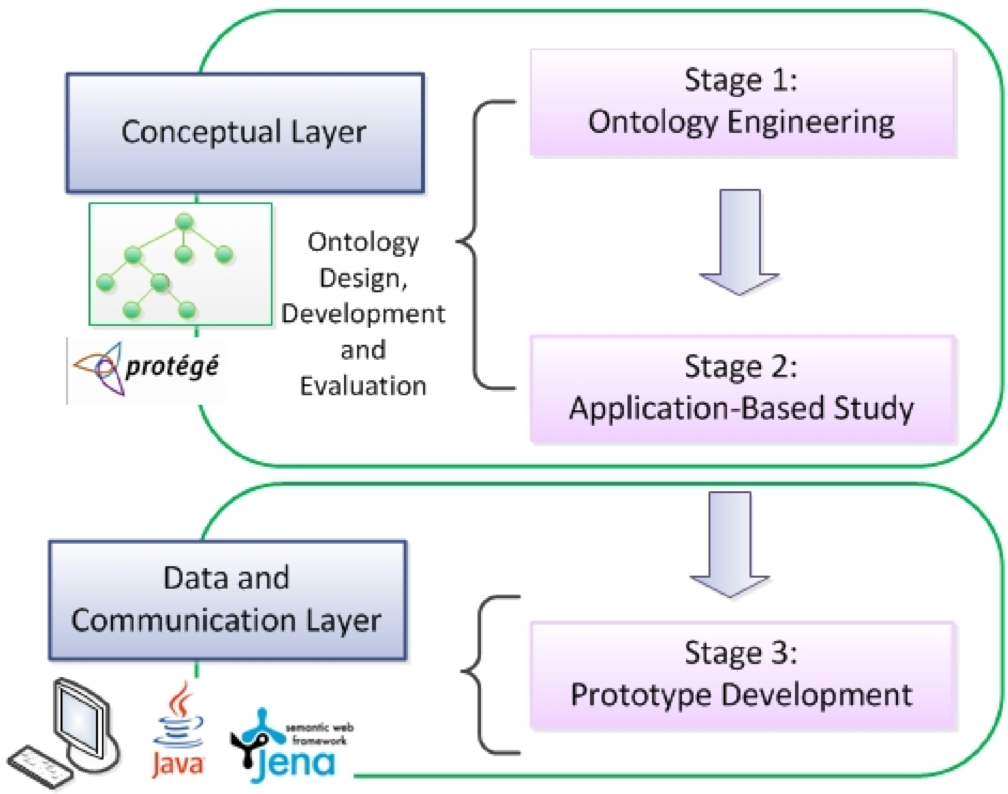

Figure 3.2: 3 stage approach for ontology application

3.2 Conceptual Layer

The developed ontology has been named HOTMES (Home Ontology for inTegrated Management in homE-based Scenarios). This ontology model is adopted as a unified common information model to provide two levels of seamless integration: data source integration and business logic integration [147]. On the one hand, integration at the data source level provides a unified view of the data provided by all the entities that work together to provide the user with a common service. On the other hand, integration at the business logic level provides a unified view of the management procedure, that is to say, a unified view of how to monitor, control and evaluate the managed information. These two levels of integration are depicted in Figure 3.3. In fact, the main idea of using the ontology is to provide a generic framework to define what has been termed in this work as a *management profile*. This can be seen as a container where all the tasks that should be performed in the HG to provide a management service are gathered together. In the case of technical management, an instance of the *management profile* could be designed to technically manage all the MDs that are linked to the HG (and even its own technical features) providing personalized technical management for each specific HG. In the case of a clinical application, an instance of the *management profile* could be designed to remotely monitor and control clinical data from a patient. In this case, the instance would be designed according to the patient's disease characteristics, providing personalized clinical management.

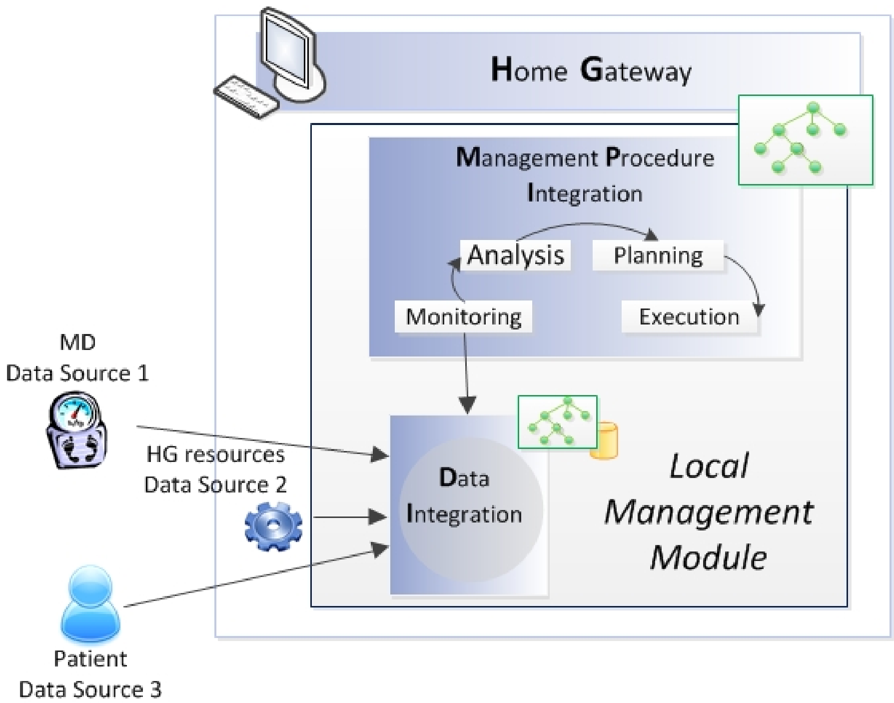

Figure 3.3: Data and Management Integration Levels

This conceptual layer includes both an ontology and the definition of rules. In particular, rules are used in combination with the ontology to provide personalized services by using them to activate and relate tasks described in the ontology. By defining instances of the proposed ontology and rules, managed elements are provided with a profile where all the tasks to be performed for the management procedure are clearly described and personalized management services can be provided. Although this ontology has been used in the work described in this thesis for clinical and technical management in telemonitoring scenarios, the proposed approach constitutes a formal and generic solution that can be used not only to represent but also to manage any type of information in home-based scenarios where data provided by the different sources are collected and managed in the HG.

3.2.1 MAPE-inspired ontology structure

The structure of the *management profile* was inspired by the MAPE process from the autonomic computing initiative. Thus, a *management profile* consists of 4 sections that define 4 different tasks: a monitoring task, an analysis task, a planning task and an execution task. By using this idea, the HG is provided with some level of autonomic behavior capability. Depending on the *management profile* content, an HG will be enabled to monitor its state or information provided by the different sources, analyze acquired information, react to abnormal findings and inform about changes in its contextual environment. Based on the statements and the steps discussed in [81], the development of the proposed ontology was an iterative process and two particular strategies were followed for its design. First, in order to determine the scope of the ontology and to set up the most important classes to describe the domain, some initial questions were formulated. These competency questions had to be answered with the ontology model and served later to evaluate the design. Second, reusing vocabularies from other ontologies was considered. Some ontologies and standards were reviewed in order to reuse vocabularies and already defined conceptual schemes instead of having to reinvent all the ontology. In the same way as was proposed in the SOUPA ontology definition [148], the strategy was to borrow terms from these ontologies but not import them, in order to limit the overhead. By reusing existing vocabularies, the mapping task with other conceptual models is made easier.

The HOTMES ontology has been written in OWL-DL using the Protégé-OWL v.4.0.2 ontology editor and checked using the FACT ++ reasoner [149]. Specifically, the popularity of this language (and consequently its extensive usage) and the rich expressiveness it offers in terms of class and property descriptions while guaranteeing total computational capacity were the main reasons for choosing this W3C-endorsed language. Pellet reasoner was also used to check the satisfiability of the ABox with respect to the TBox. It should be noted that this HOTMES ontology is the basic structure representing management tasks and that it should be extended according to the specific application. Consequently, two extensions of the HOTMES ontology for clinical and technical purposes have been proposed in this thesis. FACT ++ reasoner was also used to automatically compute the classification hierarchy and check the logical consistency of the extended versions of the HOTMES ontology. See Figure 3.4 where the relation between the data and management integration with the ontologies is depicted.

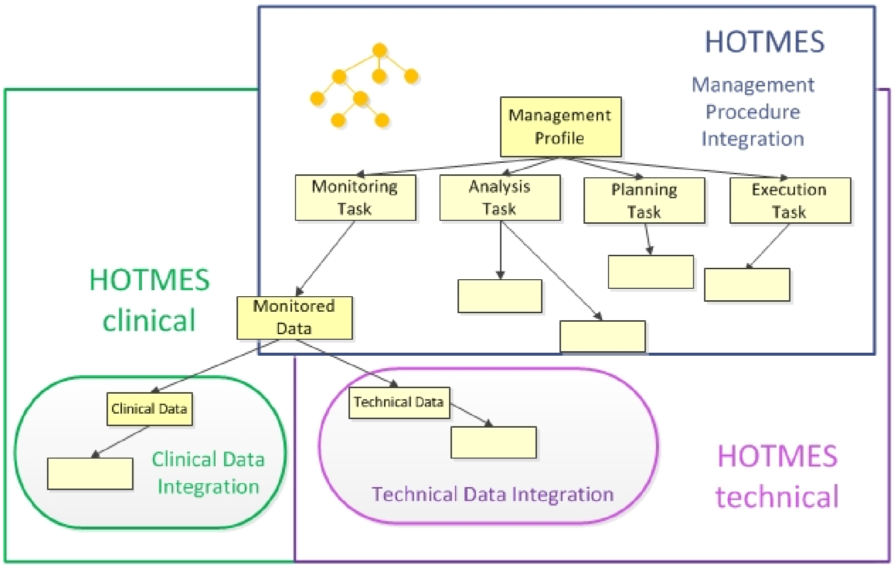

Figure 3.4: HOTMES extensions

The main components of this proposed ontology are presented in Figure 3.5 by means of UML class diagrams using the nomenclature described in chapter 2. As can be seen from Figure 3.5, the HOTMES ontology has 5 root classes: the *management profile* class and 4 classes that describe the tasks that comprise a *management profile*. Detailed information about each root class is provided in the following subsections. As shown in Figure 3.5, a *management profile* comprises at least one instance of each type of task. Furthermore, an identification, the version, a description and the state of the profile should be specified with each configured *management profile*. The HOTMES ontology includes 53 classes, 45 object properties, 43 data properties, and 13 individuals. Furthermore, 21 annotations properties have been included in the ontology to clarify some terms and relate them with well known standards or models. The size of the OWL ontology file is 81 KB and the DL expressiveness is ALCON(D). Apart from classes and properties, the ontology includes the following constructed forms: *owl:unionOf*, *owl:allValuesFrom*, *owl:oneOf*, *owl:cardinality* and *owl:disjointWith*. As will be seen in chapters 4 and 5, extensions of the HOTMES ontology considerably enlarge its size.

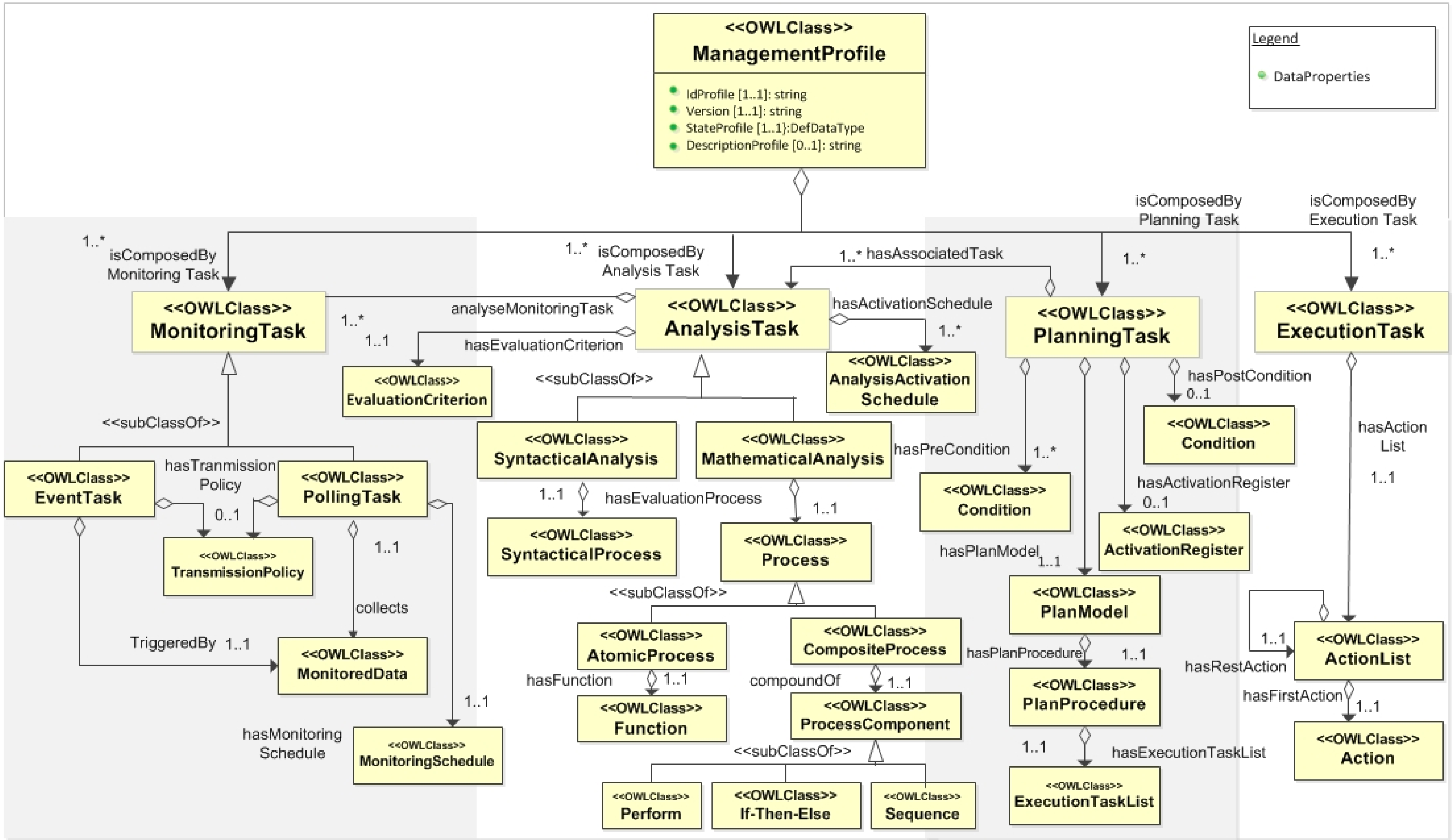

Figure 3.5: HOTMES ontology

3.2.1.1 From HOTMES OWL DL to OWL 2

OWL 2 extends OWL 1 with a small but useful set of features in order to redress the lack of various constructs often required for modeling a complex domain. The HOTMES ontology domain was successfully expressed by the resources available in the OWL 1 DL language. However, OWL 2 provides a set of resources that improves both the HOTMES ontology in terms of the number of triples required to express the same knowledge and in terms of offering an accurate definition of the model [150]. A second version of the HOTMES ontology was developed in order to take advantage of the OWL 2 resources. Although OWL 2 introduces a wide range of new possibilities, the current situation is that only some of its features are used by developers and they are not used prominently. Despite the complexity of OWL, many times, a little OWL is required and RDFS are the most prominently used [151]. Changes introduced in the ontology were related to the following constraints:

1. *Qualified cardinality restrictions (QCRs)* : qualified cardinality restrictions allow a constraint on the number of values of a particular type of property to be taken from a particular class. For example, they could be used to express that a *management profile* specifically comprises three tasks that are execution tasks. These restrictions are not available in OWL 1. When required, they were approximated by using both cardinality constraints and the value constraint *owl:allValuesFrom*. By using qualified constraints, the number of restrictions included in the ontology is reduced and the number of properties could even be reduced.

2. *Disjoint Classes (Sugar properties)* : most of the classes included in the ontology were made disjoint in order to clearly express that an individual of a class could not be a member of another class (*owl:disjointWith*). OWL 1 provides a means of stating that two classes are disjoints but not that several subclasses are pairwise disjoint. In contrast, OWL 2 defines the feature *owl:disjointClasses* for this purpose, thus allowing the complexity of the ontology definition and the number of statements included in the ontology to be reduced.

3. *Keys* : one of the main limitations of OWL 1 is the definition of *Keys* which is in fact a core feature of database technologies. They are vital for uniquely identifying individuals by a value of a key property. Using OWL 1, it is not possible to directly express that a *management profile* is uniquely identified by its property *IdManagementProfile*. This limitation was redressed in this work by using AnnotationsProperties. Properties used as keys were tagged in the ontology as “Primary Key”. Then, these conditions were considered for the implementation of the prototype and the application of defined rules. The second version of the HOTMES ontology took advantage of the *owl:hasKey* property defined in OWL2 and annotations properties used for this purpose were replaced by *hasKey* axioms.

This feature has been added to some reasoners such as Pellet, so it could be used in combination with Jena to access the ontology.

The HOTMESv2 ontology was written in OWL 2 DL and includes 55 classes, 41 object properties, 37 data properties, and 13 individuals. The size of the OWL ontology file is 79 KB. The expressiveness of the ontology was evaluated and its consistency and satisfiability were measured using the Pellet reasoner too. The expressiveness detected was ALCHOQ(D).

3.2.2 Rules

By using a set of if-then rules (which indicate what should be done according to a situation), the behaviour of individuals can be expressed inside a domain. Hence, by defining an instance of the *management profile* and some individual rules, personalized management services can be provided. These rules will not be generic. The EM (external manager that can be a clinical or technical member of staff according to the proposed application) will define a specific set of rules associated to each *management profile* instance. Each defined *management profile* will be composed of at least one instance of each type of task. Then, the relation among the tasks will be achieved by certain conditions expressed using the rules. By applying these rules over the content expressed by the instances, the agent will execute, for example, the appropriate analysis task when acquiring data specified in a monitoring task instance, or the appropriate planning task relating to the analysis of a test result. Rule-based reasoning is thus used to determine the appropriate task to be performed according to the context. These rules are expressed with the SPARQL language. Their use is related to the activation of the analysis and planning task, so they are applied over instances of information and not over the whole ontology model and, more importantly, at a certain specific moment during the closed loop performance. This was one of the main reasons for selecting the SPARQL language to express the rules.

The use of SPARQL has also been studied to complement OWL expressiveness for arithmetic functions that will be included in the model. Describing functions and including them in the ontology model provides a clear explanation about how data should be processed during the analysis phase. In this way, the ontology provides all the information required to execute described actions and releases developers from the task of implementing the described functions. The same SPARQL rule-engine can be used to execute functions and individual management rules, providing great flexibility in the definition of both types of data. In this way, functions included in the ontology may be extended without having to modify the management module implementation and different personalized management actions can be defined.

3.2.3 Data Integration

To provide data integration, the ontology describes within the monitoring task the structure of the information that is of interest for management purposes. Data provided by the different sources included in the managed scenario will be mapped into these classes and transformed into ontology instances. This information should be described individually for each application (as is shown in chapters 4 and 5). It is interesting to note that the monitoring, analysis, planning and execution tasks defined in the ontology describe a generic process. Thus, they could be used to manage any type of data described under the monitoring task in this scenario.

Note that modeling a domain is not an easy task. Different data models, standards and other external sources should be reviewed in order to provide a generalized view where all sources that are to be mapped may be represented. Furthermore, the selection of vocabulary and terminology (which should be close to the domain described), the evaluation of taxonomy thoroughness and the correct usage of the ontology language are issues to be addressed. For both clinical and technical purposes, these issues were covered during the first stage of the three stage approach depicted in Figure 3.2.

3.2.4 Management Integration

Management procedure integration is undertaken by managing data according to the tasks described in the *management profile* (see Figure 3.5).

3.2.4.1 Monitoring Task class

Performing a monitoring task generally means collecting information from a system, in order to test it and to be aware of its state [42]. Defining an instance of a monitoring task will allow control and observe of any changes that may occur over time in the state of a resource of the system. This task is subdivided into two subclasses that represent two different techniques used in traditional network management for collecting information:

- *Polling Task*: A polling-based monitoring method which consists of periodically polling devices for information retrieval.
- *Event Task*: An event can be defined as something that happens during the course of a flow process. Therefore, all information provided in an asynchronous manner should be monitored by instances of the event task.

The following ontology properties (depicted in Table 3.1) have been defined for the Polling, Event and Monitoring Task classes.

Object Property	Domain	Range	Description
hasMonitoringSchedule	Polling Task	Monitoring Schedule	Indicates when the polling task should be activated and the retrieval characteristics.
collects	Polling Task	Monitored Data	Refers to the information that should be collected.
isTriggeredBy	EventTask	Monitored Data	Refers to the monitored data that activates the event task.
hasTransmissionPolicy	Monitoring Task	Transmission Policy	Indicates when the collected information should be transferred to the remote-site (if necessary).

Table 3.1: Monitoring Task properties

As shown in Table 3.1, each monitoring task is characterized by 3 main properties: the information being monitored, the monitoring schedule and the transmission policy.

- *Monitoring Schedule*: this class indicates to the agent the specific time for retrieving some pieces of data and/or its periodicity.

- *Monitored Data*: this is a generic class that refers to the information that the agent should collect. When extending the monitoring task for each application, the new classes that describe the monitored resources will be added as subclasses of the *MonitoredData* class.

- *TransmissionPolicy*: this class indicates whether collected data should be transferred to a remote site (in real time or at a specific hour) or not.

Note that each configured task will be associated to a unique identifier by means of the data property *IdMonitoringTask*.

3.2.4.2 Analysis Task

An analysis task can be defined as a set of processes and rules that enable a complex problem to be broken down into smaller parts in order to gain a better understanding of the problem, as well as to manipulate data in order to extract additional information. The analysis task described in the proposed ontology explains how to reason with the monitoring data acquired by defining different analysis methods. Moreover, it explains which process is used to analyze a datum (a piece of data retrieved at a certain time) and also describes a set of functions that will be used to manipulate information.

Some terms from the OWL-S ontology have been borrowed to describe the analysis task ontology (defined in chapter 2). Although OWL-S was initially proposed to describe semantic web services, its use has already been studied in other applications such as in network management [100]. Specifically, classes and terms related to the Process description have been used. It should be noted that there are standard proposals for representing business models such as BPMN (Business Process Modelling Notation) and its language BPML (Business Process Modelling Language) that provide tools and generic notation for describing workflow processes [152]. Hence, these could be useful for modeling the workflow of the management procedure and thus may inspire the design of certain tasks of the *management profile*. Specifically, BPEL4WS (Business Process Execution Language for Web Services) was initially evaluated for the design of workflow processes in analysis tasks [153]. BPEL4WS provides a language to describe the behavior of business processes and interaction for web services. However, the OWL-S ontology was chosen in preference to BPEL4WS for the present work, as OWL-S already incorporates richer semantics and more meaningful descriptions for implementing automated tasks. Its successful application elsewhere and its attractive characteristics enhanced by other works motivated its selection [100, 154].

Considering the type of data that can be acquired in a home-based scenario, the analysis task has been subdivided into two subclasses that represent two different types of analysis: Syntactical and Mathematical. Initially, according to the proposed scenario, only numerical data (integer and decimal datatypes) and simple data strings were expected to be received and then processed in the HG. Nevertheless, in the same way that the monitoring task may be extended according to the application case, the task could also be extended if more complex data is acquired (e.g electrocardiographic signals) which then require analysis. These two main subclasses have been defined as follows:

- *Syntactical Analysis*: This detects specific tag values when comparing string data types in the analysis task.

- *Mathematical Analysis*: By defining an instance of the mathematical analysis task, the associated input data will be evaluated after being mathematically manipulated.

The main properties relating to the analysis task are defined in Table 3.2. As can be seen in the table, the *ActivationSchedule* class dictates when the analysis task should be run. It can be configured as a "real-time" analysis (that is to say, the analysis task will be run after acquiring the data specified in the monitoring task it is related to) or it can be configured to be executed at a certain time of the day (including in the analysis all the data acquired during the day until the set time). The *Process* class describes how the acquired data (specified in the monitoring task) is to be manipulated in order to subsequently compare the acquired result with the criteria configured as dictated by the *Evaluation Criterion*. An instance of this latter class will be characterized by a list of conditions expressed and an operator (AND or OR) that will govern the relations among them. Each condition will be characterized by the language used to express it (SPARQL in this work) and an identifier called nameRule used to indicate the rule statement. After processing the acquired data according to the defined *Process* instance, these configured rules will be applied and consequently the analysis task result will turn into a positive or negative output value. It should be noted that the notation and vocabularies used for the description of the *Process* class have been inspired by the Process of ServiceModel subclass (within the service profile) in the OWL-S ontology.

Object Property	Domain	Range	Description
analyseMonitoringTask	Analysis Task	Monitoring Task	Indicates which monitoring tasks should be analyzed according to this analysis process.
hasEvaluationCriterion	Analysis Task	Evaluation Criterion	The analysis task makes reasoning by matching processed monitoring data information (using a set of functions) with a set of pre-defined rules (that can also be combined by using "OR" and "AND" operators). These rules will constitute the criterion to indicate whether an analysis result is true or false.
hasActivationSchedule	Analysis Task	Activation Schedule	Indicates when an analysis task should be performed (in real time or at a specific hour).
hasEvaluationProcess	Syntactical Analysis	Syntactical Process	Indicates that the analysis task is associated to the evaluation of a syntactical process result.
hasEvaluationProcess	Math Analysis	Process	Indicates that the analysis task is associated to the evaluation of a mathematical process result.

Table 3.2: Analysis Task properties

As can be seen in Figure 3.5, a *Process* class is divided into two subclasses: the atomic and composite processes.

- *Atomic Process*: this is associated with an action that should be executed in a single step or interaction, thus for an input message it will do something and then will return an output. The main property of this class is hasFunction (d: AtomicProcess, r: Function). An Atomic Process will process input data according to a mathematical function (*Function* class) to provide an output value that will be evaluated according to a specified criterion.

- *Composite Process*: this is associated with actions that are required to be performed in multi-steps. Inspired by OWL-S, the ontology defines 3 types of executing process combinations (ProcessComponent from OWL-S): *Perform* class, *Sequence* class and *If-Then-Else* class. The *Perform* construct is use to reference the execution of a process as part of the composite process. The *Sequence* construct is used to reference a set of processes that should be executed in sequence. Finally, the *If-Then-Else* construct consists of a condition (expressed in SPARQL), a then and an optional else process. Consequently, the execution of the process will be done according to the positive or negative result after the application of the condition. OWL-S defines further types of *ProcessComponent* (nowadays deprecated for ConstructComponent) such as Choice, Iterate, Split, etc. Thus, if more complex execution processes are deemed necessary for performing an analysis task, the HOTMES ontology could be extended with such definitions.

The *Syntactical Process* class simply indicates that the acquired string data is to be syntactically compared with configured evaluation criteria as an instance of the *Evaluation Criterion* class.

Function Class

This class indicates how information is to be mathematically processed. A total of six functions have been explicitly described in the ontology: Abs, Avg, Division, Multiplication, Round and Sum. One of the OWL limitations is its lack of ability to express methods and function definitions. Nevertheless, this limitation can be compensated for by using some SPARQL queries based on XPath arithmetic functions [155]. To deal with this problem and provide a model where all the information is explicitly represented, some SPARQL queries that express arithmetic functions have been included in the model to indicate how the information is modified with each defined function. Therefore each described function is associated to an XpathFuncion and will have one or more operators as inputs. As these functions are to be used to evaluate monitored data and also the output of other processes, the operators required will indicate the name of the resource class (monitored data or results from processes) whose instances are going to be evaluated. Then, the hasSparql-Xpathfunction property indicates for each class the associated query that dictates the mathematical performance of the function in relation to the operators associated to it.

For example, Figure 3.6 corresponds to a SPARQL function associated with averages and divisions. It indicates that the output value of a monitoring datum expressed as an operator (X or Y) will be processed as the selected clause indicates. Take, for example, the average function. First, this SPARQL query indicates that the result variable of the function will contain an average of values "Select avg(?a) as ?result". Then, restrictions are presented within the `WHERE` clause in order to apply the function query only to limited individuals. This restriction indicates that only one operator is involved in the process. In fact it indicates the type of individuals that are going to be processed with this function "?rX hasOperator ?datX". Next, the restriction indicates the output value property of individuals whose class name is the same as that pointed out by the operator of the function which will be included and then processed into this mathematical function.

In this way, it is clearly indicated how the software module should manipulate monitoring information included in the ontology when executing an analysis process. It should be noted that this approach has a double advantage. The firrst advantage is that more functions can be defined in the ontology without changing the code in the software module that is to execute the task in the HG. The second advantage is that by providing a clear explanation of how data is manipulated, the re-usability of the ontology for other applications is enhanced. Moreover, this latter advantage could be very valuable if complex operations are to be expressed in the ontology. It is also interesting to note that input data for an atomic process is not only directly acquired by a monitoring datum, but can also be the output value of other mathematical processes executed previously, hence providing enhanced flexibility of the model.

```
1. «AVGFunction»

Xpath Function→ avg
SparqlExpression→

SELECT( avg(?a) as ?result)
WHERE {
?f     base:hasOperatorX   ?datX .
?rX   base:hasNameClass  ?datX .
?rX   base:outputValue      ?a }
```

```
2. «DIVISIONFunction»

Xpath Function→DIV
SparqlExpression→

SELECT( ?a/?b  as ?result)
WHERE {
?f     base:hasOperatorX   ?datX .
?f     base:hasOperatorY   ?datY .
?rX   base:hasNameClass  ?datX .
?rY   base:hasNameClass  ?datY .
?rX   base:outputValue      ?a
?rY   base:outputValue      ?b  }
```

Figure 3.6: SPARQL average and division functions

3.2.4.3 Planning Task

A planning task can be defined as a detailed formulation of a sequence of tasks and activities for solving a sequential decision problem [156]. The output of the planning task will be a plan specified according to an application solution criterion. The *Planning Task* model described in the ontology answers four questions. 1) What strategy are we going to follow for dealing with the detected analysis outcome? 2) Who is going to execute the plan? 3) What is the schedule for the actions of the plan? 4) Under what conditions will the plan be executed? Vocabularies and schemas used to describe the planning task such as hasPreCondition, hasPostCondition and hasPlanModel object properties have been inspired by other ontologies and models that define the concept of planning [156, 157, 158]. The main properties defined for the Planning task are shown in Table 3.3.

Pre-Conditions and Post-Conditions are defined as instances of the *Condition* class. An instance of this class is characterized by the name of the rule it refers to and its language expression. These rules expressed using SPARQL are provided to the software module together with the *management profile*. After performing an analysis task, the agent should execute the pre-condition and post-condition rules in order to execute the corresponding planning task if any is activated.

Object Property	Domain	Range	Description
hasPreCondition	Planning Task	Condition	Indicates the rule that triggers the activation of the planning task. Normally, it will refer to a positive finding in an analysis task or a failure of another planning task.
hasPostCondition	Planning Task	Condition	If the pre-condition is activated then the post-condition should be checked in order to activate the plan. If the post-condition is not achieved, the plan is not activated.
hasAssociatedTask	Planning Task	Analysis Task	This is associated to the task that activates the plan. After performing an analysis task the agent will check if there is any planning task associated to the previous analysis task performed.
hasActivationRegister	Planning Task	Activation Register	Indicates how many times the pre-condition has been activated but not the post-condition of the task. If a post-condition exists then the agent should check the activation register before activating the plan.
hasPlanModel	Planning Task	PlanModel	Indicates the model of the plan, that is to say, who is going to execute it, which task must be performed and their relation.

Table 3.3: Planning Task properties

As a property of an instance of the *PlanModel* class (depicted in Figure 3.5), the plan in model should be configured to have an instance of the *Plan Procedure* class. This class indicates the sequence list of execution tasks that should be performed in the planning task. It should be noted that all list of classes defined in the ontology comprise a first element and the rest of the elements modeled as a new list. In this way the order of the sequence is clearly indicated according to the tasks to be performed (see *ActionList* class in Figure 3.5 and Table 3.4).

3.2.4.4 Execution Task

An execution task can be defined as a set of actions performed by a specific agent [157]. In this model the *Execution task* refers to actions to be performed by the agent or the EM in order to react to a certain situation or solve a detected problem. Therefore the execution tasks will be used to comprise the planning task. Each execution task can be decomposed into small actions, thus each execution task will have a list of actions associated to it. Defined instances of actions must have a name and a description of their activity and optionally they can include some required parameters. For example, for the action of sending a mobile text message, a mobile number will be required as a parameter. Other possible actions would be to send an e-mail or to show a pop-up to the user. These will be configured as instances of the *Action* class.

Object Property	Domain	Range	Description
hasFirstAction	Action List	Action	Indicates the first action to be performed to solve a problem.
hasRestAction	Action List	Action List	Indicates the rest of the list of actions to complete the execution task.

Table 3.4: Execution Task properties

3.3 Data and Communication Layer

As explained in this chapter, the whole architecture includes the conceptual and the data and communication layers. Within the data and communication layer, two management modules and web technology modules inside the HG and the TS constitute the main parts of the telemedicine system (see Figure 3.1). This architecture is oriented to supporting any extension of the HOTMES ontology used at the conceptual layer to describe a *management profile*. Hence, the architecture is able to provide both types of management services (clinical and technical). Furthermore, its communication method could be easily reused in other implementations based on ontologies and no major changes would be required to include additional services based on the HOTMES ontology such as, for example, an environmental-sensor application.

The development of specific tools to manipulate and map incoming data into the ontology model and to exchange these data between the HG and the TS is required. As stated at the beginning of this chapter, apart from developing the theoretical model, practical applications are required to integrate data and thus address challenges that may arise [159, 160]. When working with ontologies, this practical or physical layer is frequently disregarded in the ontology-based system Consequently, few works have given details about this practical implementation of the ontology-based system. Memory, storage resource consumption, network usage cost and also execution time for performing the tasks are strongly conditioned by the practical implementation of a system and may ultimately determine its utility.

In the present case, the data and communication layer of the architecture was oriented to minimizing this consumption of resources and to providing re-usable key ideas for future ontology-based architecture developments. This proposed architecture to deal with ontologies is in line with other reported solutions based on web service technologies [86, 88]. Nevertheless, the communication method proposed in this work enables us to go one step further as it is based on the exchange of generic OWL instances, thus making the WS description of the content expressed in the ontology independent. By deploying a WS in the data and communication layer independent from the information described in the ontology used for the conceptual layer, a flexible and reusable architecture based on ontology usage is achieved for the communication between the end sites.

The modules that comprise the architecture have been developed using Java technologies. Specifically, the Jena framework (version 2.8.6) has been used to process the ontology and create new instances, data acquisition and manipulation when the SPARQL rules are applied. The Jena framework was chosen because it is open source and supports the reading and writing of OWL ontologies. The Jena engine was also used to execute the SPARQL rules (ARQ engine). Also, a TDB database (version 0.87) has been used to store the ontology model and the technical *management profile* instance in the HG [161].

3.3.1 TS Modules Description

The components of the remote management module installed in the TS are depicted in Figure 3.7. The communication between this module and that installed at the home site is established through WS technologies.

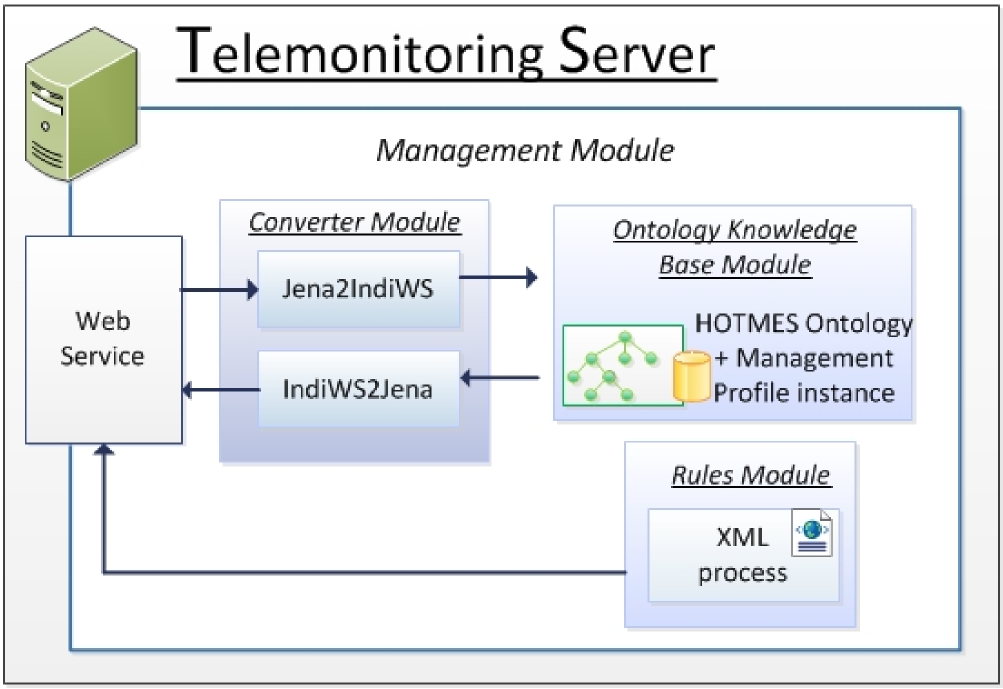

Figure 3.7: TS management module

As can be seen in Figure 3.7, this management module includes 3 components:

1. **Ontology Knowledge Base Module**: This module contains the ontology knowledge models and the instances of the registered *management profiles*. The architecture has been designed to provide both clinical and technical management services, so two ontologies have been developed for these tasks as extensions of the HOTMES ontology. Both ontology extensions and registered instances of the *management profile* (associated to all linked HGs for technical purposes and associated to monitored patients for clinical purposes) are handled in this module.

Physicians and technical managers will configure these profiles by using additional software tools stored in the knowledge base module. This information will then be transfered to the home sites when required. In the TS, the communication between the web service and the knowledge base module is established through the Converter Module. The TDB database has been chosen to store the ontology model and new instances in this knowledge base module.

TDB is a persistent graph storage layer for Jena. It is a non-transactional and fast database solution. It should be noted that the first versions of the architecture made use of a MySQL database to provide persistent storage of the ontology [162, 163]. By using the Jena SDB system, it is possible to use relational databases for storage of RDF and OWL. It supports many open source and commercial databases including MySQL and PostgradeSQL and scales to graphs of 100 million triples. Finally, however, this option was changed to a TDB database because the TDB is more suitable for the ontology store (as it was designed to store triples) and offers better performance for loading and querying (as the overheads of SQL and JDBC are avoided). Native triplestores (purpose-built databases for the storage and retrieval of triples) have the advantage of providing good performance. Other triplestores such as Mulgara could also have been selected for this purpose [164]. Mulgara offers an optimized database to store subject-predicate-object statements and can be used for storing 500M triples. Therefore, it could be used to store and manage big ontologies. This is a complete system that can be used with Jena and also provides its own query language called iTQL. In fact, depending on the profiles registered in the TS, this database could be an alternative solution to the TDB database. To scale the system and support large quantities of profiles, other databases such as Oracle could be adapted to the architecture by using the Jena SDB module.

2. **Converter Module**: The communication module of this architecture is mainly based on OWL instances exchanged generically by means of a developed object structure named *IndividualWS*. The Converter module is used to wrap and unwrap the Jena individuals into the *IndividualWS* structure used to exchange information with web clients. Furthermore, this module incorporates some reasoning tasks. Ontology-based reasoning is used in order to retrieve individuals, to check instances before including new information in the model, and to ensure the consistency of the model. These tasks are performed by means of the Jena framework and SPARQL queries. New ontology instances are added to the ontology when new managed data is acquired and when it is transfered from the HG or the TS module using the Converter module. Details of this implementation are given in the communication method subsection.

   The ontology model is initially read in both the HG and the TS from an OWL file and then stored in the TDB database. Then, the model is dump in the memory using the Jena Framework and without additional entailment reasoning. The first versions of the architecture included the Pellet reasoner (in combination with the Jena Framework) for the interaction with the ontology. However, this option was discarded because the OWL entailment and subsequent interactions with the ontology were much more resource-consuming in terms of time and memory. The ontology structure and its usage in this work do not require complex reasoning tasks. For this reason it was decided simply to use the Jena memory model and perform required reasoning tasks through the execution of SPARQL queries and rules using the ARQ Jena engine. It could be said that the implementation and the required reasoning tasks were adapted to the structure of the ontology and the required interaction.

As stated in chapter 2, there is a wide range of reasoner implementations, each one based on different reasoning algorithms such as Tableaux-based (e.g Pellet), first order logic-based (e.g KAON) or rule-based reasoners (e.g Jena). Although OWL entailment rule-based OWL reasoners are generally fast, ontology processing and reasoning tasks consume significant time and memory resources (compared to other non-semantic tools used for similar purposes). Researchers therefore continue proposing new solutions in order to reduce resource consumption for the reasoning performance [165]. Semantic tools need to be optimized in terms of resource consumption as device constraints and hardware resource limitations increase when working with small devices such as smartphones.

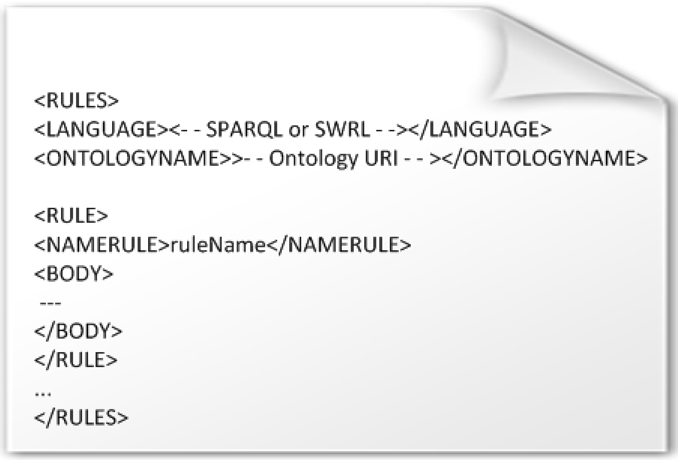

Figure 3.8: Rules-XML schema

3. **Rules Module**: This module is used to store rules associated to each *management profile*. These rules are subsequently transferred by means of an XML file. The structure of the XML file is presented in Figure 3.8. As can be seen, a list of any type of rules could be transferred to the HG. The URI of the associated ontology is indicated and then a name is assigned to each rule. This name corresponds to the DataProperty "nameRule" of the *Condition* class used to express rules inside

the ontology. Providing the rules apart from the ontology model, applying all the rules together over the model or at a certain moment in the computing process is conditioned by the required application. This structure could be used to express rules with other types of language such as SWRL.

3.3.2 HG Modules Description

The components of the local management module installed in the HG are depicted in Figure 3.9. This management module has been designated the “Semantic Autonomic Agent”. The communication between this agent and the management module installed at the remote site is established through a web client connection to the web service installed in the remote TS. As can be seen in Figure 3.9, the architecture of the agent comprises the following modules:

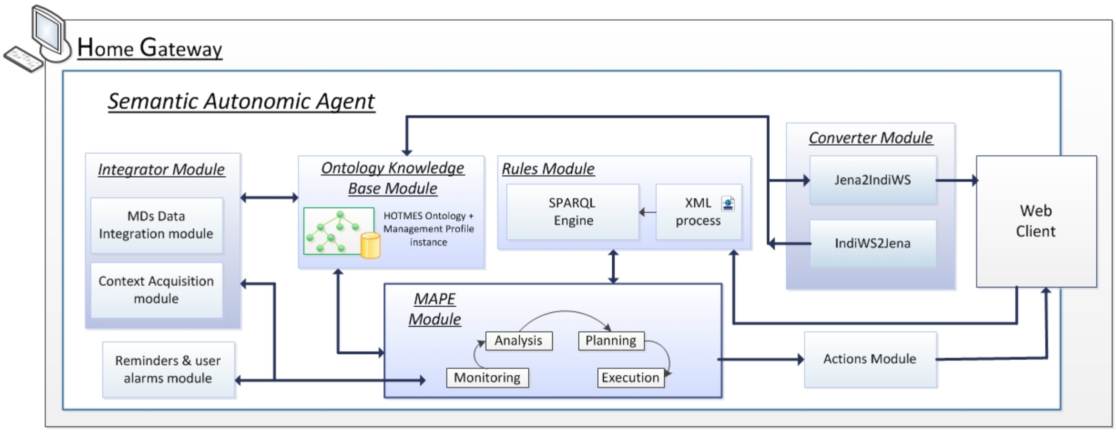

Figure 3.9: Semantic Autonomic Agent

1. **Ontology Knowledge Base Module**: This module contains the ontology knowledge model and the instance of the required *management profile*. Both the ontology and the *management profile* should be downloaded from the remote TS. Therefore, they will be acquired by a web client connection. Inside the HG, the communication between the web client and the knowledge base module is established through the Converter Module. Note that if both clinical and technical services are required, then both extensions of the HOTMES ontology should be downloaded and saved in the knowledge module. In the same way, instances of the *management profile* for technical management (normally one) and for clinical management (as many as the number of patients located in every HG to be remotely monitored) will be stored there.

2. **Converter Module**: The converter module comprises two sub-modules used to wrap and un-wrap the Jena individuals into the *IndividualWS* structure (used to

exchange the information with the web clients). It has the same functionality as in the TS. Basically, in the HG it will be used to integrate the *management profile* instance in the ontology and to wrap into the *IndividualWS* the data acquired from MDs and users.

3. **MAPE Module**: This module constitutes the computing core of the agent. It will be used to run the tasks specified in each *management profile*, hence to execute the closed loop (MAPE). Data acquired by the integrator module will be processed or not into this MAPE module according to the monitoring tasks specified in the profile. If so specified, the included analysis tasks will be carried out and also conditions will be executed in order to render the analysis task positive or negative. Some rules will be executed in order to activate or not the planning tasks configured in the profile and consequently execute the actions described where appropriate. These rules are defined individually for each *management profile*, thus providing an application-dependent mechanism to make inferences over the ontology base. In this way, the developed management module used at the home site inherits some characteristics of an autonomic system due to the fact that the agent runs the tasks described in the profile to completely manage its behaviour. Moreover, update capabilities and context-aware properties have also been included in the architecture of the agent.

4. **Integrator Module**: Information transferred by MDs and also contextual data provided by patients will be acquired in this module, which integrates data coming from different data sources. New instances will be created in order to include acquired information in the ontology, and thus in the *management profile*. As will be explained in the following chapters, according to the protocol used to retrieve information from the MDs, some mapping module implementations are required to interact with this integrator module. Furthermore, the development of an interface to interact with patients is required in order to acquire contextual information.

5. **Rules Module**: This module is used to read and extract the rules from the XML file. It also incorporates a rule engine used to execute any SPARQL sentence required by the Converter module to perform the reasoning tasks or the MAPE module to evaluate the activation of the tasks.

6. **Reminders and alarms module**: Monitoring tasks specified in the ontology can be event-driven or polling-driven. This module is used to activate polling-driven tasks. It includes clock functionalities to ask patients about data (reminders) or to collect information from a specific software resource.

7. **Actions module**: This last module is used to execute actions described within the execution tasks of the *management profile* if an abnormal finding occurs.

3.3.3 Communication method

In the data layer, the communication between the end sites is established using WS technologies (see Figure 3.1). Consequently, a WS has been designed to be placed in the TS and also a web client to be installed in the HG (to establish a communication with the TS). The WS provides the associated *management profile* to all the HGs that apply for it looks for requested data and stores new results provided by the MDs. The first implementations of the web service followed the classic RCP style over SOAP and HTTP protocols. Specifically, Apache Axis2-v1.6.2 was used to deploy both the web service and a client program used to request the server to execute an operation [166]. However, in order to enhance the scalability of the architecture and improve coupling and performance, a REST style web service was finally developed. The interaction required between clients and server is a point-to-point communication where both of them clearly understand the context. Using this alternative, the SOAP overhead is avoided and bandwidth is reduced (of interest when working with resource-constrained mobile devices). Jersey v.1.12 was used to deploy both the web service and client program [167]. This web service comprises and defines a set of operations over the following resources: an OWL **ontology**, the SPARQL **rules** (transfered by means of an XML), **OWL individuals** (sent by the *IndividualWS* structure), **properties** datatype values corresponding to an individual (identified by the URI of the individual and the URI of the property and sent in a string generic type) and **inform messages** to provide some control functions to the web pair communication. Each one of these resources was identified by an URI and a set of operations was defined for each particular resource using HTTP methods (e.g., POST, GET, PUT or DELETE). This web service interface allows information described in the ontology to be exchanged in a generic manner. A summary and graphical view of the resources and corresponding operations is depicted in Figure 3.10.

As shown in the figure, different parameters should be included in each operation and different MIME types defined to transfer data. According to the methods described on the web service, GET operations can be executed to retrieve an OWL ontology (identified by its URI), the rules (identified by the URI of the ontology where they should be applied and a profile identifier), individuals and properties values. POST and PUT methods have been included to create new individuals and properties and update registered ones. Note that a UNA assumption is made to exchange the individuals of the ontology by using the proposed methods. In this way, interchanged resources are defined in a unique manner. Finally, inspired by SNMP commands, INFORM messages have been included as resources to be exchanged. The defined Message Codes are as follows: “0” used to validate an action, “1” used to inform of a failure or error, and “2” used to notify about an update. String responses to PUT and POST follow the same nomenclature. To execute these operations, the client should provide the identifier of the *management profile* related to the specific action, the list of required parameters (according to the operation) and an HMAC code (Hash-based Message Authentication Code) generated by the user with his/her private password.

Hence, this WS provides a set of methods to exchange data independent of the information described in the ontology used for the conceptual layer, thus achieving a flexible and reusable architecture based on ontologies for the communication between the end sites. This WS design contributes to improving the scalability of the architecture. As can be seen, it is not limited to exchanging information described in a specific ontology. The main idea using this approach is that the architecture can easily be reused for other applications. It provides the solution with full scalability without having to modify the main structure of the data layer if any change is made to the ontology used in the conceptual layer.

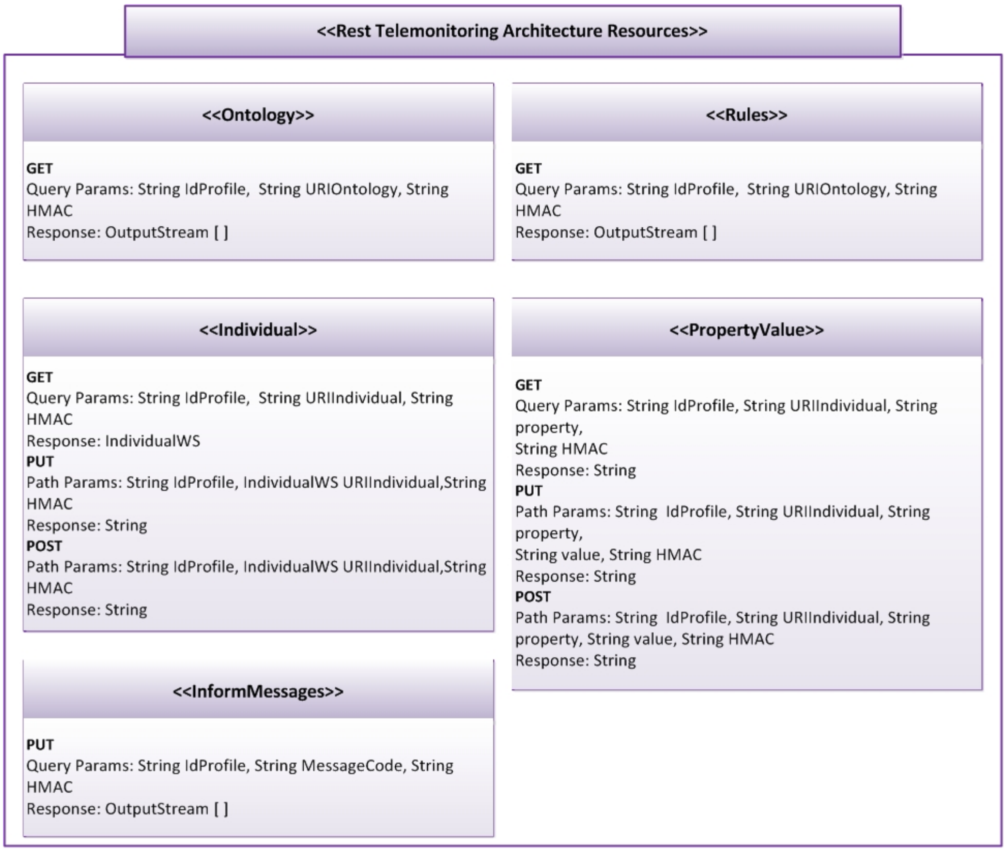

Figure 3.10: Resources and methods described within the Web Service

As mentioned in the description of the Converter module, individuals are exchanged by using a developed structure designated as *IndividualWS* and depicted in Figure 3.11. Using OWL language, an individual of the ontology can be described as a member of a class with individual axioms or facts as individual property values (data properties and object properties). Our proposed structure for exchanging information permits individuals to be described, containing the instance name, the URI of the class it belongs to, its list of data property values and the relations it has with other individuals of the ontology, in other words the list of object properties. Classes at the same level described in the ontology have been marked as *disjoint* classes, hence an individual can not be a member of two classes at the same time. As the object properties link individuals to others individuals, this object structure allows individual definitions to be chained. Thus, when a result is sent from HG to TS, not only is the result instance sent but also the individual instances related to it, putting the received result into context. Consequently, by using this structure the complete *management profile* can be transferred from the TS to the HG as an instance of the *management profile* chained to task instances through object properties. An example of these chained individuals can be seen in Figure 3.12. Not only is the instance of interest transferred to the remote site (highlighted in the Figure), but also the related information (object and data properties) required to put the transferred information completely into context.

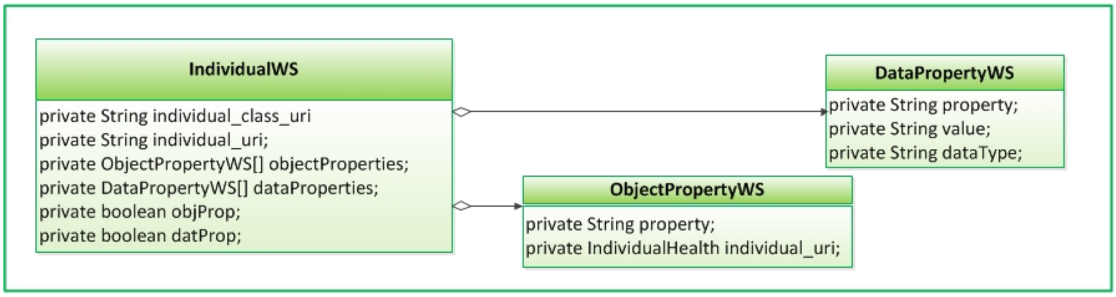

Figure 3.11: Individual WS attributes

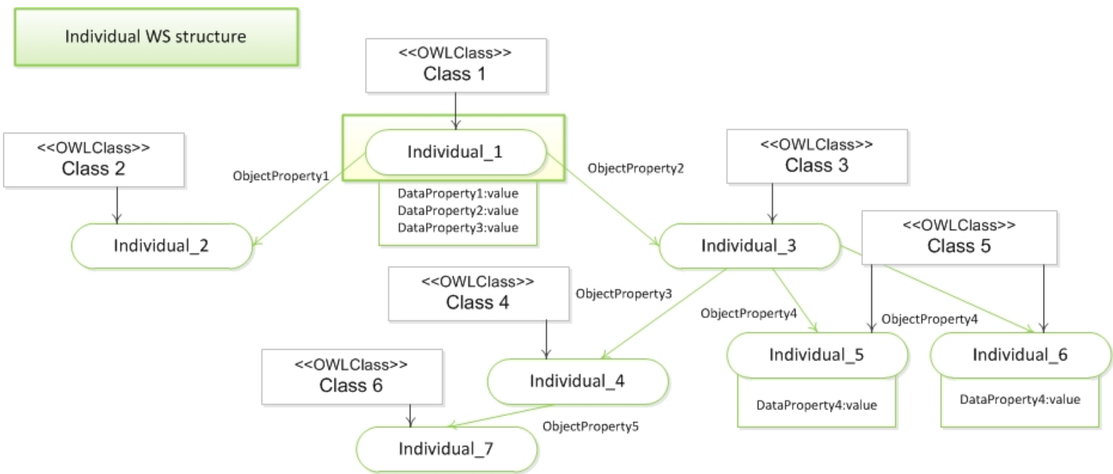

Figure 3.12: Individual WS chain structure

3.3.4 Security

Security in technology communication infrastructures is a big issue of concern, especially when dealing with personal data such as medical information. Some basic security measures have been included in this architecture for user authentication and to guarantee data integrity and reliability of exchanged data. Figure 3.13 depicts this secure system architecture. Final messages represent data transferred to the WS in each request (GET, PUT, etc). “Object” elements refer in general to all defined objects as parameters in the WS methods.

First, to provide security in the communication link, a TSLv1 protocol encryption layer with HTTP (or HTTPS) is used between the web ends points to secure the communication against eavesdroppers and thus provide confidentiality. Furthermore, a complete PKI (public-key infrastructure) has been deployed. To authenticate the web service and enable the TLSv1 usage, a X.509 certificate generated using OpenSSL libraries has been installed in the server site [133, 168]. This certificate contains the server public key signed by our own CA (certification authority). This certificate is presented initially in each HTTPS connection to verify the identity of the server domain. Although the secure layer can be used by authenticating only one side of the communication, user authentication by client certificates has been configured in the architecture. In this way, access to the service and thus to requesting a *management profile* is limited to authorized users. Hence, a X.509 certificate (containing the client’s public key signed by the CA) has been installed in the HG site. The public key of the CA was installed in both server and client trust-stores in order to validate the client and server certificate, respectively. According to the TLSv1 configuration, both web service and client use a symmetric key-based encryption to secure the communication.

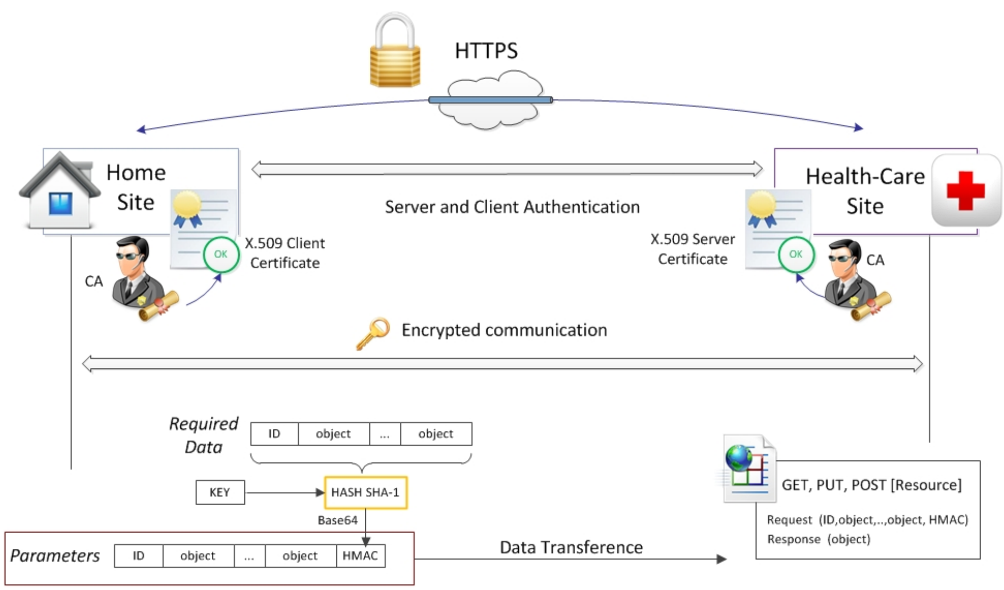

Figure 3.13: Secure Architecture

By using this authentication method, passwords for user identification could be dropped on behalf of the information included in each certificate. However, as each semantic agent (and thus client module) could be used at the home site by different people, an additional security layer has been configured regarding the web service message description. As can be seen in Figure 3.13, each client request should include an HMAC code value encoded in base64. This cryptographic code is calculated through the data that is to be transferred and using a cryptographic hash function (SHA-1 in this implementation) in combination with a secret key (personal for each user that holds a *management profile*). This code provides both data integrity and authenticity to the message. That is to say, by using this method the owner of the transferred data can be identified and it can be checked whether the information has been altered during its transfer. Once the other web service gets the client message request which includes all arguments) values and the HMAC re-generate it's HMAC based on the information transferred in the client message. To generate this HMAC, the server uses the same key that was used by the client which is associated to the user (ID information parameter). The server compares the two HMAC and if they are equal, then the server trusts the client request and provides a response to it. The structure of the message stack regarding an HTTP generic message request is depicted in Figure 3.14.

The use of transport layer security has been proposed in this work to assure data confidentiality while it is transferred in the communication network. Furthermore, the use of the HMAC codes enhances data integrity and user authentication. In this way, some advantages that WS-Security provides to a SOAP-based style have been included in this REST-based architecture. Nonetheless, additional research should be carried out to deal with privacy regarding data available in the *management profile* and its management once it is stored in the remote site.

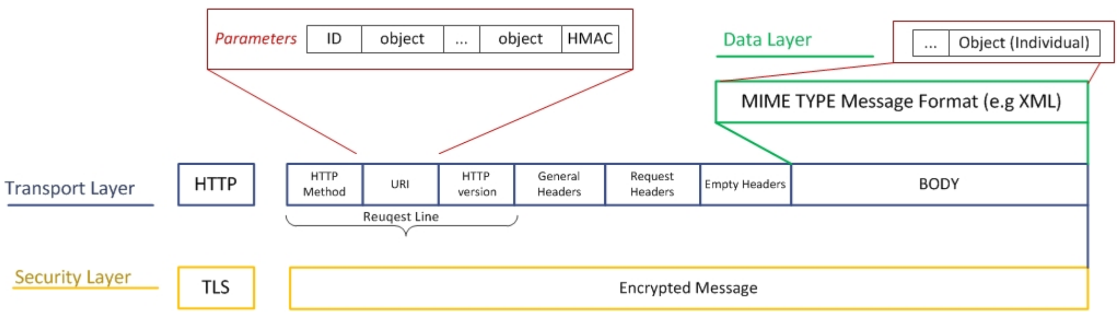

Figure 3.14: Message stack layer

3.3.5 Workflow Procedure

The workflow procedure, depicted in Figure 3.15, includes the following steps:

Step 1: Download Ontology, Profile and Rules

First of all, an instance of the *management profile* should be configured by an EM placed at a remote site (health-care site). Furthermore, a set of individual rules should be configured for each particular management purpose. Then, through a connection to the web service, the user (the patient in the telemonitoring scenario) situated in the home site will acquire the required *management profile*. The first time that he or she requests a *management profile*, the client will ask for the ontology model and the rules. For subsequent requests, the client will request first the current version (using the GET (propertyValue) method) of the profile and depending on the response will ask for an update of the profile or not. This workflow is depicted in Figure 3.15. As can be seen, the methods (mainly GET methods) involved in this step of the workflow procedure are (1) GET (ontology), (2) GET (rules), (3) GET (individual) and (4) GET (propertyValue). Note that the TLS authentication phase is not depicted in Figure 3.15, but it is initially carried out in order to allow the web client connection to the web service or not.

As depicted in Figure 3.15, the associated *management profile* is extracted from the ontology and the Jena instances are wrapped into the *IndividualWS* structure. This is a simple process where individuals and their properties are chained to other individuals and encapsulated into the *IndividualWS* structure. *Inverse* properties are specially considered in order to avoid loops. At the end, a tree-structure *IndividualWS* scheme is acquired and transferred to the HG. Once the *management profile* is in the HG, it will be processed into the Converter module, unwrapped and inserted as Jena individuals in the ontology. The instance checking function is performed in the Converter module in order to ensure that the unwrapped instances are correctly built and thus satisfy the ontology model. Only named individuals and members of a named class are accepted for checking. For each *IndividualWS* unwrapped, the existence of the class in the ontology is first checked. Then, the ontology is queried in order to retrieve necessary conditions related to the class and verify that the individual satisfies them. Both extensions of the HOTMES ontology only contain classes with necessary but not sufficient conditions, thus they are primitive classes. Hence, it can not be said that any (random) individual that satisfies the condition of class A is a member of that class. This fact simplifies the reasoning task as each unwrapped individual belongs to a class and sufficient conditions of, for example, class A are not evaluated in order to assume that an individual satisfying these conditions is a member of class A. Checking for class subsumption is a task for a logic reasoner. No doubt the introduction of complex properties (transitivity, symmetry, etc) in the ontology or the use of necessary and sufficient conditions for class descriptions would require the inclusion of a powerful reasoner in the prototype implementation.

If after the evaluation of all object and data properties included in the *IndividualWS* these conditions are not satisfied, the individual is removed from the ontology. Related individuals are added too. Nearly all the conditions included in both HOTMES ontology extensions are value and cardinality constraints.

Once the necessary conditions are extracted, the list of object and data properties is evaluated. The domain and range of associated object properties are evaluated (not only *rdfs:domain* and *rdfs:range* but also *UnionOf* domains and range classes, superclass properties and the enumeration of individuals comprising domains and ranges), the same evaluation is performed for data properties and finally the class constraints are evaluated (e.g cardinality) in order to verify that the relations provided are correctly used. OWL domains and ranges are not strictly constraints to be checked. They are used as axioms in reasoning. For example, if the hasMonitoringTask property has the domain as *MonitoringTask* class and then this property is applied to an instance of the *AnalysisTask* class, that would not be an error. It would be inferred that *AnalysisTask* is a subclass of the *MonitoringTask*. However, that would not be a correct reasoning in the HOTMES ontology where *MonitoringTask* and *AnalysisTask* are declared as disjoint classes. Nearly all the classes have been declared as disjoint, consequently an individual can not be a member of two classes, and domain and range are considered as constraints in the reasoning performed. The Jena engine and 11 SPARQL queries were required to carry out this verification process. This procedure is depicted in Figure 3.16. Note that this implementation was designed to execute the tasks described in the HOTMES OWL-DL ontology. However, only two additional SPARQL queries were required to adapt this implementation to the HOTMES OWL 2 ontology. Specifically, they were required to verify qualified cardinalities and keys.

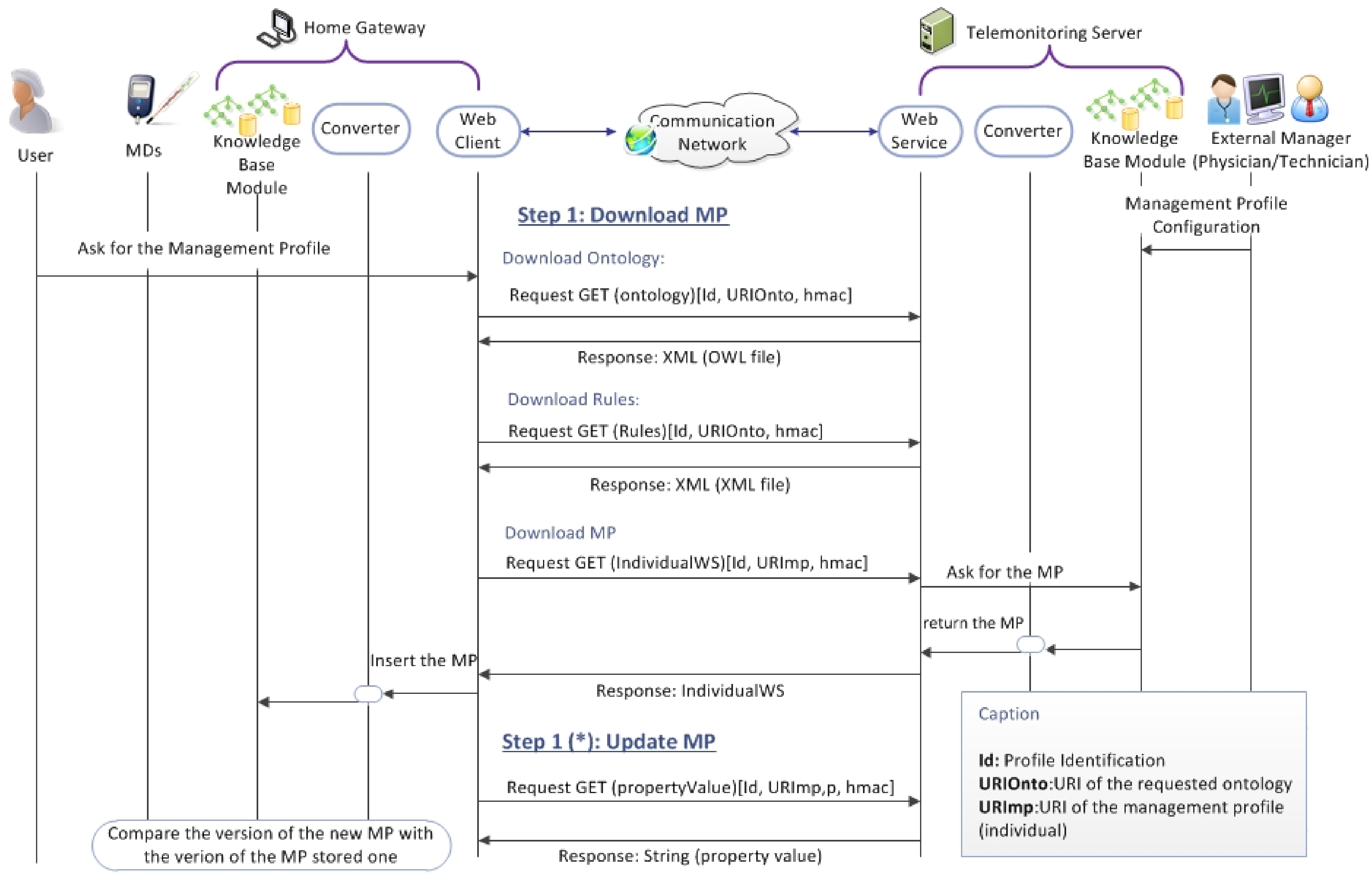

Figure 3.15: Workflow Communication Procedure (Step 1)

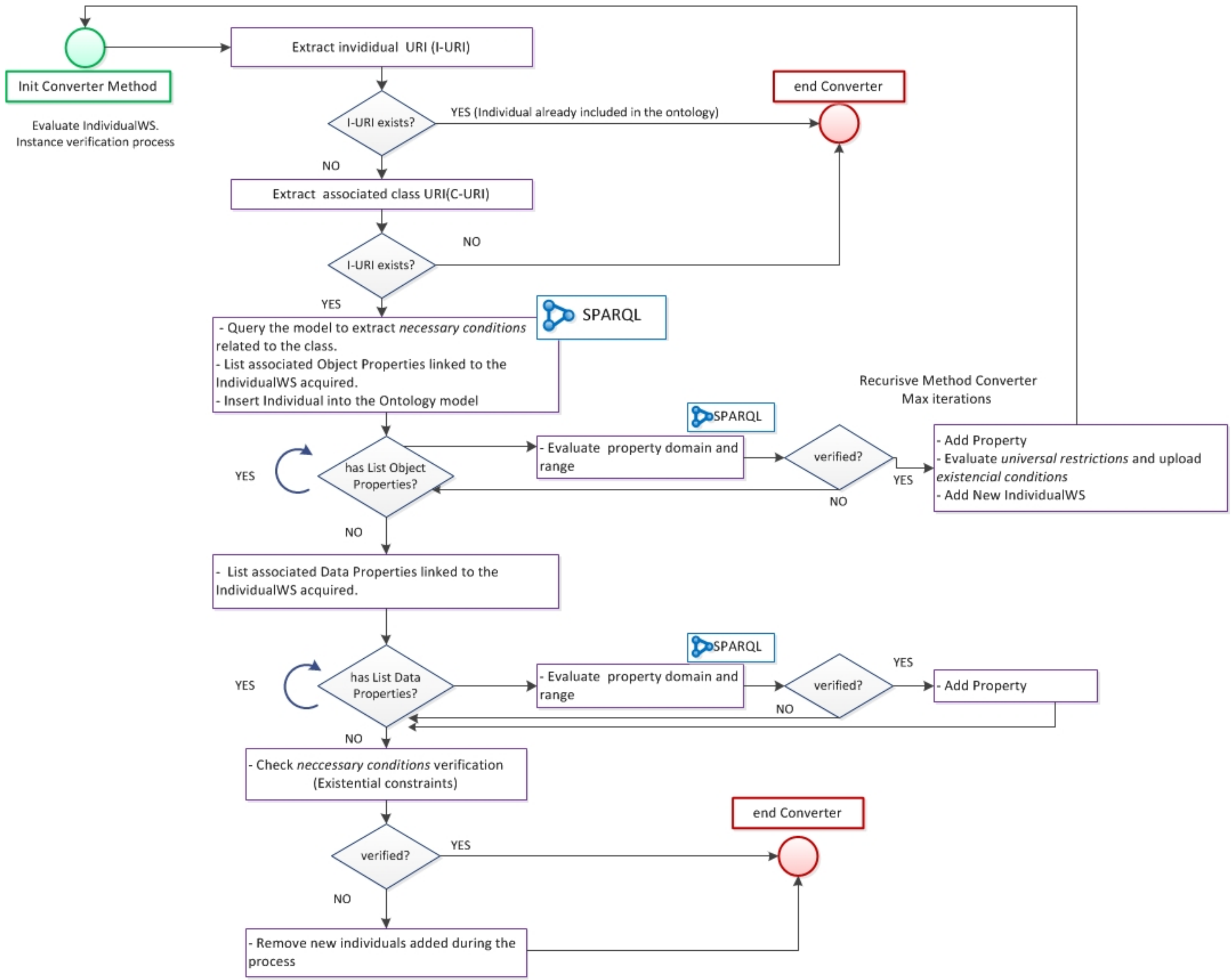

Figure 3.16: Converter Method algorithm

Step 2: Running the management profile tasks

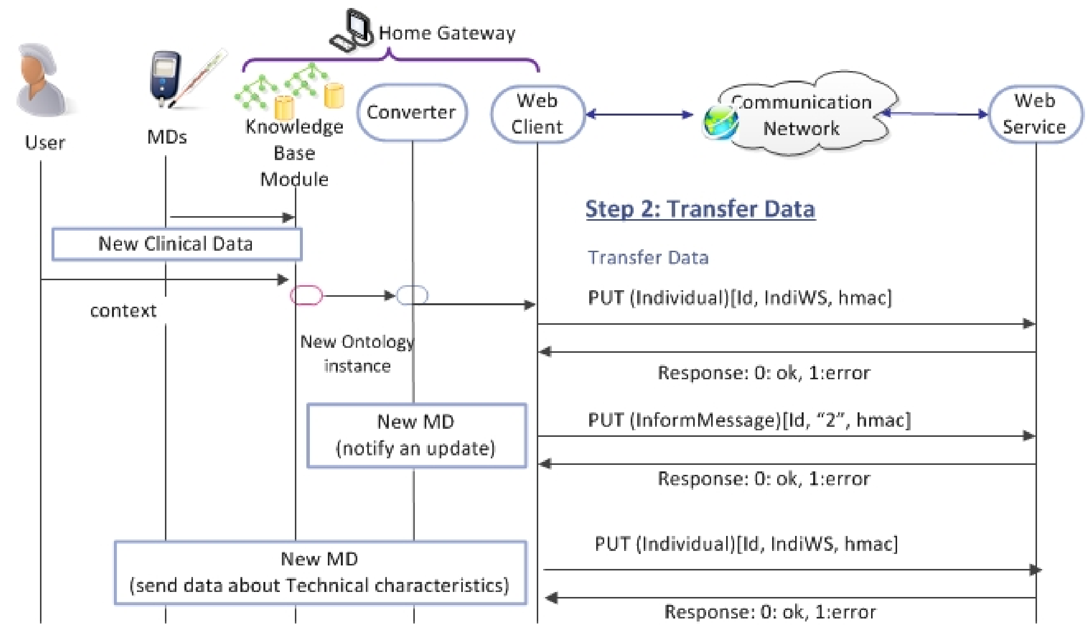

Figure 3.17: Workflow Communication Procedure (Step 2)

Once the *management profile* has been included in the ontology knowledge base module of the HG, it will be evaluated in the MAPE module and the management procedure will be performed by running the tasks specified in the profile. Under the MAPE process, first the monitoring tasks will be checked in order to know which information should be collected. These data will be transferred from different data sources (e.g MDs, sensors, HG characteristics or user feedback) to the HG by using the particular format or protocol appropriate to each one. Data provided by the different sources included in the managed scenario will be mapped into the ontology model and thus transformed into ontology instances. In this way, data integration is achieved (examples of this process are provided in chapters 4 and 5 for clinical and technical management). Then, according to the analysis task description included in the *management profile*, these monitored data will be analyzed. This analysis will have a positive or negative result after the execution of the corresponding SPARQL rules. This positive or negative value will lead to the activation of a planning task and consequently the execution of actions required to face the detected problem. In this step of the workflow procedure, the following methods are executed : (1) PUT (individual), (2) POST (individual), (3) PUT (propertyValue), (4) POST (propertyValue) and (5) PUT (informMessage). For example, the PUT (individual) method will be used to transfer information about a new result included in the *management profile*. It may be possible, by configuring the *TransmissionPolicy* related to a monitoring task, to specify that every acquired result should be transferred to the remote site or, by configuring in the analysis task, to transfer only the result of a positive analysis. For new individuals this method will be used. POST (individual) and POST (propertyValue) can be used to modify existing individuals or values. Finally, an Inform message could be used to report to the external manager that a new MD is being used by the patient and that it should be technically controlled. Figure 3.17 depicts the workflow of messages sent to the remote site if a new medical result is acquired by a new MD (thus activating both types of management, technical and clinical).

3.4 Discussion

This chapter describes a generic ontology-based solution to integrate data and its management procedure in home-based scenarios. Hence, the approach could easily be used in other applications involving remote management tasks at home sites. Furthermore, it facilitates the integration of several management services at home sites using the same software engine. The architecture has been specifically studied to support both technical and clinical services in the telemonitoring scenario. Its layer structure and its communication method provide the approach with scalability and re-usability features.

In the last decade, ontologies have become popular to represent knowledge and they have also been successfully applied to achieve data integration in scenarios such as clinical or home-smart environments where heterogeneous sources work together. While the experiences for clinical purposes [86] and for a home-smart environment [82] are good examples of ontology-based approaches to achieve data integration, they lack the provision of a clear methodology to describe and explain the sequence of tasks for processing acquired data, that is to say the workflow that dictates the management service (e.g telemonitoring) process. Our ontology-based approach was designed to represent not only data modelling but also modelling of the management procedure execution.

The idea of using ontologies to provide knowledge formalization in combination with the autonomic computing paradigm to organize processes has also been proposed in other works such as [101, 122] for different application fields (personalized services in home environments, intrusion detection and data warehouse improvement, respectively). Nevertheless, in contrast to these approaches that use ontologies as a core knowledge describing the knowledge upon which the autonomic system is to work, the idea presented in this work is to provide an ontology for describing both the data to be managed but also the tasks that make up the MAPE process. In this way, data integration is achieved but a clear explanation is also provided about how to process data, providing an abstract view about the business model. In fact, this approach of modeling the management procedure execution with an ontology and the idea of using it as a general framework to design individual *management profiles* constitute the key differences in contrast to other ontology based solutions in home-based scenarios [82, 86, 88].

The HOTMES ontology was written in OWL-DL. This language offers sufficient features and expressiveness to accurately describe a *management profile*. As seen in this chapter, some improvements could be introduced by using new constraints available in OWL 2. However, these new constraints were not deemed necessary to express the required knowledge. Hence, OWL-DL was selected in order to make the ontology easy to understand and to take advantage of available software tools for working with ontologies.

Moreover, the idea of defining *management profiles* in combination with SPARQL rules offers a flexible solution to personalize management tasks, which in fact is a difference compared with similar solutions presented for home-based scenarios [82, 86, 88]. Although this SPARQL language is not strictly speaking a rule language, it has been successfully applied to define rules over individual management tasks and it has also been used to define arithmetic functions to clarify the mathematical analysis process. It is therefore able to provide an explicit representation of all the actions involved in the management procedure.

In addition to the theoretical approach (conceptual layer), the proposed solution has been studied from a practical view point for managing data in telemonitoring scenarios (data layer). This is of interest given that while many proposals have been presented regarding the use of ontologies in the medical field, few of them have given consideration to the technologies used to physically exchange the information described on them [86]. Some proposals such as [169] aim to contribute with an architecture for optimizing the use of resources and messages exchanged among the actors involved. This work uses the ontology to capture the knowledge supporting the management of message exchange between different actors in the telemedicine system using XML files. Other works such as [86] or [88] use web service technologies to exchange the information described in the ontology. However, the services described seem to be specific to the model of the information described. In contrast to these approaches, the architecture presented is an innovative REST-inspired architecture that allows information contained in an ontology to be exchanged in a generic manner. It is a secure and scalable architecture with a simple communication method.

This architecture has been used as the baseline to provide technical and clinical services in a telemonitoring scenario. Following the three-stage approach presented in this chapter and depicted in Figure 3.2, chapter 4 describes how this solution was adapted to remotely monitor patients with chronic conditions. In essence, the HOTMES ontology was extended to this clinical purpose, as explained in section 3.2.3. The definition of *management profiles* for patients was then evaluated. Finally, the physical architecture was adapted to support these services. A similar process was performed to adapt the architecture to technical purposes. Chapter 5 explains, following the same methodology (see Figure 3.2), how the HOTMES ontology was extended and evaluated, and the prototype adapted. Furthermore, this architecture is compared with an SNMP solution.

Chapter 4

Clinical Management

The ontology-driven solution presented in chapter 3 can be used within a telemonitoring scenario to monitor remote chronic patients at their homes. Following the 3 stage approach proposed in that chapter, the main challenges of integration and personalised medical care have been met through the ontology development and evaluation from engineering and clinical points of view, both theoretical and practical. During the first stage, extensions and adjustments in the ontology have been made in order to take into account physicians' knowledge and MD interoperability. During the second stage, the application of the ontology has been studied to monitor patients with different and multiple morbidities using the ontology model to specify the monitored data and alarms configured by clinical staff. In the third and final stage, the prototype designed as part of the data and communication layer of the architecture to perform the tasks that enable remote patients to be supervised have been adapted to the clinical scenario conditions.

4.1 Introduction

Remote supervision of patient health status together with appropriate feedback provision is made possible by periodically collecting clinical data located at patients' homes and transferring them to the corresponding physician located in a health-care site. This telemonitoring process, and thus the data required to supervise the evolution of patient health status, is managed through indications or guidelines provided by physicians. Through these guidelines, physicians indicate how many measurements the patient should take, sometimes even specifying alarms to be triggered by acquired information in order to be alerted if an abnormal finding is detected and indicating the actions that should be taken. In fact, all the guidelines and actions can be implemented as tasks to be performed in the HG in order to provide remote control of patient disease supervision, that is to say, a *management profile* for patients. Consequently, this type of profile has been termed the *patient profile*. Hence, extending the HOTMES ontology for this clinical purpose, a generic framework to design *patient profiles* can be achieved. By defining instances of the monitoring task (and creating new subclasses of the *Monitored Data* class) the physician can specify the vital constant measurement (e.g blood pressure) that a patient should perform and its schedule. Then, by defining instances of the analysis task, the acquired measurement can be evaluated within a range in order to detect abnormalities and, finally, the planning and execution tasks can be configured to remind the patient to take a medicament or to be aware if something abnormal occurs. As explained previously, an instance of the *management profile* can be designed according to the characteristics of the patient's disease, providing personalized clinical management. This ontology extension named "HOTMES clinical" provides a generic framework to define *patient profiles* that can be used in any home telemonitoring scenario e.g for cases of physical diseases, psychological disorders or simply diet supervision. Thus, it is not restricted to a few chronic conditions. Furthermore, its structure describes clearly and simply the workflow of the management procedure. These factors undoubtedly enhance its re-usability and transferability.

As stated in the previous chapter, our system-design approach was divided into three stages. The first stage dealt with the ontology engineering, the second stage with the ontology application-based study and the third stage with the prototype development. Figure 4.1 depicts inputs, actions and outputs required for each stage to adapt the ontology-based architecture to clinical services. The inputs required to carry out each phase are represented on the left side of the image (materials involved in each stage), the actions performed in the centre and the outcomes or results of each phase on the right.

The data integration challenge was tackled through the ontology developed during the first stage of our study, the ontology engineering phase. The main objective was to extend the HOTMES ontology in order to achieve a generic model to monitor a patient with any type of chronic condition. An important task performed during this stage was identifying and defining all the information that could be relevant (and usually necessary) to be collected at the patient's home in order to correctly supervise the patient's health status evolution. As can be seen in Figure 4.1, both a literature review (of clinical guidelines, experiences in telemonitoring scenarios and MD data formats) and physicians' knowledge were used in the ontology engineering stage. As an outcome of this phase, a formal conceptual schema to define individual *patient profiles* was achieved in which data provided by the different sources participating in the telemonitoring process could be mapped. Furthermore, its description allowed an integrated management procedure to be carried out by including in the *patient profile* the tasks that the HG should perform in order to monitor the data defined in the physician's guidelines, analyze the data acquired and execute the appropriate action if an abnormal finding is detected.

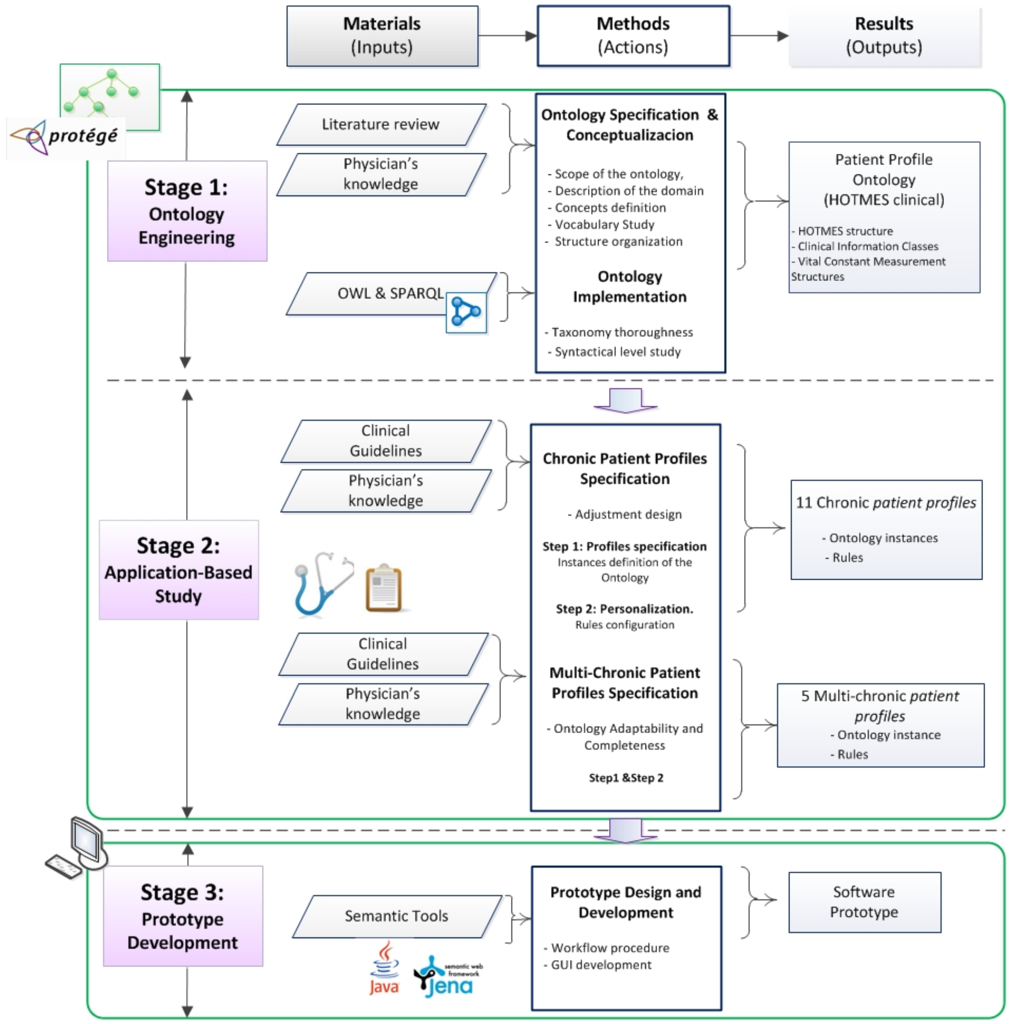

Figure 4.1: Three-stage work flow for clinical data

The main goal of the second stage was to demonstrate by defining different cases of use that the proposed ontology could be used to define any type of *patient profile*. This stage involved two sub-phases: 1) chronic *patient profile* specification, and 2) multi-chronic *patient profile* specification. Furthermore, two steps were defined in each one of these sub-stages: Step 1) Profile Specification (related to the definition of instances of the ontology), and Step 2) Personalization (related to the definition of rules). It should be noted that in order to provide personalized monitoring services, apart from an instance of the *patient profile* ontology, a number of rules should be individually defined for each patient. The main purpose of the first sub-stage was the specification of different profiles in order to adjust the design of the ontology. In fact, this process was part of the dynamic ontology development process and at the end of this stage some modifications were included in the ontology model. By using clinical guidelines and physicians' knowledge, a total of 11 chronic *patient profiles* that allowed 11 types of generic chronic patients with different chronic conditions to be monitored were defined as instances of the proposed ontology and rules configuration. In the second sub-stage, the ontology adaptability (used to measure how far the ontology anticipates its uses [79]) and completeness were studied in order to verify the use of the ontology for defining new profiles by designing *patient profiles* for the supervision of co-morbid chronic patients. Regarding the two steps involved in the *patient profile* specification process, it should be noted that first the suitable instances of the ontology should be defined and then, in order to provide personalized services (within the same chronic condition as other patients), a limited number of rules (according to each pathology) should be defined together with an instance of the *patient profile*. Hence, this stage demonstrated that it was possible to provide context-aware and personalized services with the proposed ontology-based solution.

Finally, the third stage of the approach dealt with a prototype implementation used to run the tasks specified in the *patient profile*. In particular, the Semantic Autonomic Agent located in the HG was adapted to integrate data and perform specified tasks described in this ontology extension.

4.2 Stage 1 (clinical): Ontology Engineering

As mentioned above, stage one dealt with the ontology engineering and included two sub-stages, the first addressing the specification and conceptualization, and the second its implementation.

4.2.1 Materials and Methods

After a review of home-based telemonitoring systems [7, 8, 86, 88], a list of competency questions relating to the knowledge to be expressed in the ontology was initially formulated in order to determine its scope and define its structure (see Figure 4.2) [81]. These competency questions were later used to evaluate the design of the ontology. In order to define concepts included in the ontology and the taxonomy of their relations, two main research tasks were then carried out. First, clinicians meetings were scheduled and, secondly, the X73 standard was reviewed. These tasks were performed to model concepts relating to the results of vital constant measurements that could be acquired in a home-based telemonitoring scenario. These results included information that could be provided by a MD and contextual information required to correctly assess a measurement result.

For the first task, a total of 20 hours of meetings were arranged with clinical staff (2 physicians) in order to identify two types of data: 1) information that could be acquired in a home-based telemonitoring scenario to be monitored for patient supervision (eg. vital constant measurements) and 2) contextual information that could influence a measurement result and that should be provided additionally by the patient. All these data were used to describe concepts included in the ontology of the *patient profile*. In order to derive all the data to be included in the ontology, the clinical staff identified types of chronic patient candidates to be supervised at home site. At the end of this process, the data included in the ontology could be seen as the total sum of requirements for monitoring individual patients with chronic conditions. As a consequence, the resulting model included generic data not confined to monitoring a patient with some specific chronic disease but representing combinations of a set of measurements and requirements allowing any one of them to be monitored. Considerations for the system design and its functionality were also discussed during the meetings with clinicians.

After identifying through the clinician meetings all the vital constant measurements that could be acquired with a MD placed at a patient's home, the X73 standard was reviewed in order to take into account MD data formats and interoperability issues. Although data from MDs from different vendors could be mapped into the ontology model, this standard was reviewed in order to take advantage of existing and mature models and to facilitate future integration with MDs implementing the standard. All features and constraints that dictate how the X73-PHD family of standards is applied for each personal health device agent (installed in the MD) are defined in a device specialization document. A total of 7 device specialization documents were reviewed, referring to the weighing scale, thermometer, blood-pressure monitor, pulse-oximeter, glucose meter, body composition analyzer and peak expiratory flow monitor. This information was used to describe vital constant measurement results in terms of units and data type values, and to know how much of the information that could be provided by a MD needs to be collected for a patient's evaluation. Note that 16 device specialization documents have been written (some of them are still drafts) within the X73 standard. Nonetheless, only device specialization documents related to the clinical measurements of interest for monitoring patients were considered.

Competency Questions

- Which data should be collected to supervise the patient?
- How often should the patient take a measurement or provide data?
- Should the acquired data be transmitted to the healthcare site?
- How will the data be analysed?
- Should an alarm be triggered according to the evaluation results?
- Which actions should be performed if an alarm is triggered?

Figure 4.2: Competency Questions (clinical)

4.2.2 Results: Data Integration

As described above, the information required for home-based telemonitoring was identified after meetings with clinicians and a literature review. The structure of a preliminary ontology was then designed to cover all this information and, as a final outcome of this phase, an initial version of the ontology (different from the HOTMES ontology) was developed [170]. This first version offered a generic structure to define individual *patient profiles* and a good solution to provide context-aware services and data integration from the different sources. Nevertheless, this ontology lacked a unified and integrated management procedure for the acquired monitored data. Consequently, after studying this problem from an engineering angle, a generic ontology to manage data in home-based scenarios was developed, the HOTMES ontology presented in chapter 3. This ontology offers a solution for data and management integration that can be used to manage any type of data in home-based scenarios.

4.2.2.1 HOTMES for clinical purposes

For this specific clinical purpose, the ontology model had to be extended in order to include all the information required to monitor a patient's health status within the *Monitoring Task* class. All this information was divided into 4 groups represented in the ontology as 4 classes (subclasses of the *Control Information* class). Furthermore, the *Polling Task* (subclass of the monitoring task) was slightly modified in order to include the Reminders concept (alarms directed to patients) in the ontology. As depicted in Figure 4.3, there are two ways of collecting data: by polling (*Polling Task*) or by events (*Event Task*). Through polling, patients are required to provide some information following which a reminder will be activated in order to alert them about the action they should perform. In this way, a physician can schedule a patient to perform a pressure measurement or take medication once per day. Through events, patients or MDs provide the information whenever they want.

An instance of a *syntactical analysis* is configured in order to compare string datatype values acquired in the home-based scenario such as, for example, a response provided by the patient to a question configured by the physician. An instance of a *mathematical analysis* is configured in order to compare a numeric value (normally after being mathematically manipulated though a process) with a threshold configured by the physician. These processes can be simple or complex. In this way, by defining an analysis task it is possible to evaluate e.g the temperature value provided by a thermometer in order to know if it is within a normal range for this patient or simply to configure a check of the temperature value conditioned to a pressure-measurement evaluation. In general, analysis tasks will be used to detect abnormal findings, thus to generate alarms.

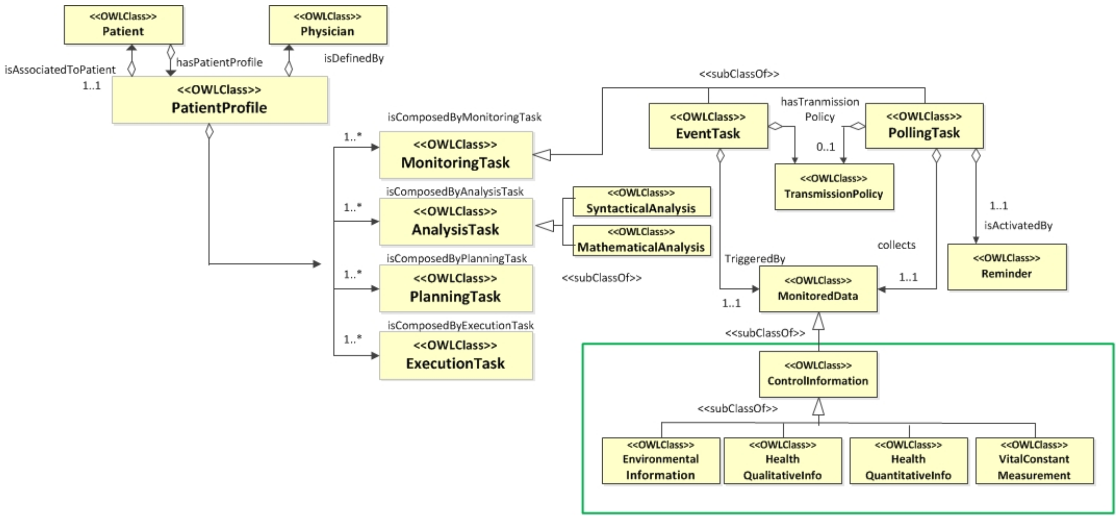

Figure 4.3: HOTMES clinical version

Each *planning task* will be activated if the associated pre and post conditions (expressed by SPARQL rules) are true. In general, they will be activated after obtaining a positive value in an analysis task in order to execute the correct actions to solve the observed problem. These actions are defined as instances of the execution tasks. For instance, a planning task consisting of warning the physician by e-mail and advising the patient to repeat the measurement can be configured.

The version of the HOTMES ontology for clinical application contains 163 classes, 121 object properties, 90 data properties and 82 individuals. Furthermore, 75 annotations properties have been included in the ontology. Of these, 49 are used to relate terms from the ontology with the X73 standard as will be explained in the following sections. The size of the OWL ontology file is 270 KB and the DL expressiveness is ALCHON(D). As shown in Figure 4.3, each *patient profile* is associated to one patient and defined by a physician.

4.2.2.2 Clinical Information Classes

As mentioned above, the data identified by physicians as relevant for monitoring patients' health status at home were classified into 4 groups described in the ontology as 4 general classes: environmental information, qualitative information, quantitative information and vital constant measurements. These classes are highlighted in Figure 4.3 and shown in detail in Figure 4.4. As can be seen in Figure 4.3, monitored data at the patient's home could include other types of information not specifically required for monitoring the evolution of the patient's disease. For example, the ontology could be used for fitness purposes, thus defining an exercise profile. If combined with the clinical purpose, then this information would be described under the monitored data at the same level of the *Control Information* class.

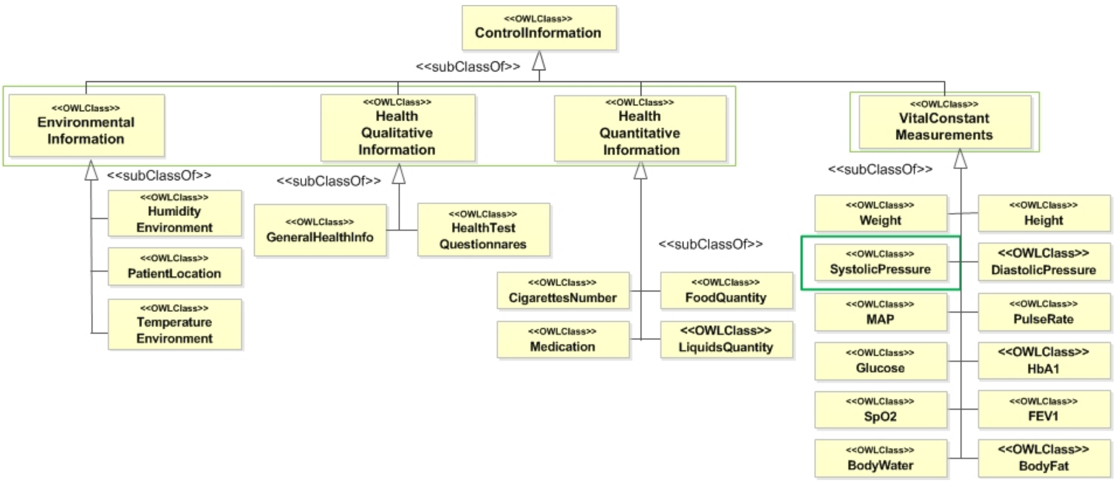

Figure 4.4: Clinical Data Classes and subclasses

The *Environmental Information* class includes the humidity environment, patient location and temperature environment subclasses. For patients with chronic obstructive pulmonary disease (COPD) or asthma diseases (spirometry patients), it may be useful to collect information about the environmental characteristics of the room where they perform the measurements.

- *Humidity Environment*: refers to the percentage of humidity measured in the room where the patient acquires clinical measurements. Each datum acquired will be expressed by percentage units and will include the date when it was obtained (data property of the datum class).

- *Patient Location*: indicates the room used by the patient to acquire measurements.

- *Temperature Environment*: indicates the temperature of the room where the patient acquires measurements. The temperature will be expressed by temperature units (Fahrenheit or Celsius).

Health Qualitative Information refers to subjective information provided by patients such as their stress level, mood or specific answers to personalized questions. This information is quite useful for example in supervising patients with mental disorders or for controlling adverse reactions caused by medication at the start of a new treatment. This section includes two subclasses: *General Health Info* and *Health Tests Questionnaires*. These classes have been depicted in Figure 4.5.

- *General Health Info*: An instance of this class could be configured in order to ask patients routine questions about stress or mood levels.

- *Health Tests Questionnaires*: An instance of this class can be configured in order to detail a sequence of questions (e.g about physical symptoms) that together with measurement data or by themselves could indicate that the patient is getting better or worse. As shown in Figure 4.5, each questionnaire will contain a list of questions (Question Health). Then, for each question the patient will provide an answer (*Answer HealthQ*). The relations between these classes are depicted in Table 4.1. Each question will be characterized by an identifier and a string containing the question itself (expressed in natural language) and each answer will be characterized by an identifier, the answer itself and the date when it was provided.

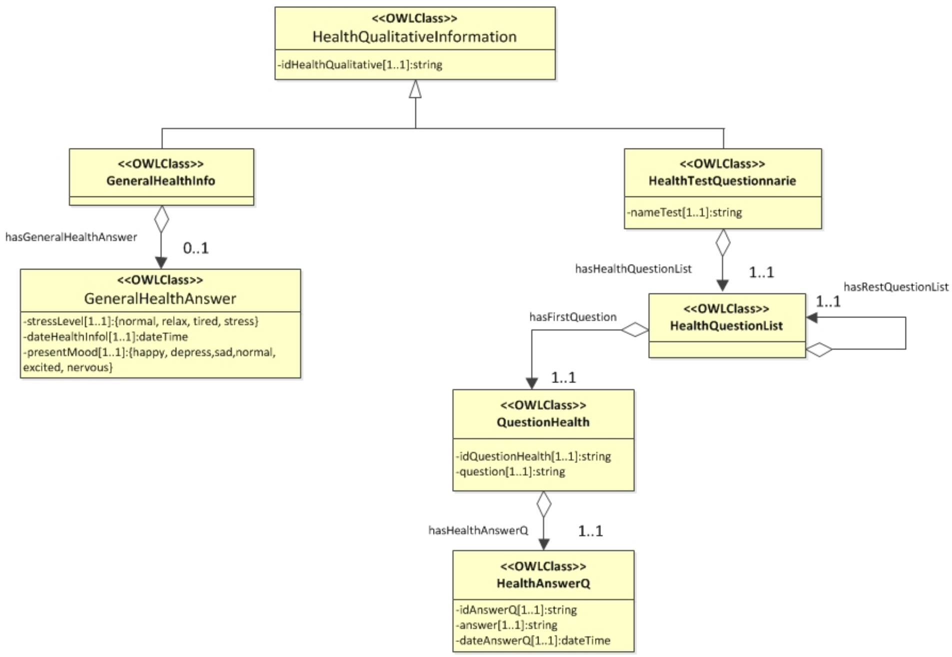

Figure 4.5: Health Test Questionnaires

Object Property	Domain	Range	Description
hasHealthQuestionsList	HealthTest Questionnaire	Health QuestionsList	Used to link the questionnaire with the list of questions it contains.
hasFirstQuestion	Health QuestionsList	HealthQuestion	First question on the list.
hasRestQuestionList	Health QuestionsList	Health QuestionsList	Rest of questions on the list to be asked.
hasHealthAnswerQ	Health Question	Health AnswerQ	Used to relate each question with the answer provided by the patient.

Table 4.1: Health Qualitative Information Properties

Health Quantitative Information refers to quantitative information that is not acquired by an MD. The following subclasses of the *Health Information* class have been defined: *cigarettes number*, *food quantity*, *liquid quantity* and *medication*. This information can be useful for clinical data measurements and patient supervision. For instance, controlling the quantity of food intake can be very interesting for following up patients with obesity or thyroid disorders.

- *Cigarettes Number*: used to ask the patient about the number of cigarettes smoked per day.

- *Food quantity*: indicates the quantity of a certain type of food (dairy, fruits, grains-nuts-seeds, meats, sweets-fats or vegetables) consumed by the patient. Each instance of acquired food datum will be expressed by a numerical value and the units in grams (gr) or milligrams (mgr).

- *Liquid quantity*: an instance of this class will be used to ask the patient about the quantity of liquids he or she usually drinks. It can be general or restricted to a certain type of liquid (alcohol, coffee, soft-drink or water). Each instance of acquired liquid datum will be expressed by a numerical value and the units in liters.

- *Medication*: an instance of the medication class will be used to ask and remind the patient about the number of pills that should be taken.

Measurement	SNOMED-CT code	X73 Device Specialization (11073-104zz)	Description
Temperature	246508008	Thermometer (-10408)	Body temperature value.
Weight	272102008	Weighing scale (-10415)	Measurement of body weight
Height	50373000	Weighing scale (-10415)	Measurement of body height.
Systolic BP	72313002	Blood Pressure Monitor (-10407)	Highest value of a blood pressure measurement during the cardiac cycle.
Diastolic BP	67726005	Blood Pressure Monitor (-10407)	Lowest value of a blood pressure measurement during the cardiac cycle.
MAP	264861000	Blood Pressure Monitor (-10407)	Mean value of a blood pressure measurement.
Glucose	36048009	Glucose meter (-10417)	Measurement of glucose in blood.
Pulse Rate	364094000	Blood Pressure Monitor (-10407) or Pulse-oximeter	Number of heart-beats per unit of time (normally measured in beats per minute).
SpO2	113080007	Pulse-oximeter (-10404)	Measurement of the amount of oxygen in the patient's body.
HbA1	259689004	Glucose meter (-10417)	Measurement of glycated hemoglobin in blood.
FEV1	74274000	Peak expiratory flow monitor (-10421)	Measurement of the forced expiratory volume in one second.
Body Water	38348009	Body composition analyzer (-10420)	Measurement of the amount of water in the patient's body.
Body Fat	248300009	Body composition analyzer (-10420)	Measurement of the amount of fat in the patient's body.

Table 4.2: Vital Constants Measurements Description

Finally, a wide range of possible measurements that can be acquired in a home telemonitoring scenario have been described in the ontology in order to provide a flexible structure for configuring profiles for multiple and specific telemonitoring cases of use. A total of 11 measurements were identified and included in the ontology as subclasses of the *Vital Constant Measurement* class. These measurement subclasses are: *temperature*, *weight*, *height*, *blood pressure* (*systolic*, *diastolic* and *MAP*, modeled as 3 different subclasses), *glucose*, *pulse rate*, *SpO2*, *HbA1*, *FEV1*, *body water* and *body fat*. Table 4.2 gives descriptions of these terms and their corresponding SNOMED-CT code. Furthermore, each measurement has been related to the X73 device-specialization used to model the ontology regarding the measurement data. Relating concepts described in the ontology with widely accepted standards makes their descriptions clearer and their mapping easier with other knowledge models with similar contents that overlap. This strategy is in line with the recommendations of Brochhausen et. al. [90] about the need to continually work towards harmonization through ontologies to achieve future unified semantic models. Specifically, concepts described in the ontology were related to standardized terms in medicine using the SNOMED-CT nomenclature. These codes were included in the ontology as individuals of subclasses of the *Measurement Code* class, one for each type of measurement represented in the ontology (eg. SystolicPressureCode). These classes were defined as Enumerated classes (using the *one Of* construct) with one unique individual, the corresponding SNOMED-CT code. Although initially no other individuals can be declared to belong to these classes, the ontology could be extended to include other standardized codes if desired.

4.2.2.3 Vital Constant Measurement

The information used to describe the clinical concepts (located under the monitoring task) has been organized in two levels: 1) the Configuration level and 2) the Data Level. The first level gathers properties used to characterize a clinical concept. An instance of these classes should be configured by the physician when defining the *patient profile* related with a monitoring task. The second level gathers properties used to characterize the corresponding clinical datum (a piece of data retrieved at a certain time). Hence, it describes all the data eventually retrieved from the patient's supervision. Vital constant measurement data were modelled in the ontology as a collection of two types of information: data acquired and provided by the MD and contextual information that could influence a measurement result provided by the patient. An example of a vital constant measurement model (for Systolic pressure, highlighted in Figure 4.4) can be seen in Figure 4.6.

As is depicted, the information represented is classified into two levels. The concepts associated to the first level correspond to configured information provided by the physician. All defined concepts are related to their corresponding SNOMED-CT code, in this case, the Systolic Pressure Measurement itself (*SystolicPressure*).

The second level corresponds to the datum acquired at a certain moment by using a MD and the additional contextual information required to evaluate each measurement (*SystolicPressureDatum*). For example, in the case of blood pressure, the physicians indicated that apart from reporting the data value of the pressure measurements, patients should provide information about which arm they used to take the measurement (left or right), the patient's position (sitting down, standing up or lying down) and if he/she had taken any stimulating substances during the hour prior to taking the measurement (such as coffee, medication, tobacco) (DataProperties: *patientPosition*, *pressureMeasurementLocation* and *stimulating Substances*). Each concept may be related to a storage of that data measurement(*hasStorageSPData*). It should be noted that the range for data properties describing context information has been defined as an enumerated datatype, thus describing a list of possible string values. Each datum acquired by an MD (*DatumMD* class) is characterized by the value of the datum acquired (DataProperty: *outputValue*), the date when it was acquired (DataProperty: dateDatum) and technical information about the MD used to acquire the measurement (DataProperties: *idSystem*, *manufacturer*, *model* and *systemType*). Table 4.3 explains the properties related to the *SystolicPressure* class. The same model structure was used to represent the other vital constant measurements.

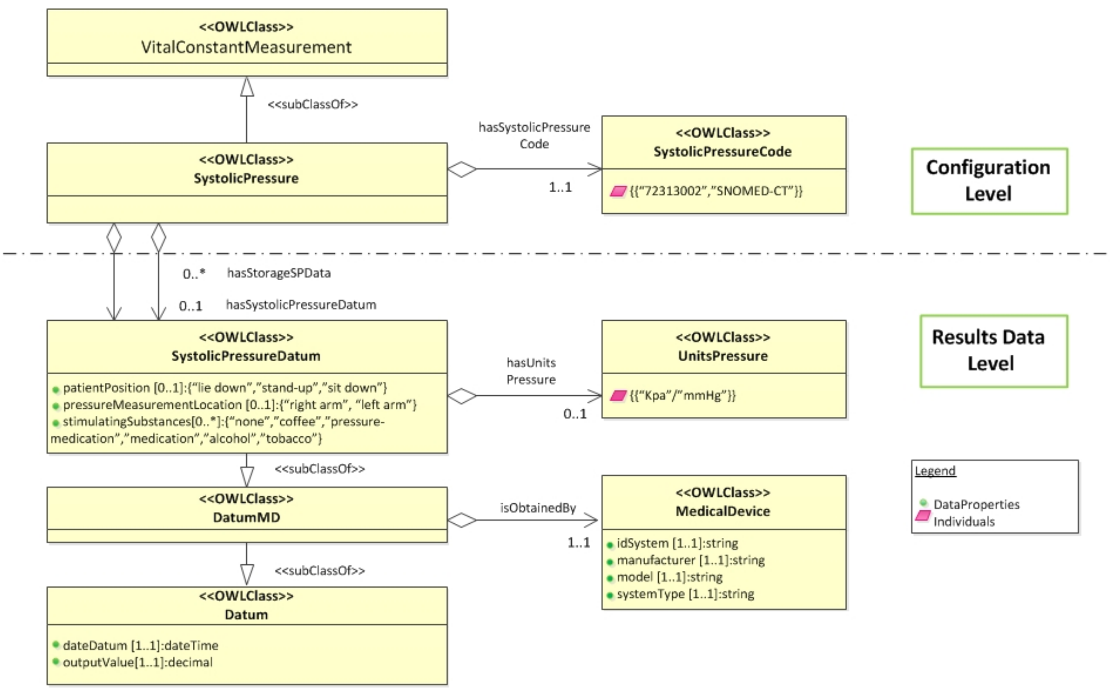

Figure 4.6: Systolic Pressure Model

Object Property	Domain	Range	Description
hasSystolicPressureCode	Systolic Pressure Datum	Systolic Pressure Code	All defined instances of the systolic pressure measurement are associated to the SNOMED-CT code "72313002".
hasSystolicPressureDatum	Systolic Pressure	Systolic Pressure Datum	A piece of data related to the systolic pressure acquired at a certain moment and with some particular conditions. Integrates data provided by the MD (measurement value) and data provided by patients (contextual data).
hasStorageSPData	Systolic Pressure	Systolic Pressure Datum	Used to store a list of data related to the systolic pressure measurement. Equivalent property of hasSystolicPressureDatum (same domain and range but different meaning).
isObtainedBy	Datum MD	Medical Device	Indicates that the measurement has been acquired by using a specific MD. Information regarding the general characteristics of the MD is also collected (e.g type of system: blood pressure monitor).
hasUnitsPressure	Systolic Pressure Datum	Units Pressure	Each acquired datum is associated to its corresponding units. As can be seen in Figure 4.6, mmHg (millimeter of mercury) and KPa (kilopascal) units were included as possible blood pressure units.

Table 4.3: Object properties related to the Systolic Pressure Measurement

Apart from the relation with the SNOMED-CT nomenclature, by using annotation properties the relation between the classes described in the ontology and the attributes and nomenclature used in the X73 standard was incorporated into the ontology model. See for example Table 4.4 were these relations have been depicted for the systolic pressure measurement model. Note that these relations were used for the X73 mapping module implementation within the prototype development. Classes and properties from the ontology are represented in column one. The nomenclature code used to identify each X73 object (column two) is: Class:attribute.subattribute. Class indicates the object class inside the DIM, attribute indicates an attribute inside the class and, in the same way, subattribute indicates that the object is a subattribute. The third column indicates the value of the attribute reported by the X73 agent. This value can be a specific X73 tag or the syntax attribute expressed in terms of X73 notation which is in fact ASN.1 notation. It should be noted that O.String stands for OctetString, Int-x stands for Integer using x bits and Bits-x stands for bit string size x. Finally, the fourth column offers a text description of the presented object.

Ontology Class - Data Property	X73 Attribute Name (Class:attribute.subattri	X73 Attribute Value (tagdatatype)	Description
SystolicPressure	metric::type	MDC_PRESS_BLD_NONINV	Systolic measurement
	metric:: Metric-Id-List	MDC_PRESS_BLD_NONINV_SYS	
SystolicPressureDatum - outputValue	numeric:: Simple-Nu - Observed-Value	FLOAT	Refers to a systolic datum
- DateDatum	metric::Absolute-Time-Stamp	DATE	Date when the measurement was acquired
Units Pressure	metric::type (MDC_ATTR_ID_TYPE)	MDC_DIM_MMHG or MDC_DIM_KILO_PASCAL	Data units of this constant measurement datum value
MedicalDevice			Information about the MD used
- idSystem	MDS: System-Id	O. String	MD identification
- manufacturer	MDS:System-Model.manufacturer	O. String	Manufacturer of the MD
- model	MDS: System-Model.model	O. String	Model of the MD
-systemType	MDS:System-Type-Spec-List	MDC_DEV_SPEC_PROFILE_BP	Indicates the type of MD

Table 4.4: Relation between Patient Profile ontology and the X73 standard

4.3 Stage 2 (clinical): Ontology Application-Based Study

Hartman et. al stressed that ontology evaluation should be a continuous process starting at the design phase and continuing through the development process [171]. This policy was adopted in the present work and continuous improvements were included in the ontology after the Chronic Patient Profile specification sub-stage included in the application-based study stage. The main goal was to make the required adjustments in the ontology in order to be able to use it to supervise patients at home with any type of chronic condition. In fact, this second stage could also be seen as part of the design phase given that some classes of the ontology model were modified after this validation phase.

4.3.1 Materials and Methods

4.3.1.1 Chronic Patient Profile Specification

For the purposes of this stage a physician who did not participate in the ontology engineering phase identified 11 different types of patients with one general chronic condition as possible candidates for remote control. The physician configured 11 *patient profiles*. These configured *patient profiles* (chronic *patient profiles*) were used to supervise patients with the following chronic conditions: 1) COPD, 2) obesity, 3) thyroid disorders, 4) ischemic heart disease (IHD), 5) asthma, 6) hypertension (HTA) or high blood pressure, 7) osteoporosis, 8) heart failure (HF), often called congestive heart failure (CHF), 9) diabetes mellitus (DM), 10) dyslipidemia (DLP) and 11) osteoarthritis. As can be seen in Table 4.5, these specific *patient profiles* where configured due to the fact that they cover diseases prevalent in a high percentage of population. Mortality rates can be considerably reduced if patients are permanently monitored. The profiles were defined according to published standard clinical guidelines used for the normal supervision in healthcare centres of patients with these diseases [172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182].

This Patient Profile specification stage included two steps: 1) Profile Specification and 2) Personalization. The first step consisted of defining measurements to be monitored for each type of patient, reminders, data to be analyzed and actions to be taken. The data defined in this step was used to identify the instances of the ontology required to configure a *patient profile*. In order to define the profile, a Patient Profile questionnaire was designed and then completed by the physician in order to describe all the requirements for monitoring each type of patient. It aims to provide with a guide for the *patient profile* definition according to the information acquired for the ontology engineering phase (through meetings with clinician staff). Table 4.6 shows an example of the Patient Profile questionnaires used in this stage. It should be noted that if the physician required more information in order to define the correct profile, such information was subsequently included in the ontology.

Profile	Description and Symptoms	Impact on population
COPD	A chronic lung disease characterized by persistent airflow limitation. Dyspnea, chronic cough and sputum production are the main symptoms of COPD.	- More than 10% of the European population suffers COPD, the fifth cause of mortality. - More than 75% are undiagnosed and untreated. - 90% of cases are caused by smoke inhalation.
Obesity	A chronic disease characterized by an increase in body fat stores that can lead to reduced life expectancy and/or increased health problems.	- Is responsible for about 35% of ischemic heart diseases, 80% of cases of type 2 diabetes and 55% of hypertensive diseases.
Thyroid Disorders	These are common disorders of the thyroid gland including hyperthyroidism, hypothyroidism and thyroid nodules.	- Around 7% of the general population suffers a thyroid disease and prevalence increases with age. - Thyroid disorders are 5 to 7 times more common in women than in men.
IHD	Patients with IHD have reduced heart pumping due to coronary artery disease. The risk increases with age, smoking, high cholesterol levels, diabetes, obesity and hypertension.	- The first cause of death in Europe. - Nearly 50% of deaths in Europe are caused by heart diseases. - One in 8 men and one in 17 women die before the age of 65 because of heart disease.
Asthma	A common chronic inflammatory lung disease that inflames and narrows the airways.	- About 5% of the population suffers asthma. - It is one of the most common chronic diseases of childhood (10%).
HTA	Common cardiac medical condition in which the systemic arterial pressure is elevated (over 140/90). HTA is a risk factor for IHD, stroke, myocardial infarction, HF, chronic kidney disease, cognitive decline and premature death.	- At least one quarter of adults (and more than half of those older than 60) have high blood pressure. - With each 2 mmHg rise in systolic blood pressure, the risk of mortality from IHD or strokes is increased by 7% and 10% respectively.
Osteoporosis	Osteoporosis is the consequence of continued bone loss throughout adulthood, low peak bone mass, or both.	-Anyone can develop osteoporosis, but it is common in older women (15% of women over 65 years old are affected). - 25-30% of fractures are due to osteoporosis.
HF	Heart failure is a chronic and progressive condition in which the heart muscle is unable to supply enough blood flow to meet the needs of the body.	- Around 2% of adults suffer from heart failure, but in those over the age of 65, this increases to 6-10%. - Heart failure is a serious condition, and usually there is no cure.
DM	A metabolism disorder where levels of blood glucose are too high. There are 4 types of diabetes: type 1 diabetes, type 2 diabetes, gestational diabetes and other types due to genetic defects.	- Diabetes is a leading cause of death and disability. -Type 2 diabetes is the most common type of diabetes (90% of patients). The World Health Organization (WHO) estimated that 6.4% of the world's adult population had diabetes in 2010.
Dyslipidemia	Dyslipidemia is a disorder of the amount of lipoprotein (overproduction or deficiency) in the blood. It is a major, modifiable risk factor for cardiovascular disease.	Middle-aged, white men are likely to be diagnosed and treated for dyslipidemia; however, the disease is most prevalent in post-menopausal women.
Osteoarthritis	This disease is related with abnormalities involving degradation of joints including articular cartilage and subchondral bone.	- Osteoarthritis is not caused by aging and does not necessarily deteriorate.

Table 4.5: Patients Profile (Chronic) Diseases Information

1. Vital Constant Measurements

Select required measurements to be controlled for this patient's supervision.

Weight		Pressure (D/S)		HbA1c	
Height		SpO2		MAP	
Pulse Rate		FEV1		Body Water	
Temperature		Glucose		Body Fat	

2. Subjective Information

Is it relevant to ask the patient about any physiological symptoms or additional qualitative information for his/her supervision? E.g: stress level or pains? Select the required qualitative information.

General Health Information		Ask patient about his/her general health state regarding stress or humor	
Health Test Questionnaires		Request patient to answer specifically the following list of questions	

3. Environmental Information

Could the patient environment affect the acquired measurement result? Should the patient be asked about additional environmental information? Select the categories required.

Patient Location		Room Humidity	
Room Temperature		Temperature Humidity	

4.Quantitative Information

Is there any additional quantitative information required for his/her supervision? Select the required categories.

Food Quantity		Cigarettes Smoked	
Liquids Quantity		Medication Room	

5. Reminders for Patients

Should patients be reminded about acquiring measurements or providing additional information?
- How often? (Specify the hour when the reminder should be activated and the time between activations eg. 1 day, 15 days)
- What message should appear to remind the patient to perform the specified task?

6. Clinical alarms

a. Should the supervising physician be warned if an out of range measurement occurs or if a suspicious answer is provided when asking about quantitative or qualitative information?.
- Can this alarm be conditioned to other monitored results (eg. an out of range measurement value or a positive answer in a health questionnaire test)? Please specify.
-Is any mathematical factor related to the specified measurements useful for triggering an alarm (eg. Body Mass Index, BMI)? Please specify.
b. Which actions should be performed if an alarm is triggered?

Table 4.6: Patient Profile Questionnaires

During the second step, the physician was asked about the specific data values for triggering a defined clinical alarm. The data defined in this step were related to the SPARQL rules definition for the analysis task and planning task activation. The rules were used to provide personalized care and to distinguish between each *patient profile* with the same definition of instances for two patients with the same pathology (e.g COPD) but with different levels of restrictions in their supervision. For instance, for a COPD patient one physician can configure one single rule in order to detect serious COPD, such as "IF FEV1 is <30% AND FEV1>49% THEN activate alarm". However, for another patient, the physician can limit the supervision further and configure rules to detect serious, mild and slight COPD for different FEV1 values. Apart from the specific values, the physician was also asked if the clinical alarm should be conditioned by the contextual information relating to the acquired measurement. This information is converted into a set of IF/THEN rules.

4.3.1.2 Multi-Chronic Patient Profile Specification

The main objective of this phase was to verify that the ontology could be used to define new and more complex *patient profiles*. Consequently, the quality of the ontology was measured in terms of adaptability and completeness by defining *patient profiles* that had not specifically been considered by physicians during the design phase of the ontology. In fact, patients normally suffer more than one chronic condition (e.g an obese patient with high blood pressure) and each patient needs his or her own specific monitoring guidelines. As the purpose of the ontology was to be able to monitor patients with any type of chronic condition, it was necessary to ensure that the domain of the ontology appropriately covered these types of patients and consequently to verify how far the ontology could be used for new cases.

A total of 5 *patient profiles* (multi-chronic patient profiles) for patients with co-morbidities were designed in this phase: 1) MCD (multi-chronic diseases) 1: DM + ischemic, 2) MCD 2: obesity + DLP, 3) MCD 3: osteoporosis + HTA, 4) MCD 4: COPD+ HF, 5) MCD 5: Obesity+DLP+DM+HTA. Table 4.7 gives descriptions and characteristics of patients with these co-morbidities and the importance of controlling them. The physician designed these profiles by using the same questionnaires as those used to define the chronic *patient profiles* in the previous sub-stage, and the same steps were followed. These are, in fact, the steps for generating new *patients profiles*. It should be noted that the ontology could be easily extended to include additional information or more complex analysis processes if required. New data to be monitored could be modeled as subclasses of the *Control Information* class and analysis processes as subclasses of the *Analysis Task*. In the same way, new execution tasks or actions could easily be included in the ontology. The generic structure of the ontology in defining how to manage and evaluate monitored data by adding new subclasses enables its application domain to be extended easily.

Profile	Description and Symptoms
MCD1	The macroangiophatic consequences of diabetes can effect the coronary vessels and peripheral vascular disease. Indeed, the most common cause of death in adults with diabetes is coronary heart disease (80% of morbidities).
MCD2	Obesity is creating concern because of its impact on morbidity and mortality. This risk increases for people with cardiovascular diseases and is frequently caused by the adverse effect of obesity on lipoproteins. Patients with sedentary lifestyles and poor eating habits often have both pathologies.
MCD3	HTA and osteoporosis are two pathologies with many common characteristics. Both increase with age and affect women overall. Hypertensive menopausal women suffer a greater prevalence of osteoporosis than normotensive women.
MCD4	Comorbidity for patients with COPD is not properly defined. Nevertheless, there are studies that reveal high rates of prevalence in people with COPD and HF (up to 59%).
MCD5	Insulin resistance is a physiological condition linked to other diseases such as high blood pressure and obesity or to a sedentary lifestyle. In fact, one pathology affects the other and it is sometimes difficult to establish which was the original disease.

Table 4.7: Multi-chronic Patient Profiles (Complex)

4.3.1.3 From the Patient Profile questionnaires to MAPE tasks ontology configuration

Figure 4.7 illustrates the relation between the information contained in the Patient Profile questionnaires together with the IF/THEN rules for alarm configurations as indicated by the physician and the tasks described in the ontology. As can be seen, monitoring patient data (including vital constant measurements, quantitative, qualitative and environmental information) is done by defining instances of the *Monitoring task* class. Reminders for patients are related to the *reminders* class. Hence, according to the ontology, reminders will alert patients about the monitoring data they should measure. Clinical Alarms are directly related to *Analysis task* which will be associated to one monitoring task, and consequently patient's data. Furthermore, some rules should be configured in order to evaluate the analysis result within a range and in this way determine the threshold that will trigger the alarm. Actions to be performed after clinical alarm detection are related to *Planning* and *Execution task.* By using rules, they will be linked to the activation of the analysis task.

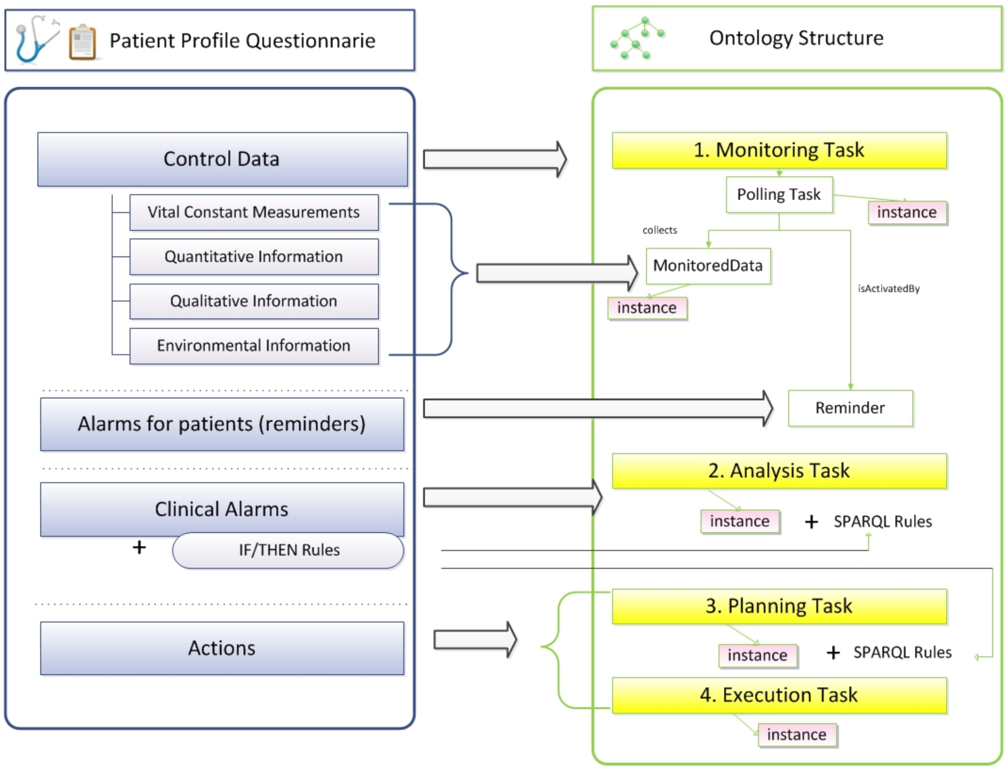

Figure 4.7: Relation between Patient Profile questionnaire and Ontology tasks

4.3.2 Results: Chronic and multi-chronic Patient Profiles

In order to monitor each patient at home sites, the physician should define an instance of the *patient profile* (which comprises a set of monitoring, analysis, planning and execution tasks) and a set of rules to activate the analysis and planning tasks and consequently provide personalized medical care.

4.3.2.1 Patient Profile Instance Example: COPD patient profile

In order to illustrate the design of a *patient profile* (ontology instances and rules), we examine in detail the COPD *patient profile* configuration. COPD patients were identified as candidates to be monitored at home sites. Some estimations suggest that up to 600 million people worldwide may currently suffer from COPD and 75% of them in Europe are undiagnosed and consequently untreated. COPD includes two main illnesses: chronic bronchitis and emphysema, a pair of commonly co-existing diseases of the lungs in which the airways become narrowed. This leads to a limitation of the flow of air to and from the lungs, causing shortness of breath. There is no cure for COPD and it is almost always caused by cigarette smoking [172].

The requirements for the supervision of this type of patient configured by the physician are depicted in Table 4.8. As explained in section 4.1.3, the data to be monitored directly corresponds to the required instances of the monitoring tasks. All of them taken together are described as the *Polling Task*, and *Reminders* have been configured to ask the patient for the relevant clinical data. The schedule of each reminder in terms of days (d) and hours (h) has been included in Table 4.8. Then, defined alarms associated to each configured data have been included in the table. The activation of any of the alarms (except the exacerbation alarm) would alert both the patient and the remote physician and warn the patient to repeat the measurement after 10 minutes in a relaxed manner.

This analysis of requirements and then the mapping process into ontology instances were performed by the physician for the 16 *patients profiles* defined in Table 4.5 and Table 4.7. This information has been gathered in the Annex A.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $>$ 25 $\rightarrow$ A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60$<$ pulse rate $<$ 100 $\rightarrow$ A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value $<$70 AND SP value $<$ 100 mmHg Alarm 2: DP value $>$90 and SP value $>$1 40 mmHg TAD alarm: DP $>$ 120 mmHg Each alarm detection $\rightarrow$ A1, A2 and A3
SpO2	1d, 10h	SpO2$<$90% $\rightarrow$ A1, A2 and A3
FEV1	1d, 10h	Testing with spirometry determines the presence and severity of the airway obstruction. Configured alarms: Alarm 1: FEV1 $<$ 80%: COPD SLIGHT
Glucose	30d, 8h	A glucose value out of this range will trigger an alarm: Glucose $<$80 mg/dl or glucose $>$ 126 mg/dl
Medication	1d, 10h	- No alarms defined
Number of Cigarettes	30d, 10h	- No alarms defined
Patient Location	15d, 10h	- No alarms defined
Room Temperature	15d, 10h	- No alarms defined
Room Humidity	15d, 10h	- No alarms defined
Health Test Questionnaire (COPD specific)	15d, 10h	Exacerbation alarm: This alarm is activated if the patient answers YES to any of the questions included in the COPD test.
The COPD test includes the following questions: 1) Do you have a persistent or increasing cough? 2) Have you experienced increased mucus in the last few days? 3) Have you suffered shortness of breath or increased wheezing in the last few days? 4) Do you feel shivering or sleepiness?		

Table 4.8: Monitoring requirements for a generic COPD patient

All this information was mapped into monitoring, analysis, planning and execution task instances of the HOTMES ontology. According to the mapping schema presented in Figure 4.7, a total of 15 monitoring tasks (one for each monitored data although it should be noted that 3 monitoring tasks should be configured to control blood pressure), 11 analysis tasks (one for each data to be evaluated within a range), 9 planning tasks (depending on the analysis tasks that should be evaluated together to perform an execution task) and 3 execution tasks were required to configure the *patient profile* for a COPD patient. Furthermore, 18 rules also had to be configured. Regarding the number of execution tasks, only 3 actions were included in the evaluation and in all the configured *patient profiles*: 1) A1: warning physician, 2) A2: warning patient and 3) A3: indicate patient to repeat a measurement. An example of the instances included in the COPD *patient profile* instance regarding alarm 1 for blood pressure measurement can be seen in Figure 4.9. It can be seen that both systolic and diastolic pressure measurements were included for monitoring by using a *PollingTask*. Both of them were activated by the same reminder (Reminder_COPD_1). Then, two mathematical analyses were associated to perform an atomic process (one step process) which analyses the absolute value of the diastolic and systolic pressure datum, respectively. The evaluation criterion was then executed by means of two SPARQL rules (see Figure 4.8) which turn the result of the analysis task into a positive value). As can be seen, the first analysis will be positive if the systolic value is under 100 mmHg and the second one if the diastolic value is under 70 mmHg. Finally, the planning task which comprises the execution of the three actions (A1, A2 and A3) described this section will be activated, reporting that a blood pressure alarm has been triggered if both analysis tasks are positive. This statement is expressed as two preconditions of the activation of the planning task (see Figure 4.8).

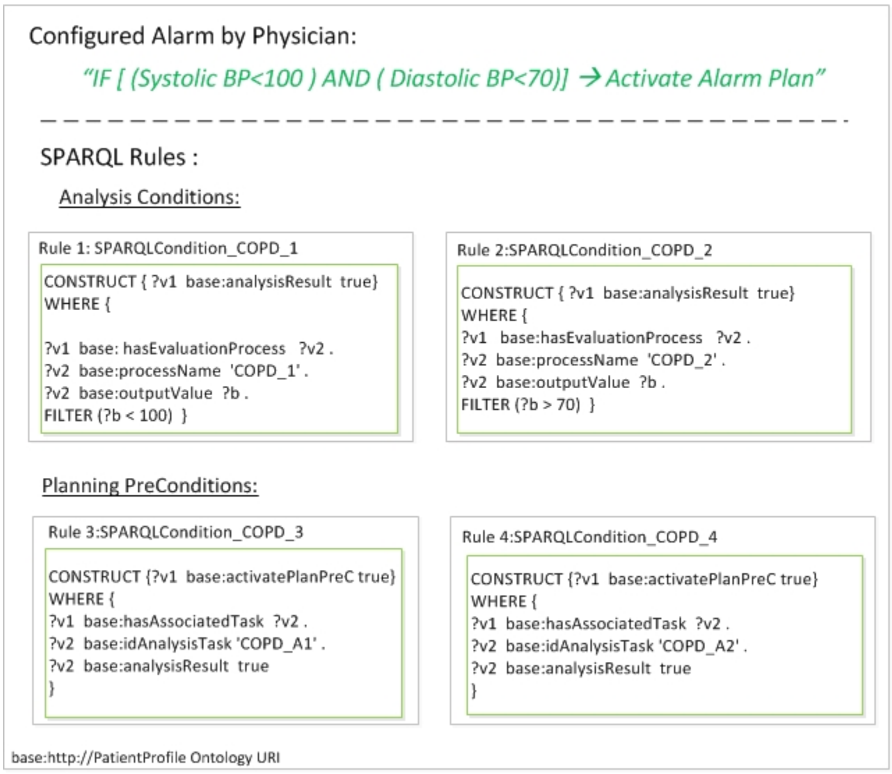

Figure 4.8: SPARQL Condition related to a blood pressure alarm

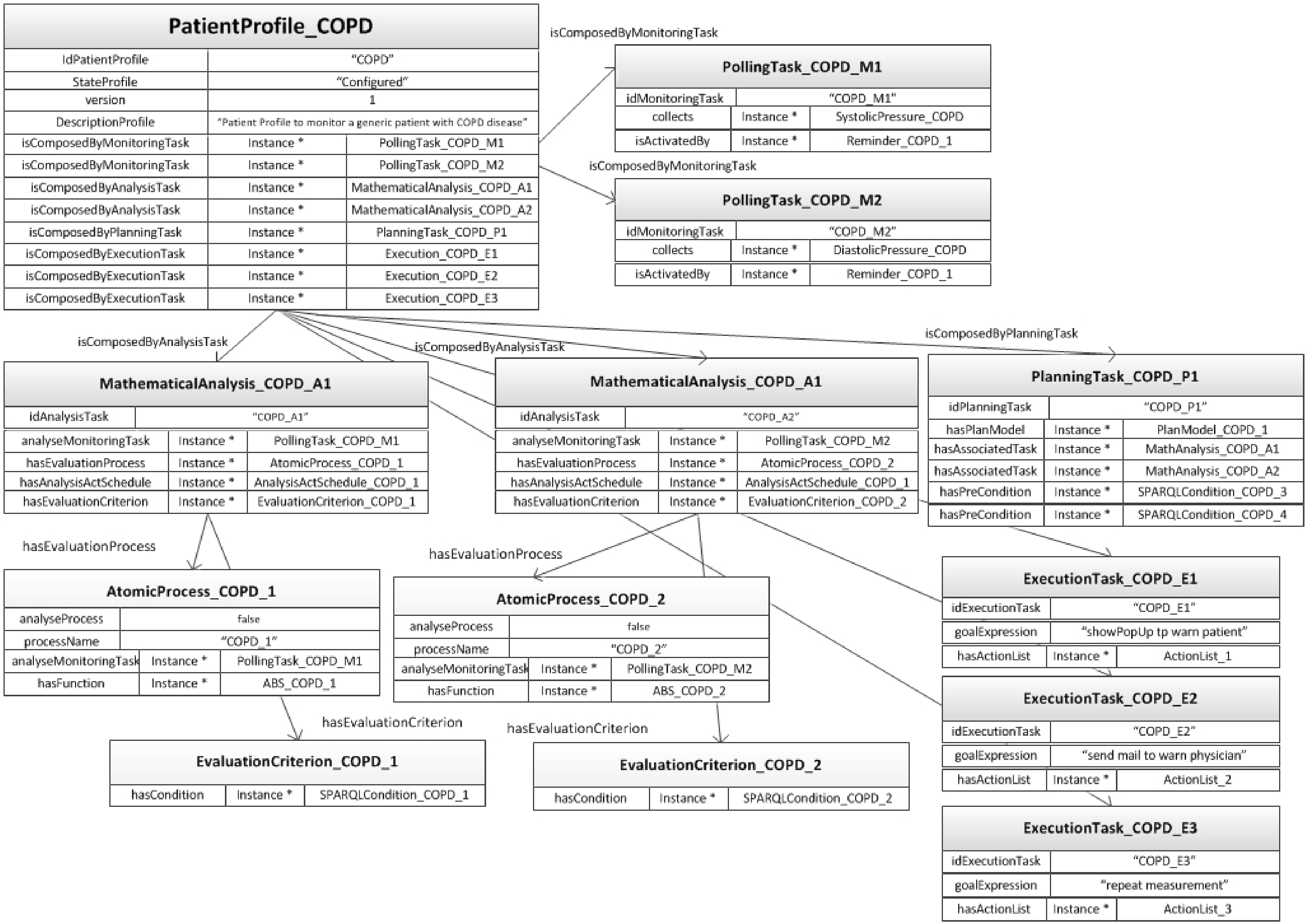

Figure 4.9: Patient Profile Instance Example

4.3.2.2 Tasks and rules for chronic and multi-chronic Patient Profiles

Both chronic and multi-chronic *patient profiles* were successfully configured by using the proposed ontology. After the Chronic Patient Profile specification sub-stage, some adjustments were introduced into the ontology in order to be able to define all the *patient profiles* proposed in Tables 4.5 and Table 4.7. It was then verified that multi-chronic profiles could also be successfully configured. According to the profiles defined by the physician, the corresponding monitoring, analysis, planning and execution tasks were configured in order to define the required tasks (information to be controlled and alarms) for each type of defined profile. Table 4.9 depicts for each profile (first column) the relation between the information required to be monitored as defined in the Patient Profile questionnaires (second column) and the number of tasks required from the ontology to configure each *patient profile* instance (third column).

It should be noted that the mapping process between the monitoring requirements specified in each profile and the tasks contained in the ontology was done manually by engineers using the Protégé editor. Additional research would be required to develop an interface that would automatically map physicians' information into the MAPE tasks, thus making it easier for physicians to define the profiles through the ontology model.

Profile	Questionnaires: Required Information					Ontology Instances Tasks			
	VC	QNI	QLI	EI	Alarms	MT	AT	PT	ET
COPD	7	2	1	3	9	15	11	9	3
Obesity	8	3	1	0	10	14	13	8	3
Thyroids	7	3	3	0	9	15	13	10	3
Ischemic	6	2	1	0	8	11	10	8	3
Asthma	8	2	1	3	9	16	11	9	3
HTA	6	2	1	0	8	11	10	7	3
Ost	2	1	2	0	2	5	2	2	2
HF	5	2	2	0	8	11	10	8	3
Diabetes	6	4	1	0	7	13	9	7	3
Disp	4	2	1	0	6	9	8	6	3
OA	4	1	1	0	5	8	7	5	3
MCD1	6	4	1	0	7	13	9	7	3
MCD2	6	4	2	0	11	16	14	9	3
MCD3	6	2	3	0	9	13	11	8	3
MCD4	7	3	3	3	11	17	13	11	3
MCD5	9	4	4	0	11	19	13	11	3

*VC: vital constant; QNI: quantitative information; QLI: qualitative information; EI: environmental information; MT: monitoring task; AT: analysis task; PT: planning task; ET: execution task

Table 4.9: Relation between Patient Profile questionnaires and Ontology tasks

As can be seen in Table 4.9, there are significant differences regarding the number of monitoring, analysis and planning tasks required to configure each profile. However, the number of execution tasks required to supervise each type of patient seems to be constant and independent of the patient's disease (except for the Osteoporosis case where only two actions were configured).

Regarding the simple *patient profiles*, the mean number of monitoring tasks configured in each profile was 12 tasks, the osteoporosis profile requiring the least tasks (5) and the asthma profile requiring the most (16). For example, for the asthma profile, 8 vital constants were selected for monitoring but environmental information, number of cigarettes and medication also had to be controlled. Furthermore, an asthma exacerbation detection questionnaire test was included (Health Test Questionnaire). This questionnaire included questions such as: "Have you got signs of a cold or allergies (sneezing, runny nose, cough, nasal congestion, sore throat, and headache)?" or "Do you feel chest tightness, pain, or pressure?". A positive answer was configured to be detected through an analysis task and consequently used to trigger an alarm.

Regarding MCD profiles, it can be seen that the same number of tasks were required to monitor a patient with diabetes as a patient with diabetes and ischemic. The main difference between these profiles lies in the Health Test Questionnaire configuration. While normally ischemic patients feel chest pains when a heart attack occurs, diabetes patients often experience stomachache or epigastrium for the same heart attack event. Consequently, this question should be modified in the profile. MCD 2, 3 and 4 profiles included the mixture of requirements of both related diseases. As can be seen in Table 4.9, the main difference between chronic and multi-chronic profiles lies in the amount of quantitative and qualitative information rather than in the constant vital measurements performed. Nevertheless, after completing Step 1 (instances definition) some alarm values can be modified and be made more restrictive during Step 2 (rules configuration) if one pathology affects another (e.g the blood pressure for obesity patients).

PP type	Instances	Rules
COPD	144	18
Obesity	142	23
Thyroid	171	22
Ischemic	120	17
Asthma	136	19
HTA	121	17
Osteoporosis	58	4
Heart Failure	129	17
Diabetes	124	15
Dyslipidemia	100	14
Osteoarthritis	88	12
MCD 1	124	15
MCD 2	159	24
MCD 3	154	18
MCD 4	181	21
MCD 5	186	19

Table 4.10: Instances and rules for each patient profile

Table 4.10 depicts the number of required instances and rules to configure each profile. The number of rules is directly related to the number and complexity of configured alarms in each profile (also with the number of analysis tasks defined in each one). Thyroid disorders, obesity and asthma profiles require a greater quantity of rules to completely configure a profile for the supervision of these diseases. It should be noted that using the same type of *patient profile* (e.g COPD), it is possible to particularize and personalize supervision of two different patients by simply modifying the rules. Details about each *patient profile* specification can be found in Annex A.

Although most of the alarms configured by physicians were related to the evaluation of the absolute value of a vital constant measurement within a range, such as in the blood pressure case, alarms conditioned to the evaluation of several measurements or after a mathematical manipulation process (such as the BMI alarm) were also described within the Patient Profile questionnaire and then successfully addressed with the HOTMES clinical ontology structure. For example, for a generic patient with thyroid problems a thyrotoxic alarm was configured and triggered when the patient's temperature was over 38 C and a positive answer was found after completing the thyrotoxic crisis test. Figure 4.10 depicts the SPARQL condition to activate each mathematical analysis.

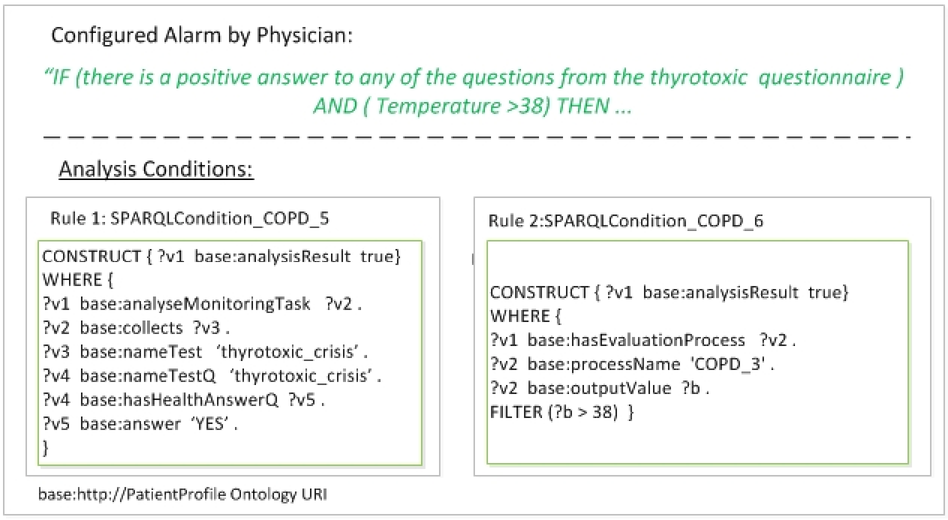

Figure 4.10: SPARQL Condition related to the thyrotoxic alarm

It should be noted that as vital constant measurement data modeled in the ontology also include contextual information, alarms can be defined conditioned to this context information. For example, different BP alarms can be configured (execution tasks and threshold values) conditioned to the answer to the question about stimulating substances taken before performing a measurement (see Figure 4.11).

Furthermore, complex management tasks could have been defined regarding alarm activation. For example, for an obesity patient with suspected hypertension, we could define an alarm associated to detecting during 3 days 141/91 mmHg in pressure measurements. As a consequence of its activation, the patient would be assigned the HTA profile in addition to the obesity profile for his supervision. In this case, apart from the mathematical analysis configuration as depicted in Figure 4.9, a post condition regarding the planning task activation should be defined (see Figure 4.10).

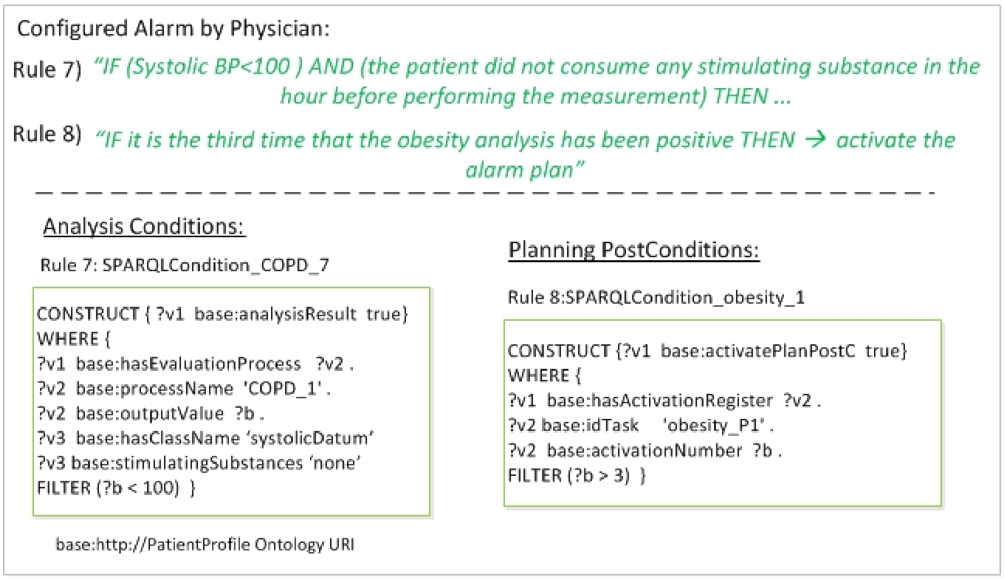

Figure 4.11: SPARQL Condition related to complex alarm definition

4.4 Stage 3 (clinical): Prototype Development

This stage dealt with the data and communication layer of the system architecture presented in chapter 3. A tool was required to manipulate and map incoming data into the new format (data integration) and to run the management tasks specified in the *patient profile* (management integration). As stated in chapter 3, these tasks were carried out by the Semantic Autonomic Agent placed at the home site in the HG. This agent includes the following modules: 1) Knowledge Base Module (where the HOTMES clinical ontology is stored), 2) Converter Module, 3) MAPE module, 4) Integrator Module, 5) Rules module, 6) Reminders and alarms module and 7) Actions module (see Figure 3.9). In order to adapt the architecture to the clinical data management, the main enhancements and improvements were made in this last integrator module. This was used to integrate data acquired by MDs with contextual data provided by patients according to the ontology model. Specifically, this module was adapted to the X73 MDs medical data format.

4.4.1 Results: Implementation

In order to develop the integrator module and test the Semantic Autonomic Agent, data transferences by the MDs using the X73 standard were simulated. In fact, data transferences from an X73 manager to the semantic agent were simulated by using XML files over sockets technology. A proxy module was developed and installed in the HG adapted to the X73, named the proxyX73-agent. The XML-based communication protocol proposed to enable an interoperable and organized information exchange mechanism between the proxy X73-agent (or X73 mapping module in this implementation) and the X73 manager had already been used in an SNMP-X73 solution which will be explained in chapter 5 [183]. The exchange protocol has been defined to minimize and optimize the information exchange by taking advantage of the inherent X73 data events and structure. The structure of the XML files follows a similar organization in terms of attributes and nomenclature to that proposed in [50] for the transfer of information from an X73 manager to an EHR (electronic health record). In addition, a GUI was developed in order to include contextual information from patients and also a means to download the *patient profile* and start and stop the monitoring process. As stated in chapter 3, all these modules were built using Java technologies.

An image of the Semantic Autonomic Agent interface can be seen in Figure 4.12. This Figure depicts two views of the agent: the principal view and a secondary view showing information contained in a *patient profile*. Patients access the principal view by introducing a pair of keys (login and password). The buttons used for this clinical application have been highlighted in the image. The "Download Clinical MP" button is used to download the *patient profile*. The second button, "Patient Profile Information" is used to access information contained in each *patient profile* available in the HG. The "Active Reminders button" is used to view active tasks that have not yet been performed by the patient. Finally, the "View Clinical Alarms Button" is used to view alarms that have been triggered after the evaluation of measurements.

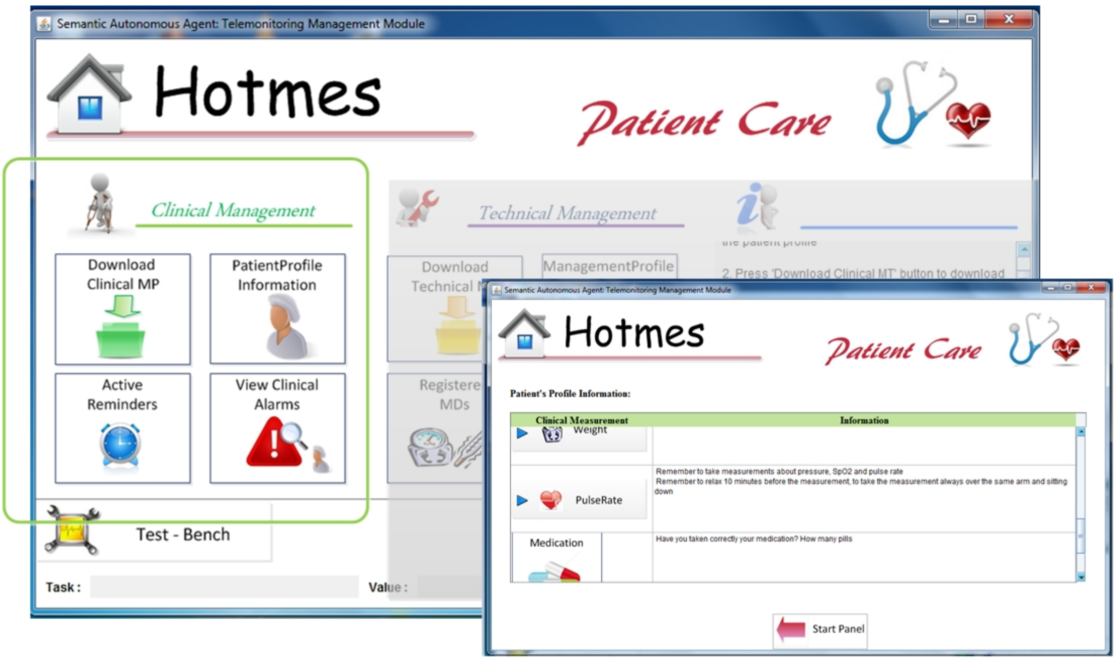

Figure 4.12: Semantic Autonomic Agent interface (principal view)

The reminders view interface is depicted in in Figure 4.13. As explained previously, once the profile is downloaded by the patient the reminders are automatically configured and the patient can visualize and provide solicited information through this reminders module. It should be noted that the displayed reminders correspond to the COPD *patient profile*. The upper part of the interface contains buttons to enable the patient to introduce solicited information. For example, pressing the "Questionnaire" button triggers pop-ups with questions to be shown to the patient in order to complete (in this case) the Health Test Questionnaire for the COPD disease. In the same way, the number of cigarettes smoked or data about the location can be provided by the patient. On the other hand, the lower part of the interface shows in the form of information only the vital constant measurements that the patient should perform. As these will be acquired by using a MD, information tags without buttons are displayed in this interface. Once the patient provides the required information, the buttons and tags disappear from this panel.

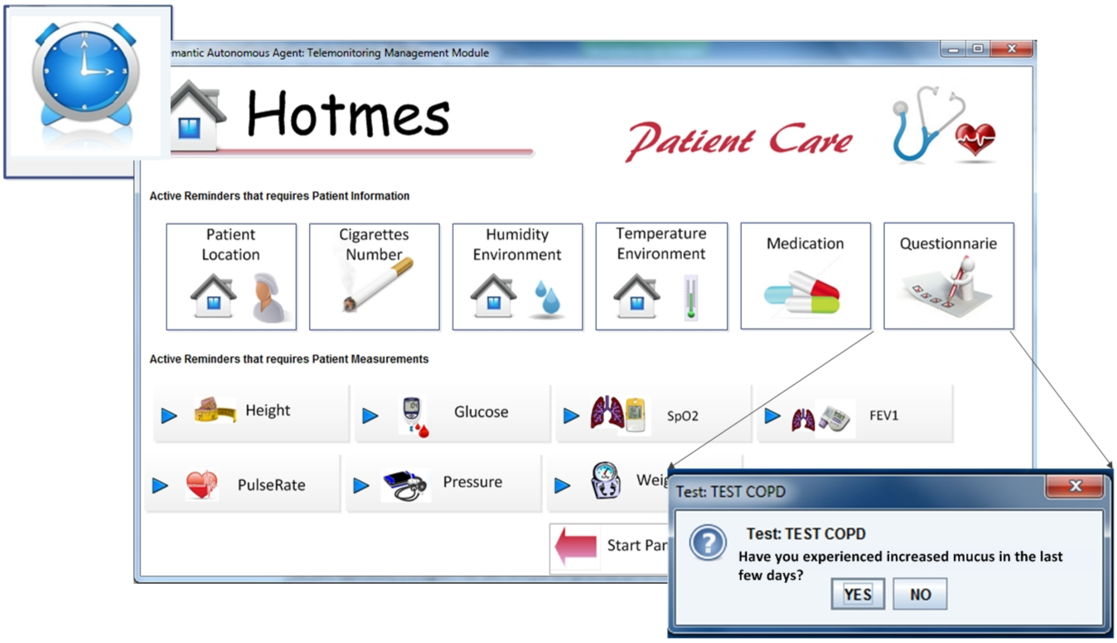

Figure 4.13: Semantic Autonomic Agent interface (reminders view)

4.4.2 Workflow Procedure

Once the *patient profile* is downloaded by the patient, the configured reminders are initiated. Every time that data is transferred or introduced by the patient (data that is not acquired by an MD), the prototype working mode follows the workflow depicted in Figure 4.14. The relation to some object properties of the ontology model has also been indicated in Figure 4.14. It can be seen that when a new datum is introduced in the system, the prototype checks first if there is any monitoring task associated to collecting that type of data. In the case of a vital constant measurement, then some pop-ups will be activated in order to ask the patient about the contextual information required to create a new datum instance according to the ontology model (see Figure 4.13 where pop-ups associated to a glucose measurement have been included). It will then check if there is any analysis task in the profile associated to a previous monitoring task. If there is, then the mathematical or syntactical process will be performed and after that the rules will be applied over the model (in the example, the analysis task is associated to detecting a glucose measurement over 126 mg/dl). As a consequence, planning tasks can be activated if pre-conditions and post-conditions are fulfilled. Finally, execution tasks will be performed (in the example, to pop-up an alarm to the patient and send an email to notify the corresponding physician about this finding).

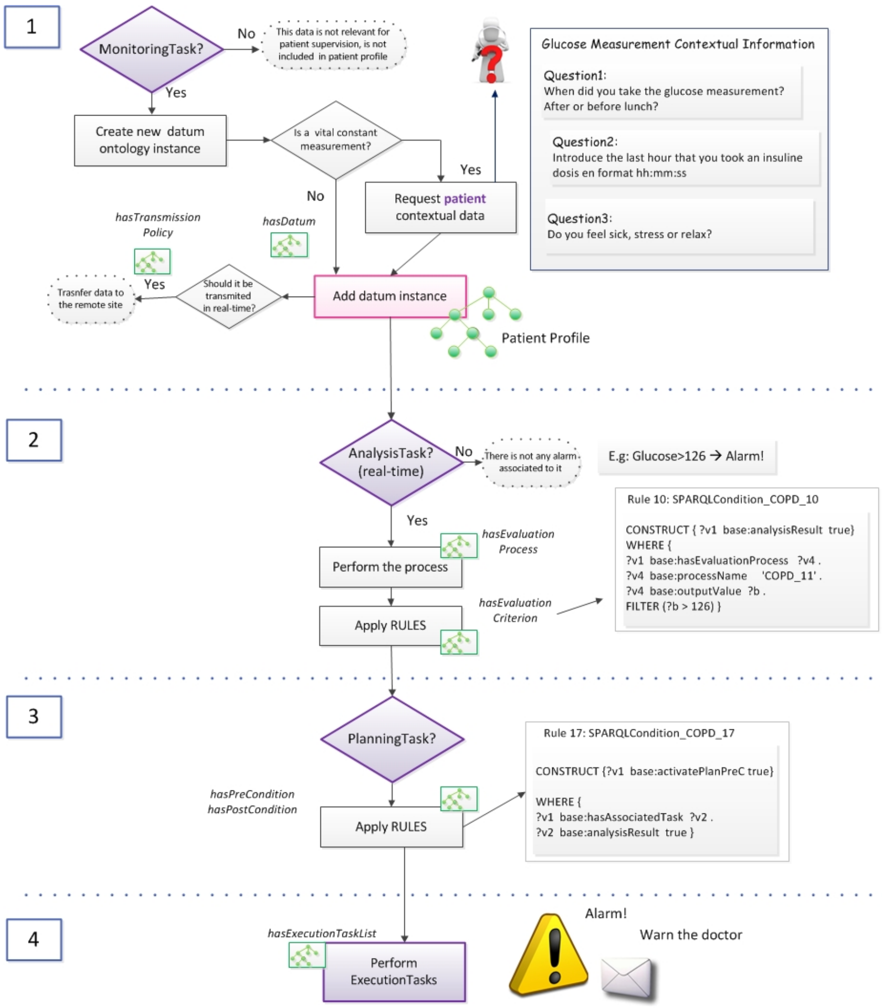

Figure 4.14: Workflow of the agent prototype execution

4.4.3 Evaluation Performance

There is no standard criterion to evaluate the effectiveness of an ontology-based application. Most of the common scenarios used for the evaluation of software implementations assess three principal cost metrics: memory cost, computational cost and network cost [127, 184]. The performance of the architecture (and in particular the Semantic Autonomic Agent) has been evaluated in terms of these metrics. In addition, the storage cost required and the execution time have been measured. The goal of this evaluation was to test the performance of this particular implementation of the semantic agent in order to measure the required resources and to assess whether its implementation could be possible in other types of constraints devices such as a mobile. Details about these metrics and how they were measured are given below:

- *Memory Cost*: Memory cost refers to the amount of memory allocated to the agent process. In the case of data memory, it can be static (memory is allocated as the program starts and remains constant during the lifetime of the program) or dynamic (memory is allocated after a request is received and released after the response is transmitted). The memory allocated to the agent process was measured using the *Windows Task Manager*. The initial required memory allocated to the agent was measured (static memory). Then, for all the tests performed, memory cost was measured in terms of increments related to this static memory.

- *Computational Cost*: This refers to the processing activity at the entity. This parameter was also measured using the *Windows Task Manager*. Specifically, an average of the maximum CPU observed for each test was considered.

- *Network Cost*: This metric refers to the total amount of information that needs to be transmitted through the network between two entities to perform an action. Traffic measurements were done using *RawCap* and *Wireshark* software. Network usage cost has been measured by gathering the total amount of exchanged KB (IP data) between the web points in order to download the ontology, the *patient profile* instance (PP) and the rules (R). The first action led to the same quantity of bytes for all the evaluated profiles as the ontology is always the same. This download process involves the exchange of 77,38 KB between the two end points.

- *Storage Cost*: This refers to the size of the database observed during the management procedure to hold the ontology and the instances. It was directly measured by examining the size of the tables included in the database.

- *Execution Time*: The execution time refers to the time required by the agent to perform a specific task. This metric was measured by running a simple Java code (using the sentence *System.currentTimeMillis()*).

In order to test the functionality of the architecture for telemonitoring patients, some tests have been performed using a simulated scenario. The test scenario included the developed Semantic Autonomic Agent with the web client module and the proxy X73 (that is to say, modules located in the HG) located on a 1.6 GHz i7 Intel Core running Windows 7. Data transferences for 7 different MDs were simulated by configuring the appropriate XML files. The web service plus its management module (that is to say, modules located in the MS) were located on the same machine. All the *patient profiles* developed to evaluate the applicability of the ontology for monitoring different patient types were used to test the architecture performance. This evaluation included two separate tests. Test 1 consisted of performing the first tasks of the monitoring process, i.e, downloading the ontology, the *patient profile* instance and the corresponding rules. Test 2 consisted of running the agent in order to perform the tasks described in the profile, thus monitoring the patient.

In order to test the limits of the prototype and its efficiency in holding more than one *patient profile* (which could be of interest if more than one patient is at the same home or in the case of a nursing home), a third test was performed. This test consisted of evaluating the performance of the agent in running the tasks included in 11 *patient profiles.*

Test 1: Download Ontology, Profile and Rules

Initially, the Semantic Autonomic Agent application downloads the OWL file into the TDB database whose initial size (without the *patient profile* instance) is 685 KB (a set of tables are created in the database in order to hold the ontology model). As explained in chapter 3, this involves executing the GET (ontology) command request in the web service. This ontology OWL file is 270 KB in size. Then, according to the *patient profile* identifier provided by the patient, the agent asks for the corresponding *patient profile* and the rules.

Table 4.11 shows the average results of the cost metrics evaluated for the agent performance in requesting the ontology, the *patient profile* and the rules. "Δ Mem" indicates the maximum increment of memory (dynamic) allocated for the agent during this test. The initial memory (static memory) allocated for the agent is 83.476 KB. Note that this initial required memory greatly depends on the interface design, thus it could be adapted according to the conditions of each particular scenario if there is any limitation in terms of memory resources. "CPU max" indicates the average of the maximum CPU percentage consumption observed. This CPU value took values of around 13% at some specific moments of the process when inserting the *patient profile* instance into the ontology model. "Network cost" measurement is depicted for the *patient profile* (PP) and the rules (R). "TDB ini" indicates the initial database size with the ontology and the *patient profile* instance. Finally, the mean execution time required by the Converter module to unwrap and dump the *patient profile* into the TDB is depicted in the column titled "E.Time (ms) $\pm$ σ". This measurement was repeated several times and the standard deviation measurement found for all the measurements is also shown in the Table. This execution time depends on the number of instances and the property information that comprise each profile. As can be seen in Table 4.11, the required execution time in the Converter module is directly proportional to the number of instances in each profile given that this module is responsible for unwrapping these instances and inserting them into the ontology.

Compared to the first SOAP-based versions of the architecture, one of the main improvements of the REST inspired-architecture was network cost. A reduction of 24% (average) in the KB exchanged to transfer a *Patient Profile*, the rules and the ontology was measured.

Profile	Δ Mem (KB)	CPU max	Network cost (R)(KB)	Network cost (P)(KB)	TDB. ini (MB)	E.Time ± σ (ms)
COPD	23.092	2	9,09	262,26	1,30	2.129 ± 138
Obesity	26.572	2,06	10,45	324,86	1,54	2.069 ± 14,38
Thyroids	33.640	2,58	10,33	288,15	1,66	2.189 ± 50,18
Ischemic	33.188	2	8,8	223,14	1,26	1.793 ± 56,32
Asthma	29.008	3	9,34	211,25	1,28	1.963 ± 24,78
HTA	29.284	3,09	8,8	190,43	1,30	1.822 ± 71,97
Ost	19.166	2,78	5,56	136,87	1,23	994 ± 21,40
HF	21.464	2,44	8,97	210,66	1,28	1.960 ± 101,41
Diabetes	28.944	2,7	8,41	189,29	1,30	1.773 ± 41,57
DSP	20.420	2,46	7,96	198,61	1,24	1.523 ± 19,61
OA	19.776	3,08	7,35	173,44	1,25	1.374 ± 38,19
Average	**25.869**	**3**	**8,64**	**219**	**1,33**	**1.781 ± 342**

Table 4.11: Cost metrics evaluation test 1

Test 2: Running the tasks to monitor a patient

For this evaluation test, the semantic agent was running during 1 hour including periods with high and low activity. Polling tasks were configured to retrieve data every two minutes (hence, reminders for patients to take measurements or provide some information were activated every two minutes). Events were manually simulated in periods with irregular activity (5 minute active periods every 1 minute and every 5 minutes). 50% of the simulated transferences led to the activation of an alarm. The cost metrics results of this evaluation test are shown in Table 4.12. "Δ Max Mem" indicates the maximum increase in memory (referring to the initial static memory) allocated to the agent during 1 hour's evaluation. "CPU max" indicates the mean maximum CPU percentage consumption observed during the tests. Network usage cost was measured in order to know the KB exchanged with the web service when the client sends data results or alarm messages to the remote site. The mean KB exchanged was 9,14 ± 1,5 KB. This value was measured considering all the data results that can be sent to the web service with no dependence on the *patient profile* that is being run in the agent. Finally, "TDB 1h." indicates the database size after one hour of evaluation.

The mean execution time required by the MAPE module to run each configured task time was not measured in this test because this time was strongly affected by the number of pop-up messages that the patient should attend to in order to provide contextual information. The execution time required to evaluate tasks that do not require interaction with the patient was considerably lower. For instance, the mean execution time required to evaluate an SpO2 measurement was 4198 ms (no pop-up messages) while to complete the evaluation of a BP measurement took 12.160 ms (3 pop up messages asking for information related to the BP measurement).

Note that after including the *patient profile* and storing one datum result in the TDB database, it stops growing. This is because after it is analyzed only the last value result is stored in the knowledge base module in the patient's home.

Profile	Δ Max Mem (KB)	CPU max	TDB 1h. (MB)
COPD	53.112	0,72	1,65
Obesity	48.658	0,66	1,7
Thyroids	59.396	0,7	2,15
Ischemic	55.876	0,52	2,15
Asthma	52.692	0,8	1,65
HTA	49.196	0,75	1,69
Ost	40.454	0,85	1,53
HF	43.012	0,42	2,13
Diabetes	51.248	0,8	2
Disp	36.036	0,67	1,59
OA	41.922	0,82	1,57
Average	**48.327**	**0,7**	**1,8**

Table 4.12: Cost metrics evaluation test 2

Test 3: Multiple Patients Profiles

In order to test the performance of the semantic agent in running several *patient profiles*, a total of 11 profiles were downloaded. Specifically, the generic *patient profiles* for COPD, obesity, thyroids, IHD, asthma, HTA, Osteoporosis, HF, DM, dyslipidemia and osteoarthritis were downloaded (following this sequence). After including all these *patient profiles* (and thus a total of 1333 new instances), the TDB database increased to 20,8 MB.

The agent loads the whole database in the memory when it is initialized. Consequently, the memory cost was slightly increased as well. Specifically, an increment of 20.816 KB was measured compared to the maximum memory increase observed when running one profile (the thyroid profile case). When more than one *patient profile* is being run in the agent, before processing a measurement or showing active reminders the patient should provide his or her corresponding *patient profile* identifier in order to evaluate the incoming data according to his/her registered guidelines.

The results of the third test show that the database size was considerably increased when several profiles were added to it. In fact, if the semantic agent is located in one home where no more than 4 or 5 patients are to be monitored with the HOTMES tool, the TDB database is appropriate. However, other databases should be studied if the tool is used to monitor hundreds of patients, e.g if it is located in a nursing home. The same issue should be considered for the database used at the remote site.

4.5 Discussion

Ontologies provide a high level of abstraction and expressiveness to define concepts and knowledge with precision. This facilitates data integration in scenarios such as telemonitoring where diverse heterogeneous sources are involved in the same process. Following the three simple stages presented in chapter 3 (see Figure 3.2), this chapter deals with adapting the proposed HOTMES-based architecture to clinical purposes, thus addressing data and management integration for context-aware and personalized monitoring services for patients with chronic conditions in the mentioned scenario. The three stages cover both conceptual (conceptual layer) and practical issues (data and communication layer).

As emphasized in [87], it is equally important to illustrate how to capture and organize knowledge in the biomedical domain as well as to identify and evaluate principles of quality in the ontology. Hence, different levels of the ontology were evaluated during the first and second phases. Stage one dealt with the ontology design and implementation while stage two dealt with the application-based study of the ontology. Specifically, this latter stage allowed us to evaluate how good the proposed ontology is for the application it was designed for, which is monitoring patients with any type of chronic condition. Similar to the methodology proposed in [87], users' knowledge (physicians' knowledge in our case) was captured for the design of the ontology. It was then implemented and evaluated in the health domain, and its ability to solve the problem it was designed for was studied. Finally, stage three dealt with the practical application of the telemonitoring system used to run the tasks specified in the *patient profile* ontology. Supervising patients with profiles has many advantages not only for monitoring health status but also for classifying patients into different groups under the same health care area and controlling several types of disease.

Ontologies have already been successfully applied in the telemonitoring domain, and as discussed in chapter 2 other solutions have been reported in the last decade regarding remote control of patients with chronic conditions and decision support tools [85, 87, 86, 88, 89, 90, 98, 99]. Our solution described in this chapter makes some additional key contributions to ontology-based system developments for monitoring patients with chronic conditions. First, not only was clinical knowledge taken into account for the ontology development but also the interoperability and integration of MDs was considered. Patients' contextual data was also included in the profile. In this way, the final ontology design aims to integrate the different heterogeneous sources involved in the telemonitoring process. Furthermore, the ontology model design describes a generic *patient profile* not limited to monitoring patients with any particular chronic condition. It has the advantage of creating, as instances of the *patient profile* class, monitoring profiles to monitor patients with different conditions and co-morbidities, as demonstrated in the application-based study phase. Hence, the design structure and the use of SPARQL rules provide a good and flexible solution for personalized monitoring services. In addition, our ontology-based approach was designed to represent not only data modelling but also modelling of the management procedure execution, that is to say the workflow that dictates the management service and the automation of the patient care tasks. The application of the MAPE paradigm to the remote management of patients provides a simple, clear and easily understandable way of controlling and evaluating patients' clinical data, and its combination with ontologies constitutes an innovative application in the healthcare domain.

Patient profiles described in this chapter provide a simple guidance to evaluate patient's conditions that by themselves or in combination with others are able to conclude whether or not something is going wrong with the patient's health. An interesting point of discussion and future research would be how this model can be used to evaluate a deterioration in health that could lead to triggering an alarm. This means extracting important conclusions not through a measurement value itself but from its evolution. This may require storing data results and analysing them in terms of averages (which could easily be done with the HOTMES ontology after a few adjustments) or may require including additional more complex processing modules in the Semantic Autonomic Agent. These approaches would lead to an increase in both complexity and resource consumption in the HG. However, the advantages and disadvantages should be evaluated for each particular disease case before introducing additional complexity.

As explained in the results section of the application-based study stage, our ontology approach could be used to monitor patients with any type of chronic condition. By performing Steps 1 and 2 as described, new *patient profiles* can be generated. Instances of the monitoring, analysis, planning and execution tasks have been created for each individual *patient profile* according to the number of vital constant measurements, qualitative and quantitative information and alarm configurations required. A mean of 122 instances of the ontology and 12 rules have to be created to define a complete *patient profile*. Furthermore, simple and complex or conditioned analysis (e.g vital constant data conditioned to a subjective patient answer) could be configured by a combination of the ontology instances (analysis task) and the definition of the SPARQL rules.

The results of the evaluation performance show that this prototype implementation does not consume many resources. Although ontology processing and also reasoning technologies usually consume significant memory resources (compared to other non semantic tools used for similar purposes), the size of this ontology is not very large and neither are the reasoning tasks required.

As can be seen in Table 4.11, the process of downloading the *patient profile* requires exchanging an average quantity of (219 ± 51,3) KB in contrast to the 9,14 ± 1,5 KB required to transfer a data result. Nevertheless, this is a not significant quantity of bytes. The network usage to download a *patient profile* (average) could be around 0.01 secs using a conventional home Internet connection of 20 Mbps. As explained in the results section, memory and computational costs were not high in either of the two tests performed. These results lead us to conclude that this prototype implementation could be viable in wearable and small devices. Nevertheless, advanced research into its specific implementation should be carried out in order to substantiate this assessment.

Our three-stage approach contributes to addressing the proposed solution from both theoretical and practical points of view. Furthermore, the practical details involved in the three stages facilitate both the understanding and application of this particular solution, and also its transferability to other ontology-based solutions for different domains or applications. In fact, these three stages were followed to adapt the architecture for technical purposes, as will be explained in chapter 5.

Chapter 5

Technical Management

An important task in telemonitoring scenarios is to guarantee that the MDs and HG are working correctly. Their remote technical management is a new challenge in the healthcare domain. This chapter explores how traditional and new proposals for network management can be successfully applied in the telemonitoring scenario by presenting and comparing two management solutions to address this new challenge. The first is based on the most popular emerging technology for knowledge representation, ontologies, combined with the autonomic computing paradigm and WS technologies. Following the 3 stage-approach presented in chapter 3, the HOTMES ontology has been extended and the prototype adapted to this type of data management. The second is based on the most popular TCP/IP management architecture: SNMP. The use of this architecture has been investigated to provide remote technical management by designing and developing an SNMP agent and its corresponding MIB-module. Specifically, integration with the X73 standard has been studied for both proposals and their characteristics have been compared in terms of efficiency, interoperability, security, complexity and semantic expressiveness.

5.1 HG and MDs technical management

Technical management concerns are important in the telemonitoring scenario. Apart from ensuring that MDs are safe for patients, any technical problem in the MD or in the HG (e.g abnormal resources consumption or measurement levels) that could prevent or hinder patients' monitoring must be detected. Two simple questions may be considered: would the patient's clinical supervision be possible if HG resources were overused or if an MD was not working properly? Could an effective clinical service be provided through an infrastructure that is not technically viable?

The important information to be remotely monitored and required for controlling MDs and the HG technical status is generalized and presented in Table 5.1 as a set of attributes classified according to 7 categories of data. The definition of these attributes has been inspired both by the data information model (DIM) of the X73 (regarding MD technical characteristics and data that can be reported by MDs) and the MIB II from SNMP (most information being inspired by the HostResources MIB (HR-MIB)) which were reviewed in order to import general device resources modelling. These general attributes could be used to technically manage MDs and the HG through a general OO (Object Oriented) technique. Some of these technical features can be controlled in both devices (HG and MD), such as the battery level (when the HG works on battery power). In contrast, other resources described refer to technical features that can be monitored in only one of them (HG or MD). For example, the NumberLinkedMD attribute which refers to the number of MDs that are connected to the HG will be a monitored resource associated only to the HG. The general attribute number and its proposed name are depicted in columns one and two Table 5.1. Columns three and four indicate whether they can be used to monitor the HG or the MDs. A text description of the attribute is presented in column five.

As shown in Figure 5.1, both SNMP and X73 standards have been used as sources of information to define relevant technical data to be managed in the telemonitoring scenario. This information was used in the HOTMES solution (A) to inspire the design of the ontology. Then, two mapping modules to both standards were developed as a specific application case of the ontology-based implementation. As shown in Figure 5.1, other mapping modules could be implemented to integrate data coming from other standards into the HOTMES architecture. In a similar way, this information was used in the SNMP solution (B) to design a MIB-module where the information managed by the agent is stored. In contrast to previous solution, this implementation consists of the integration of the X73PHD data into the SNMP architecture by a proxy module.

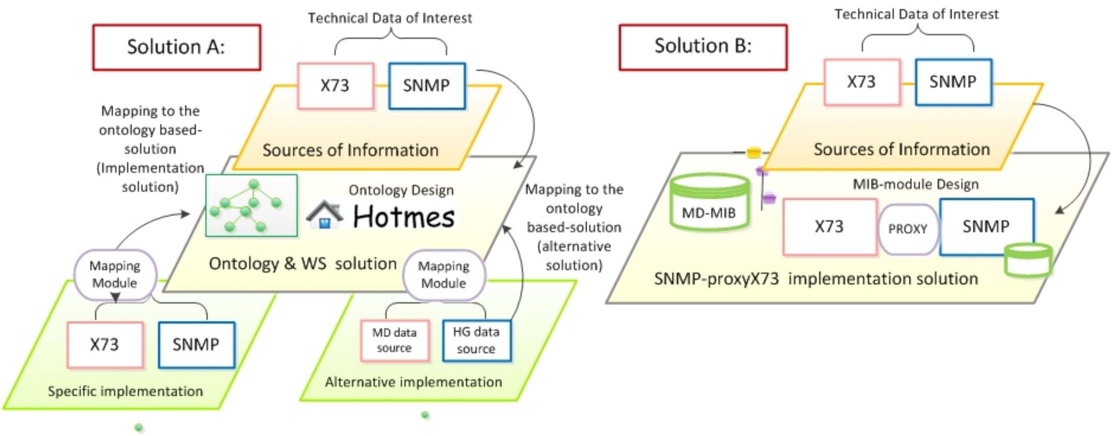

Figure 5.1: HOTMES solution vs SNMP solution

Eight device specializations from X73 were reviewed to extract the data: weighing scale, thermometer, blood-pressure monitor, pulse-oximeter, glucose-meter, body composition analyzer, peak expiratory flow monitor and medication monitor. In addition to available information common to all MD devices, some specifications define additional technical data (for extended configurations) related to the specific operating mode or singularities of a specific MD. For example, the pulse-oximeter specification defines information related to its sensor condition, such as faulty sensor detection or signal irregularities. This information included in the device and sensor annunciation status object (defined by the PulseOxDevStat entity) has been related to attributes 23, 24, 25 and 26 described in Table 5.1.

Apart from the technical data located in the DIM class, agent state information is required in order to monitor when a new MD is linked to the HG, when each MD is active (connected or transferring data) and, most important, to detect when it is removed from the scenario and consequently is not transferring clinical data to the healthcare site. Specifically, attribute 14 was included in Table 5.1 to report this information. The X73 standard defines a synchronized finite state machine both for the agent and the manager. This state machine defines the dynamic behaviour of both entities by describing the states and substates while they are passing through a communication process. Detailed information about transitions can also be found in the standard, but for both solutions (HOTMES and SNMP) only the main states of the agent were considered. These states are: disconnected (physically disconnected but registered in the manager list), connected (physically connected to the X73 manager), associated, (connected, recognized and prepared agent) and operating (indicates that the MD is associated and transferring data).

Regarding SNMP data, system (OID: 1.3.6.1.2.1.1), interfaces (OID: 1.3.6.1.2.1.2) and HR-MIB (OID: 1.3.6.1.2.1.25) groups were considered. The first group defines a list of objects that includes information about the system's hardware, operating system, its administrative name or the services that the entity offers. The list of objects that comprises this group contains the "sys" chain in its attribute name. The interfaces group provides information about each interface on the managed entity. Information about the total number of octets exchanged, current bandwidth or packets discarded is available in this MIB group. The list of objects that comprises this group contains the "if" chain in its attribute name. Finally, the HR-MIB provides information about the resources of the managed entity (e.g: memory or running processes consumption). Specifically, information of interest comes from the *hrStorage* and the *hrSWRunPerf* groups. The list of objects that comprises this group contains the "hr" chain in its attribute name.

N	Attribute Name	HG	MDs	Description
	Device Information			General information of the device characteristics
1	- DeviceType	x	x	It refers to the type of MD (e.g thermometer) or HG (eg. PC)
2	- Manufacturer	x	x	Device manufacturer or vendor
3	- Model	x	x	Device model
4	- Software info	x	x	Information about the software used by the device (type and version)
5	- Hardware info	x	x	Information about the hardware used by the device (type and version)
6	- Firmware info	x	x	Information about the firmware used by the device (type and version)
7	- IPAddress	x		IP Address for remote access to this device
8	- SystemId	x	x	Unique identifier of the device
9	- Configuration		x	Configuration used by the MD (e.g standard or extended in X73)
	PowerResources			Power resources consumed by the device
10	- SupplyType	x	x	Reports the type of supply used by the device (e.g battery or on mains)
11	- BateryLevel (%)	x	x	Reports the percentage of remaining battery
12	- RemainderBatery (time)	x	x	Operational time left in the battery
	CommunicationIssues			Communication issues regarding the MD and HG connection
13	- communicationProtocol		x	Communication protocol used by the MD to transfer data to the HG
14	- CommunicationState	x	x	Refers to the communication state of the devices (e.g operating)
15	- NumberOfMDslinked	x		Number of MDs that are linked to the HG
	MemoryResources			Memory resources consumed by the device (percentage units)
16	- PhysicialMemory	x		Refers to the physical memory consumed by the device
17	- VirtualMemory	x		Refers to the virtual memory consumed by the device
	StorageResources			Storage resources consumed by the device (percentage units)
18	- DiskSpace	x		Refers to the occupied disk space in the device
	CPUResources			CPU resources consumed by the device (percentage units)
19	- Total CPU (%)	x		Refers to the total of CPU cycles consumed by all active processes
	NetworkResources			Total amount of network resources consumed by the device
20	- BandwidthUsed	x		Total amount of network bandwidth used by the device (percentage)
	WorkingMode			Operational state of the device regarding working mode
21	- GeneralWorkingMode	x	x	Refers to the general working mode of the device (correct or not)
22	- MeasurementData		x	Refers to the evaluation of a clinical measurement level
23	- SensorOperationalState		x	Reports if any failure has been detected in the sensor linked to the MD.
24	- SensorConnection		x	Reports if any connection problem is detected in the sensor linked to the MD
25	- SensorJamming		x	Reports if there is any interference.
26	- SignalFailures		x	Reports if the MD detects any problem with acquired signal.

Table 5.1: General HG and MD technical data

The relations between the defined attributes and both standards used to define all this information have been gathered in Table 5.2. The nomenclature code used to identify each X73 object is: "Class:attribute.subattribute". "Class" indicates the object class inside the DIM, "attribute" indicates an attribute inside the class and, in the same way, "subattribute" indicates that the object is a subattribute. The value of the attribute reported by the X73 agent corresponding to each attribute is also indicated. This value can be a specific X73 tag or the syntax attribute expressed in terms of X73 notation which is in fact ASN.1 notation. It should be noted that "O.String" stands for"OctetString", "Int-x" stands for "Integer" using x bits and "Bits-x" stands for bit string size x. Regarding SNMP MIB relations, the name of the related object has been directly included. Other attributes are not directly related to any of these standards.

N	Attribute Name	[X73 Attribute Name & X73 Attribute Value] // [SNMP relation] [(Class:attribute. subattribute) & (tag/datatype)] // [SNMP relation]
1	DeviceType	**X73** → MDS:SystemTypeSpecList & MDC_DEV_SPEC_PROFILE_xxx // **SNMP** → sysDescr or sysServices
2	Manufacturer	**X73** → MDS:System-Model.manufacturer & O.String
3	Model	**X73** → MDS:System-Model.model & O.String
4	Software info	**X73** → MDS: Production-Specification & ProductionSpec::=INT-U16 sw-revision(4) // **SNMP** → sysDescr
5	Hardware info	**X73** → MDS: Production-Specification & ProductionSpec::=INT-U16 hw-revision(3) // **SNMP**→ sysDescr
6	Firmware info	**X73** → MDS: Production-Specification & ProductionSpec::=INT-U16 fw-revision(5) // **SNMP** → sysDescr
7	IPAddress	**SNMP** → ifPhysAddress
8	SystemId	**X73** → MDS:System-Id & O.String
9	Configuration	**X73** → MDS:Dev-Configuration-Id & ConfigId::=Int U16 e.g standard(1)
10	SupplyType	**X73** → MDS:Power-Status & PowerStatus::=Bits-16 onMains(0) or onBattery (1))
11	BatteryLevel	**X73** → MDS:Battery-Level & Int-U16 ::=Integer (0..65535)
12	RemainderBattery	**X73** → MDS:Remaining-Battery-Time & BatMeasure::= Value/Unit Float/e.g MDC_DIM_MIN
13	CommunicationProtocol	11073 for MDs
14	CommunicationState	**X73** → From 1 to 7 it indicates different states of the X73 finite state machine, eg. (1) Disconnected
15	NumberOfLinkedMD	—
16	PhysicalMemory	**SNMP** → Virtual memory, physical memory and hard disk space usage will be measured as instances of the hrStorageGroup as the storage used against max size of the storage resource being filtered by the hrStorage type → hrStorageUsed/hrStorageSize. For Physical memory, hrStorageType=PhysicalMemory
17	Virtual Memory	**SNMP** → hrStorageType=VirtualMemory
18	DiskSpace	**SNMP** → hrStorageType= C:/
19	Total CPU(%)	**SNMP** → Total CPU usage is related to hrProcessorLoad from HR-MIB. It will be calculated as the sum of the number of centi-seconds of the total system's CPU resources consumed by the active processes in the HG → $\sum$ hrProcessorLoad ($\sum$ active process)
20	BandwidthUsed	**SNMP** → Calculated with information from MIB II in this way: (Δ ifOutOctets + Δ ifInOctets in t) *8 /(ifSpeed *t) [t=60 s]
21	GeneralWorkingMode	—
22	MeasurementData	**X73** → in general for all MDs: numeric:: Simple-Nu-Observed-Value // FLOAT
23	SensorOperationalState	**X73** → Aa & AVb → PulseOxDevStat::=Bits-16 sensor-malfunction (1)
24	SensorConnection	**X73** → Aa & AVb → e.g sensor-disconnected(0)
25	SensorJamming	**X73** → Aa & AVb → e.g sensor-interference(5)
26	SignalFailures	**X73** → Aa & AVb → e.g signal-poor(11)

Aa → Enumeration: Enum-Observed-Value- Basic-Bit-Str & AVb → PulseOxDevStat or GlucoseDevStat

Table 5.2: Mapping to X73 and SNMP

5.2 HOTMES ontology based solution

The main purpose of this practical application was to study how the architecture presented in chapter 3 could be used to define a profile with a set of tasks to manage technical information in the telemonitoring scenario. To achieve this aim, the HOTMES ontology was extended ("HOTMES technical") to define a *management profile* for devices, thus a *technical management profile*. Similar to the clinical application case, the *technical management profile* comprises all the tasks that the HG should perform in order to control the general working mode of the HG and the MDs that are connected to it.

The HOTMES architecture was applied to MDs and HG technical management following the three-stage approach presented in chapter 3 (see Figure 3.2). The first stage dealt with the ontology engineering, the second stage with the ontology application-based study and the third stage with the prototype implementation. Figure 5.2 depicts inputs, actions and outputs required for each stage to adapt the ontology based architecture to remote technical management tasks. The inputs required to carry out each phase are represented on the left side of the image (materials involved in each stage), the actions performed in the centre and the outcomes or results of each phase on the right.

As with the clinical application, the data integration challenge was tackled through the ontology extension developed during the first stage of our study, the ontology engineering phase. The HOTMES ontology was extended to describe all the information that may need to be managed in relation to the technical characteristics of both MDs and HG. An important task performed during this stage was identifying and defining all the information that could be of interest, already summarized in Table 5.1. As a final outcome of this phase, an extension of the HOTMES ontology ("HOTMES technical") to define individual *technical management profiles* was achieved. As in the case of the clinical application, technical data provided by MDs and the HG are mapped into ontology instances to be processed.

The second stage dealt with the application-based study of the developed ontology. The goal of this stage was to verify that the ontology could be used to define different types of required management tasks. Consequently, a *technical management profile* containing simple, medium and complex management tasks was configured. After this stage, the ontology was adjusted in order to be able to define all required management actions, and its adaptability and completeness was demonstrated. The definition of the management tasks involved two steps: 1) defining the instances of the ontology and 2) defining the rules required to activate the analysis and planning tasks. Finally, the third stage of the approach dealt with the practical implementation of the prototype able to run the tasks specified in the *technical management profile*. For this purpose, the Semantic Autonomic Agent was modified in order also to integrate technical data.

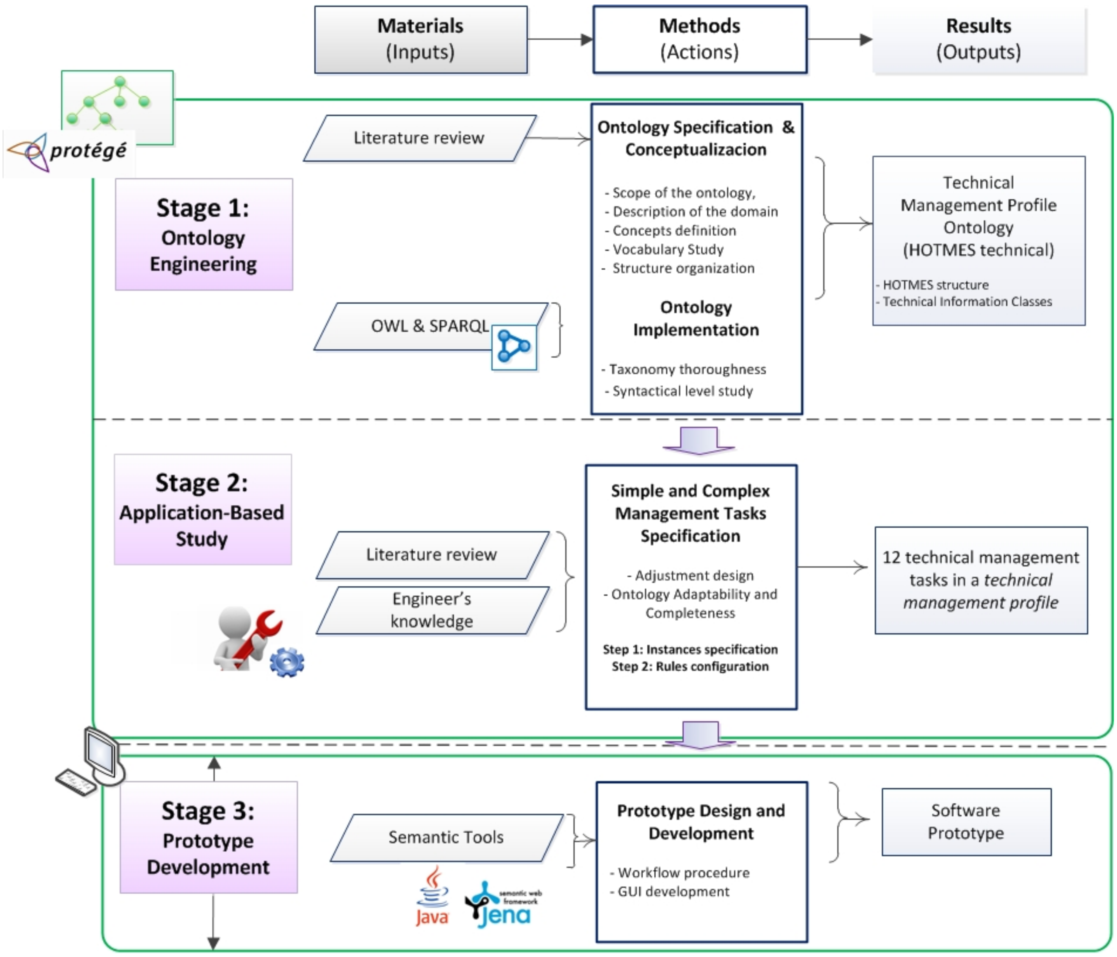

Figure 5.2: Three-Stage workflow for technical data

5.2.1 Stage 1 (technical): Ontology Engineering

As shown in Figure 5.2, the HOTMES technical ontology specification, conceptualization and implementation was addressed at this first stage.

5.2.1.1 Materials and Methods (X73, SNMP)

The information to be monitored regarding HG and MD technical features as shown in Table 5.1 was included in the HOTMES ontology extension for technical management. Furthermore, some of these terms were directly included in our ontology and these sources were also used to describe defined resources in terms of units and datatypes. On the one hand, vocabularies and tags described in X73 were used to describe technical features of MDs. Their relation to the standard was included by means of annotation properties. For all the reviewed device specializations, the technical information provided from MDs was practically the same. Therefore, adding new MDs in the telemonitoring scenario would not lead to having to modify the structure of the ontology, as the information required to control their technical status would be the same as the information already considered for the devices included. On the other hand, vocabularies and schemes described in the ontology regarding the HG characteristics were directly related to fields and information from MIB II of the SNMP architecture (see Table 5.3). Again, this relation was included in the ontology by means of annotation properties. By relating concepts from the ontology to widely known standards, the nature of the terms is clearer, and the mapping task process for data integration purposes is made easier. It should be noted that the CoDaMos ontology presented in [185] was used to inspire the organization of the information in our ontology. This ontology aims to provide a basis for the expression of the most important aspects of context information related with users and their preferences, environment, and activities, and descriptions of the hardware and software platform.

The list of competency questions to be answered with the knowledge to be expressed in the ontology is given in Figure 5.3.

Competency Questions

- Which data should be collected to control the functionality of the HG and the MDs ?
- How often should these data be retrieved?
- Should the remote technical manager be alerted if a new MD is added to the scenario?
- How will the data be analysed?
- Should an alarm be triggered according to the evaluation results?
- Which actions should be performed if an alarm is triggered? Should the remote technical manager be alerted?

Figure 5.3: Competency Questions (technical)

5.2.1.2 Results: Data Integration

The HOTMES ontology presented in chapter 3 was extended to include all the above-mentioned technical data and was thus applied to manage technical information in a home-based scenario by defining a *technical management profile*. All this information was divided into 4 groups represented in the ontology as 4 classes (subclasses of the *Technical Information* class). As can be seen in Figure 5.4, a *Clinical Data* class was included in the ontology in order to consider an out-of-range clinical data measurement as indicative of an error in the operational state of the MD.

In addition to the root classes that comprise a *technical management profile*, new classes related to the HG and MD devices were included in the ontology in order to link the *technical management profile* to the HG and provide a description of the devices presented in the scenario. As depicted in Figure 5.4, a *technical management profile* is configured for each HG. Management information about the MDs linked to it are specified in the same *technical management profile*.

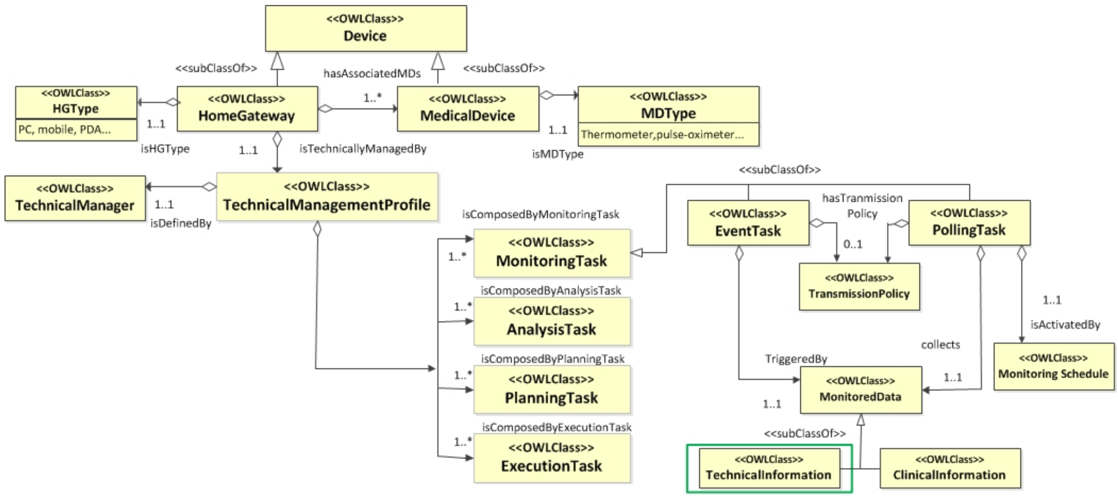

Figure 5.4: HOTMES technical version

By configuring a *technical management profile* and defining instances of the *monitoring tasks*, the general working mode of MDs can be monitored or HG resource consumption (e.g memory) can be controlled periodically. Then, by defining instances of the *analysis task*, this acquired information can be evaluated to detect an abnormal performance in the devices and, by defining instances of the *planning* and *execution tasks*, alarms can be triggered to an external manager if the performance of any of the devices can compromise patient supervision.

New classes included as subclasses of the *Technical Information* class are depicted in Figure 5.5. Instances of subclasses of the *Resources Information* class are included in a *technical management profile* in order to be aware of memory, CPU, network (bandwidth), storage and battery consumption. The operational and communication state of the devices are controlled by instances of the *HGstate* and *MDState* classes. Information about the general working mode of the devices and the sensors connected to the MDs are controlled by instances of classes *DeviceWorkingMode* and *SensorWorkingMode*. Finally, the number of MDs linked to the HG may be evaluated (*NumberLinkedMD*). The relation between these classes and the technical information depicted in Table 5.1 has been gathered in Table 5.3.

The version of the HOTMES ontology for technical application contains 120 classes, 85 object properties, 86 data properties and 56 individuals. Furthermore, 53 annotation properties have been included in the ontology. These have been used to relate terms from the ontology with the X73 standard and the MIBs of SNMP, to clarify some terms and to indicate whether a property was used as a primary key property. The size of the OWL ontology file is 214 KB and the DL expressiveness is ALCHON(D).

Ontology Class	*Attribute Name (Attribute Number)*
Memory Resource	Physical Memory (16)
	Virtual Memory (17)
Virtual, physical and hard disk space data can be reported by an instance of this class	
Storage Resource	Disk Space (18)
CPU Resource	Total CPU(%) (19)
Bandwidth	Bandwidth used (20)
Several instances can be configured according to the interface evaluated	
PowerResource	Supply type (10)
BatteryLevel	BattteryLevel (11)
RemainderBattery	RemainderBattery (12)
HGState	CommunicationState (14)
Communication state related to the MDs: communicating (0) or idle (1)	
MDState	CommunicationState (14)
Related to the states defined for the X73 standard	
NumberLinkedMD	NumberOfLinkedMD (15)
DeviceWorkingMode	GeneralWorkingMode (21)
State of the HG: operative (0) or damaged (1)	
SensorWorkingMode	SensorOperationalState (23), SensorConnection (24) SensorJamming (25), SignalFailure (26)
HomeGateway *Medical Device*	For both: Manufacturer (2), Model (3), SoftwareVersion (4), HardwareVersion (5), FirmwareVersion (6), SystemId (7) and Configuration (8)
MDType	DeviceType (1)
HGType	DeviceType (1)

Table 5.3: Technical Classes Description

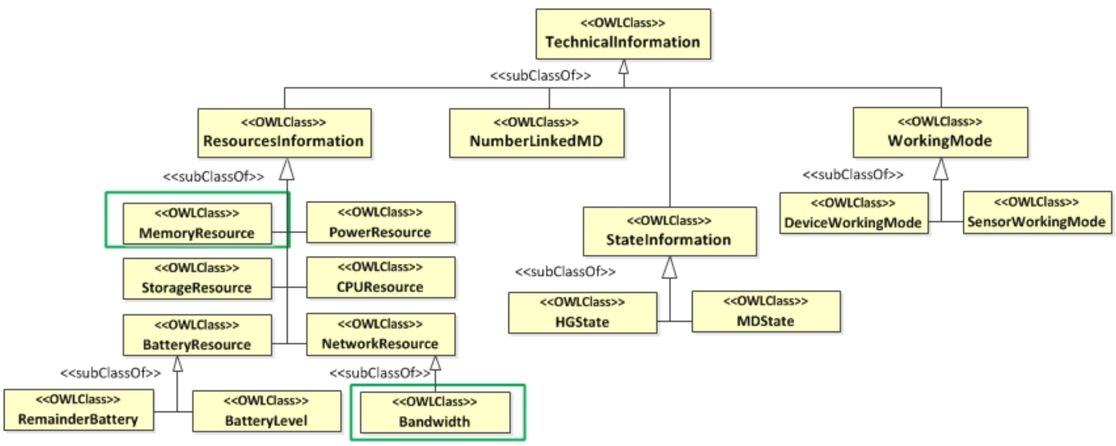

Figure 5.5: Technical Data Classes and Subclasses

As with the clinical data defined in the clinical HOTMES ontology, information used to describe the technical concepts described above has been organized in two levels: 1) the Configuration level and 2) the Results Data Level. As can be seen in Figure 5.6, the first level gathers properties used to characterize a resource. An instance of these classes should be configured by the external manager when defining the *technical management profile*. For example, regarding the classes displayed in Figure 5.6 (memory resource and bandwidth usage), in order to monitor the bandwidth used in a link, the IP address of the interface and the type of interface should be configured previously. The second level contains properties that describe and characterize a datum associated to a previous resource description. These data will be provided by the different sources involved in the management procedure. Therefore, this second level describes all pieces of data eventually retrieved from the MDs and the HG. In the example given above, the maximum capacity of the bandwidth link (*BandwidthMaxCapacity* class), the date, the time and the output value of the measurement, together with the speed units (*SpeedUnit* class), will characterize the bandwidth datum related to the previously configured information.

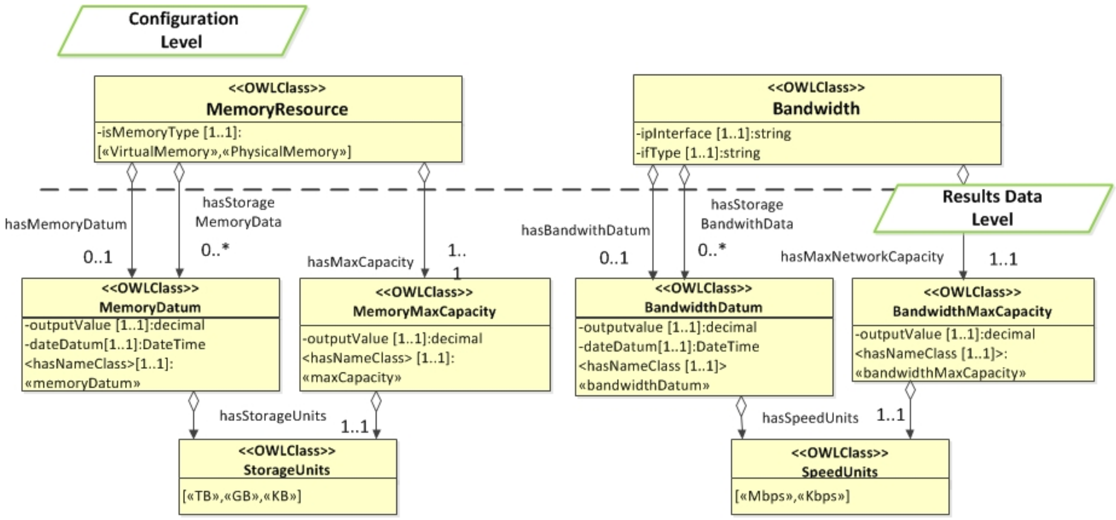

Figure 5.6: Memory Resource and Bandwidth Usage classes

5.2.2 Stage 2 (technical): Ontology Application-Based Study

The main goal of this stage was to test the applicability of our approach by configuring management actions in a *technical management profile*. These are different types of typical management actions (simple, medium and complex) of interest for controlling the performance of a device. This stage was part of the design process as it led to some adjustments after it was completed.

5.2.2.1 Materials and Methods

Technical Management Profile Example
Simple Actions
Action 1: ***Evaluate the state of a MD*** Warn both the EM and the patient if the MD is detected as disconnected
Action 2: ***Evaluate the communication state of the HG*** Warn the patient if the HG is receiving information from an MD
Action 3: ***Evaluate the state of a sensor connected to a MD*** Warn the patient if the sensor is not working properly
Action 4: ***Evaluate the battery level of a MD*** Warn the patient if the battery level of a MD is detected to be under 15%
Action 5: ***Evaluate the BW (Bandwidth) used*** Warn the EM if an overuse of the available BW (BW/MaxCapacity in %) is detected
Action 6: ***Evaluate the CPU daily usage*** Warn the patient if the average of the CPU daily usage is over a threshold.
Medium Actions
Action 7: ***Evaluate a clinical measurement value*** Warn the EM if it is the third time that a clinical temperature measurement is out of a numeric range
Action 8: ***Evaluate periodically RAM memory usage and the disk space usage*** Warn the EM if both memory and disk space usage are over a threshold (e.g.72% and 90% respectively)
Action 9: ***Evaluate the number of MDs linked to the HG conditioned to the memory used*** If the RAM memory usage is detected to be over 80% of its capacity, then evaluate the total number of MDs that are linked to the HG and warn the EM if there are more than 5
Complex Actions
Action 10: ***Evaluate the CPU/CPU average*** This action measures the absolute CPU value acquired in comparison with the last CPU average calculated. If this measurement is over a threshold then the EM will be alerted
Action 11: ***Evaluate the BW%/BW% average*** This action measures the calculated BW used value in comparison with the last BW average calculated. If this measurement is over a threshold then the EM will be alerted.
Action 12: ***Evaluate the BW%/BW% average conditioned to BW%*** If the calculated BW is over a threshold then the BW/BW average will be evaluated as in the previous case

Table 5.4: Management Action Examples

For this evaluation stage, a total of 12 action tests were defined in order to evaluate a wide range of possible management situations and alarm definitions. These actions were included in a single *technical management profile*. The tests demonstrated that by defining an instance of the *technical management profile*, it was possible to configure the agent to warn the patient when, for example, it detects that a registered MD is not connected to the HG or when the battery level of a MD is under the 15%. Table 5.4 shows the configured actions used to evaluate the ontology and the agent management capabilities.

5.2.2.2 Results: Management Profiles Specification (complex and simple)

The mapping process between defined actions and instances of the HOTMES technical ontology follows similar rules to those shown in Figure 4.7 for clinical management. An instance of the *monitoring task* class (event or polling) is configured in order to monitor the state of a resource (e.g the state of the MD or RAM memory consumption). Then, an instance of the *analysis task* class (mathematical or syntactical) is configured in order to analyze these data and, according to the acquired result, warn the external manager, the patient or both (*Planning* and *Execution* task). Note that a complex or a two-step analysis requires the execution of *CompositeProcess* to be configured, while a simple analysis can be done by instances of *AtomicProcess*. Rules are used to render the result of an analysis positive or negative and also to activate a *Planning Task*.

The actions shown in Table 5.4 led to the definition of a total number of 11 monitoring tasks, 12 analysis tasks, 6 planning tasks and 2 execution tasks. The number of required tasks (MT: monitoring task, AT: analysis task, PT: planning task, ET: execution tasks) instances and rules to configure each action are shown in Table 5.5.

Action	MT	AT	PT	ET	Individuals	Rules
Action 1	1	1	1	2	20	2
Action 2	1	1	1	2	17	2
Action 3	1	1	1	2	17	2
Action 4	1	1	1	2	18	2
Action 5	1	1	1	2	18	2
Action 6	1	1	1	2	18	2
Action 7	1	1	1	2	19	4
Action 8	2	2	1	1	28	4
Action 9	2	3	1	1	27	4
Action 10	1	2	1	1	22	2
Action 11	1	3	1	1	22	2
Action 12	1	4	1	2	28	4

Table 5.5: Technical Classes Description

Only atomic processes are involved in the configured mathematical analysis of the simple action examples. Furthermore, planning tasks are activated by only one pre-condition. As can be seen, only two rules are involved in each of these management tasks. It should be noted that the sixth action requires a non-real time analysis. This involves having to store a set of individuals during a day and to perform the analysis at a certain programmed hour. Medium management actions require configuring more complex tasks. For example, to add several pre and post conditions to the planning task and composite process, (If then Else) is required to perform the last task. The number of tasks and rules are also increased. Finally, Table 5.5 includes complex management tasks that can be configured within a *technical management profile*. The number of tasks and rules is similar to that of medium actions. However, in the case of complex actions, apart from the number of rules and tasks required to monitor just one resource, the output of an analysis process will be used as the input for another analysis task. The complexity of the analysis tasks is thus increased because their performance requires more than one process to be executed (atomic or composite). As shown in Table 5.5 there are no significant differences regarding the number of individuals required to configure each action. This number is increased when several resources are monitored as part of the same analysis (e.g Action 8) or when it is required to configure a *CompositeProcess* in the analysis (e.g Action 12). The number of individuals can also be increased if several execution tasks are configured within a plan (e.g Action 1).

To illustrate the use of the *technical management profile* and the instance task definition, a simple example of an action configured in Table 5.4 is presented in detail. Let us assume that we want to monitor when a thermometer is not working properly by controlling the temperature measurement value that it provides. A monitoring task (event type) associated to this clinical measurement must be defined in the profile (see Figure 5.7). It will be determined that a clinical measurement out of the range 32-43 C will indicate that the thermometer is not working properly, so an analysis task (mathematical type) to evaluate the absolute value of a clinical temperature datum must be included in the profile. This analysis will be related to two conditions expressed by two SPARQL rules (rules 1 and 2 in Figure 5.7): one to detect when the measurement datum is over a threshold (43 C) and the other to detect when it is under a threshold (32 C). A positive result in this analysis will activate the planning task (condition expressed by rule 3 in Figure 5.7). Nevertheless, in order to be sure that the out-of-range measurement is not isolated, a post-condition will be defined (expressed by rule 4 in Figure 5.7) to indicate that the execution tasks that compose the planning task should only be performed if the out-of-range measurement is detected 3 times. This planning task composed of two execution tasks (send an e-mail to the manager and show a pop-up to the patient) will be included in the *technical management profile*. Therefore, as depicted in Figure 5.7, an instance of the *technical management profile* to run this test will be composed of 5 tasks and 4 rules will have to be defined.

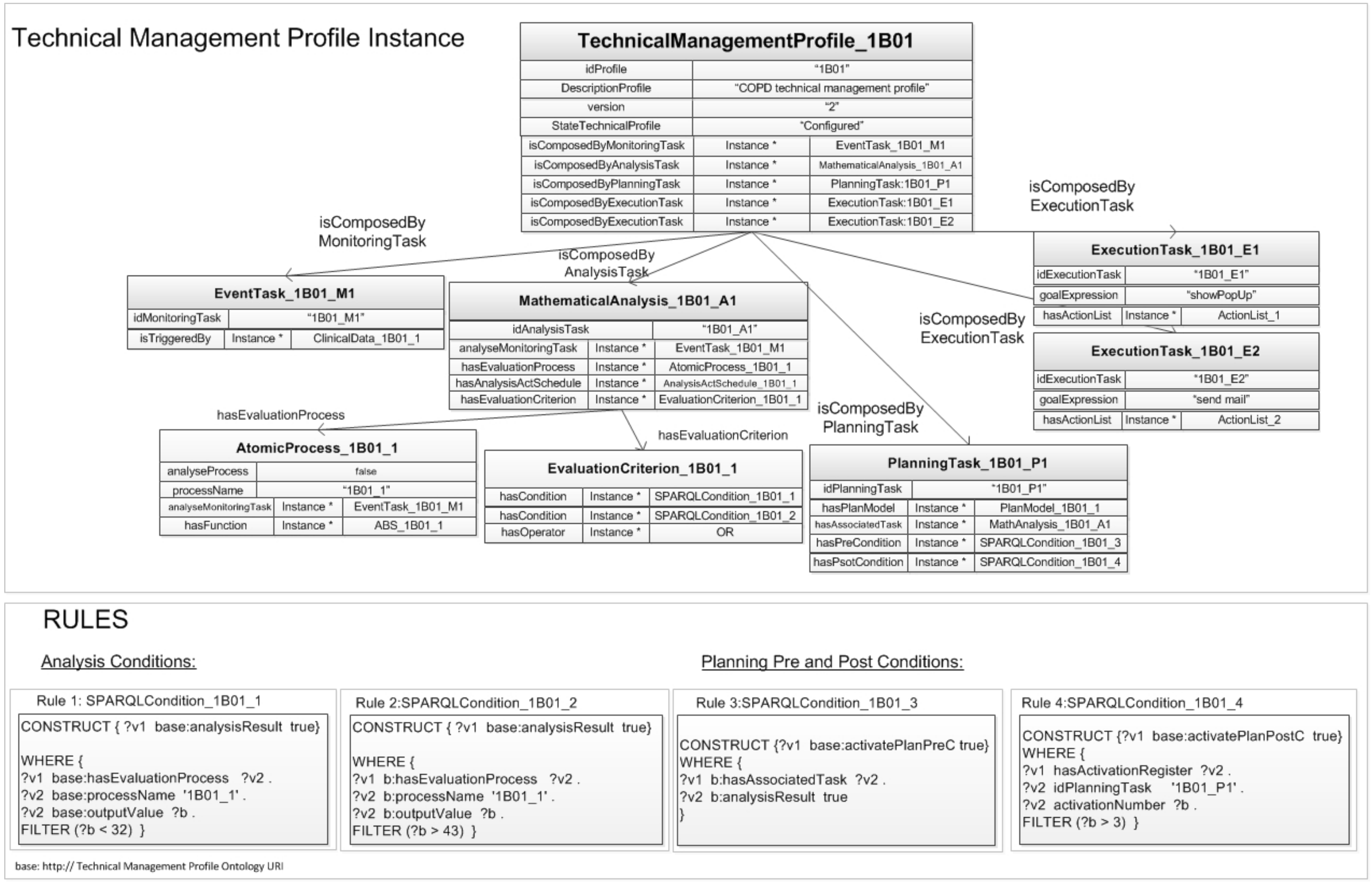

Figure 5.7: Technical Management Profile Instance Example + SPARQL rules

5.2.3 Stage 3 (technical): Prototype Development

5.2.3.1 Results: Implementation

The Semantic Autonomic Agent presented in chapters 3 and 4 was slightly modified in order to support technical management tasks and thus run the tasks specified in a *technical management profile*. In particular, the Knowledge Base Module (1) and the Integrator Module (7) were modified. The former was enhanced in order to hold the HOTMES technical ontology and the *technical management profile*. The latter was modified in order to translate data acquired from the MDs and the HG into ontology datum instances. In order to test the agent's performance, two mapping modules were included in the implementation in order to integrate data transfered from an MD using the X73 and data collected from the HG resources using SNMP (see Figure 5.8). Hence, in our implementation, data (clinical and technical) from an MD was simulated to be transferred from an X73 manager and technical data was transferred from an SNMP agent (see Figure 5.8).

The GUI was also modified in order to allow users to download the *technical management profile*. It was not necessary to modify any of the other modules already developed for the clinical purpose. Figure 5.8 shows these proxy entities and the modules that required modifications.

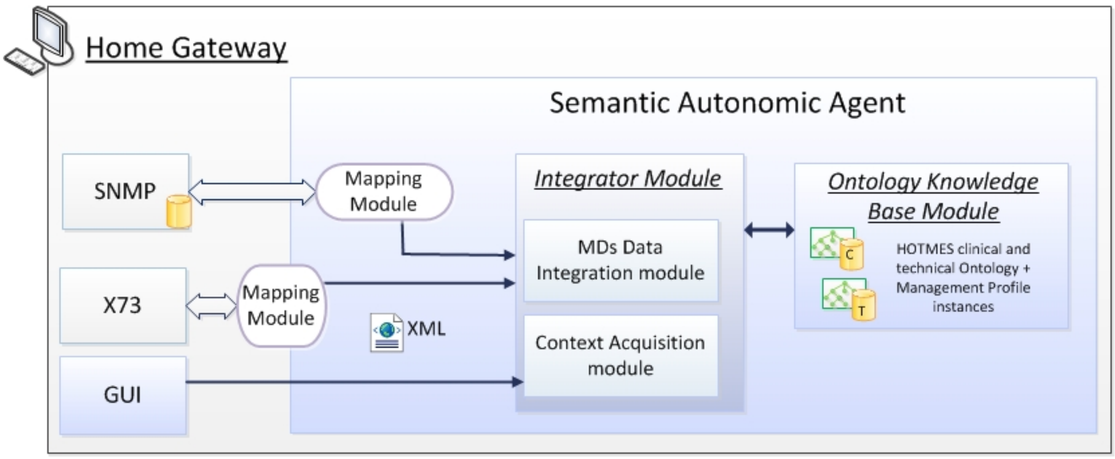

Figure 5.8: Modified modules in the agent implementation

Figure 5.9 depicts the principal view of the interface of the semantic autonomic agent where technical buttons have been highlighted. The "Donwload technical MP" button is used to download the *technical management profile*. The "management profile information" button is used to access information registered in the *management profile*. Information related to registered MDs can be seen by pressing button "Registered MDs". Finally, registered technical alarms can be reviewed after pressing button "View Technical Alarms".

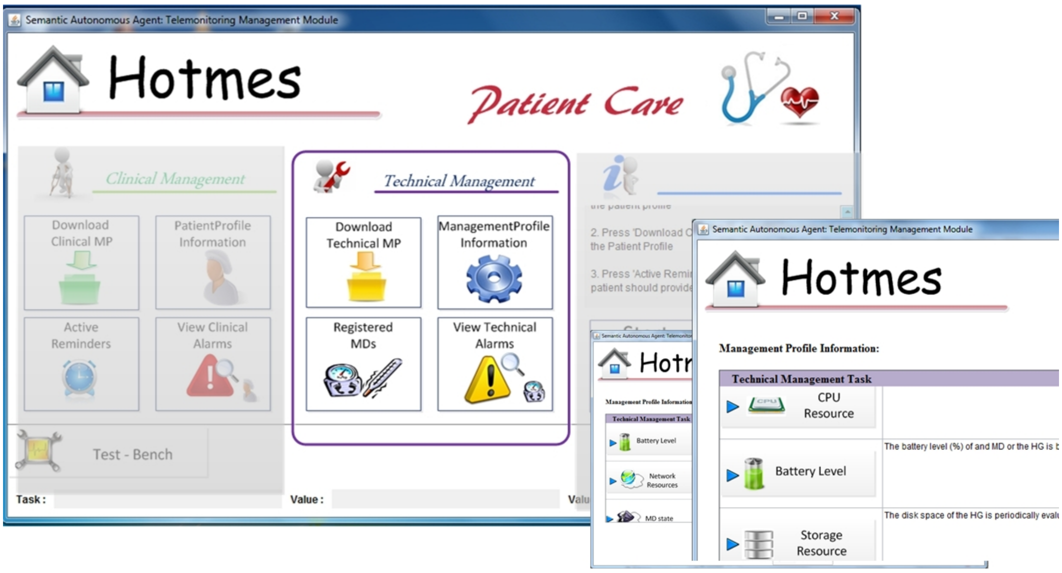

Figure 5.9: Semantic Autonomic agent interface (technical view)

5.2.3.2 Workflow Procedure

The Semantic Autonomic Agent follows the same workflow procedure to run a *technical management profile* as that followed to run a *patient profile*. One *technical management profile* is configured for each HG. Although several *patient profiles* can be located in the same HG (to monitor different patients), there is only one *technical management profile* associated to each HG. Once it is downloaded by the patients, the configured polling tasks are initiated. Every time that new data is retrieved both from the HG and the MDs, the Semantic Agent runs the MAPE closed loop which analyses the acquired data and executes the associated planning and execution tasks. It should be noted that every time that a new MD is detected to be transferring information to the HG, it will be communicated to the remote site. An instance of the *Medical Device* class will be created containing the information of the new MD and will be transferred by using the *PUT(Individual)* method defined in the remote web service.

5.2.3.3 Evaluation Performance

As explained in chapter 4, the performance of the Semantic Autonomic Agent has been evaluated in terms of memory cost, computational cost, network cost, storage cost and execution time. The Semantic Agent was located in the engine equipment described in chapter 4 (1.6 GHz i7 Intel Core running Windows 7 machine).

In order to check the performance of the agent for technical management purposes, a *technical management profile* was tested containing simple actions required to monitor the resources of MDs used by a COPD patient and to control HG resources. The first test consisted of running this *technical management profile*. A second test was performed running both a *patient profile* for a COPD patient and its *technical management profile* together. For the evaluation test, the agent was run during short periods of high activity. Polling tasks were configured to retrieve data every two minutes and events were manually simulated in periods with irregular activity (5 minute active periods every 1 minute and every 5 minutes).

Test 1: "COPD" technical management profile

This *technical management profile* was designed to monitor the state of a weighing scale, a blood pressure monitor, a pulse-oximeter and a glucometer. Basic monitoring actions were also configured in order to monitor the state of the HG. Specifically, the actions listed in Table 5.6 were configured.

"COPD" Technical Management Profile Example
The following actions are configured for each required MD. Specifically for a COPD case, the MDs required are: weighing scale, blood pressure monitor, pulse-oximeter and glucometer
Action 1: *Evaluate the state of a MD* Warn both the EM and the patient if any of the 4 MDs is detected as disconnected
Action 2: *Evaluate the battery level of a MD* Warn both the EM and the patient if the battery level of any of the 4 MDs is detected to be under 15%
Action 3: *Evaluate a clinical measurement value* Warn both the EM and the patient if it is the third time that a clinical blood pressure measurement is out of an extreme numeric range
The following actions are configured for the HG
Action 4: *Evaluate periodically by polling the BW (Bandwidth) used* Warn both the EM and the patient if an overuse of the available BW (BW/MaxCapacity in %) is detected. Specifically if this value is over 60%.
Action 5: *Evaluate the CPU* Evaluate if the the CPU% usage is over a threshold (e.g 70%). Warn both the EM and the patient if this event often happens.
Action 6: *Evaluate periodically by polling RAM memory usage* Warn both the EM and the patient if the RAM memory usage is over a threshold (e.g. 72%)
Action 7: *Evaluate periodically the disk space usage* Warn both the EM and the patient if the disk space usage is over a threshold (e.g. 90%)

Table 5.6: Actions included in the COPD *technical management profile*

These actions led to configure 13 monitoring tasks, 7 analysis tasks, 7 planning tasks and 2 execution tasks. In addition, 11 rules were configured and 83 new instances were required to be configured in the *technical management profile.*

Initially, the Semantic Autonomic Agent application downloads the OWL file into the TDB database whose initial size (without the *technical management profile* instance) is 684 KB . The size of the ontology OWL file is 214 KB. To download the ontology involves exchanging 69,50 KB with the remote site. Then, the agent asks for the corresponding *management profile* and the rules. The identifier to download the *technical management* *profile* is the same as that used to initiate the semantic agent tool. In summary, this first step consists of downloading the ontology, downloading the *technical management profile* instance and downloading the rules. Table 5.7 summarizes the resources consumption results after this first step. Specifically, the table shows the memory cost in terms of the maximum increment (Δ Max Mem) observed over the initial memory required (83.476 KB), the network cost (the total amount of exchanged KB, IP data, between the web points in order to download the ontology, the *technical management profile* instance, PP, and the rules, R), the initial TDB size and the required execution time in the Converter module.

Profile	Δ Max Mem (MB)	Network cost (R)(KB)	Network cost (PP)(KB)	TDB. ini (KB)	E.Time ± σ (ms)
COPD Technical	16.396	9,71	164,40	988	1.444 ± 26.19

Table 5.7: Cost metrics evaluation COPD technical test (step 1)

The second step involves starting the agent to run the tasks specified in the profile. After one hour running the agent with periods with high and low activity, the maximum increase in memory was 51.524 KB. After this period of time the TDB was 2,46 MB and the average maximum CPU observed was 0.9%. Regarding network usage cost, the average of KB exchange when a datum was acquired or an alarm was transferred was 5,2 ± 497,7 KB.

Test 2: "COPD" technical and clinical management profile

As depicted in Figure 5.8, the Semantic Autonomic Agent contains two different databases. One holds the clinical ontology plus the *patient profiles* and the other holds the technical ontology plus the management profile. Running the *patient profile* and the *technical management profile* together does not affect the agent performance in terms of storage cost or network cost as there are no dependencies in the process of downloading the *management profile* and the rules are stored in different databases. Similarly, the execution time required in the converter module is not affected. In fact, running both profiles at the same time only affects the memory cost, and very slightly. After running both the COPD *patient profile* and the *technical management profile*, the observed increase in maximum memory was 67.000 KB.

The COPD *technical management profile* was modified in order to introduce some complex management tasks and then study the resulting performance of the agent. Although more instances are required to express complex tasks, there were no significant differences in terms of execution time or memory cost. The only increase worth mentioning is the TBD storage size. This was increased when executing actions that involve an average analysis because more than one individual must be stored in order to be able to calculate the required average.

5.2.4 Discussion

Following the three stages presented in chapter 3 (see Figure 3.2), section 5.2 has examined the adaptation of the proposed HOTMES-based architecture for technical management in home-based scenarios in order to detect any technical problem in the MD or in the HG performance. The application and re-usability of the HOTMES solution for technical purposes has been demonstrated.

The combination of ontologies and the autonomic computing paradigm is an interesting approach that takes advantage of the main benefits that both solutions offer in terms of knowledge representation (to address integration and interoperability challenges), workflow organization and self-management capabilities. This last capability implies that the EM does not need to poll periodically the HG to control it, thus mobility and independence are enhanced while management tasks for the EM and traffic between the two end points are reduced.

The idea of defining *management profiles* in combination with SPARQL rules offers a flexible solution to personalize management tasks according to the demands of each particular scenario. This is of interest given that few studies have been conducted into technical management in home-based telemonitoring scenarios. Most published solutions deal only with clinical data [86]. Furthermore, to the best of our knowledge, this is the first approach dealing with ontologies and technical management of MDs in telemonitoring scenarios. Hence, it constitutes an innovative and integrated management solution in the field of healthcare.

5.3 SNMP based solution

The SNMP solution was based on the development of an SNMPv3 agent and its MIB-module. This SNMP entity was installed into the HG and stored into its MIB-module information about the HG technical resources and MD's data that were transferred there. Then, by using SNMP protocol an SNMP manager externally located (for example at hospital) could establish a communication with the developed agent in order monitor the HG and MDs technical status. Specifically SNMP version 3 was selected in order to ensure security in data access and transferences. Noteworthy that this HG module can be termed also as CE (Compute Engine).

Specifically, the integration of the X73 and the SNMP architecture was studied in order to provide a unified, global and integrated technical management solution (see Figure 5.1, solution B). The main purpose was to take advantage of the relevant characteristics that both protocols offer to solve this management problem. This proposal avoids increasing computational power and energy consumption in MDs, providing a solution that optimizes efficiency and increases flexibility. The integration of an SNMP agent in a HG enables the telemonitoring system to be accessed via any network management tool that understands SNMP. To the best of our knowledge, this is the first approach dealing with the integration of X73 information into an SNMP architecture to technically manage MDs and HG and, moreover, one of the few SNMP approaches applied in a telemonitoring scenario [139, 140].

5.3.1 General Architecture Description

The integration of both standards is leveraged by an architecture based on the development of an agent which integrates into a MIB-module two information sources: one provided by MDs that use X73 to transfer information to the HG and one related to the HG resources information. In fact, the agent designed is not only, in terms of SNMP, an SNMPv3 agent entity to establish a communication with another SNMPv3 manager that can be externally located for remote management. Additionally, it also implements an X73 interface that works as a proxy with the X73 manager. Hence, as the functionalities of this agent have been enhanced it will be more appropriately designated as “SNMPv3-proxyX73 agent”. The general architecture of the developed agent is depicted in Figure 5.10. This architecture can be described by its communication and data models.

1. *Communication model*: On the one hand the use of X73 is proposed to be used by the MDs to communicate data to the HG. On the other hand, the communication between the proposed agent installed in the HG and the EM is done through SNMPv3 messages. The developed agent works as a conventional SNMPv3 agent being capable of answering SNMPv3 requests from an EM and also to send asynchronous messages to notify important changes (traps). As can be seen in

Figure 5.10, the HG may be managed by several EMs. Furthermore this SNMPv3 agent has proxy functionality in order to make possible the communication between the X73 manager and the developed agent. The communication between the X73 manager and the SNMPv3 agent is a bidirectional communication through a proxy module using XML files exchange. The X73 manager can transfer data to the agent in both a polling and an event-driven manner. Hence, it can send messages to the SNMPv3 agent in an asynchronous manner and can also send response messages to the agent requests. It should be noted that when the SNMPv3 agent requests information from the X73 manager, the manager provides information that it has in its memory concerning the MD in question.

2. *Data model*: The architecture data model relies on the development of a MIB-module used to hold the information both from the MDs (provided by the X73 manager) and the HG's own resources. The developed MIB-module has been designated as Medical Devices-Management Information Base (MD-MIB). Furthermore, the MIB-module provides a set of structured data to define alarms and events in order to control abnormal values related to MD and HG performance. Technical data considered for the design of the MIB-module is shown in Table 5.1.

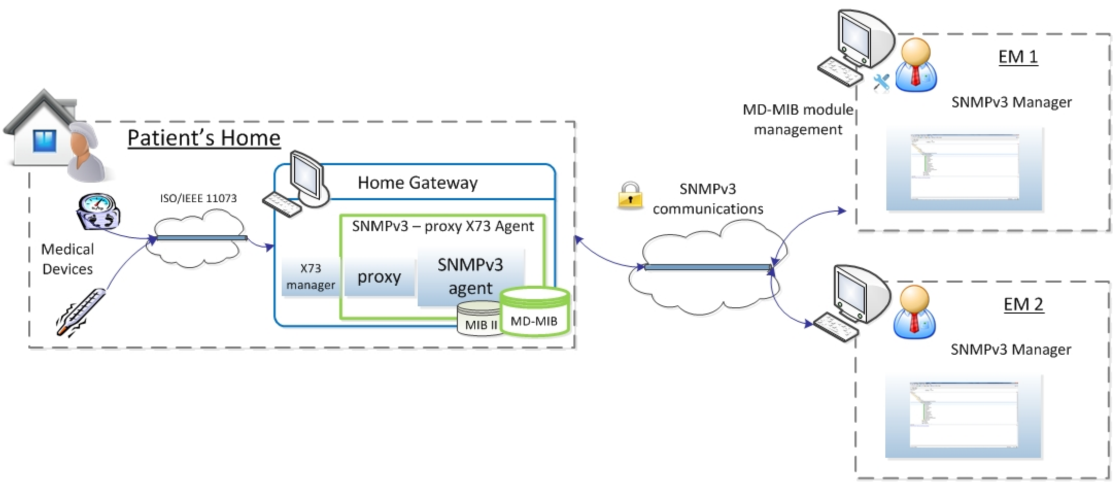

Figure 5.10: General SNMP architecture proposal

5.3.1.1 Communication and Technical Data

The general operating mode of the system (data flows and interactions between the modules) is depicted in Figure 5.11. In the proposed scenario, the information provided by MDs is collected in the HG by the X73 manager using a communication based on X73. This information, together with information about the communication state of the MDs, is analyzed and then transferred from the X73 manager to the SNMPv3-proxyX73 agent to be remotely accessed. This information is translated by using the proxy module, then transferred to the SNMPv3 agent and stored in the MD-MIB-module. In the same way, information about the HG resources is collected periodically by using the information available in the MIB II of the SNMP and stored in the designed MIB-module. As a matter of fact, previous proxy modules deployed for the HOTMES solution presented in section 5.2 were adapted to the SNMP solution. On the other side, an EM can access the HG to manage technical information provided by both MD and CE devices through an SNMPv3 communication. It should be noted that both communications (X73 and SNMPv3) are independent. Hence, SNMP methods are not directly related to X73 commands. Nevertheless most of information managed through SNMP should be provided by the X73 manager and consequently first by the MDs (through X73 communication).

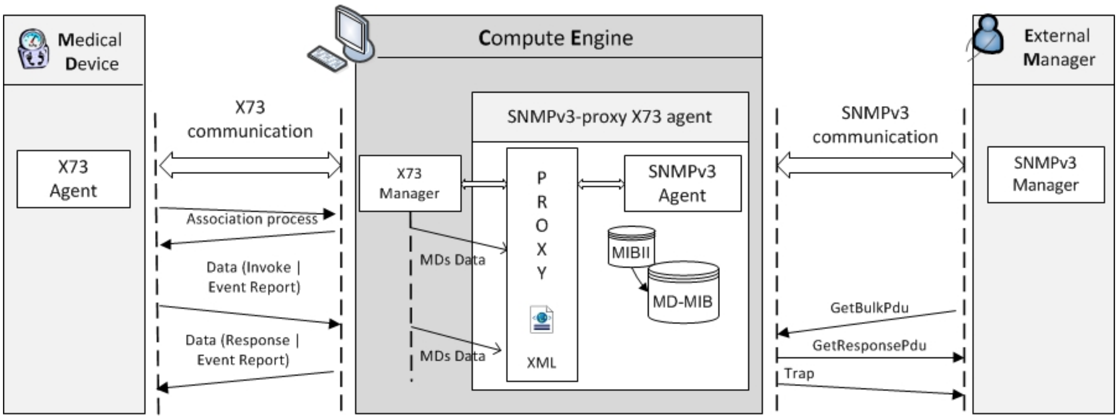

Figure 5.11: Communication model (SNMP-X73 solution)

5.3.1.2 MIB-Module Design

The proposed MIB-module was designed to make X73 information accessible in the SNMP architecture, thus the MIB-module includes all the information presented in Table 5.1. Therefore, the MIB design was based on the X73 standard information. Moreover this MIB-module includes technical aspects related to the HG capabilities. Apart from monitoring information provided by MDs and monitoring the operating mode of the HG, the developed MIB-module enables the EM to configure alarms and define associated events.

1) MIB-overview

The developed MD-MIB-module is located inside private MIB groups under the top-level OID assigned to our research group: zaragozaNetworkManagementResearchGroup OID: 1.3.6.1.4.1.28308. The MD-MIB-module has been given the OID: 1.3.6.1.4.1.28308.4. Its definition provides scalability to the proposed agent design and it could be easily extended if the functionalities of the agent were also extended in the future. As can be seen in Figure 5.12, the proposed MIB is composed of 5 sections (tables and groups that can also be organized in groups). These groups are: computeEngineControlInfo, medicalDeviceInfo, alarmsTable, eventsTable, and managerTable. Furthermore, most of the tables that compose the described groups are inspired by RMON MIB organization control and data tables and also working mode rules. A control row has been added in the alarmsTable, eventsTable and managerTable groups in order to control the creation and modification of alarms and associated event actions and to register new technical managers.

SNMP provides no specific mechanism for issuing a command to an agent to perform an action. However, it is possible to use the Set command to issue an operation that is called an action invocation. A number of these objects have been included in the MD-MIB-module. They are located within the computeEngineControlInfo and the medicalDeviceInfo and are used in order to update information about the resources of the HG and to ask the X73 manager for an update of MD data.

Tables 5.8, 5.9, 5.10, 5.11 and 5.12 follow SMIv2 notation to describe main objects from the designed MD-MIB. Syntax field contains primitive datatypes defined in ASN.1, application datatypes such as Ip-Address ("Ip-Add" in tables) used to define IP address or defined types with next structure: type::=int32 option0 (0), option1(1) in order to limit attributes values to X73 data . Apart from that, for the MIB-MD definition, datatype called "DateTime" used to express together the date and time format (Octet String based) has been defined.

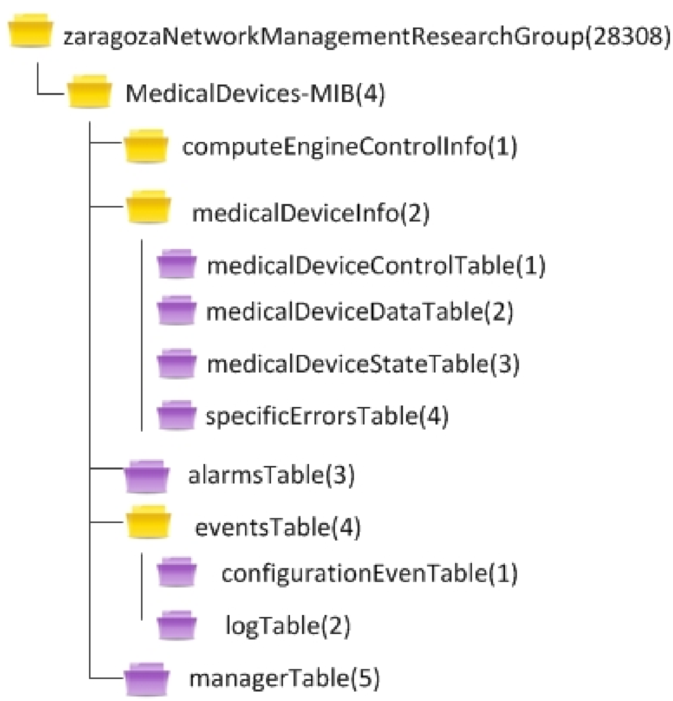

Figure 5.12: Medical Devices MIB

2) ComputeEngineControlInfo

This group contains scalar objects. In fact it contains information related to the HG performance and its technical features presented in Table 5.1. A detailed explanation about the objects that comprise this group is provided in Table 5.8. As is shown in this Table, some information corresponds to some data from the MIB II (interfaces and HR-MIB group) which is periodically updated (default interval request is 1 minute) in a transparent process for the EM.

ObjectName (Id)	*Syntax*	*Access*	*Description/Mapping Fields*
idComputeEngine(1)	O.String	RO	CE/HG identification sequence
deviceType (2)	O.String	RO	CE/HG device type (mobile, PC)
manufacturer (3)	O.String	RO	Mapping to → Table 5.1: attribute N 2
model (4)	O.String	RO	Mapping to → Table 5.1: attribute N 3
workingState (5)	WS:=Int32	RO	Mapping to → Table 5.1: attribute N 21. State of the CE: operative (0) or damaged (1)
communication State (6)	CS:=Int32	RO	Mapping to → Table 5.1: attribute N 14. Communication state related to the MDs: communicating (0) or idle (1)
associatedMDs (7)	Int32	RO	Mapping to → Table 5.1: attribute N 15
supplyType (8)	ST:=Int32	RO	Mapping to → Table 5.1: attribute N 10 (onMains (0) or onBattery (1))
batteryLevel(9)	Int32	RO	Mapping to → Table 5.1: attribute N 11
remainderBattery (10)	Int32	RO	Mapping to → Table 5.1: attribute N 12
bandwidthUse (11)	Int32	RO	Mapping to → Table 5.1: attribute N 20
cpuUse (12)	Int32	RO	Mapping to → Table 5.1: attribute N 19
hardDiskMemory (13)	Int32	RO	Mapping to → Table 5.1: attribute N 18
virtualMemory (14)	Int32	RO	Mapping to → Table 5.1: attribute N 16
physicalMemory (15)	Int32	RO	Mapping to → Table 5.1: attribute N 17
userContactInfo(16)	O.String	RO	Contact information about HG owner
hostIpAddress(17)	Ip-Add	RO	Corresponds with the CE ip address
updateRequest-StateMDs(18)	Int32	R/W	Works as an invocation object. By setting 1 in this field, the agent will ask the x73 manager for an update of the state of all the devices that are linked to it
updateRquest-TechnicalDataMD(19)	Int32	R/W	Works as an invocation object. Ask the manager for an update of the dynamic technical data of the MDs
dateTimeControl (20)	DT	RO	Indicates the date-time of the last data update

Table 5.8: Compute Engine Control Info Group

3) MedicalDeviceInfo

This group contains tabular objects. It contains information about the MDs that are linked to the HG. Therefore, several instances of each object are necessary to show information of all the MDs. In a similar manner, the rest of the groups require this organizational structure to present information.

This group is composed of one control table and three data tables. The control table contains a row for each one of the MDs that are linked to the HG, identified by a unique idMDControl value. As is shown in Table 5.9, medicalDeviceControlTable contains static information. The first data table (medicalDeviceDataTable) contains dynamic data provided by the MD and it is always updated when a new message with technical information arrives from the X73 manager. MedicalDeviceStateTable provides a record of the MD states. The defined possible states correspond to the X73 device state machine described in section 5.1. Finally, there is a third table that stores information related to specific reported MD errors. When the X73 manager detects a new MD linked to it, the event is communicated to the SNMPv3 agent and a new entry with the available information is created in the medicalDeviceControlTable. As long as the X73 manager retrieves more information from the MD, it is included in the previous table to complete the entry. Regarding the data tables, a record of 20 entries is reserved for each MD in order to store a list of the most recently provided data. This number of entries was chosen because it was considered sufficient to provide a repository of technical data (of interest in order to know the technical evolution over time) but not an excessive burden added to the agent. Nevertheless, this parameter could easily be modified in the database. A description of the objects or their relation to X73 attributes shown in Table 5.1 is provided in the last column of Tables 5.9, 5.10, 5.11 and Table 5.12. Apart from described MDs states in section 5.1 two additional states were considered in the MIB-module design: Available (identified MD in the X73 manager list) and NotAvailable (MD removed from manager list).

These tables do not in fact work exactly like RMON control and data tables because the information described in the control table will be mostly filled by the agent and not by the EM. Nevertheless, the distribution of the information follows the design philosophy mentioned as the first group (control table) contains static information about the identified MDs and the data tables contain several data for each MD specified in the control table. The relationship between medicalDeviceControlTable and medicalDeviceDataTable is straightforward. For each MD specified in the control table there will be from 0 to 20 rows in the data table. Each row of the data table will be identified by a unique combination of the idMDData index and the idMDCapture. The relation between the control table with the other data tables (state and errors table) is the same.

ObjectName (Id)	*Syntax*	*Access*	*Description/Mapping Fields*
idMDControl(1)	O.String	RO	Uniquely identifies each MD linked to the HG
manufacturer (2)	O.String	RO	Mapping to X73 → Table 5.1: attribute N 2
model (3)	O.String	RO	Mapping to X73 → Table 5.1: attribute N 3
systemId(4)	O.String	RO	Mapping to X73 → Table 5.1: attribute N 8
mdType (5)	O.String	RO	Mapping to X73 → Table 5.1: attribute N 1
protocolType (6)	O.String	RO	Mapping to X73 → Table 5.1: attribute N 13
configurationType (7)	CT:=Int32	RO	Mapping to X73 → Table 5.1: attribute N 9 (standard (0) or extended (1))
updateRequestState(8)	Int32	R/W	Works as an invocation object. By setting 1 in this field, the agent will ask the X73 manager for an update of the state of this MD
updateRequest-TechnicalData(9)	Int32	R/W	Works as an invocation object. Ask the manager for an update of the dynamic technical data of this MD
deviceDate (10)	DT	RO	Indicates the date-time when the MD was registered

Table 5.9: Medical Device Control Table

ObjectName (Id)	*Syntax*	*Access*	*Description/Mapping Fields*
idMDData(1)	O.String	RO	Indicates each MD described in the control table, is related to idMDControl
idCapture (2)	Int32	RO	Indicates the number of registered data captures for that MD
supplyType (3)	ST:=Int	RO	Mapping to X73 → Table 5.1: attribute N 10
batteryLevel(4)	Int32	RO	Mapping to X73 → Table 5.1: attribute N 11
remainderBattery (5)	Int32	RO	Mapping to X73 → Table 5.1: attribute N 12
errors (6)	O.String	RO	If it exists, it indicates the OID of the row from specific errors table
ownerMD (7)	O.String	RO	It indicates the patient that the data belongs to
dateTimeData (8)	DT	RO	Indicates the date-time when the MD data was provided

Table 5.10: Medical Device Data Table

ObjectName (Id)	*Syntax*	*Access*	*Description/Mapping Fields*
idMDState(1)	O.String	RO	Indicates each MD described in the control table, is related to idMDControl
idState(2)	Int32	RO	Indicates the number of registered data states for that MD
deviceState (3)	DS:=Int32	RO	Mapping to X73 → Table 5.1: attribute N 14
dateTimeState (4)	DT	RO	Indicates the date-time when the MD data state was provided

Table 5.11: Medical Device State Table

ObjectName (Id)	*Syntax*	*Access*	*Description/Mapping Fields*
idMDError(1)	O.String	RO	Indicates each MD described in the control table, is related to idMDControl
idError (2)	Int32	RO	Indicates the number of registered data captures for that MD
deviceWorking(3)	O.String	RO	If a clinical measurement provided by the MD is out of a predetermined range, this field will indicate that the MD is not working properly. E.g the established range for a temperature measurement is [33-43 °C]. Mapping to → Table 5.1: attribute N 22
sensorWorking(4)	SW:=Int32	RO	Mapping to → Table 5.1: attribute N 23
sensorConnection(5)	SC:=Int32	RO	Mapping to → Table 5.1: attribute N 24
sensorJamming (6)	SJ:=Int32	RO	Mapping to → Table 5.1: attribute N 25
signalFailures (7)	SF:=Int32	RO	Mapping to → Table 5.1 attribute N 26

Table 5.12: Medical Device Errors Table

4) AlarmsTable

This group is used to define a set of thresholds to control the MD and HG performance. The table enables 3 different types of alarms to be configured: up-threshold, down-threshold and absolute value alarms. Therefore, an alarm can be triggered by obtaining a value above an up-threshold, below a down-threshold or because a specific action occurs. For example, an alarm can be generated if the battery level for one MD is under 10% or if the virtual memory used by the HG is over its 85% of capacity or if a MD changes its state to "disconnected". As has been shown, these alarm configurations refer to the fields described in previous groups and their activation will be associated to the execution of an event described in the EventsTable group. This model provides flexibility in the definition of alarms, and it is possible to configure different alarms for each one of the MDs. Each alarm defined in this group will be uniquely identified by the pair idMDAlarm and idAlarm. In this way, independent alarms will be described for each one of the devices. The alarm table and event group design has been based on the alarm and event group from RMON, and their functionality follows the same rules. The main purpose of these tables is to provide a set of simple tools to generate alarms specifically applied to the telemonitoring domain.

5) EventsTable

This group is composed of two tables: ConfigurationEventTable and LogEventTable. The event group supports the definition of actions to be performed by the agent after detecting that an alarm has been triggered. Possible event types to be configured in this table are: (1) log, (2) snmp-trap and (3) log-and trap. A log event implies registering a new entry in the LogEventTable, snmp-trap implies sending a specific trap to the manger and log-and-trap involves performing both actions. Therefore, if an event is logged, a new entry will be created in the LogEventTable. This provides a record of all the alarms that have been triggered for which this event was defined. The event table works in the same way as a RMON control table, so all the objects are read and write. On the other hand, the LogEventTable works as a RMON data table, so all the objects are read only.

6) Managers Table

Finally, this table contains the information of all the managers that control the devices in the HG and for which trap notification is activated. This group gathers information required to send a trap to a specified manager whose personal data is also described there: name, ip address, phone number, working place and permissions.

In order to notify the manager about abnormal changes in the working mode of a HG or the MDs, the MIB provides the agent with 5 different traps as shown in Table 5.13.

Trap Type	*Description*
System Over Charged	This trap will notify an overuse of the resources With this trap information about the HG resources used and the date of the identified event will be sent to the manager
wrongDevice	Working This trap will be used in order to notify a generic failure in any of the devices Only information about the date time will be transferred
newMD	This message informs about a new MD detection
Specific Errors	This trap informs about specific errors in some MDs such as sensor or signal problems Information about all possible errors that could have triggered this alarm will be reported
warning	This is a generic trap that can be used where no specific error is to be reported

Table 5.13: Traps Table

5.3.2 Prototype Development

The SNMPv3-X73proxy agent has been developed using Java technologies. Specifically, the snmp4j framework [186] has been used to implement the SNMPv3 communications required. Figure 5.13 depicts the interface of the SNMP developed agent. The upper part of the interface is used to manage registered managers who are allowed to access data available in the MD-MIB module. The lower part of the interface is used to initialize the MD-MIB, the agent (IP address and port) and to start running it.

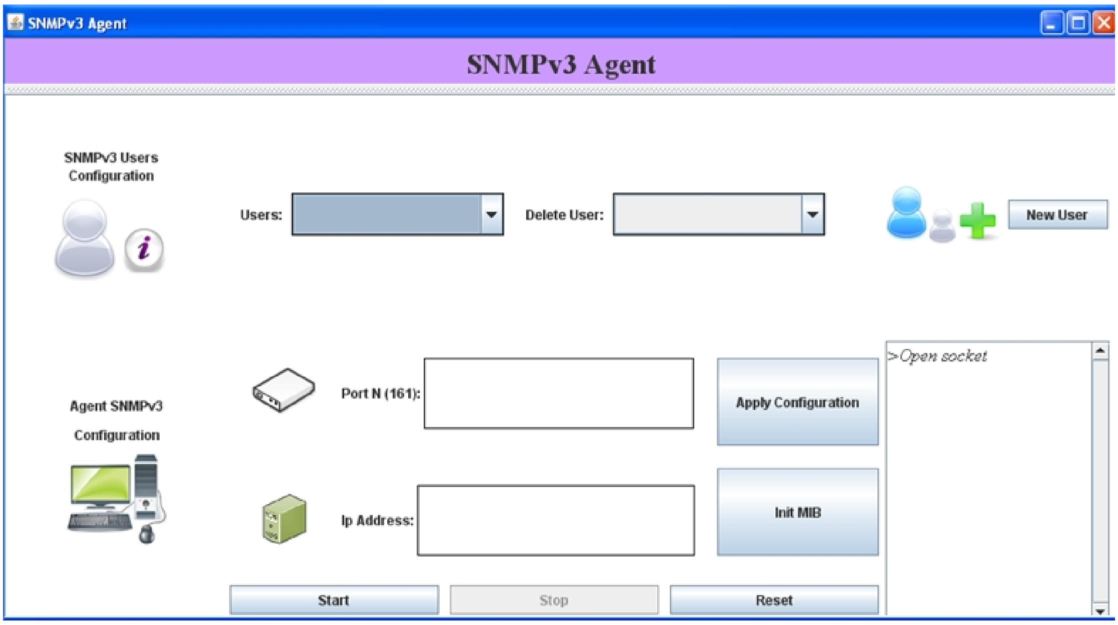

Figure 5.13: SNMP agent interface

The workflow procedure of the SNMP management tool is as follows: first, the EM establishes an SNMP communication in order to review available data (sending Get, GetNext and GetBulk commands) in the MIB-module and configure required alarms (using Set commands). This checking process can be done periodically in order to control when new devices are added to the scenario and in order to review data acquired. On the one hand, information about the HG resources is collected and updated (every 1 minute) in the ComputeEngineControlInfo group. Then, the agent checks if there is any alarm associated to the evaluation of the HG consumption of resources. If there is and if the evaluation result satisfies the conditions to trigger an alarm, an SNMP trap will be transferred to the remote site or a log will be recorded in the Log Table.

On the other hand, when an MD transfers information about its state or about its technical data, new entries are included in the medicalDeviceControlTable (if it is the first time the MD is connected to the HG), the medicalDeviceDataTable (in order to transfer dynamic information e.g battery level) the medicalDeviceStateTable (in order to update information about the MD communication state) and/or the SpecificsErrorsTable (if an error is reported in the sensor connected to it). Alarms are then checked and evaluated in the same way as for the HG resources evaluation.

5.3.3 Evaluation Performance

This SNMP entity (agent) was evaluated, similarly to the HOTMES solution, in terms of memory, computational and network cost. The memory allocated to the agent process and CPU consumed cycles (by percentage) were measured using the Windows

Task Manager. During the tests, the traffic between the two endpoint entities was captured (using the Wireshark software) and processed to analyze the number of packets exchanged in all the actions.

To test the functionality of the agent, it was assumed that a COPD patient using 4 MDs (blood pressure monitor, pulse-oximeter, glucometer and weighing-scale) was located at a home site. The idea was to implement the same management tasks as described for the COPD *technical management profile* with the SNMP tool (see Table 5.6). To achieve this goal, some alarms had to be configured in the MD-MIB. These tests have been performed using a simulated scenario. However, no significant differences are expected in a real scenario. The test scenario included the developed SNMPv3-proxy X73 agent and an SNMPv3 manager in two separate virtual machines (256 MB Windows XP). Virtual Box software was used to hold both virtual machines: the SNMPv3-proxy X73 agent and also an interface developed to simulate X73 manager performance. This interface allowed the simulation of X73 data transfers from different MDs. The second virtual machine contained an MIB Browser from MG-Soft used as a SNMPv3 manager.

Tests performed during this evaluation were divided into two phases: configuration and monitoring. The first phase of the tests involved checking the information provided by the agent (HG information). Also, the MD connections were simulated in order to check that new entries were being created and updated correctly. During this phase, different alarms were also configured. The configuration phase involved a continuous flow of packets between the two entities. In order to configure the required monitoring tasks listed in Table 5.6 with the SNMP solution, 13 different alarms were configured. Four alarms were configured in order to detect when the state of the 4 MDs was 'disconnected', four other alarms were configured to detect when the battery level was under 15% and one alarm was defined in order to control that the blood pressure monitor was working correctly (hence to detect any abnormal out-of-range measurement). Four alarms were then configured to evaluate memory, CPU, bandwidth and storage consumption. It should be noted that configuring one alarm involves 3 steps: 1) a new row must first be added to the ManagerTable in order to configure the IP address where the traps should be sent (if this has not been previously configured), 2) a new event must be created associated with sending a trap to the manager and 3) the alarm should be specified in the Alarms Table. However, the same manager and event can be used to configure an alarm.

This configuration process involves an exchange of 111,83 KB. Eight messages were required to be exchanged between the SNMPv3 agent and the SNMPv3 manager (after completion of the initial authentication phase) in order to modify an object value from the MIB-module (required to configure the desired actions). This involved a total of 15 variable bindings and 1065 bytes (IP data, using security constraints) exchanged between the two entities. For this test, a new row must first be added to the ManagerTable. As can be seen in Table 5.14, this requires modifying at least 4 objects: the status object (twice, in order to create the entry and to validate it), and the owner and IP objects. A total of 4260 bytes are exchanged between the entities. Then, a new event must be created. To completely configure the event entry, a total of 40 messages must be exchanged between the entities. Finally, the desired alarm must be configured, for which 7 objects need to be modified. The defined event is associated to the configured alarm through the idEventValue object field. Therefore, a total of 17,04 KB data (IP data) need to be exchanged to completely configure an alarm. Then, if it is generated, a trap will be sent to the EM. Note that the *max-repetitions* and *non-repeaters* parameters for a GetBulk request were configured as 0 and10 respectively.

New Entry Tables	N objects	N messages	VB	Octets Exchanged
Manager Table	4	32 (2x8)	60 (4x15)	4.260
Events Table	5	40 (5x8)	75 (5x15)	5.325
Alarms Table	7	56 (7x8)	105 (7x15)	7.455
Total	16	120	240	17.040

Table 5.14: Messages for modifying an object

The second phase of the tests was devoted to checking the functionality of the agent in terms of MD data communication, alarm activation and event action execution. During this phase MDs were configured to transfer 20 clinical measurements and each alarm was activated twice (all of them involved sending traps). These tests were run during one hour.

The mean results for different cost metrics are summarized in Table 5.15. The static memory allocated to the agent was 32.500 KB and the dynamic memory increased this value to nearly 48.000 KB for the configuration phase. The maximum increase of memory allocated to the agent was measured for each phase and is depicted in column two of Table 5.6. Regarding computational cost, the CPU consumed cycles (in percentage terms) was less than 3% for all the actions performed. Finally, network usage represents the total amount of KB exchanged between the entities in each phase. During the configuration phase 4,41 KB were exchanged to visualize HG and MD information. Then, during the monitoring phase 42,02 KB were exchanged to visualize all registered MD data. Finally, 1,58 KB were captured during each alarm activation.

The relational database management system MySQL was chosen to support the implementation of the proposed MD-MIB-module, which has an initial size of 30,5 KB. The initial configuration of the database only contains some generic control tables. Eventually, at the time the MDs are connected to the HG, the database size is increased as tables are created and new rows are added in the control tables. Also, the memory for the max 20 entries that can be registered for each MD in the data, state and error tables is reserved. After a simulation with four MDs linked to the HG and 13 alarms configured, the database size was 262 KB.

Evaluation Phase	*Δ Max. Memory Cost (KB)*	*Max. CPU %*	*Network Usage (KB)*	*DDBB (KB)*
(1) Configuration				
HG and MD info	500	1	4,41	97,7
Alarms Conf	14.000	3	111,83	262
(2) Monitoring				
MDs data	16.000	3	42,02	267
Alarm	16.000	2	1,58	273

Table 5.15: SNMP solution: evaluation performance results

5.3.4 Discussion

The results have shown that this management tool does not demand many resources. Computational cost is very low (CPU $< 3\%$) and maximum RAM memory consumption is around 20% of the total RAM available in a device with a limited 576 MB memory. The SNMPv3 agent application requires less than 5 MB, which is a small memory storage cost for a common laptop (nowadays they normally have more than 80 GB). Furthermore, little traffic is required to configure and then monitor the HG. As stated in the introduction, the tool is very effcient for transferring the small quantities of data required for technical management in home-based scenarios.

This technical management may be performed remotely from the healthcare site using, for example, a commercial SNMP manager, the same manager that could be used by technical staff in the hospital for network supervision and control. Nowadays, most leading management software packages used in the network management arena such as HPOpenView, CiscoWorks, Cacti, Zennos or OpManager are also used in hospitals to manage IT (Information Technology) resources (e.g WAN and LAN links, routers, and personal computers) both locally and remotely. Some of these tools, some proprietary and others OpenSource, integrate RRDtool data storage (Round-Robin Database tool) which is a very useful feature for plotting the evolution of managed parameters in a simple manner. All these tools provide mechanisms for network monitoring based on SNMP. Therefore, our proposed solution could be seamlessly integrated in a real healthcare scenario as an additional module to be managed through the management application already in use at the hospital.

The integration architecture presented in this work (X73 plus SNMPv3) takes advantage of the main benefits that both protocols provide in terms of interoperability (elements can communicate and exchange data accurately independently of the software technology implementation) and efficiency for remote technical management. On the one hand, the use of X73 at home sites provides an interoperable and widely accepted solution for MD data transfer in e-health environments. On the other hand, using an architecture for TCP/IP network management offers the advantage of using outcomes to solve interoperability challenges (among others) that researchers have been addressing since the early 1990's in TCP/IP networks.

5.4 Comparative study

This section presents a comparison between both SNMP and HOTMES ontology based solutions in terms of meeting the main challenges presented by the telemonitoring scenario. A general overview of the evaluation points that have been compared and their classification as key strengths or key limitations for this management purpose are depicted in Figure 5.14. Both qualitative and quantitative metrics have been evaluated.

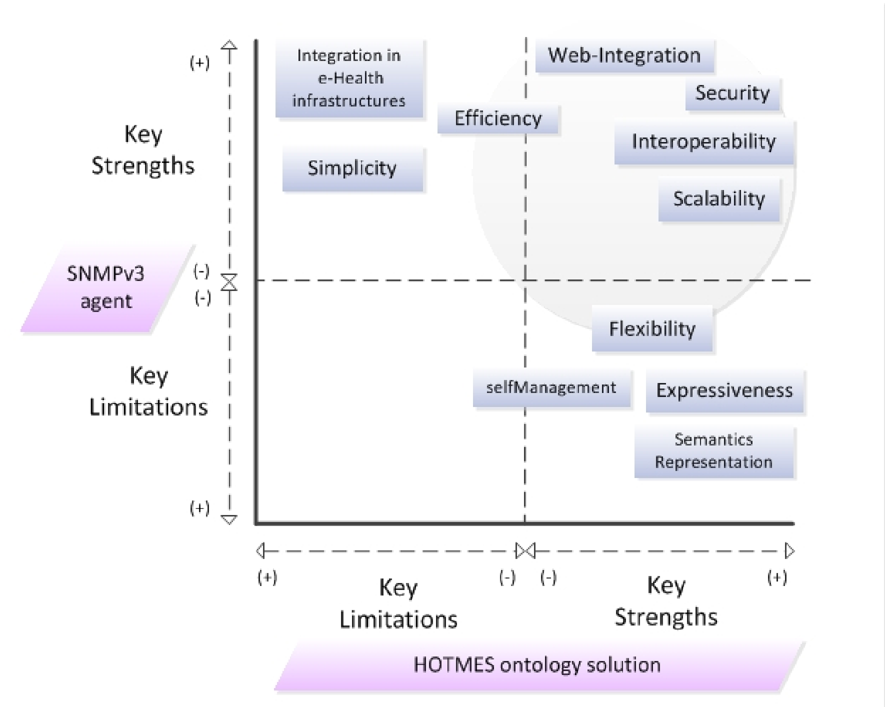

Figure 5.14: Key Strengths and Limitations Comparison

5.4.1 Qualitative Metrics

5.4.1.1 Semantics representation and Interoperability

The HOTMES ontology-based solution offers benefits in terms of modelling managed entities, their technical features and, furthermore, the management procedure (explicitly describing the tasks that comprise the MAPE process and using rules). While SNMP provides a data model representation with a clear organization of monitored data, it lacks semantics representation. Furthermore, management behaviour is not explicitly described. On the other hand, the HOTMES ontology provides knowledge representation. This higher level of abstraction provides more expressiveness and a clear understanding of presented data which eases interoperability. Hence, it facilitates the harmonization process when MDs using different protocols are used together to monitor a patient's health status and makes clear how management tasks (e.g monitoring and evaluation data) should be performed.

5.4.1.2 Integration in e-Health infrastructures

There is another important integration issue that should be considered. Apart from integrating the management of heterogeneous devices, the management modules should themselves be integrated in hospital communication management networks. Since most common software applications used in hospitals to manage their WAN and LAN communication networks such as Cacti, Zennos or proprietary management tools include SNMP management, there is no doubt this solution would fit better in e-health infrastructures. Nevertheless, it should be noted that ontologies are not as mature a technology as SNMP, and neither is the development of ontology applications. Hence this could be a future issue to investigate further as research into the implementation of ontology solutions advances.

5.4.1.3 Web integration

A different integration solution for both management solutions (SNMP and HOTMES) is to develop a web-manager. Including the remote SNMP manager in a web environment or providing access to the web service through a web browser would make the management of the devices easier for the EM and would also enhance its integration in the management scenario. In fact, an SNMP web-manager has been developed in order to provide the EM with a means to access SNMP information in a user-friendly way and enhance its remote control from every location [187]. As shown in Figure 5.15, the communication between the SNMP agent and the SNMP manager located in the remote health care site is effected by means of SNMPv3 communications (SNMP solution). The communication between the web manager and the EM is established by means of a commercial web browser and HTTPS communication. Hence, this web manager comprises two parts: the SNMP manager entity and the web module. On the one hand, as a traditional SNMP manager, this entity includes modules to make requests to the remote agent as well as modules to process asynchronous messages (thus traps) sent by the agent. On the other hand, this SNMP manager is integrated in a web environment which was developed using the MVC (Model View Controller) programming paradigm and the Struts2 framework [188]. The web manager includes graphical tools to depict information stored in the MD-MIB module, a complete environment for alarm configuration and an AJAX (Asynchronous JavaScript And XML) module for reporting traps. As shown in Figure 5.15, this solution could be extended to integrate the HOTMES solution into this web environment. In a similar way, the web application would provide an intuitive graphical interface and make easier the *technical management profile* configuration (and rules) as well as the visualization of the information contained in the ontology. The communication between the home-site and the remote site would be established by means of HTTPS and the web service technologies. Then, as in the case of the SNMP solution, a connection would be established between the HOTMES manager and the MVC web module. Furthermore, both solutions could be integrated together in the same management scenario. This option would allow the EM to choose the best management option (HOTMES vs SNMP) according to the scenario conditions and thus exploit the advantages that each one offers.

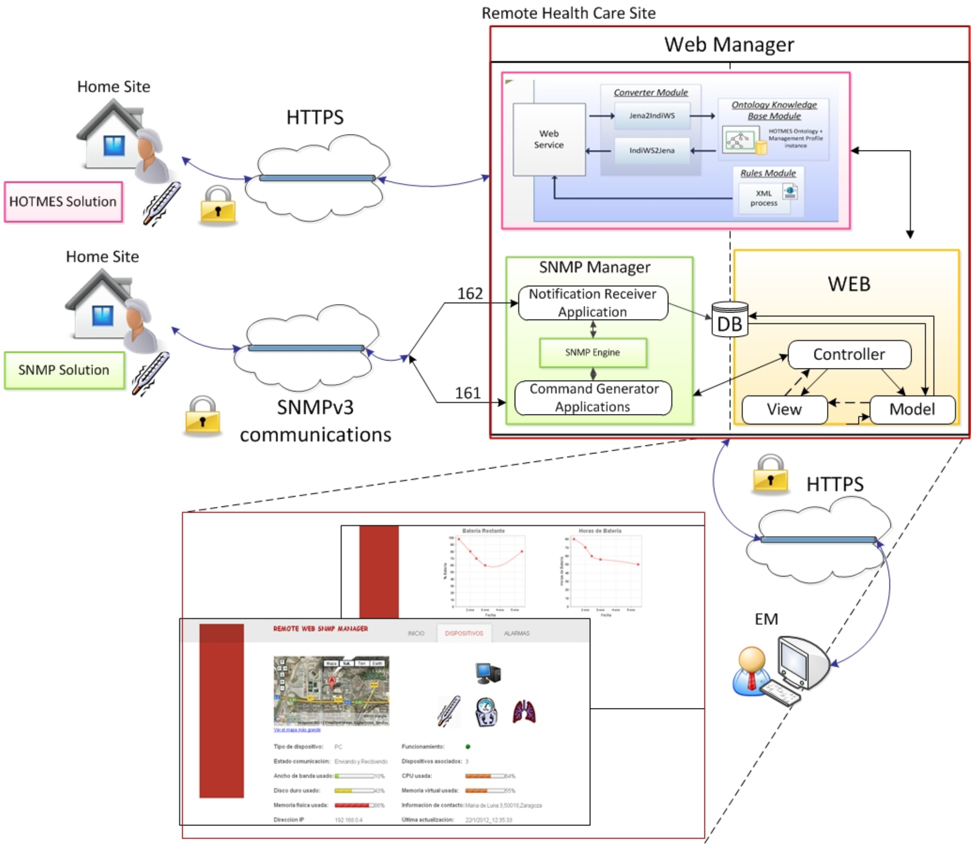

Figure 5.15: SNMP Web Manager

5.4.1.4 Expressiveness and Flexibility

The HOTMES ontology solution offers more flexibility to express management tasks, and to define alarms and data evaluation processes. Although traditionally SNMP has been used to monitor the state of a resource, the MD-MIB module design enhances these management capabilities by including alarms and event tables. However, while with the SNMP solution only alarms that detect upper or lower thresholds or specific values can be configured, with the combination of the rules and the ontology model, more complex evaluation processes and alarms can be defined. For example, it is possible to configure the evaluation of both the HG memory and CPU consumption conditioned to detect a number of linked MDs, e.g over 4. In the same way, not only notification events can be configured (SNMP solution) when an abnormal performance is detected. The ontology model offers more expressiveness to configure different types of execution tasks. For example, different planning tasks can be configured such as performing a second analysis with other managed resources.

5.4.1.5 Self-Management Capabilities

The SNMP solution provides a traditional agent-manager structure where the communication between the entities is mainly based on periodically polling the agent, although asynchronous messages can also be sent by the latter. In contrast, the HOTMES ontology solution is based on the definition of *technical management profiles* that contain all the information that is required to manage the HG and the MDs. It should be noted that this management procedure differs from SNMP performance. As it is based on running tasks from a profile and it is the HG which first establishes the communication, some kind of self-management capabilities are provided to the HG. This fact could be seen as an advantage in e-health scenarios because more independence is provided. Nevertheless, it implies the drawback of reducing the remote manager's control over monitoring.

5.4.1.6 Scalability

Both solutions are good in terms of scalability. Both the ontology and the MIB-module could be easily extended if managing more technical data is required. Nevertheless, the HOTMES ontology offers a higher level of abstraction describing generic management tasks and thus the managed data. This solution could be easily extended to integrate more types of management at the home site running over the same agent prototype without requiring complex modifications. In fact, this has been demonstrated in chapters 4 and 5 where extending the HOTMES ontology both for clinical and technical purposes is described. Furthermore, this HOTMES solution could be applied to other types of management such as for fitness and exercise control.

5.4.1.7 Simplicity

Working with ontologies and rules is not as simple as working with the SNMP architecture. This limitation leads to more complex *technical management profile* configuration than alarm definition in the MIB-module. However, by developing a GUI interface that would map this interaction into simple queries, this limitation could be solved.

5.4.1.8 Security

There are no significant differences in the security level that both solutions provide. On the one hand, version 3 of SNMP was selected and thus authentication, encryption and access control functions were implemented. On the other hand, for this particular solution using WS, TLSv1 (Transport Layer Security) protocol over HTTP was used between the web end points and thus integrity and reliability of exchanged data were guaranteed. Also authentication was guaranteed by using both server and client X.509

certificates. In addition, the HMAC code included in each request allows data integrity to be checked. At the application layer, both proposals make use of similar security constraints to guarantee confidentiality and authentication. Regarding the transport layer, it could be said that the SNMP solution is not as reliable as the HOTMES solution. While SNMP relies on UDP as its transport layer, HTTP requests (used by the WS) rely on TCP. TCP offers robust errors-recovery among other features. Therefore, for this second option it is guaranteed that if an error occurs during the transmission, the data will be retransmitted and both end sites will be aware if the communication fails. Note that SNMP can also be transported via TCP. Nevertheless, the unreliable UDP solution requires low overheads and the impact on the network is reduced. The unreliable nature of UDP is not a real problem as the SNMP manager, by using a timeout, can determine if datagrams are lost and if it is desirable to repeat the request. The worst case would be for sending traps.

5.4.2 Quantitative Metrics

The COPD technical management tasks (summarized in Table 5.6 and discussed in sections 5.2.3.3 and 5.3.3) were considered in order to compare both solutions. Quantitative metrics involved in the comparison comprise network cost (both for the configuration and monitoring phase in terms of KB exchanged), storage cost, memory cost (measured in terms of increments in dynamic memory, Δ) and CPU cost.

Note that for measuring the differences in terms of memory and CPU cost, the HOTMES solution was installed in the same scenario as the SNMP solution (Virtual Machines using Windows XP, 256 MB). It should also be noted that the HOTMES solution was tested in section 5.2.3.3 using different software, so some implementation-dependent metrics vary from those discussed there. The SNMP solution provided better results in terms of resources usage, and thus in effciency. Nevertheless, the ontology based solution does not consume many resources either. A graphical view of the quantitative metrics comparison is shown in Figure 5.16. The scale colour bars show a graphical estimation for each solution in terms of the consumption of resources, considering other applications running in the device and the available resources in the evaluation device. The numerical values obtained during the test performance are summarized in Table 5.16.

5.4.2.1 Network Cost

As is shown in Table 5.16, 105 objects need to be modified in order to configure the desired alarms. Note that only one entry was created in the Manager Table and two events were configured (send an overused resources trap and a general warning trap) associated to all the alarms using the SNMP-based solution. Hence, 111,83 KB are exchanged between the SNMPv3 agent and a commercial SNMP manger during the configuration phase.

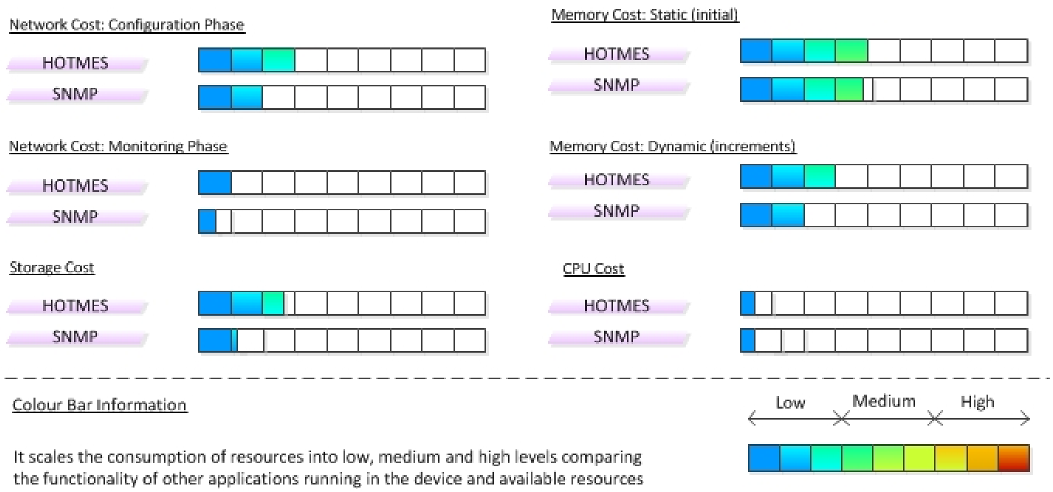

Figure 5.16: Qualitative metrics colour bar comparison

In contrast, just two request messages were necessary to download the technical management profile and the rules from the web service, involving an exchange of 174,11 (9,71 + 164,40) KB (IP data). If the ontology is required, then 243,61 KB is exchanged. On the other hand, 83 instances need to be configured to complete the *technical management profile*. Regarding the monitoring phase, 5,2 KB are required on average to send a data result from the client to the WS in the HOTMES solution. In SNMP, 1,58 KB are required to send a trap in order to inform about an alarm. Information about the devices and stored in the tables from the MD-MIB module can be checked by the EM at any time. To visualize data contained in this table requires an exchange of 42 KB. According to these results (shown in Table 5.16), the SNMP solution seems to be more efficient in terms of KB exchanged. However, the configuration process is much faster as the *technical management profile* has been previously configured and the interaction between the end sites requires less time.

5.4.2.2 Storage Cost

To hold the ontology and the *technical management profile* requires more storage resources than the implementation of the MIB-module. After executing the configuration tasks, the MIB-module database size was 262 KB. After 1 hour running during the monitoring evaluation phase (and executing the tasks described above), this value increased to 273 KB. On the other hand, the TDB used to hold the ontology and the *technical management profile* was initially 988 KB in size. Then, after 1 hour of agent performance, the database size was 2,46 MB. According to these results, the SNMP MIB module seems to require less storage resources than the HOTMES solution. However, these results depend on the database selected and the internal structure design used to hold tables on it.

HOTMES ontology-based solution	*SNMP-proxy-X73 solution*
Network cost : Configuration phase	
N Individuals: 83 N Rules: 11	N Objects: 105
Download Ontology: 69,50 KB Download TMProfile: 164,40 KB Download Rules: 9,71 KB	Alarms configuration: 111,83 KB
Network cost : Monitoring phase	
Send Result (Data): 5,2 KB (one single piece of data) Send Result (Alarm): 5,2 KB	Data exchange to visualize tables: 42 KB (all the tables) Trap: 1,58 KB
Storage Cost	
Initial database size: 988 KB Final database size: 2,46 MB	Initial database size: 262 KB Final database size: 273 KB
Memory Cost	
Initial memory: 36.800 KB Δ max memory (configuration): 17.292 KB Δ max memory (monitoring): 25.300 KB	Initial Memory: 32.500 KB Δ max memory (configuration): 14.000 KB Δ max memory (monitoring): 16.000 KB
CPU	
Max CPU % (configuration): 3 % Max CPU % (monitoring): 0.8 %	Max CPU % (configuration): 3% Max CPU % (monitoring): 2%

Table 5.16: Quantitative Metrics Comparison

5.4.2.3 Memory and CPU Cost

The memory metric greatly depends on the implementation of the solutions. More static memory is required to launch the Semantic Autonomic Agent (36.800 KB for the HOTMES solution vs 32.500 KB for the SNMP solution). Furthermore, higher increments in dynamic memory are required for the HOTMES agent once it starts working (see Table 5.16). Regarding the CPU cost consumption, there were no major differences worth mentioning. Both solutions require higher CPU for the configuration process and less for the monitoring phase.

Note that the HOTMES implementation uses JENA and loads the ontology from the TDB in the memory to work with it. However, a particular implementation of the MIB-module has been carried out where individual connections to the MySQL database are made to access information stored there. This implementation difference may cause differences in memory and CPU consumption during the monitoring phase.

5.4.3 Discussion: Overall view

Remote technical management of MDs and the HG is an important task to be considered within telemonitoring scenarios. In order to address this new challenge, two management proposals based on network management field techniques have been presented and compared in this chapter.

The SNMP solution offers considerable power and efficiency for transferring small amounts of data and its information model structure provides a clear organization of monitored data. Using this architecture for our technical management purpose would provide substantial benefits in terms of integration into healthcare scenarios, network costs and interoperable solutions, offering an easy and scalable development of the new data model and a simple way of managing the devices operating in the telemonitoring scenario.

On the other hand, the HOTMES ontology-based solution provides a knowledge model representation with richer semantics. It offers benefits in terms of flexibility, expressiveness, interoperability and scalability. The developed ontology does not depend on the implementation or method used to transfer data and, more importantly, it is in line with current e-health trends and follows new tendencies in management solutions. Furthermore, it can be used in combination with the clinical management solution described in chapter 4, and can thus be integrated in the same architecture.

Chapter 6

Discussion and Conclusions

The underlying hypotheses and research objectives presented in the first chapter have been raised and discussed during this thesis. Integration, management, context-awareness and personalization are fundamental issues in home-based telemonitoring scenarios and the main challenges addressed in this investigation. In this last chapter, the main outcomes are discussed and final conclusions presented in order to set the contributions of this work within the context of the big e-health picture. Finally, future challenges are enumerated in this chapter. This is a growing research field where new challenges arise as society evolves.

6.1 Research Objectives Achieved

The first chapter of this thesis presented evidence of the main benefits that telemedicine systems in general and home-based telemonitoring systems in particular can provide to patients and healthcare organization as a whole. Then, the challenges and critical issues involved were set out. Various research questions were raised ranging from the general issues that should be taken into account in order to set-up a telemedicine system to the particular issues to be addressed during each phase of the set-up (see Figure 1.1). In particular, this thesis contributes to providing an answer to the main research question identified in chapter 1: how the development of an architecture system based on knowledge-oriented technologies can be used to successfully address the main challenges presented by the telemonitoring scenario in terms of remote clinical and technical data management and data integration. In support of this research goal and to answer this statement, this thesis has met the research objectives presented in chapter 1, section 1.3, as discussed below.

Research objective 1: *To conduct reviews on the state of the art in general aspects of ontology design, languages and implementation. Specifically, to study the application of ontologies in the clinical sphere and to examine other solutions proposed in telemonitoring scenarios. To study the potential of ontologies for representing knowledge and addressing integration and interoperability challenges.*

The purpose of the first objective was to review ontology-based solutions in the e-health domain, review and study ontology languages, study their potential to represent knowledge in the telemonitoring scenario and explore the possibilities of reasoning over them by using rules in order to provide personalized services. This was approached in chapter 2 which provides a review of the state of the art regarding ontologies in e-health followed by an explanation of ontology languages, the OWL language features in particular, and related issues such as reasoning and ontology evaluation. Throughout these studies it was understood that the OWL language could be used to express required knowledge in the telemonitoring scenario and it was concluded that the SPARQL language could be used effectively to express rules over the content expressed in the ontology. Furthermore, reviews on reasoning and ontology evaluation led to the definition of the modules required in the architecture system aiming to exploit the advantages offered by ontologies, leading to the overall design of the methodology as proposed in chapter 3. This methodology was examined in chapters 4 and 5 with a view to adapting the ontology development and implementation to its particular purpose.

In fact, chapter 2 provides a description of all the elements involved in the proposed architecture solution and their combination. Not only are ontology features analyzed in this chapter, but also standards in e-health, the SNMP architecture, the autonomic computing paradigm and web services technologies. In summary, a general and particular view is offered of how the combination of all these elements could provide a successful solution to meet the challenges presented in the telemonitoring scenario.

Research objective 2: *To design and develop an end-to-end telemonitoring architecture taking into account both conceptual issues (thus the domain modelling) and the transference of data between the end points (thus communication and message exchange). Hence the architecture should include two layers: a conceptual layer and a data and communication layer.*

To achieve this second objective, two sub-objectives were addressed. First it was required to design and develop an ontology to support the conceptual layer of the proposed architecture in order to provide data and management integration in general home-based scenarios. This research task led to the study of the autonomic computing paradigm and its combination with ontologies which was addressed in chapter 2. As an outcome of these research tasks, the HOTMES ontology was designed and presented in chapter 3. Its combination with SPARQL rules is presented to provide a flexible solution to personalise management tasks. Furthermore, the methodology depicted in Figure 3.2 was proposed as an outcome of this research, comprising 3 simple stages to adjust and apply the HOTMES ontology to different management purposes.

To achieve the second sub-objective, the use of WS was investigated to support the exchange of information defined in the conceptual layer of the architecture. WSs were studied in chapter 2. Based on these studies, an architecture based on a client-server pair was proposed in chapter 3. The solution presented in chapter 3 offers a generic ontology-based solution to integrate data and its management procedure in home-based scenarios. As an outcome of these studies, an innovative REST- inspired architecture was designed that allows the exchange of information contained in an ontology in a generic manner. This approach could easily be used in other applications involving remote management tasks at home sites. Specifically, the use of this architecture is been examined in chapters 4 and 5 as a support for both technical and clinical services in the telemonitoring scenario. Its layer structure and its communication method provide the approach with scalability and re-usability features.

This approach of modelling the management procedure execution with an ontology and the idea of using it as a general framework to design individual *management profiles* constitute the key differences in contrast to other ontology-based solutions for home-based scenarios.

Research objective 3: *To investigate and adapt the ontology-based architecture design to provide clinical management in the home-based telemonitoring scenario*

One of the main research questions posed in chapter 1 was how knowledge-oriented techniques could be used to provide personalized services according to patients' chronic conditions. Furthermore, how could they be used to integrate data provided by all the sources involved in the telemonitoring process and how could remote management be achieved Theses issues were mainly approached in chapter 4. Ontologies provide a high level of abstraction and expressiveness to define concepts and knowledge with precision. This facilitates data integration in scenarios such as telemonitoring where diverse heterogeneous sources are involved in the same process. Following the simple three-stage methodology presented in chapter 3 (see Figure 3.2), the proposed HOTMES-based architecture is adapted to clinical purposes. Thus, data and management integration for context-aware and personalised monitoring services for patients with chronic conditions is addressed. This research implied a multi-disciplinary collaboration where clinicians had an essential role in defining both the ontology as well as *patient profiles*. Furthermore, reviews on current standards in e-health scenarios were conducted in order to address integration challenges. As an outcome of this chapter, the HOTMES ontology was adapted to the clinical management purpose and 16 clinical *patient profiles* were designed to validate the proposed approach. The HOTMES clinical ontology includes descriptions of the information to be monitored in order to control the state of a patient. This information was classified into 4 data categories: vital constant measurements, environmental information, quantitative information and qualitative information. The vital constant measurements model described in the ontology includes information gathered by a MD and additional information that should be provided by the patient in order to set the context of the measurement and make a positive assessment.

The *patient profiles* described in chapter 4 provide a simple guide to evaluate the patient's conditions that alone or in combination with other conditions at a certain time lead to the conclusion that something is going wrong with the patient's health. These profiles were designed according to clinical guidelines aiming to monitor types of diseases of interest at the home site. Specifically, 11 chronic *patient profiles* were configured and then 5 multi-chronic profiles were designed in order to demonstrate that the ontology solution could be used to monitor any type of patient disease.

Apart from the ontology extension, the WS architecture was adapted to support this type of management. Specifically, the modules used to integrate data coming from the different sources at the home site (patients and MDs) were adapted to create new ontology data instances by combining data provided by the ISO/IEEE 11073 standard and contextual information provided by patients.

In summary, following the 3 stage approach proposed in chapter 3, the main challenges of integration and personalised medical care have been met through the ontology development and evaluation from engineering and clinical points of view, both theoretical and practical.

Research objective 4: *To address the technical management challenge in home-based telemonitoring scenarios studying two different solutions: an ontology-based solution and an SNMP based solution*

An additional challenge present in health scenarios and identified in chapter 1 is technical management. To guarantee that MDs and HG are working correctly is an important task in telemonitoring scenarios. Throughout chapter 5, research objective 4 is mainly addressed by extending the HOTMES solution to this type of management. Specifically, this chapter explores and compares how this ontology-based solution (which combines the autonomic computing paradigm and WS) and a solution based on the most popular TCP/IP management architecture, SNMP, can be effectively used to address this technical challenge.

Initially, the X73 and the SNMP architecture were reviewed to inspire the ontology model design covering MD technical features and generic network resource-monitored specifications. On the one hand, following the three stages presented in chapter 3 (see Figure 3.2), section 5.2 describes how the proposed HOTMES-based architecture is adapted to technical management in home-based scenarios. Hence, it demonstrates the application and re-usability of the HOTMES solution for technical purposes. New classes were added in the ontology, complex and simple management tasks were configured to test it in the application domain, and finally the prototype was adapted to integrate the technical data.

On the other hand, the SNMP solution was specifically adapted to the X73 standard. The integration architecture presented in this work (X73+SNMPv3) takes advantage of the main benefits that both protocols provide in terms of interoperability (elements can communicate and exchange data accurately independently of the software technology implementation) and efficiency for remote technical management. On the one hand, the use of X73 at the home site provides an interoperable and widely accepted solution for MD data transfer in e-health environments. On the other hand, using the SNMP architecture offers the advantage of using outcomes to solve interoperability challenges (among others) that researchers have been addressing since the early 1990's in TCP/IP networks.

Finally, both proposals and their characteristics were compared in terms of efficiency, interoperability, security, complexity and semantic expressiveness. This study led us to conclude that the SNMP solution provides a powerful benefit in terms of integration into healthcare scenarios, scalability and network cost. On the other hand, the HOTMES ontology-based solution provides advantages in terms of flexibility, expressiveness, interoperability and scalability. Although nowadays an SNMP solution would be easy to integrate and deploy in real scenarios, the HOTMES ontology solution is in line with current e-health trends and follows new tendencies in management solutions.

Research objective 5: *To develop a prototype and evaluate its efficiency for supporting clinical and technical management tasks in telemonitoring scenarios. Specifically, to implement its application to X73 and SNMP standards.*

This objective was addressed in chapters 3, 4 and 5. The design and structure of the HOTMES solution was oriented to support both types of management on the same system architecture. Then, the prototype development was addressed at the third stage of the three simple stages presented in chapter 3 (see Figure 3.2) for both clinical and technical solutions. This prototype was named the Semantic Autonomic Agent.

Furthermore, evaluations were carried out in order to test the performance of the prototype in terms of memory, CPU consumption, execution time and bandwidth required. Although there is no standard criterion to evaluate the performance of an ontology- based application, these metrics are currently being used in the state of the art to evaluate performance in similar prototypes. The evaluation results led us to conclude that the semantic agent does not consume many resources and that it could be installed in a resource-constrained device such as a mobile device.

6.2 Contributions and Conclusions

The general approach of this thesis was to research and make contributions in the field of home-based telemonitoring scenarios. Specifically, the focal aim of this research was to investigate the application of knowledge oriented frameworks based on ontologies to address the main challenges presented in the first chapter: clinical data integration and clinical management, the provision of personalised care services taking into account the patient's context, and integrated technical management of all the devices that comprise the telemonitoring scenario. This focal aim has been covered and the overall objectives have been fulfilled. The major contribution of this thesis is the proposed generic system architecture design based on the combination of the HOTMES or extended HOTMES ontology (conceptual level) and WS technologies (data and communication layer) to support both clinical and technical remote management in home-based telemonitoring scenarios. The combination of ontologies and the autonomic computing paradigm is an interesting approach that takes advantage of the main benefits that both solutions offer in terms of knowledge representation, workflow organization and self-management capabilities. The potential application of this architecture was demonstrated by studying its application for managing and transferring clinical and technical data.

Taking into account MD interoperability, clinical knowledge and engineering issues, the HOTMES ontology was extended following a proposed methodology based on three simple stages. The HOTMES clinical ontology describes the definition of a generic patient profile not limited to monitoring patients with just one particular chronic condition. It provides the advantage of creating, as instances of the *patient profile* class, monitoring profiles to monitor patients with different conditions and co-morbidities, as demonstrated in the application-based study phase. Hence, the design structure and the use of SPARQL rules provide a good and flexible solution for personalised monitoring services. In addition, our ontology-based approach was designed to represent not only data modelling but also modelling of the management procedure execution, that is to say, the workflow that dictates the management service and the automation of the patient care tasks. The application of the MAPE paradigm to the remote management of patients provides a simple, clear and easily understandable way of controlling and evaluating patients' clinical data, and its combination with ontologies constitutes an innovative application in the healthcare domain.

Regarding technical management, two contributions have been presented in this thesis. On the one hand, the HOTMES ontology-based solution was applied to this type of management providing self-management capabilities to the HG. This last capability implies that the EM does not need to poll periodically the HG to control it, thus mobility and independence are enhanced while management tasks are reduced for the EM. On the other hand, an SNMP agent together with its MIB-module were designed to provide a means to access remote MDs and provide HG management based on TCP/IP traditional management techniques.

The underlying hypotheses set out in chapter 1 have been proven. First, the need has been demonstrated for greater investment in home-based telemonitoring system development since these systems allow patients to be remotely monitored and guided by their physicians (by the configuration of *patient profiles*) while being able to manage their own condition in any place at any time. Second, it has been proven that ontologies can be successfully used to provide clear descriptions of managed data (both clinical and technical) and also of the way of managing that information (by describing the MAPE tasks). Third, it has been proved that by the incorporation of individual rules in combination with the knowledge expressed in the ontology, individual and personalised tasks can be provided. This provides great flexibility to the solution. No two patients are the same and thus individual and personalised management guidelines should be provided for each one of them. Finally, it has been proved that the same knowledge-oriented architecture could be used to support technical management thus offering both services running on the same engine. Furthermore, it has been shown that traditional TCP/IP management architectures also offer a good solution for remotely managing the working state of MDs.

6.3 Future Work

Going back to chapter 1 and the methodology depicted in Figure 1.1, it seems clear that after solving these technical challenges, further research steps remain to be taken before integrated solutions for viable long-term telemedicine systems can be made generally available. Furthermore, the way ahead is not straightforward. Multiple choices have be explored to solve new problems arising while others are solved. The methodology presented in this thesis is focused on the technical phase. Specifically, the proposed architecture is a first attempt to provide an integrated solution for remote clinical and technical management services. While a knowledge-oriented solution to link the heterogeneous sources involved in the telemonitoring process has been found, new challenges for future research opportunities have also been identified.

1. ***Medical Devices and Electronic Health Record Integration***: the HOTMES solution aims to provide a means to communicate the home-site and the remote health-care site. Nevertheless, at both end sites of the architecture work remains to be done to integrate data provided by the different data sources. This information should be mapped as instances of the developed ontology, as proposed in this thesis, with the adaptation of the X73 standard. Other modules should be developed in order to integrate data coming from MDs that use other proprietary or standardised protocols. A general mapping method could even be deployed to make this mapping process easy. Furthermore, the information monitored should be integrated in the patient's EHR. Hence, additional research should be conducted in

order to map knowledge expressed in the ontology with standards used to transfer information from patient's EHRs, such as 13606, OpenEHR or HL7. As shown in Figure 6.1 , the HOTMES module would act as the intermediate key piece able to relate different standards used at both end sites, thus avoiding one to one relations.

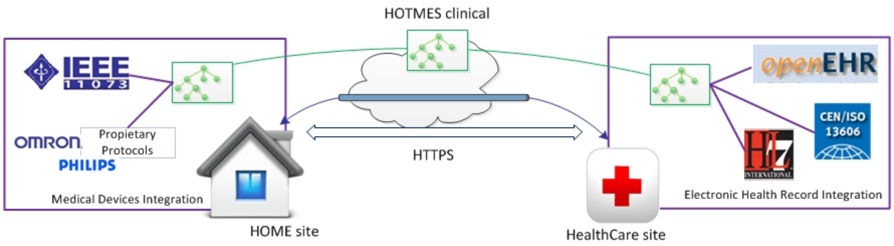

Figure 6.1: Integration of data models at both end sites

2. ***Physician-Ontology Interaction***: working with ontologies and rules is not an easy task and engineers may have difficulty in quickly deploying new *management profiles*. This problem is even greater for physicians who need to be able to configure patient profiles easily. This issue clearly requires the development of a software tool to map the concepts expressed in the HOTMES ontology into images and simple questions easily understood by users. This tool could subsequently be used to manage acquired information, and it should be evaluated and tested in a clinical scenario. This research point could be extended to the general interaction of physicians with ontologies. As long as ontologies are eventually used in the medical domain, the development of new software tools to deal with their adaptation to the clinical scenario and clinical users will be necessary. Note that this research should be done not only for ontology interaction but also for the configuration of rules.

3. ***Enhancement of clinical evaluation complexity***: the *patient profile* that can be designed with the HOTMES clinical ontology allows control of the patient's health status by performing simple evaluation analyses. While more complex analysis tasks could be configured in the ontology (as shown in chapter 5 dealing with technical management), as stated in chapter 4, an interesting future subject of research would be to find out how this model can be used to evaluate a deterioration in health that could lead to triggering an alarm. This would involve extracting important conclusions not through a measurement value itself but from its evolution.

This research has addressed technical issues concerning integration and seamless communication within a home-based telemonitoring system. However, further research should be undertaken with a view to setting up a real system considering the interaction of technical, clinical and social issues. Once the technical problems have been solved, the next phase according to Figure 1.1 is a clinical viability study. The objective of this study would be to demonstrate that the same or better results can be obtained for patient monitoring and care than that obtained without using the home-based telemonitoring system. This study should be carried out individually for each type of chronic patient in order to define real candidates to be monitored at home and discard those for whom no better outcomes are achieved. Addressing the social aspects, the impact on physicians, patients and the healthcare organization should be considered and measured because, as stated in the first chapter, these aspects can strongly determine the long-term implementation of a real system.

At the beginning of this thesis, it was asserted that the guarantee of technical viability is a necessary but not sufficient condition for success in the long-term implementation of a telemedicine system. The contribution of this research thesis aims to bring integrated home-based telemonitoring systems closer to reality. This would represent a further step in facing the incidence and prevalence of chronic diseases that affect an ageing society and consequently impose increasing demands on sanitary services.

Bibliography

[1] European Chronic Disease Alliance Policy Report. A Unified Prevention Approach, 2010. [Available online] `http://www.idf.org/webdata/docs/idf-europe/Chronic-disease-alliance-Final.pdf`.

[2] United Nations (UN). World Population Prospects: The 2010 Revision, 2010. [Available online] `http://www.un.org/esa/population/`.

[3] G. Eysenbach. What is e-health? *J Med Internet Res*, 3(2):e20, 2001.

[4] W.A. Hig, M.S. Houston, S.D. Calobrini, L.A. Drage, and M.T. McEvoy. Assessment of the accuracy of low-cost store-and forward teledermatology consultation. *J. Am. Acad. Dermatol.*, 42 (5 (Pt 1)):776–783, 2000.

[5] J. Cuadros and G. Bresnick. EyePACS: An Adaptable Telemedicine System for Diabetic Retinopathy Screening. *J Diabetes Sci Technol*, 3(3):509–516, 2009.

[6] M. Lewin, C. Xu, M. Jordan, H. Borchers, C. Ayton, D. Wilbert, and S. Melzer. Accuracy of paediatric echocardiographic transmission via telemedicine. *J Telemed Telecare*, 12:416–421, 2006.

[7] N. Maglaveras et al. The citizen health system (chs): A modular medical contact center providing quality telemedicine services. *IEEE Trans Inf Technol Biomed*, pages 353–362, 2005.

[8] J. Basilakis, N.H. Lovel, and B.G. Celler. A decision support architecture for telecare patient management of chronic and complex disease. *Proc. in Engineering in Medicine and Biology Society Conference, EMBS*, pages 4335–4338, 2007.

[9] J. Luftman. Assessing business-it alignment maturity. *Comm Assoc Inform Syst*, 4(14):1–49, 2000.

[10] R. Van de Wetering, R. Batenburg, J. Versendaal, R. Ledermanand, and L Firth. A balanced evaluation perspective: picture archiving and communication system impacts on hospital workflow. *J Digit Imaging*, 19 (Suppl. 1):10–17, 2006.

[11] M. Bush, A.L Lederer, X. Li, J. Palmisano, and S. Rao. The alignment of information systems with organizational objectives and strategies in health care. *Int J Med Informat*, 78 (7)):446–456, 2009.

[12] R.L. Wears and M. Berg. Computer technology and clinical work: still waiting for godot. *JAMA*, 293 (10):1261–1263, 2005.

[13] A.G. Ekeland, N. Bowes, and A.S.A. Flottorp. Effectiveness of telemedicine: a systematic review of reviews. *Int J Med Informat*, 79:736–771, 2010.

[14] MethoTelemed Web Site, 2010. [Available online] `http://www.telemed.no/methotelemed`.

[15] K. Kidholm, A. Bowes, S. Dyrehauge, A.G. Ekeland, S.A. Flottorp, L.K. Jensen, C.D. Pedersen, and J. Rasmussen. Mast model for assessment of telemedicine, 2010. [Available online] `http://www.telemed.no/methotelemed`.

[16] Universidad Politécnica de Madrid (Grupo de Bioingeniería y Telemedicina). *Bases metodológicas para evaluar la viabilidad y el impacto de proyectos de telemedicinaMethodologies*. 2001.

[17] C. Campos, E. Caudevilla, A. Alesanco, N. Lasierra, O. Martinez, J. Fernández, and J. García. Setting up a telemedicine service for remote real-time video-EEG consultation in La Rioja (Spain). *Int J Med Informat*, 81(6):404–414, 2012.

[18] N. Lasierra, A. Alesanco, Y. Gilaberte, R. Magallón, and J. García. Lessons learned after a three-year store and forward teledermatology experience using internet: Strengths and limitations. *Int J Med Informat*, 81:332–343, 2012.

[19] R. Van de Wetering and R. Batenburg. A PACS maturity model: A systematic meta-analytic review on maturation and evolvability of PACS in the hospital enterprise. *Int J Med Informatics*, 78:127–140, 2009.

[20] World Health Organization. *WHO Quality of Care: A process for making strategic choices in health systems*, 2006.

[21] WHO, World Health Organization. [available online] `http://www.euro.who.int/en/home`.

[22] J. L. Monteagudo and O. Moreno. E-health for patient empowerment in Europe. *World Wide Web electronic publication*, 2009. [Available online] `http://ec.europa.eu/informationsociety/newsroom/cf/itemdetail.cfm?itemid=3448`.

[23] M. Brito, L. Vale, P. Carvalho, and J. Henriques. A sensor middleware for integration of heterogeneous medical device. *Proc. in Engineering in Medicine and Biology Society Conference, EMBS*, 2010.

[24] D. Zhang, Z. Yu, and C.Y. Chin. Context-aware infrastructure for personalized healthcare. *Stud. Health Technol. Inform*, 117:154–163, 2005.

[25] N. Saranummi. It applications for pervasive, personal, and personalized health. *IEEE Trans Inform Tech Biomed*, 12:1:1–4, 2008.

[26] POCT1-A2, Point-of Care Connectivity: Approved Standard-Second Edition.

[27] ISO/IEEE11073 - Personal Health Devices standard (X73-PHD). Health informatics. [P11073 - 104xx. Device specializations] [P11073-20601. Application profile - Optimized exchange protocol], 2009. [Available online] `http://standards.ieee.org/`.

[28] Integrating the Healthcare Enterprise [IHE]. [Available online] `http://www.ihe.net`, 2009.

[29] Continua Health Alliance. [Available online] `http://www.continuaalliance.org/index.html`, 2010.

[30] HL7. Health Level Seven. Devices Special Interest Group. [available online] `http://www.hl7.org/Special/committees/healthcaredevices/index.cfm`.

[31] Extensible Markup Language (XML). [Available online]: `http://www.w3.org/XML/`.

[32] OpenEHR. [available online] `http://www.openehr.org/home.html`.

[33] M. Uschold and M. Gruninger. Ontologies: Principles, methods and applications. *Knowl Eng Rev*, 11(2):93–136, 1996.

[34] openEHR. Archetypes openehr. [Available online]: http://www.openehr.org/knowledge/.

[35] R. Studer, V.R Benjamins, and D. Fensel. Knowledge engineering: principles and methods. *Data Knowl Eng*, 25:161–197, 1998.

[36] J. O. Kephart and D.M. Chess. The vision of autonomic computing. *IEEE Comput*, 36(1):41–50, 2003.

[37] A.K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Int J Hum Comput Interact*, 16:2–4, 2001.

[38] R. Gijsen, N. Hoeymans, FG. Schellevis, D. Ruwaard, WA. Satariano, and GA. Van den Bos. Causes and consequences of comorbidity: a review. *J Clin Epidemiol*, 54:661–674, 2001.

[39] American Diabetes Association. Executive summary: Standards of medical care in diabetes, 2011.

[40] World Health Organization. Medical device regulations: Global overview and guiding principles, 2003.

[41] M. Martínez-Espronceda, I. Martínez, L. Serrano, S. Led, JD. Trigo, A. Marzo, J. Escayola, and J. García. Implementation Methodology for Interoperable Personal Health Devices with Low-Voltage Low-Power Constraints. *IEEE Trans Inform Tech Biomed*, 15(3):398–408, 2001.

[42] W. Stallings. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2 (3rd Edition)*. 2000.

[43] J. E. López de Vergara, A. Guerrero, V. A. Villagrá, and J. Berrocal. Ontology based networking management: Study cases and lessons learned. *J Netw Syst Manag*, 17:234–254, 2009.

[44] W3C Working Group. Web Services Architecture, 2004. [Available online]: `http://www.w3.org/TR/ws-arch/`.

[45] VG. Koutkias, I. Chouvarda, A. Triantafyllidis, A. Malousi, GD. Giaglis, and N. Maglaveras. A personalized framework for medication treatment management in chronic care. *IEEE Trans Inf Technol Biomed*, 14(2):464–472, 2010.

[46] EJ. Gómez, ME. Hernando, A. García, F. Del Pozo, J. Cermeño, R. Corcoy, E. Brugués, and A. De Leiva. Telemedicine as a tool for intensive management of diabetes: the diabtel experience. *Comput Methods Programs Biomed*, 69(2):163–177, 2002.

[47] B. Maric, A. Kaan, A. Ignaszewski, and S. A. Lear. A systematic review of telemonitoring technologies in heart failure. *European Journal of Heart Failure*, 11:506–517, 2009.

[48] J. Basilakis, NH. Lovell, and BG. Celler. A Decision Support Architecture for Telecare Patient Management of Chronic and Complex Disease. *Conf Proc IEEE Eng Med Biol Soc, EMBS*, pages 4335–4338, 2007.

[49] M. V. M. Figueredo and J. Dias. Mobile telemedicine system for home care and patient monitoring. *Conf Proc IEEE Eng Med Biol Soc, EMBS*, pages 3387–3390, 2004.

[50] I. Martinez et. al. Seamless integration of ISO/IEEE11073 personal health devices and ISO/EN13606 electronic health records into an end-to-end interoperable solution. *Telemed J E Health*, 16(10):993–1004, 2010.

[51] S. Nizami, JR. Green, and C. McGregor. Service oriented architecture to support real-time implementation of artifact detection in critical care monitoring. *Conf Proc IEEE Eng Med Biol Soc*, pages 4925–4928, 2011.

[52] CAN in Automation (CiA) CANopen. [Available online]: `http://www.can-cia.org/index.php?id=canopen`.

[53] J. Escayola. Contribuciones a estándares y tecnologías de comunicación en dispositivos médicos para e-Salud. Technical report, Universidad de Zaragoza, 2012. PhD Dissertation.

[54] IHTSDO: SNOMED CT. [Available online]: `http://www.ihtsdo.org/snomed-ct`.

[55] M. Clarke, D. Bogia, K. Hassing, L. Steubesand, T. Chan, and D. Ayyagari. Developing a standard for personal health devices based on 11073. *Conf Proc IEEE Eng Med Biol Soc, EMBS*, pages 6174–6176, 2007.

[56] J. Yao and S. Warren. Applaying the ISO/IEEE 11073 standards to wearable home health monitoring systems. *J Clin Monit Comput*, 19:427–436, 2005.

[57] M. Martinez-Espronceda, I. Martínez, L. Serrano, S. Led, J. D. Trigo, A. Marzo, J. Escayola, G. Barron, and J. García. Lessons Learned Implementing the ISO/IEEE l1073 Standard into Wearable Personal Devices. *10th IEEE International Conference on Information Technology and Applications in Biomedicine (ITAB)*, 2010.

[58] J. D. Trigo, F. Chiarugi, A. Alesanco, M. Martínez Espronceda, L. Serrano, C. E. Chronaki, J. Escayola, I. Martínez, and J. García. Interoperability in digital electrocardiography: harmonization of ISO/IEEE x73-PHD and SCP-ECG. *IEEE Trans Inf Technol Biomed*, 14:1303–1317, 2010.

[59] I. Martínez et. al. Implementation experiences of ISO/IEEE11073 standard applied to new use cases for e-health environments. *Conf Proc IEEE Eng Med Biol Soc, EMBS*, pages 1679–1682, 2009.

[60] A. Aragués, J. Escayola, I. Martínez, P. del Valle, P. Muñoz, J. D. Trigo, and J. García. Trends and Challenges of the Emerging Technologies toward Interoperability and Standardization in e-Health Communications. *IEEE Communications Magazine*, pages 182–188, 2011.

[61] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *The Semantic Web, Scientific American*, pages 35–43, 2001.

[62] A. Maedche and S. Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

[63] O. Corcho, M. Fernández López, and A. Gómez Pérez. Methodologies, tools and languages for building ontologies. Where is their meeting point? *Data and Knowledge Engineering*, 46:41–64, 2003.

[64] Antonio Guerrero Casteleiro. Especificación del comportamiento de gestión de red mediante ontologías. Technical report, Universidad Politécnica de Madrid, 2007. PhD Dissertation.

[65] A. Gómez Pérez and O. Corcho. Ontology Languages for the Semantic Web. *IEEE Intelligent Systems*, 17(1):54–60, 2002.

[66] H. Zhu and S. Madnick. A lightweight ontology approach to scalable interoperability. *VLDB Workshop on Ontologies-based techniques or DataBases and Information Systems,(ODBIS06)*, 2006. Seoul, Korea.

[67] Ontologies tutorial. [Available online]: `http://www.obitko.com/tutorials/ontologies-semantic-web/ontologies.html`.

[68] O. Corcho and A. Gómez Pérez. A Roadmap to Ontology Specification Languages. *12th Int Conf. Knowledge Eng. and Knowledge Management, Lecture Notes in Artificial Intelligence*, pages 80–96, 2000.

[69] Resource Description Framework (RDF) Model and Syntax Specification /W3C Recommendation. [Available online]: `http://www.w3.org/TR/REC-rdf-syntax`.

[70] OWL Web Ontology Language Guide. W3C Recommendation, 2004. [Available online]: `http://www.w3.org/TR/owl-guide/`.

[71] OWL 2 Web Ontology Language, 2004. [Available online]: `http://www.w3.org/TR/2009/REC-owl2-new-features-20091027/`.

[72] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A Practical OWL-DL Reasoner. *J. Web Sem*, 5(2):51–53, 2007.

[73] C. Milena Rodríguez, W. Cano Montaño, and J. Mauricio Martínez. Razonadores semánticos: un estado del arte. *INGENIUM Revista de la Facultad de Ingeniería*, 21, 2010.

[74] Thomas Henry Briggs. Constraint generation and reasoning in owl. Technical report, University of Maryland, 2008. PhD Dissertation.

[75] Pellet: OWL 2 Reasoner for Java. [Available online]: `http://clarkparsia.com/pellet/`.

[76] Racer Renamed Abox and Concept Expression Reasoner. [Available online]: `http://www.sts.tu-harburg.de/~r.f.moeller/racer/`.

[77] I. Horrocks. The FaCT system. *Proc. Automated Reasoning with Analytic Tableaux and Related Methods: Int Conf. Tableaux 98, Lecture Notes in Artificial Intelligence*, (1397):307–312, 1998.

[78] Hermit OWL Reasoner. [Available online]: `http://www.hermit-reasoner.com/`.

[79] D. Vrandecic. *Ontology Evaluation*. PhD thesis, Fakultät für Wirtschaftswissenschaften, Karlsruhe, 2010.

[80] J. Brank, M. Grobelnik, and D. Mladenic. A survey of ontology evaluation techniques. *Proceedings of the Conference on Data Mining and Data Warehouses, SiKDD*, 2005.

[81] N. F. Noy and D. L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical Report SMI-2001-0880, Stanford Medical Informatics, 2012.

[82] T. Gu et al. An ontology based context model in intelligent environments. *Proc. Communication Networks and Distributed Systems Modeling and Simulation*, 2004. Conf Soc. for Modeling and Simulation Int.

[83] J. Strassner, S. van der Meer, D. O'Sullivan, and S. Dobson. The Use of Context-Aware Policies and Ontologies to Facilitate Business-Aware Network Management. *J. Netw. Syst. Manag*, 17:255–284, 2009.

[84] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *Knowledge Engineering Review*, 18(1):1–31, 2003.

[85] F. Latfi, B. Lefebvre, and C. Descheneaux. Ontology-based management of the telehealth smart home, dedicated to elderly in loss of cognitive autonomy. *Third International Workshop on OWL: Experiences and Directions, OWLED*, 2007.

[86] F. Paganelli and D. Giuli. An Ontology-based System for Context-aware and Configurable Services to Support Home-based Continuous Care. *IEEE Trans Inform Tech Biomed*, 15(2):324–333, 2011.

[87] C. E. Kuziemsky and F. Lau. A four stage approach for ontology-based health information system design. *Artif Intell Med*, (50):133–148, 2010.

[88] V.F.S.Fook, S.C. Tay, and D. Zhang M. Jayachandran, J. Biswas. An ontology-based context model in monitoring and handling agitation behavior for persons with dementia. *Proc. of the Fourth Annual IEEE Int. Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, 2006.

[89] D. Riaño et al. An ontology-based personalization of health-care knowledge to support clinical decisions for chronically ill patients. *J Biomed Inform*, 45(3):429–446, 2012.

[90] M. Brochhausen et al. The ACGT Master Ontology and its applications - Towards an ontology-driven cancer research and management system. *J Biomed Inform*, 44(1):8–25, 2011.

[91] A. Valls, K. Gibert, D. Sanchez, and M. Batet. Using ontologies for structuring organizational knowledge in home care assistance. *Int J Med Inform*, 79(5):370–387, 2010.

[92] C. Cornalba, R. G. Bellazzi, and R. Bellazi. Building a Normative Decision Support System for Clinical and Operational Risk Management in Hemodialysis. *IEEE Trans Inf Technol Biomed*, 2(5):678–686, 2008.

[93] M.M. Bouamrane, A. Rector, and M. Hurrell. Development of an Ontology of Preoperative Risk Assessment for a Clinical Decision Support System. *Proc. of the 22nd IEEE International Symposium on Computer-Based Medical Systems, CBMS, Albuquerque, US*, 2009.

[94] Health Informatics - Electronic Healthcare Record Communication. Parts 1-5, CEN/ISO EN13606, 2010.

[95] I. Berges, J. Bermudez, and A. Illarramendi. Towards Semantic Interoperability of Electronic Health Records. *IEEE Trans Inf Technol Biomed*, 16 (3):424–431, 2012.

[96] EJ. Ko, HJ. Lee, and JW. Lee. Ontology-based context modeling and reasoning for U-HealthCare. *IEICE Trans Inf Syst*, E90-D(8):1262–1270, 2007.

[97] K. Gibert, A. Valls, and D. Riaño. Knowledge Engineering as a support for building an Actor Profile Ontology for Integrating Home-Care Systems. *Proc MIE, Conference of the European Federation for Medical Informatics, Goteborg, Sweden*, 2008.

[98] F. Campana, A. Moreno, D. Riaño, and L. Varga. K4Care: Knowledge-Based Homecare e-Services for an Ageing Europe. *Chapter in book Agent Technology and e-Health, Whitestain Series in Soft Agent Technologies and Autonomic Computing*, pages 95–115, 2008.

[99] T. Sampalli, M. Shepherd, and J. Duffy. A patient profile ontology in the heterogeneous domain of complex and chronic health conditions. *Proc. of the 44th Hawaii International Conference on System Sciences (HICSS '11)*, 2011.

[100] A. Guerrero, V. A. Villagrá, J.E. López de Vergara, and J. Berrocal. Ontology-based integration of management behavior and information definitions using SWRL and OWL. *Proc. of the 16th IFIP/IEEE International Workshop on Distributed System, Barcelona, Spain*, 2005.

[101] J. E. Lopez de Vergara and V. A. Villagra and C. Fadon and J. M. Gonzalez and J. A. Lozano and M. Alvarez-Campana. An autonomic approach to offer services in OSGi-based home gateways. *Computer Communications*, 31(13):3049–3058, 2008.

[102] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML, 2004. [Available online]: http://www.w3.org/Submission/SWRL/.

[103] H. Boley et. al. Rule-ML Specification . [Available online]: http://ruleml.org/1.0/.

[104] B. Motik, U. Sattler, and R. Studer. Query Answering for OWL-DL with Rules. *J. of Web Semantics*, 3(1):4160, 2005.

[105] M. Jang and J.C. Sohn. Bossam: an extended rule engine for OWL inferencing. *Third International Workshop on Rules and Rule Markup Languages for the Semantic Web, RuleML 2004, Hiroshima, Japan*, 3323:128–138, 2004.

[106] A. Kuusik, E. Reilent, I. Loobas, and M. Parve. Semantic formal reasoning solution for personalized home telecare. *2nd International Conference on Mechanical and Electrical Technology (ICMET)*, pages 72–76, 2010.

[107] A. Jovic, M. Prcela, and D. Gamberger. Ontologies in Medical Knowledge Representation. *Proc. of Int. Conf. Information Technology Interfaces*, pages 535–540, 2007.

[108] SPARQL Query Language for RDF. W3C Recommendation, 2008. [Available online]: http://www.w3.org/TR/rdf-sparql-query/.

[109] A. Polleres. From SPARQL to rules (and back). *InProc. of WWW-2007*, 2007.

[110] SPIN SPARQL, 2008. [Available online]: http://spinrdf.org/.

[111] B. Glimm. Using SPARQL with RDFS and OWL entailment. *Reasoning Web. Semantic Technologies for the Web of Data*, pages 137–201, 2011.

[112] SPARQL 1.1, 2012. [Available online]: http://www.w3.org/TR/sparql11-query/.

[113] Jena Framework . [Available online]: http://jena.sourceforge.net/.

[114] SPARQL 1.1 Entailment Regimes. [Available online]: http://www.w3.org/TR/2012/WD-sparql11-entailment-20120105/.

[115] M. Parashar and S. Hariri. Autonomic computing: an overview. *UPP 2004, LNCS 3566*, pages 247–259, 2005.

[116] J. Strassner, N. Agoulmine, and E. Lehtihet. FOCALE - A Novel Autonomic Networking Architecture. *ITSSA Journal*, 3(1):64–79, 2007.

[117] R. Sterritt, M. Parashar, H. Tianfield, and R. Unland. A concise introduction to autonomic computing. *Advanced Engineering Informatics*, 19(1):181–187, 2005.

[118] S. Dobson et al. A survey of autonomic communications. *ACM Transactions Autonomous Adaptive Systems (TAAS)*, 1(2):223–259, 2006.

[119] G. Tesauro and et al. A Multi-Agent Systems Approach to Autonomic Computing. *IBM Press*, pages 464–471, 2004.

[120] J. Keeny, D. Lewis, and D. O'Sullivan. Ontological Semantics for Distributing Contextual Knowledge in Highly Distributed Autonomic Systems. *J. Netw. Syst. Manag*, 15(1):75–86, 2007.

[121] D.Lewis, D. O'Sullivan, R. Power, and J. Keeney. Semantic Interoperability for an Autonomic Knowledge Delivery Service. *proc of WAC, Athens, Greece*, 2005.

[122] L. Stojanovic, J. Schneider, A. Maedche, S. Libischer, R. Studer, T. Lumpp, A. Abecker, G. Breiter, and J. Dinge. The role of ontologies in autonomic computing systems. *IBM Systems Journal*, 43(3), 2004.

[123] R. Ribeiro de Azevedo et. al. An Autonomic Ontology-Based Multiagent System for Intrusion Detection in Computing Environments. *International Journal for Infonomics (IJI)*, 3(1):182–189, 2010.

[124] W3C Working Group. Web Services Description Language (WSDL), 2001. [Available online]: `http://www.w3.org/TR/wsdl`.

[125] X. Feng, J. Shen, and Y. Fan. REST: An alternative to RPC for Web services architecture. *First International Conference on Future Information Networks(ICFIN)*, pages 14–17, 2009.

[126] W3C Working Group. Web Application Description Language, 2009. [Available online]: `http://www.w3.org/Submission/wadl/`.

[127] A. Pras, T. Drevers, R. van de Meent, and D. Quartel. Comparing the performance of snmp and web services based management. *IEEE Electron. Trans. Netw. Serv. Manag*, 1(2):72–82, 2004.

[128] G. Pavlou, P. Flegkas, S. Gouveris, and A. Liotta. On Management Technologies and the Potential of Web Services. *IEEE Communications Magazine*, 42(7):58–66, 2004.

[129] R. Neisse, R. L. Vianna, L. Z. Granville, M. J. B. Almeida, and L. M. R. Tarouco. Implementation and Bandwidth Consumption Evaluation of SNMP

to Web Services Gateways. *IEEE/IFIP Network Operations and Management Symposium, NOMS*, pages 715–728, 2004.

[130] D. Martin et. al (W3C Working Group). OWL-S: Semantic Markup for Web Services, 2004. [Available online]: `http://www.w3.org/Submission/OWL-S/`.

[131] J. de Brujin et. al (W3C Working Group). Web Service Modeling Ontology (WSMO), 2001. [Available online]: `http://www.w3.org/Submission/WSMO/`.

[132] OASIS Standard 200401. Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), 2004.

[133] M. Cooper et. al (Network Working Group). Internet X.509 Public Key Infrastructure: Certification Path Building. RFC 4158.

[134] Object Management Group. CORBA. Common Object Request Broker Architecture and Specification, Ver. 3.1, OMG Document Number formal/2008-01-04, 2008. [Available online]: `http://www.corba.org/`.

[135] J. Yu and I. Al. Ajarmeh. An Empirical Study of the NETCONF Protocol. *Proc. 6th Int. Conference on Networking and Services, ICNS*, pages 253–258, 2010.

[136] Q. Gu and A. Marshall. Network management performance analysis and scalability tests: SNMP vs. CORBA. *in Proc. IEEE/IFIP Netw. Operat. Manag. Symp., Salvador, Brazil*, pages 701–714, 2004.

[137] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP), May 1990. RFC 1157.

[138] J. Case, R. Mund, D. Partain, and B. Stewart. Introduction to Version 3 of the Internet-standard Network Management Framework, April 1999. RFC 2570.

[139] J. Seo, J. Heo, S. Lim, J. Ahn, and W. Kim. A study on the implementation of a portable u-Healthcare system using SNMP and AODV. *in Proc. 29th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.,Lyon, France*, pages 1519–1522, 2007.

[140] Y. Lim, M. Messina, F. Kargl, L. Ganguli, M. Fischer, and T. Tsang. SNMP-proxy for wireless sensor network. *5th Int. Conf. on Information Technology: New Generations (ITNG), Las Vegas*, pages 148–159, 2008.

[141] K. McCloghrie, D. Perkins, J. Schoenwaelder, J. Case, M. Rose, and S. Waldbusser. Structure of Management Information Version 2 (SMIv2), Apr. 1999. RFC 2578.

[142] K. McCloghrie and M. Rose. Management Information Base for Network Management of TCP/IP-based Internet: MIB-II , March. 1991. RFC 1213.

[143] S. Waldbusser. Remote Network Monitoring Management Information Base, May 2000. RFC 2819.

[144] W. Stallings. SNMPv3: A security enhancement to SNMP. *IEEE Commun Surv*, 1(1):2–17, 1998.

[145] U. Blumenthal and B. Wijnen. User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3), Dec. 2002. RFC 3414.

[146] B. Wijnen, R. Presuhn, and K. McCloghrie. View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), Dec. 2002. RFC 3515.

[147] M. Fowler. *Patterns of Enterprise Application Architecture*. 2003.

[148] H. Chen, T. Finin, and A. Joshi. The SOUPA Ontology for Pervasive Computing. *in Series in Software Agent Technologies Springer*, 2005.

[149] Protégé-OWL Editor. [Available online]: `http://protege.stanford.edu`.

[150] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P.Patel-Schneider, and U. Sattler. OWL2: The Next Step for OWL. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322, 2008.

[151] B. Glimm and A. Hogan and M. Krotzsch and A. Polleres. OWL: Yet to arrive on the Web of Data? [Available online]: `http://www.slideshare.net/aidhog/owl-yet-to-arrive-on-the-web-of-data`.

[152] OMG (Object Management Group). Business Process Model and Notation. [Available online]: `http://www.bpmn.org/`.

[153] T. Andrews et. al. Specification: Business Process Execution Language for Web Services Version 1.1, 2003. [Available online]: `http://www-106.ibm.com/developerworks/library/ws-bpel/`.

[154] M.A. Aslam, S. Auer, J. Shen, and M. Herrmann. Expressing Business Process Models as OWL-S Ontologies. *Proc. of Int. Workshop on Grid and Peer-to-Peerbased Workflows (GPWW)*, 2006.

[155] A. Berglund et. al (Eds). XML Path Language (XPath) 2.0 (Second Edition) W3C Recommendation, 2010. [Available online]: `http://www.w3.org/TR/xpath-datamodel/`.

[156] Y. Gil and J. Blythe. Planet: A shareable and reusable ontology for representing plans. *AAAI Workshop on Representational Issues for Real-World Planning Systems*, 2000.

[157] D. Rajpathak and E. Motta. An ontological formalization of the planning task. *Proc. 3-d Int. Conf. on Formal Ontology in Information Systems (FOIS'04)*, 2004.

[158] Y. Shahar, S. Miksch, and P. Johnson. A Task-Specific Ontology for Design and Execution of Time-Oriented Skeletal Plans. *Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop*, 1996.

[159] I. Davies, P. Green, M. Rosemann, M. Indulska, and S. Gallo. How do practitioners use conceptual modeling in practice? *Data & Knowledge Engineering*, 58:358–380, 2006.

[160] H.J. Happel and S. Seedorf. Applications of Ontologies in Software Engineering. *Proc. Second Int'l Workshop Semantic Web Enabled Software Eng.*, 2006.

[161] TDB database. [Available online]: `http://openjena.org/TDB/`.

[162] MySQL. [Available online]: `http://www.mysql.com/`.

[163] N.Lasierra, A.Alesanco, and J.García. Home-Based Telemonitoring Architecture To Manage Health Information Based on Ontology Solutions. *10th IEEE International Conference on Information Technology and Applications in Biomedicine (ITAB)*, 2010.

[164] Mulgara: Semantic Store. [Available online]: `http://www.mulgara.org/`.

[165] W. Tai, R. Brennan, J. Keeney, and D. O'Sullivan. An Automatically Composable OWL Reasoner for Resource Constrained Devices. *Proc. of Int. Conf. on Semantic Computing (ICSC), Berkeley, CA, USA*, pages 495–502, 2009.

[166] Apache Axis 2. [Available online]: `http://axis.apache.org/axis2/java/core/`.

[167] Jersey API. [Available online]: `http://jersey.java.net/`.

[168] OpenSSL: Criptography and SSL/TLS tool. [Available online]: `http://www.openssl.org/`.

[169] J. Fayn E. Nageba and P. Rubel. A model driven ontology-based architecture for supporting the quality of services in pervasive telemedicine applications. *Proceedings the 3rd International Conference on Pervasive Computing Technologies for Healthcare*, pages 1–8, 2009. New York.

[170] N. Lasierra, A. Alesanco, and J. García. An ontology approach to manage individual patient profiles in home-based telemonitoring scenarios. *10th IEEE International Conference on Information Technology and Applications in Biomedicine (ITAB)*, 2010.

[171] J. Hartmann, P. Spyns, A. Giboin, D. Maynard, R. Cuel, and MC. Suárez-Figueroa et al. Methods for ontology evaluation. *Knowledge Web Deliverable*, D1.2.3, 2004.

[172] The global initiative for chronic obstructive lung disease (GOLD) pocket guide to COPD diagnosis, management, and prevention, 2011.

[173] C. Tsigos et al. Management of obesity in adults: European clinical practice guidelines. *Obesity Management Task Force of the European Association for the Study of Obesity. Obesity Facts*, 1:106–116, 2008.

[174] Guidelines for management of clinical endocrine disease. [Available online] `http://www.endotext.org/guidelines.htm`.

[175] Spanish Society of Cardiology. Clinical guidelines for ischemic heart disease. [Available online]: `http://www.secardiologia.es/practica-clinica-investigacion/guias-practica-clinica-cardiologia/cardiopatia-isquemica`.

[176] Guidelines for the diagnosis and management of asthma (epr-3). [Available online]: `http://www.nhlbi.nih.gov/guidelines/asthma/`.

[177] Cg127 hypertension: Nice guideline. [Available online]: `http://guidance.nice.org.uk/CG127/NICEGuidance`.

[178] ICSI Health Care Guideline: Diagnosis and Treatment of Osteoporosis, 2011.

[179] Spanish Society of Cardiology. Clinical guidelines for ischemic heart disease. [Available online]: `http://www.secardiologia.es/practica-clinica-investigacion/guias-practica-clinica-cardiologia/insuficiencia-cardiaca-y-miocardiopatia`.

[180] American Diabetes Association. Executive summary: Standards of medical care in diabetes, 2011.

[181] J. Genest and et. al. Canadian cardiovascular society/canadian guidelines for the diagnosis and treatment of dyslipidemia and prevention of cardiovascular disease in the adult. *Can J Cardiol*, 25:567–579, 2009.

[182] NICE clinical guideline 59 : Osteoarthritis: the care and management of osteoarthritis in adults, 2008.

[183] N. Lasierra, A. Alesanco, and J. García. An SNMP based solution to enable remote ISO/IEEE 11073 technical management. *IEEE Trans Inform Tech Biomed*, 16(4):709–719, 2012.

[184] L. Andrey, O. Festor, A. Lahmadi, A. Pras, and J. Schonwalder. Survey of SNMP performance analysis studies. *Int. J. Netw. Manage*, 19:527–548, 2009.

[185] D. Preuveneers et al. Towards an extensible context ontology for ambient intelligence. *Lecture Notes in Computer Science: Second European Symposium on Ambient Intelligence, LNCS 3295*, pages 148–159.

[186] SNMP4j framework. The Object Oriented SNMP API for Java Managers and Agents. [Available online]: `http://www.snmp4j.org/`.

[187] V. García. Desarrollo de un gestor snmpv3 con interfaz web para la gestión de información técnica en escenarios de telemonitorización domiciliaria. Technical report, University of Zaragoza, March 2012. Master Thesis (Supervisors: N. Lasierra and A. Alesanco).

[188] The Apache Software Foundation. Struts 2, 2010. [Available online]: `http://struts.apache.org/2.2.1/index.html`.

Annex A

Patients Profiles Requirements & Specification

During the application-based study of the clinical HOTMES ontology, the physician Sara Guillén who works in the Arrabal primary care center (located in Zaragoza) and contributes for the health research unit in primary care of Aragón, configured 11 chronic *patients profiles* and 5 multi-chronic *patients profiles* (merged as the combination of simple profiles). Supervising patients with profiles reports many advantages. They can be used not only to monitor patient's health status but also to classify patients into different groups under the same health care area and control types of population diseases. Using the questionnaire depicted in chapter 4, the physician included in each *patient profile* guidelines and questions to monitor subjective or qualitative, environmental, quantitative and vital constant measurements information.

The qualitative information provides relevant information to monitor a chronic patient. As it shown in Table 4.9, the physician included in each *patient profile* at least one specific test to collect from patients this type of data. These tests included questions about the presence of physiological symptoms (specific for each disease) in order to detect a risk for the patient. For example, the hypothyroidism test (included in the thyroid disorder profile) contained questions about sleepiness, fatigue, weakness, menstrual periods, dry skin or puffy face, lacks of memory and intolerance to cold temperatures among others. This profile, as well as the heart failure profile, included more than one test. In contrast, other profiles, such as the dysplidemia profile, included an instance of a test containing just two questions. Asking patients questions every day or every week (as configured in each profile) about his/her health status help them to identify and remember symptoms related to their disease.

To collect information about environmental conditions was only relevant to monitor COPD patients and patients with asthma. This fact is related with the FEV1 measurement supervision included in both profiles.

The physician remarked that to correctly supervise a patient was important to collect quantitative information such as the quantity of food and liquids took by the patient, the medication and the number of cigarettes smoked by him. Although the physicians did not define any specific alarm to alert about this information, most of the profiles included questions to ask the patient about this information.

An average total number of 6 vital constant measurements were included to be monitored in each profile. Weight, Height (for BMI calculation), pulse rate and pressure measurements were included in nearly all profiles. Other measurements such as FEV or HbA1c were included only in specific profiles (COPD, asthma and diabetes respectively). The monitoring frequency of these data is conditioned to the patient's disease. For example, the glucose measurement should be daily controlled for a diabetes patient and monthly controlled for a patient with ischemic heart disease. The details of each *patient profile* requirements are exposed in next subsections. These configured profiles could be reused to evaluate other approaches aiming to monitor patients with chronic or multi-chronic conditions.

Multi-chronic profiles were designed by using the same methodology as the combination of individuals measurements configured in the related chronic profiles. Then, subjective questions were modified to take into account the co-morbidity and alarms were adjusted to be more restrictive.

A.1 COPD - chronic obstructive pulmonary disease

COPD patients were identified as candidates to be monitored at home sites. Some estimates suggest that up to 600 million people worldwide may currently suffer from COPD and 75% of them in Europe are undiagnosed and consequently untreated. COPD includes two main illnesses: chronic bronchitis and emphysema, a pair of commonly co-existing diseases of the lungs in which the airways become narrowed. This leads to a limitation of the flow of air to and from the lungs, causing shortness of breath. There is no cure for COPD and it is almost always caused by cigarette smoking [172]. Actions included in the evaluation and in all the configured *patient profiles* are: 1) A1: warning physician, 2) A2: warning patient and 3) A3: indicate patient to repeat a measurement.

The clinical requirements to monitor a generic COPD patient are depicted in Table A.1 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.2.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI > 25 → A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60< pulse rate < 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value <70 and SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	1d, 10h	SpO2<90% → A1, A2 and A3
FEV1	1d, 10h	Testing with spirometry determines the presence and severity of the airway obstruction. Configured alarms: Alarm 1: FEV1 < 80%: COPD SLIGHT
Glucose	30d, 8h	A glucose value out of this range will trigger an alarm: Glucose <80 mg/dl or glucose > 126 mg/dl
Medication	1d, 10h	- No alarms defined
Number of Cigarettes	1d, 10h	- No alarms defined
Patient Location	15d, 10h	- No alarms defined
Room Temperature	15d, 10h	- No alarms defined
Room Humidity	15d, 10h	- No alarms defined
Health Test Questionnaire (COPD specific)	15d, 10h	Reagudization alarm: This alarm is activated if the patient answers YES to any of the questions included in the COPD test.
The COPD test includes the following questions: 1) Do you have a persistent or increasing cough? 2) Have you experienced increased mucus in the last few days? 3) Have you suffered shortness of breath or increased wheezing in the last few days? 4) Do you feel shivering or sleepiness?		

Table A.1: Monitoring requirements for a generic COPD patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
COPD	1,30	144	18	15	11	9	3

Table A.2: Cost metrics and resources required for a COPD patient

A.2 Obesity

Obesity is a medical condition that means having too much body fat what can lead to reduce life expectancy and/or increase health problems. The growing prevalence of obesity in most OECD countries is a cause of concern. According to 2010 data, obesity rate in Spain is (14%) a lower rate than obesity rate in United States (34%) or Mexico (30%) where the concern is increased. Obesity occurs over time when you eat more calories than you use. Factors that might tip this balance include genetic makeup, overeating, eating high-fat foods and not being physically active. Obesity increases the likelihood of various diseases such as heart disease, type 2 diabetes, obstructive sleep apnea, certain types of cancer, and osteoarthritis.

The clinical requirements to monitor a generic patient with obesity are depicted in Table A.3 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.4.

Monitored Data	Reminder	Alarm
Height Weight	7d, 10h	BMI > 25 and < 29.9 Overweight Alarm BMI >30.0 and < 34.9 Class I obesity Alarm BMI >35.0 and < 39.9 Class II obesity Alarm BMI >40.0 Class III obesity Alarm → A1, A2 and A3
Pulse Rate	7d, 10h	A pulse rate measurement value out of this range: 60< pulse rate < 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	7d, 10h	Alarm 1: DP value <70 and SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	7d, 10h	SpO2<90% → A1, A2 and A3
Glucose	7d, 8h	A glucose value out of this range will trigger an alarm: Glucose <80 mg/dl or glucose > 126 mg/dl
Liquids Quantity	1d, 10h	- No alarms defined (Provide information about liters of liquid drank during the day)
Food Quantity	1d, 10h	- No alarms defined (Provide information about gr. eaten during the day)
Medication	1d, 10h	- No alarms defined
BodyFat	30d, 10h	- No alarms defined
BodyWater	30d, 10h	- No alarms defined
Health Test Questionnaire (Obesity specific)	15d, 10h	Obesity Risk alarm: This alarm is activated if the patient answers YES to any of the questions included in the Obesity test. An instance of this test will be used to ask the patient about symptoms that could lead to conclude that there is a risk for the patient.
The Obesity test includes the following questions: 1) Do you have difficulties for sleeping or do you sweat a lot? 2) Do you have problems for breathing or feeling very tired every day? 3) (for women) Do you have hair in predominant masculine areas? 4) Do you have back and joint pains?		

Table A.3: Monitoring requirements for a generic obesity patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Obesity	1,54	142	23	14	13	8	3

Table A.4: Cost metrics and resources required for a obesity patient

A.3 Complex Management Task

More complex management task could be defined in an obesity profile. For example, for an obesity patient with hypertension suspect, we could define an alarm associated to detect along 3 days 141/91 mmHg in pressure measurement and if that occurs, then to assign the patient the HTA profile in addition to the obesity profile for his supervision. In terms of MAPE tasks it would be translated into defining:

1. Two Analysis Tasks → one for Systolic and one for Diastolic Pressure
2. One Planning Task with 2 pre-conditions (systolic pressure $>$ 141 and diastolic pressure $>$ 91) with one post-condition (be activated 3 times).
3. One execution task will be added to the normal execution task defined for all patients → download the HTA profile.

As another example, for an obesity patient with dyslipidemia suspect, we could define more complex tasks in order to detect a glucose value over 126 mg/dl so to download a dyslipidemia profile for that patient.

In terms of MAPE tasks it would be translated into defining:

1. One Analysis Task: In order to detect a glucose datum over 126.
2. One Planning Task with one pre-condition (glucose $>$ 126 true) and one post-condition (be activated 3 times).
3. One execution task will be added to the normal execution task defined for all patients' download the dyslipidemia profile.

A.4 Thyroid Disorders

Thyroid disorders are common disorders of the thyroid gland. Thyroid disorders include hyperthyroidism (abnormally increased activity), hypothyroidism (abnormally decreased activity) and thyroid nodules, which are generally benign thyroid neoplasms, but may be thyroid cancers. Increased production of thyroid hormone in these thyroid disorders results in a stimulation or quickening of the body's metabolism and symptoms, such as nervousness, anxiety and hypertension. Thyroid disorders that result in hypothyroidism result in a slowing of the body's chemical processes and metabolism and symptoms, such as weight gain, fatigue and depression.

Thyroid disease is common in the general population (7% of general population suffer a thyroid disease), and the prevalence increases with age. Thyroid disorders are more common in women than in men. Thyroid disorders may also lead to serious, potentially life-threatening complications.

The clinical requirements to monitor a generic patient with thyroids disorders are depicted in Table A.5 and Table A.7 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.6.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	- No alarms defined
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60< pulse rate < 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value <70 and SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	1d, 10h	SpO2<90% → A1, A2 and A3
Temperature	1d, 10h	if 35 >T or T >38 → A1, A2 and A3
Glucose	7d, 8h	A glucose value out of this range will trigger an alarm: Glucose <80 mg/dl or glucose > 126 mg/dl
Liquids Quantity	1d, 10h	- No alarms defined (Provide information about liters of liquid drank during the day)
Food Quantity	1d, 10h	- No alarms defined (Provide information about gr. eaten during the day)
Medication	1d, 10h	- No alarms defined

Table A.5: Monitoring requirements for a generic thyroid disorder patient(I)

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Thyroids Disorders	1,66	171	22	15	13	10	3

Table A.6: Cost metrics and resources required for a thyroids disorder patient

Monitored Data	Reminder	Alarm
Test Questionnaire 1	15d, 10h	Hypothyroidism risk alarm: This alarm is activated if the patient answers YES to any of the questions included in the test.
The Hypothyroidism test includes the following questions: 1) Do you feel sleepiness, fatigue or weakness? 2) Do you have heavier than normal menstrual periods? 3) Do you feel cold intolerance (you can't tolerate cold temperatures like those around you)? 4) Have you felt weight gain or increased difficulty losing weight? 5) Do you have muscle cramps and frequent muscle aches? 6) Have you felt a decreased libido? 7) Are you more irritable or depressed? 8) Are you pale, dry skin or with puffy face? 9) Do you have memory loses? 10) Do you feel hoarse voice?		
Test Questionnaire 2	15d, 10h	Hyperthyroidism risk alarm: This alarm is activated if the patient answers YES to any of the questions included in the test.
The Hyperthyroidism test includes the following questions: 1) Are you very nervous and continuously sweeting? 2) Have you got frequent palpitations or fast heart rate? 3) Have you got insomnia or breathlessness? 4) Have you got light or absent menstrual periods? 5) Have you got muscle weakness or fatigue? 6) Have you got trembling hands? 7) Have you got weight loss?		
Test Questionnaire 3	1d, 10h	Thyrotoxic crisis risk alarm: This alarm is activated if the patient answers YES to any of the questions included in the test AND Temperature $> 38\ ^{\circ}C$
The Thyrotoxic test includes the following questions: 1) Are you very nervous and excited? 2) Have you got frequent palpitations or fast heart rate? 3) Do you have ophthalmological disorders?		

Table A.7: Monitoring requirements for a generic thyroid disorder patient (II)

A.5 Ischemic heart disease

Patients with ischemic heart disease are patients who have reduced heart pumping due to coronary artery disease. Ischemic cardiomyopathy results when the arteries that bring blood and oxygen to the heart are blocked. It is the most common cause of death in most Western countries, and a major cause of hospital admissions. Ischemic heart disease in USA affects approximately 1 out of 100 people, most often middle-aged to elderly men. Its risk increases with age, smoking, hypercholesterolaemia (high cholesterol levels), diabetes, obesity and hypertension (high blood pressure), and is more common in men and those who have close relatives with ischaemic heart disease. Symptoms of heart failure usually develop slowly over time. However, sometimes symptoms start very suddenly and are severe. The goal of treatment for patients with ischemic heart disease is to relieve symptoms and treat the cause of the condition. If symptoms are severe, you may need to stay in the hospital.

The clinical requirements to monitor a generic patient with thyroids disorders are depicted in Table A.8 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.9.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $>$ 25 → A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60$<$ pulse rate $<$ 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value $<$70 AND SP value $<$ 100 mmHg Alarm 2: DP value $>$90 and SP value $>$140 mmHg TAD alarm: DP $>$ 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	1d, 10h	SpO2$<$90% → A1, A2 and A3
Glucose	30d, 8h	A glucose value out of this range will trigger an alarm: Glucose $<$80 mg/dl or glucose $>$ 126 mg/dl
Medication	1d, 10h	- No alarms defined
Number of Cigarettes	7d, 10h	- No alarms defined
Health Test Questionnaire (Ischemic specific)	15d, 10h	ACS (Acute Coronary Syndrome) alarm: This alarm is activated if the patient answers YES to any of the questions included in the Ischemic test.
The Ischemic test includes the following questions: 1) Have you got severe chest pain and over the left arm or the jaw? 2) Have you got shortness of breath? 3) Have you got pain, numbness, weakness or coldness in your legs or arms?		

Table A.8: Monitoring requirements for a generic ischemic patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Ischemic	1,26	120	17	11	10	8	3

Table A.9: Cost metrics and resources required for a ischemic patient

A.6 Asthma

Asthma is a common chronic inflammatory lung disease that inflames and narrows the airways. The airways are tubes that carry air in and out of your lungs. This can cause wheezing, coughing, chest tightness and trouble breathing, especially early in the morning or at night. Asthma affects a total of 300 million of people worldwide and it causes 250,000 deaths per year worldwide. Asthma is treated with two kinds of medicines: quick-relief medicines to stop asthma symptoms and long-term control medicines to prevent symptoms. It is very important to treat symptoms when you first notice them. This will help prevent the symptoms from worsening and causing a severe asthma attack. Severe asthma attacks may require emergency care, and they can be fatal.

The clinical requirements to monitor a generic patient with asthma are depicted in Table A.10 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.11.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI > 25 → A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60< pulse rate < 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value <70 AND SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	1d, 10h	SpO2<90% → A1, A2 and A3
FEV1	1d, 10h	Testing with spirometry determines the presence and severity of the airway obstruction. Alarm 1: FEV1 < 80%: COPD SLIGHT
Temperature	1d, 10h	if 35 <T or T >38 → A1, A2 and A3
Glucose	30d, 8h	A glucose value out of this range will trigger an alarm: Glucose <80 mg/dl or glucose > 126 mg/dl
Medication (inhalers)	1d, 10h	- No alarms defined
Number of Cigarettes	7d, 10h	- No alarms defined
Patient Location	15d, 10h	- No alarms defined
Room Temperature	15d, 10h	- No alarms defined
Room Humidity	15d, 10h	- No alarms defined
Health Test Questionnaire (asthma specific)	1d, 10h	Reagudization alarm: This alarm is activated if the patient answers YES to any of the questions included in the asthma test.
The Asthma test includes the following questions: 1) Do you have chest tightness, pain or pressure? 2) Do you have cough especially at night? 3) Do you have signs of a cold or allergies (eg: sneezing, runny nose,sore throat or/and headache)? 4) Do you have wheezing or shortness of breath?		

Table A.10: Monitoring requirements for a generic asthma patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Asthma	1,28	136	19	16	11	9	3

Table A.11: Cost metrics and resources required for a asthma patient

A.7 Hypertension (HTA) or high blood pressure

High blood pressure is a common cardiac chronic medical condition in which the systemic arterial pressure is elevated. It means that the force of the blood against your artery walls is high enough that it may eventually cause health problems, such as heart disease. The more blood your heart pumps and the narrower your arteries, the higher your blood pressure. In the year 2000 it was estimated that nearly the 26% of the adult population had hypertension worldwide. High blood pressure is typically developed over years, and it affects nearly everyone eventually. Persistent hypertension can cause serious problems such as stroke, myocardial infarction, heart failure and arterial aneurysm and kidney failure. High blood pressure has no symptoms and an uncontrolled hypertension can cause serious problems. However it can be easily detected and controlled by your doctor.

The clinical requirements to monitor a generic patient with HTA are depicted in Table A.12 and the required resources in terms of individuals, tasks and rules to specify a *patient profile* are gathered in Table A.13.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI > 25 → A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60< pulse rate < 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value <70 AND SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
SpO2	1d, 10h	SpO2<90% → A1, A2 and A3
Glucose	30d, 8h	A glucose value out of this range will trigger an alarm: Glucose <80 mg/dl or glucose > 126 mg/dl
Medication	1d, 10h	- No alarms defined
Number of Cigarettes	7d, 10h	- No alarms defined
Health Test Questionnaire (HTA specific)	1d, 10h	Hypertension alarm: This alarm is activated if the patient answers YES to any of the questions included in the HTA test.
The HTA test includes the following questions: 1) Have you got chest pain or palpitations? 2) Have you got ophthalmological disorders? 3) Have you got dizziness or sickness? 4) Have you notice an evident decrease in the urine?		

Table A.12: Monitoring requirements for a generic HTA patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
HTA	1,30	121	17	11	10	7	3

Table A.13: Cost metrics and resources required for a HTA patient

A.8 Osteoporosis

Osteoporosis is a disease of the bones that makes your bones more likely to be broken. In this disease, the bone mineral density is reduced leading to the bone microarchitecture to be deteriorated. Hence the bone is more fragile and there is an increased risk for fractures. Anyone can develop osteoporosis, but it is common in older women. It usually appears after the menopause. It is estimated that osteoporosis affect 200 million women worldwide (75 million people in Europe). Osteoporosis is a silent disease. For people with osteoporosis, risks can be reduced by lifestyle change including diet, exercises and preventing falls.

The clinical requirements to monitor a generic patient with thyroids disorders are depicted in Table A.14 and the required resources in terms of individuals, tasks and rules to specify a patient profile are gathered in Table A.15.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $> 25 \rightarrow$ A1, A2 and A3
Medication	1d, 10h	- No alarms defined
Health Test Questionnaire (Calcium)	1d, 10h	- No alarms defined
The Osteoporosis test includes the following questions: 1) How many aliments rich in calcium did you eat yesterday during breakfast? List them 2) How many aliments rich in calcium did you eat yesterday during lunch? List them 3) How many aliments rich in calcium did you eat yesterday during supper? List them		
Health Test Questionnaire (Falls)	1d, 10h	Falls Alarm: YES in pain question and > 2 in falls question
The Falls test includes the following questions: 1) How many falls did you have yesterday? 2) Have you got wrists or vertebral pains?		

Table A.14: Monitoring requirements for a generic osteoporosis patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
HTA	1,23	58	4	5	2	2	2

Table A.15: Cost metrics and resources required for a osteoporosis patient

A.9 Heart failure (HF) often called congestive heart failure (CHF)

Heart failure is a chronic and progressive condition in which the heart muscle is unable to supply enough blood flow to meet the needs of the body. Heart failure does not mean that your heart has stopped or is about to stop working. It means that your heart can not keep up with that workload; that is not able to pump blood the way it should. Heart failure is a common condition. In developed countries, around 2% of adults suffer from heart failure, but in those over the age of 65, this increases to 6-10%.

Heart failure is a serious condition, and usually there is no cure. Heart failure cause a number of symptoms including shortness of breath, persistent coughing or wheezing, buildup of excess fluid in body tissues (feet, ankles and legs) (called edema), fatigue, lack of appetite or nausea, impaired thinking, or an increased heart rate.

The clinical requirements to monitor a generic patient with HF are depicted in Table A.16 and the required resources in terms of individuals, tasks and rules to specify a patient profile are gathered in Table A.17.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $> 25 \rightarrow$ A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60$<$ pulse rate $< 100 \rightarrow$ A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value $<$70 AND SP value $<$ 100 mmHg Alarm 2: DP value $>$90 and SP value $>$140 mmHg TAD alarm: DP $>$ 120 mmHg Each alarm detection $\rightarrow$ A1, A2 and A3
SpO2	1d, 10h	SpO2$<$90% $\rightarrow$ A1, A2 and A3
Medication	1d, 10h	- No alarms defined
Liquids Quantity	1d, 10h	- No alarms defined
Health Test Questionnaire (Liquids)	1d, 10h	- This alarm is activated with a positive value in the question and high liquids quantity detected ($>$2l)
The liquids test includes the following questions: 1) Have you noticed that the quantity of urine has been decreased?		
Health Test Questionnaire (HF)	1d, 10h	HF Alarm: This alarm is activated if the patient answers YES to any of the questions included in the HF test.
The HF test includes the following questions: 1) Have you got shortness of breath, persistent coughing or wheezing? 2) Have got any edema in feet, ankles or legs? 3) Have you got fatigue, lack of appetite or nauseas? 4) Have you notice that your heart rate increases?		

Table A.16: Monitoring requirements for a generic HF patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
HF	1,28	129	17	11	10	8	3

Table A.17: Cost metrics and resources required for a HF patient

A.10 Diabetes mellitus

Diabetes mellitus often simply referred to as diabetes, is a disorder of metabolism in which a person has levels of blood glucose (or sugar) too high. There are 3 types of diabetes: type 1 diabetes (results when the body cannot produce insulin), type 2 diabetes (results when the body is resistance to insulin) and gestational diabetes (results when a pregnant woman who has never had diabetes before have high levels of glucose in blood during the pregnancy). The World Health Organization (WHO) estimated that 285 million people, corresponding to 6.4% of the world's adult population will live with diabetes in 2010. This number is expected to grow to 438 million by 2030.

Diabetes is a leading cause of death and disability. It can damage your eyes, kidneys, and nerves. Diabetes can also cause heart disease, stroke and even the need to remove a limb. Nevertheless, all forms of diabetes can be controlled with insulin or medications (type 2). The classical symptoms of diabetes are frequent urination, increase of thirst and increased hunger. Symptoms of Type 2 diabetes may include fatigue, thirst, weight loss and blurred vision.

The clinical requirements to monitor a generic patient with diabetes are depicted in Table A.18 and the required resources in terms of individuals, tasks and rules to specify a patient profile are gathered in Table A.19.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $>$ 25 $\rightarrow$ A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60$<$ pulse rate $<$ 100 $\rightarrow$ A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value $<$70 AND SP value $<$ 100 mmHg Alarm 2: DP value $>$90 and SP value $>$140 mmHg TAD alarm: DP $>$ 120 mmHg Each alarm detection $\rightarrow$ A1, A2 and A3
Glucose	1d, 8h	(This remainder should be activated each day for patients with insulin. OAD's (oral antidiabetic drugs) patients don't need to take this measurement every day. Neither do patients with diet.) A glucose value out of this range will trigger an alarm: Glucose $<$80 mg/dl or glucose $>$ 126 mg/dl
Medication	1d, 10h	- No alarms defined
Liquids Quantity	1d, 10h	- No alarms defined
Food Quantity	1d, 10h	- No alarms defined
Cigarettes Number	1d, 10h	- No alarms defined
HbA1c	30d, 10h	Remember to take a HbA1c measurement. Patients treated with insulin and OADs should take this measurement every month. Patients with diet once each 6 months. HbA1c $>$ 7%
Health Test Questionnaire (Diabetes)	1d, 10h	Diabetes Alarm: This alarm is activated if the patient answers YES to any of the questions included in the HF test.
The Diabetes test includes the following questions: 1) Have you got pain when walking short distances or fatigue? 2) Have got chest or epigastric pains? 3) Have you notice a decrease in skin sensitiveness or numbness? 4) Have you got ulcers in your feet? 5) (for men) Have you got urine incontinence or impotence?		

Table A.18: Monitoring requirements for a generic diabetes patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Diabetes	1,24	124	15	13	9	7	3

Table A.19: Cost metrics and resources required for a diabetes patient

A.11 Dyslipidemia

Dyslipidemia is a disorder of the amount of lipoprotein (overproduction or deficiency) in the blood. Dyslipidemias may be manifested by elevation of the total cholesterol, the "bad" low-density lipoprotein (LDL) cholesterol and the triglyceride concentrations, and a decrease in the "good" high-density lipoprotein (HDL) cholesterol concentration in the blood. Dyslipidemia further raises the risk of atherosclerosis in people with diabetes (it affects more to people with type 2). Dyslipidemia has no symptoms, so the diagnosis is made by a blood test called a lipid profile. Treatment of dyslipidemia usually involves a multimodal approach, including diet and exercise in all cases, lipid-modifying medication, and consideration of changes in ARV (antiretroviral) medication.

The clinical requirements to monitor a generic patient with dysplidemia are depicted in Table A.20 and the required resources in terms of individuals, tasks and rules to specify a patient profile are gathered in Table A.21.

Monitored Data	Reminder	Alarm
Height Weight	15d, 10h	BMI $>$ 25 → A1, A2 and A3
Pulse Rate	1d, 10h	A pulse rate measurement value out of this range: 60$<$ pulse rate $<$ 100 → A1, A2 and A3.
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value $<$70 AND SP value $<$ 100 mmHg Alarm 2: DP value $>$90 and SP value $>$140 mmHg TAD alarm: DP $>$ 120 mmHg Each alarm detection → A1, A2 and A3
Medication	1d, 10h	- No alarms defined
Cigarettes Number	7d, 10h	- No alarms defined
Health Test Questionnaire (Dyslipidemia)	7d, 10h	Dyslipidemia Alarm: This alarm is activated if the patient answers YES to question 1.
The Diabetes test includes the following questions: 1) Have you got strong chest pains or distressing? 2) Do you follow any type of diet?		

Table A.20: Monitoring requirements for a generic dyslipidemia patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Dyslipidemia	1,24	100	14	9	8	6	3

Table A.21: Cost metrics and resources required for a dyslipidemia patient

A.12 Osteoarthritis

Osteoarthritis is the most common form of arthritis. This disease is related with abnormalities involving degradation of joints including articular cartilage and subchondral bone. It causes pain, swelling and reduces motion in your joints. It can occur in any joint, but usually it affects your hands, knees, hips or spine. Osteoarthritis occurs more frequently as we age. Before 45 it occurs more frequently in men, but after 55 it occurs more frequently in women. Despite the frequency of the disease, the cause is not still completely known and there is no cure. Factors that may cause osteoarthritis include: being overweight, getting older, injuring a joint and genetics.

The clinical requirements to monitor a generic patient with osteoarthritis are depicted in Table A.22 and the required resources in terms of individuals, tasks and rules to specify a patient profile are gathered in Table A.23.

Monitored Data	Reminder	Alarm
Height Weight	30 d, 10h	BMI > 25 → A1, A2 and A3
Blood Pressure (Systolic) Blood Pressure (Diastolic) Blood Pressure (MAP)	1d, 10h	Alarm 1: DP value <70 AND SP value < 100 mmHg Alarm 2: DP value >90 and SP value >140 mmHg TAD alarm: DP > 120 mmHg Each alarm detection → A1, A2 and A3
Temperature	30d, 10h	Alarm: T >38 → A1, A2 and A3
Medication	1d, 10h	- No alarms defined
Cigarettes Number	7d, 10h	- No alarms defined
Health Test Questionnaire (Osteoarthritis)	7d, 10h	Osteoarthritis Alarm: This alarm is activated if the patient answers YES to question 1.
The osteoarthritis test includes the following questions: 1) Do you have articular pains? 2) Do you have any redness or inflammation area?		

Table A.22: Monitoring requirements for a generic osteoarthritis patient

Profile	TDB inic (MB)	Instances	Rules	MT	AT	PT	ET
Osteoarthritis	1,25	88	12	8	7	5	3

Table A.23: Cost metrics and resources required for a osteoarthritis patient