Study on Semantic Assets for Smart Appliances Interoperability

D-S2: SECOND INTERIM REPORT

A study prepared for the European Commission
DG Communications Networks, Content & Technology
by:

TNO innovation for life

Digital Agenda for Europe

This study was carried out for the European Commission by

Authors:

Laura Daniele
laura.daniele@tno.nl

Frank den Hartog
frank.denhartog@tno.nl

Jasper Roes (Project manager)
jasper.roes@tno.nl

Internal identification

Contract number: 30-CE-0610154/00-11

SMART number: 2013/01077

Version	Date	Comments
0.9	22 Augustus 2014	For review by the Expert Group
1.0	24 September 2014	For public release and comments

Summary

About two thirds of the energy consumed by buildings originates from the residential sectors and thus household appliances. Household appliances or home appliances are electrical/mechanical machines which accomplish some household functions. Nowadays, appliances are not stand-alone systems anymore. They are often highly intelligent ("smart") and networked devices, that form complete energy consuming, producing, and managing systems. Reducing the use of energy and production of greenhouse gasses is therefore not only a matter of increasing the efficiency of the individual devices, but managing and optimizing the energy utilization on a system level. The systems will therefore inevitably consist of devices and sensors from different vendors, and open interfaces enabling further extensions. The interfaces need to be properly standardized and offer external access on a semantic level both to any manageable and controllable function of the system as a whole, and to any device that is part of the system.

However, the problem is not the lack of available standards. Actually, there already exist many standards, too many really, all dealing with a smaller or larger part of the problem, sometimes overlapping and competing. Various workshops and projects already explored this field and concluded that defining a useful and applicable reference data model should in principle be possible. One single, reference ontology could be created to cover the needs of all appliances relevant for energy efficiency, and it can be expanded to cover future intelligence requirements. The European Commission therefore issued a tender for a Study on "Available Semantics Assets for the Interoperability of Smart Appliances. Mapping into a Common Ontology as a M2M Application Layer Semantics", defining 3 tasks:

- **Task 1:** Take stock of existing semantic assets and use case assets
- **Task 2:** Perform a translation exercise of each model (or use case) to a common ontology language and a mapping or matching exercise between all the models
- **Task 3:** Propose a reference ontology and document the ontology into the ETSI M2M architecture

TNO was invited to perform this study. This document, *D-S2 Interim Study Report*, presents the results of task 2. It analyses in detail the semantic assets that are on the short list defined in task 1, presents an OWL ontology for each of these assets, and proposes an initial mapping between these ontologies.

In task 1 we have analysed 43 semantic assets and we have defined their initial semantic coverage. Moreover, we have created a visual representation of the key terms used by each asset, and provided a visual representation of the most recurring key terms among all assets. In this way, we were able to short-list 20 semantic assets that provide a good basis for further development of a reference ontology for the smart appliances domain. In task 2 we have translated the assets in the short list to corresponding OWL ontologies, which are presented in this deliverable, and we have created initial mappings among these ontologies. The purpose of these mappings is to relate the 20 assets using their most recurring concepts (core concepts), as initially identified in task 1, in terms of the ontologies presented here. Moreover, in order to reduce the number of mappings, a reference ontology is proposed with mappings to/from the ontologies of the specific assets. This reference ontology will be developed in task 3, but an initial proposal of the concepts that can constitute its core is presented in this document.

Table 1 lists, in alphabetical order, the OWL ontologies described in this document. For each ontology we provide a *title*, the *source* document used as a main reference to create the ontology, a *description* of the main classes and properties, and, eventually, *observations* necessary to better understand the choices underlying the ontology design and suggestions for its future extension. This document also provides an initial mapping of the ontologies by means of a number of concepts that we have identified as most relevant in the smart appliances domain and in the scope of this study. These concepts provide the basis for creating the reference ontology in the next phase of the project (task 3).

The ontologies presented here are available online at the smart appliances website[1]. This website provides a page for each of the 20 semantic assets in the short list, with the URL to download the corresponding ontology and a human-readable explanation to describe its main classes and properties. In order to guarantee transparency during the process and take into account the feedback of the stakeholders, especially the "owners" of the considered assets, each page includes a tab for posting comments (available when logged on to the website with a Google-account). Moreover, a second stakeholders' workshop will take place on the 15th of October 2014 at the ETSI premises in Sophia Antipolis in order to officially present this *D-S2 Interim Study Report* and collect feedback from the stakeholders, especially to check whether the meaning they intended has actually being captured by the ontologies we have created for them. It is possible to comment on this *D-S2 Interim Study Report* also at the ETSI website[2].

Table 1. Ontologies presented in this document

Acronym	Source	URL
DECT ULE	'HF-Protocol', 'HF-Service', 'HF-Interface', 'HF-Profile', version1.0, 23 January 2014	https://sites.google.com/site/smartappliancesproject/ontologies/dect_ule-ontology
ECHONET	ECHONET Specifications Appendix 'Detailed Requirements for ECHONET Device Objects' Release C, 31 May 2013	https://sites.google.com/site/smartappliancesproject/ontologies/echonet-ontology
eDIANA	'D2.2-A Ontology for Device Awareness', 30 November 2009	https://sites.google.com/site/smartappliancesproject/ontologies/ediana-ontology
EnOcean	'EnOcean Equipment Profiles (EEP)', Version 2.6, 17 December 2013	https://sites.google.com/site/smartappliancesproject/ontologies/enocean-ontology
FAN	'Interface description: Interface report' , Version 1.0 (final), 7th January, 2014	https://sites.google.com/site/smartappliancesproject/ontologies/fan-ontology
FIEMSER	'D5 FIEMSER Data Model', February 2011	https://sites.google.com/site/smartappliancesproject/ontologies/fiemser-ontology
FIPA	'FIPA Device Ontology Specification', document number SC00091E , 3 December 2002	https://sites.google.com/site/smartappliancesproject/ontologies/fipa-ontology
HYDRA	'Deliverable D6.6 Updated MDA Design Document', version 1.0, 20 August 2009	https://sites.google.com/site/smartappliancesproject/ontologies/hydra-ontology
KNX	• 'KNX System Specifications Interworking Datapoint Types', Version 1.07.00, 26	https://sites.google.com/site/smartappliancesproject/ontologies/knx-ontology

[1] https://sites.google.com/site/smartappliancesproject/ontologies
[2] http://sap.etsi.org

	April 2012 • 'KNX Advanced Course-Interworking_E1209b'	
MIRABEL	'D7.5 MIRABEL-ONE: Initial draft of the MIRABEL Standard', version1.0, 22 December 2011	https://sites.google.com/site/smartappliancesproject/ontologies/mirabel-ontology
OMA LW M2M	'OMA Lightweight Machine-to-Machine Technical Specification', candidate version 1.0, 10 December 2013	https://sites.google.com/site/smartappliancesproject/ontologies/oma-lightweight_m2m-ontology
OMS	'Open Metering System Specification Vol.2 – Primary Communication Issue 4.0.2', 27 January 2014	https://sites.google.com/site/smartappliancesproject/ontologies/oms-ontology
OSGi DAL	'RFC-196 OSGi Alliance Device Abstraction Layer, Draft', 30 January 2014	https://sites.google.com/site/smartappliancesproject/ontologies/osgi_dal-ontology
PowerOnt	• Politecnico di Torino, e-Lite research group webpage (http://elite.polito.it/dogont) • D. Bonino, F. Corno, 'DogOnt - Ontology Modeling for Intelligent Domotic Environments'.	https://sites.google.com/site/smartappliancesproject/ontologies/dogpower-ontology
SEEMPubs	'Deliverable D5.1, Data Format Definition', version 1.0, 30 September 2012	https://sites.google.com/site/smartappliancesproject/ontologies/seempubs-ontology
SEP2	'Zigbee Alliance/HomePlug Alliance Smart Energy Profile 2 Application Protocol Standard, ZigBee Public Document 13-0200-00, April 2013'	https://sites.google.com/site/smartappliancesproject/ontologies/sep2-ontology
SmartCoDE	'Deliverable D1.1.2 -Model of local energy resource cluster', 31 December 2012	https://sites.google.com/site/smartappliancesproject/ontologies/smartcode-ontology
UPnP	'UPnP Device Architecture 1.1', 15 October 2008	https://sites.google.com/site/smartappliancesproject/ontologies/upnp-ontology
W3C SSN	W3C Semantic Sensor Network Incubator Group webpage (http://www.w3.org/2005/Incubator/ssn/ssnx/ssn)	https://sites.google.com/site/smartappliancesproject/ontologies/w3c_ssn-ontology
Z-Wave	'Z-Wave Technical Basics - Chapter 4: Application Layer', 1 June 2011	https://sites.google.com/site/smartappliancesproject/ontologies/z-wave-ontology

Contents

Abbreviations

3G	Third Generation
AMM	Automated Meter Management
API	Application Programming Interface
BACnet	Building Automation and Control Networks
BACS	Building Automation and Control Systems
BEMO-COFRA	Brazil-Europe - Monitoring and Control Frameworks
BEMS	Building Energy Management Systems
BIM	Building Information Model
CECED	European Committee for Domestic Equipment Manufacturers
CEM	Customer Energy Managers
CEN	European Committee for Standardization
CENELEC	European Committee for Electrotechnical Standardization
CLC	CENELEC
CoAP	Constrained Application Protocol
COSEM	Companion Specification for Energy Metering
CSEP	Consortium for SEP2 Interoperability
DCP	Device Control Protocol
DECT	Digital Enhanced Cordless Telecommunications
DEHEMS	Digital Environment Home Energy Management System
DHCP	Dynamic Host Configuration Protocol
DLMS	Device Language Message Specification
DomoML-env	An ontology for Human Home Interaction
DPWS	Devices Profiles for Web Services
E2BA	Energy Efficient Buildings Association
Ebbits	Enabling business-based Internet of Things and Services
EC	European Commission
ECHONET	Energy Conservation and HOmecare NETwork
eDiana	Embedded Systems for Energy Efficient Buildings
EE	Energy Efficiency
EEP	EnOcean Equipment Profiles
ELC	European Lamp Companies Federation
EMU	Energy Management Unit
ENV	Environmental and Contextual data
EP	Energy Profile
EPI	Energy Performance Indicators
ERP	EnOcean Radio Protocol
ESCO	Enery Service Company
ESO	European Standardization Organisation
ETSI	European Telecommunications Standards Institute
EU	European Union
eu.bac	European building automation controls association
EupP	Energy using and producing Product

FAN	FlexiblePower Alliance Network
FIEMSER	Friendly Intelligent Energy Management Systems in Residential Buildings
FIPA	Foundation for Intelligent Physical Agents
FP7	European 7th Framework Program
FPAI	Flexible Power Application Infrastructure
FttH	Fiber to the Home
GENA	General Event Notification Architecture
GHz	Gigaherz
HAN	Home Area Network
HAN FUN	Home Area Network FUNctionality
HFC	High Frequency Communication
HGI	Home Gateway Initiative
HTTP	Hypertext Transfer Protocol
HVAC	Heating, ventilation, and air conditioning
Hydra	Heterogeneous physical devices in a distributed architecture
ICT	Information and Communication Technologies
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IES	Illuminating Engineering Society
IETF	Internet Engineering Task Force
IFC	International Foundation Classes
IoP	Internet of People
IOPTS	Internet of People, Things and Services
IoS	Internet of Services
IoT	Internet of Things
IP	Internet Protocol
IPR	Intellectual Property Rights
kbps	kilobit per second
KNX	Konnex
LDN	Logical Device Name
LEP	Local Energy Providers
LWM2M	Lightweight M2M
M2M	machine-to-machine
ME3GAS	Middleware for Energy Efficient Embedded Services & Smart Gas Meters
MDA	Model Driven Architecture
MIRABEL	Micro-Request-Based Aggregation, Forecasting and Scheduling of Energy Demand, Supply and Distribution
MUC	Multi Utility Communication
OASIS	Organization for the Advancement of Structured Information Standards
oBIX	Open Building Information Exchange
OBIS	Object Identification System
OMA	Open Mobile Alliance

OMS	Open Metering System
OSGi	OSGi Alliance / OSGi technology
OWL	Web Ontology Language
OpenIoT	Open Source cloud solution for the Internet of Things
PC	Personal Computer
PHEV	plug in hybrid electric vehicle
PLC	Power Line Carrier
R&D	Research & Development
RDF	Resource Description Framework
REST	REpresentational State Transfer
RF	Radio Frequency
RFC	Request for Comments
SCL	The Service Capability Layer
SD	Study Document
SDK	Software Development Kit
SDO	Standard Development Organization
SEEMPubS	Smart Energy Efficient Middleware for Public Space
SEIPF	Semantic Energy Information Publishing Framework
SensorML	Sensor Model Language
SEP2	Smart Energy Profile 2.0
SG-CG	Smart Grid Coordination Group
SIG	Special Interest Group
SKOS	Simple Knowledge Organization System
SmartCoDe	Smart Control of Demand for Consumption and Supply to enable balanced, energy-positive buildings and neighbourhoods
SML	Smart Message Language
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SSDP	Simple Service Discovery Protocol
SSN	Semantic Sensor Network Ontology
SUMO	Suggested Upper Merged Ontology
SWE	Sensor Web Enablement
TC	Technical Committee
TM	Technical Memorandum
TNO	Netherlands Organisation for Applied Scientific Research TNO
TR	Technical Report
TRV	Thermostat Radiator Valves
TV	Television
ULE	Ultra-Low Energy
UML	Universal Markup Language
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USR	User Preferences
VoCamp	Vocabulary Camp

W3C	World Wide Web Consortium
WG	Working Group
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Network
xDSL	x Digital Subscriber Line
XML	eXtensible Markup Language
XSD	XML Schema Definition Language

1. Introduction

1.1. Context

Achieving higher energy efficiency is an important goal for the European society. The residential and tertiary sector, the major part of which are buildings, accounts for more than 40% of the final energy consumption in the European Community and is expanding, a trend which is bound to increase its energy consumption and hence its carbon dioxide emissions [1]. It is not so much the buildings as such that consume energy and produce greenhouse gasses, but the so-called Energy using and producing Products (EupP), also called “appliances”, inherently present in the buildings’ ecosystems, and the people using them.

An appliance is an instrument or device designed for a particular use or function. About two thirds of the energy consumed by buildings originates from the residential sectors and thus household appliances. Household appliances or home appliances are electrical/mechanical machines which accomplish some household functions, such as cooking or cleaning. The broad definition allows for nearly any device intended for domestic use to be a home appliance, including stoves, refrigerators, toasters, air conditioners as well as TVs, PCs, and light bulbs. Home appliances can be classified into major appliances (or White goods), small appliances (or Brown goods), and consumer electronics (or Shiny goods).

Nowadays, appliances are not stand-alone systems anymore. They are often highly intelligent (“smart”) and networked devices, that form complete energy consuming, producing, and managing systems. Therefore, reducing the use of energy and production of greenhouse gasses is not only a matter of increasing the efficiency of the individual devices, but managing and optimizing the energy utilization at a system level. One of the requirements for making such systems adopted by the mass market, is the flexible and dynamic extension with new smart devices and applications, based on the user’s needs and available budget. The systems will therefore inevitably consist of devices and sensors from different vendors, and open interfaces enabling further extensions. An open interface is a public standard for connecting hardware to hardware and software to software. Said otherwise, networked devices can be managed for energy saving measures if there is a system that can be flexibly enhanced. They also need to be able to communicate with service platforms from different service providers.

In such a system, the interfaces need to be properly standardized and offer external access on a semantic level both to any manageable and controllable function of the system as a whole, and to any device that is part of the system. However, the problem is not the lack of available standards. Actually, there already exist (too) many standards, all dealing with a smaller or larger part of the problem, sometimes overlapping and competing [2]. What is needed is a reference ontology, a shared data model.

Various workshops and FP7 projects already have explored this field and concluded that defining a useful and applicable reference data model should be possible in principle. Several of those exploratory discussions were held at the Energy Efficiency research community at the 2nd (2011) and 3rd (2012) Workshop on eeBuildings Data Models (Energy Efficiency Vocabularies and Ontologies).

These workshops presented results of FP7 and Artemis funded projects[3] related to energy efficiency with different approaches and solutions to bridge over the connectivity standards "jungle" for the smart appliances, but more importantly, explored expanded semantic ontologies to cover broader areas of interactions (more intelligent machine-to-machine "conversations") as the ones covered by the traditional control networks. The conclusion from these workshops were the following: Indeed, one single, reference ontology can be created to cover the needs of all appliances relevant for energy efficiency; indeed, this ontology can be designed in a way that it can be expanded to cover future intelligence requirements; and indeed, this ontology is a rather simple ontology as compared to the state of the art ontology engineering level of complexity. The workshops also concluded that these models show high mapping correlations, and that all what is needed is a formal agreement, a recognised standard and combined efforts of standardization organizations.

However, before launching a formal exercise, the industry was consulted to discover their support and their perception of this need. On 24 September 2012 the European Commission (EC) hosted a workshop on a roadmap for the standardization of smart appliances, inviting all relevant stakeholders:

Stakeholders associations

- Energy Efficient Buildings Association (E2BA)
- CECED, European Committee for Domestic Equipment Manufacturers
- eu.bac, European building automation controls association
- ELC, European Lamp Companies Federation (now succeeded by LightingEurope)
- Smart Grid Task Force

Standardisation Bodies and Organisations

- ETSI M2M (now called ETSI Smart M2M)
- CENELEC TC59x WG7, Smart Grid/Smart Home Activities
- HGI Home Gateway Initiative
- buildingSmart International
- OASIS Open Building Information Exchange (oBIX)
- OSGi Alliance

The main recommendation of this meeting consisted of two objectives:

1. Propose a high-level semantic modelling of information to be exchanged (API-like) – the first step is a common vocabulary for appliances product information, commands, signals (like price or sensor information) and feedback.
    a. Take stock of the existing semantic assets, across different stakeholders and standardisation efforts, and perform a translation exercise. Agree on a nuclear vocabulary.
    b. Discuss a complete range of use cases, covering all devices (white goods, HVAC, plumbing, security and electrical systems, lightings, sensors and actuators (windows, doors, stores), micro renewable home solutions (solar panels, solar heaters, wind, etc.), multimedia and home computer equipment and all Building Energy

[3] E.g. SmarCoDe (www.fp7-smartcode.eu), eDiana (www.artemis-ediana.eu), ENERsip (www.enersip-project.eu), and FIEMSER (www.fiemser.eu)

Management Systems (BEMS), Building Automation and Control Systems (BACS), Customer Energy Managers (CEM), and Energy Boxes as defined by the Consumer Electronics industry, finding the messages and signals they may need to share. Extend the nuclear vocabulary.

2. With regard to connectivity, agree on an abstract architecture with a clear horizon and considering the world's machine-to-machine (M2M) standards, approaches and architectures to bridging the manifold communication layers already available.
    a. Propose available architectures that go in that direction
    b. Create open repositories of reusable pieces

With regard to objective 1, the European Commission has the intention to launch a standardisation exercise at ETSI to propose this high-level model, an ontology for smart appliances, as an ETSI standard. With regard to objective 2, the results should be integrated in the abstraction layer of the ETSI M2M architecture for the Home and Building environment.

1.2. Goal and objectives of this study

To provide this ETSI working group with the relevant background, the European Commission issued a tender for a Study on "Available Semantics Assets for the Interoperability of Smart Appliances. Mapping into a Common Ontology as a M2M Application Layer Semantics" [3] , defining 3 tasks:

- **Task 1:** Take stock of existing semantic assets and use case assets
- **Task 2:** Perform a translation exercise of each model (or use case) to a common ontology language and a mapping or matching exercise between all the models
- **Task 3:** Propose a common ontology and document the ontology into the ETSI M2M architecture

The study will thus contribute with recommendations for a reference ontology, based on semantic assets defined and examined within this study.

TNO was invited to perform this study. The study aims to provide the material needed to define these tools and data models, for the collection of devices that helps the EU to reach its 2020 goals regarding the reduction of greenhouse gas emission and buildings' energy consumption, being the said appliances. The work packages and tasks defined in the study will fulfil the following objectives:

- An overview of existing explicit or implicit semantic assets and use case assets.
- Detailed analysis of the existing semantic assets or requirements in an exhaustive way.
- Proposal for a reference ontology to be contributed to ETSI for consideration as a future standard.
- Documentation of the proposed ontology I nto the ETSI M2M architecture.

The first document, *D-S1 Interim Study Report*, presented the results of task 1 "take stock of existing semantic assets and use case assets". DS-1 was first reviewed by the project's Expert Group, and later on discussed in the 1st stakeholders' workshop that took place in Brussels on May 27/28, 2014. Important changes in DS-1 after the 1st stakeholders' workshop will be addressed in the *D-S4 Final Study report*. This document, *D-S2 Second Interim study report,* covers a translation of the most relevant assets identified in task 1 into OWL ontologies and an initial mapping between these ontologies. The ontologies described here are also published online[4], where the "owners" of the

[4] https://sites.google.com/site/smartappliancesproject/ontologies

corresponding assets can validate whether the meaning they originally intended for their assets is actually reflected in our ontologies. The website provides a page for each of the 20 semantic assets in the short list, with the URL to download the corresponding ontology and a human-readable explanation to describe its main classes and properties. Each page also includes a tab for posting comments (available when logged on to the website with a Google-account). The content of this document has been reviewed by the project's Expert Group. Moreover, DS-2 will be officially presented at the 2nd stakeholders' workshop that will take place at the ETSI premises in Sophia Antipolis on October 15, 2014. This 2nd workshop provides an additional opportunity to discuss the ontologies described here with the stakeholders of the smart appliances domain. It is possible to comment on this *D-S2 Interim Study Report* also at the ETSI website[5]. Any eventual change after the 2nd workshop and until the end of the project in March 2015 will be covered in the online version of the ontologies, and major changes will be addressed in the *D-S4 Final Study report*. The next deliverable, *D-S3 Third Interim study report*, will cover the definition of the smart appliances reference ontology and a description of this ontology within the ETSI M2M architecture. The *D-S4 Final Study report,* will include all the results described in the previous reports, as well as an executive summary.

It should be emphasized that this report, D-S2, is an *Interim* study report. The *D-S4 Final Study report*, to be published next year, is the final result of the study and only D-S4 will be officially passed to ETSI Smart M2M for further development into, as is currently foreseen, a Technical Specification. In D-S4 the results of DS-2 will be updated with the newest insights. This will include an assessment of how the additional assets considered for inclusion in the short list after the first stakeholders' workshop (see section 2.3.2) eventually fit into the reference ontology.

1.3. Structure of the document

Chapter 2 describes the scope of the study and in particular of this document. Moreover, it gives an introduction about ontologies and OWL, which is the language chosen to represent our ontologies. Finally, it outlines the 20 ontologies that are described in this document and the additional assets that were brought to our attention during the first stakeholders' workshop.

Chapter 3 elaborates on the approach that we have followed in task 2 to translate the assets in the short list into OWL ontologies and create an initial mapping between these OWL ontologies.

Chapter 4 is the core of this document in which we present, in alphabetical order, the OWL ontologies corresponding to the assets in the short list. The description of the ontologies follows the same template: for each ontology we provide a *title*, the *source* document used as a main reference to create the ontology, a *description* of the main classes and properties, and, eventually, *observations* necessary to better understand the choices underlying the ontology design and suggestions for its future extension.

Chapter 5 shows an initial mapping of the ontologies by means of a number of concepts that we have identified as most relevant in the smart appliances domain and in the scope of this study. The purpose of this mapping is to relate the 20 assets in the short list using their most recurring concepts (core concepts), as initially identified in task 1, in terms of the ontologies presented in Chapter 4. In order to reduce the number of mappings, a reference ontology is proposed with mappings to/from

[5] http://sap.etsi.org

the ontologies of the specific assets. This reference ontology will be developed in task 3, but an initial proposal of the concepts that can constitute its core is presented in this document.

Chapter 6 presents our conclusions and outlines the activities that will be carried out in task 3.

2. Scope

2.1 Sectors, use cases and appliances

Our study mainly addresses the consumer (mass) market of the home, private dwellings, but also common public buildings and offices, and the standard appliances used in that environment. Elevators and other special equipment are out of scope.

The following appliances are covered:

- Home and buildings sensors (temperature, humidity, energy-plugs, energy clams, energy meters, water-flow, water quality, presence, occupancy, air monitors, environmental sensors, CO_2 sensors, weather stations, etc.) and actuators (windows, doors, stores). Sensors belonging to appliances are treated individually.
- White goods, as classified by CECED[6]
    - Rinsing and Cleaning
    - Cooking and Baking
    - Refrigerating and Freezing
    - Vacuum Cleaning
    - Washing and Drying
- HVAC; heating, ventilation, and air conditioning, plumbing, security and electrical systems, as classified by Eu.bac[7]
- Lighting, with use cases as defined by LightingEurope[8] (f.k.a. ELC)
- Micro renewable home solutions (solar panels, solar heaters, wind, etc.)

Multimedia and home computer equipment devices will be explored only with respect for semantic requirements for the energy relevant operations (switch on, standby), but not for the content management (i.e. channel choice).

The study further covers the following interoperability use cases:

- Interoperability with construction design tools (product information, product performance and product behaviour)
- Interoperability with Facility Management and Energy Management Systems
- Interoperability with Building Control systems
- ESCO (Energy Services) systems
- Interoperability with the Smart Grid

As primary stakeholders the manufacturers of the following home energy producing and consuming products are consulted:

- Manufacturers of white goods
- Manufactures of HVAC, plumbing, security and electrical systems
- Manufacturers of lightings
- Manufacturers of sensors and actuators (windows, doors, stores)
- Manufacturers of micro renewable home solutions (solar panels, solar heaters, wind, etc.)
- Manufacturers of multimedia and home computer equipment

Furthermore stakeholders from directly linked industries are consulted:

[6] European Committee of Domestic Equipment Manufacturers, www.ceced.org
[7] European building automation controls association, www.eubac.org
[8] www.lightingeurope.org, the successor of the former ELC (European Lamp Companies federation)

- Construction industry
- Facility Management and Building Control industry
- ESCO (Energy Services Providers)
- Utilities and operators of the power grid

2.2 About ontologies and OWL

An ontology is here defined as a formal specification of a conceptualization, used to explicit capture the semantics of a certain reality [5,6,7]. As such, we regard an ontology as:

- a set of concepts used to describe the reality under consideration e.g., the concepts of 'household appliance' and 'function';
- precise definitions of these concepts in natural language e.g., 'an appliance is an instrument or device designed for a particular household function, such as cooking or cleaning';
- instances of these concepts, e.g., the specific household appliance of type 'washing machine' from manufacturer 'A and with serial number '123xyz';
- relations among these concepts e.g., a household appliance of type 'washing machine' realizes the function 'cleaning'; and
- axioms to constrain the intended meaning of these concepts, e.g., special conditions under which an appliance should function, such as a specific timeslot during the night when the energy costs are reduced

In this study, ontologies are used to improve the communication among stakeholders, providing a shared understanding that reduces ambiguities and confusion in the terminology adopted in the smart appliances domain. Ontologies are also used here to provide an interpretation to data and, therefore, facilitate interoperability between systems and devices provided by different vendors, providing a reference model that allows translation and mapping among different assets (models/standards/software) from different parties [4].

Ontologies require a language that is suitable to represent the ontology concepts. In practice, people often refer to ontologies as what are in fact specifications of conceptualizations loosely expressed in an informal language, such as natural language. These are not ontologies according to the definition adopted here. In contrast, we consider ontologies as formal specifications expressed using formal semantics and axioms [8]. Informal specifications may lead to ambiguities, and systems that are based on such specifications are more error-prone than systems based on formal ontologies, which, in contrast, allow automated reasoning and consistency checking. Therefore, ontologies expressed using formal semantics are engineering artifacts that can be processed and checked by machines.

It is important to choose a suitable language depending on the purpose of an ontology. Several ontology forms are currently used on the Web, with varying expressiveness. These ontologies range from simple vocabularies, to taxonomies of concepts related by hyponym-hypernym relationships (i.e., *Is-a* relationships), to complete representations of concepts related by complex relationships and axioms. These axioms provide semantics by allowing systems to infer additional information based on the information explicitly provided in the ontology. The Semantic Web[9] provides several languages and technologies that can be used to express ontologies with different degrees of expressiveness. They can be summarised as follows:

[9] http://en.wikipedia.org/wiki/Semantic_Web

- SKOS[10] can be used to model simple lists of terms, such as controlled vocabularies, taxonomies or thesauri
- RDF[11] allows to represent data models as objects (web resources) and relations in terms of (subject, predicate, object) triples, while RDF Schema[12] extends RDF for describing properties and classes of the RDF resources. RDF Schema can be used to express the so called 'lightweight ontologies'.
- OWL[13] is suitable to express formal semantics and, therefore, ontologies. It provides more primitives than RDF Schema to express properties, classes, and axioms to constrain the usage of these properties and classes, allowing a higher degree of semantic reasoning. There are several variants of OWL with a different degree of expressiveness, namely OWL Lite, OWL-DL and OWL Full[14]. OWL Lite was conceived as a lighter version that could support the definition of simple constraints (e.g., cardinality restricted to zero or one), but in practice is not adopted. OWL-DL is the most used and is based on description logic[15], allowing a high degree of expressiveness and reasoning. OWL full is based on a different semantics than the other variants, and was conceived to preserve some compatibility with RDF Schema, however, there is no software able to perform complete reasoning for it (the language is undecidable).

The language chosen in the smart appliances study to express the ontologies corresponding to the semantic assets is OWL-DL, since it provides formal semantics to explicitly represent the meaning intended for these assets, and allows a high degree of semantic reasoning, being supported by a large number of software reasoning tools. In an OWL-DL ontology, the concepts used to describe the reality under consideration are called *classes*, the natural language definitions of these classes can be annotated as *comments*, the instances are called *individuals*, the relations are called *properties*, and the axioms are called *restrictions*.

Several vendors in the smart appliances domain adopt XML to exchange their data. XML provides a basic syntax that allows formatting data and documents, which can be therefore exchanged between machines. However, XML is not a language to provide ontologies because it does not associate any semantics (meaning) to the data being exchanged. In other words, XML covers the syntactical level, while an ontology language, such as RDF Schema or OWL, provides the primitives to express semantics. It is possible to map XML to RDF Shema and/or OWL, and vice versa [9]. A number of tools is available for this purpose[16].

2.3 Ontologies described in this document

2.3.1 Short List

In task 2 we have created ontologies for the following semantic assets, which were part of the short list identified in task1:

- DECT ULE
- DogPower (previously SEIPF)
- Mirabel
- OMA Lightweight M2M

[10] http://www.w3.org/TR/2009/NOTE-skos-primer-20090818/
[11] http://www.w3.org/TR/2004/REC-rdf-primer-20040210/
[12] http://www.w3.org/TR/2004/REC-rdf-schema-20040210/
[13] http://www.w3.org/TR/2009/REC-owl2-primer-20091027/
[14] http://en.wikipedia.org/wiki/Web_Ontology_Language
[15] http://en.wikipedia.org/wiki/Description_logic
[16] http://www.w3.org/wiki/ConverterToRdf#XML

- ECHONET
- eDIANA
- EnOcean
- FAN
- FIEMSER
- FIPA
- Hydra
- KNX
- OMS
- OSGi DAL
- SEEMPubs
- SEP2
- SmartCoDE
- UPnP
- W3C SSN
- Z-Wave

2.3.2 Additional assets identified after the workshop

The results of task 1 "take stock of existing semantic assets and use case assets" presented in the *DS-1 Interim Study Report* were first reviewed by the project's Expert Group, and later on discussed in the first stakeholders' workshop that took place in Brussels on May 27/28, 2014. As a result of the discussions and suggestions made during and after the workshop, we have identified the following additional assets that were missing in DS-1:

- CENELEC-CEM, 'Technical Report IEC 62746-2', Draft version,
  ' Technical Standard of CLC TC205 WG18', Draft version
- DomoML-env , available at http://dl.acm.org/citation.cfm?doid=2031325.2031327
- OpenIoT, available at http://vmusm03.deri.ie/?q=ontology/ns
- ZigBee Home Automation
- ZigBee Light Link

Therefore, the long list presented in the *DS-1 Interim Study Report* will be extended in DS-4 with these additional assets. These assets are currently not translated into OWL ontologies. In the next phase (task 3) we will assess their possible contribution to the study and eventually consider them for taking part in creating the reference ontology for the smart appliances domain as the final output of this project.

3. Approach

In task 1 we have analysed 43 semantic assets that needed to be included in our study given the scope as set out by the European Commission. In the *DS-1 Interim Study Report*, we have defined the initial semantic coverage of these 43 assets, created a visual representation of the key terms used by each asset, and provided a visual representation of the most recurring key terms among all assets. In this way, we were able to short-list 20 semantic assets that provided a good basis for further development of a reference ontology for the smart appliances domain.

Task 2 started from the results of task 1 with the goal of analysing in detail the 20 semantic assets in the short list by 1) performing a translation exercise of each asset into an OWL ontology, and 2) create an initial mapping between the assets described by the different ontologies. In order to perform these tasks, we have followed a systematic approach that allowed us to deal with the quantity of ontologies to be created and their complexity.

The first step was to translate the 20 semantic assets in the short list into ontologies in OWL in order to formally capture their semantics and be able to automatically reason about their content. Out of the 20 assets:

- 4 assets were already expressed in OWL, namely eDIANA, Hydra, PowerOnt (called SEIFP in the previous *DS-1 Interim Study Report*), and W3C SSN. However, only W3C SSN provided an URL to the corresponding OWL file, while the other assets provided detailed documentation about their OWL ontologies, but not an URL. Therefore, we contacted the authors of these ontologies in order to acquire the original OWL file, which are essential for us to make sure that the reference ontology is based on the actual models that were defined in the projects or organizations, and not on our own interpretations of the available documentation.
- 16 assets needed to be translated into OWL from scratch.
    - Some of these assets provided UML-like data models from which we could (more or less) straightforward create a corresponding ontology. Some of these data models were built reflecting specific structures of the underlying implementation languages. We therefore needed to make an extra effort to abstract from these implementation details and capture the actual semantics.
    - Other assets consisted of technical specifications in terms of natural language descriptions, often supported by tables, from which we had to capture the semantics originally intended for these assets by their creators . While some of these specifications were clear and well-structured, supporting us to a great extent in our translation task, other specifications were ambiguous and inconsistent, requiring a major effort in the translation task.
    - Although having XML schema representations was one of the criteria we adopted to select the assets in the short list, we did not use these schemas, when available, to automatically generate OWL from XML. This was a specific design choice we made, given the resources that could be allocated for the translation task. In fact, it was more effective to create the OWL ontologies top-down, extracting the semantics from the natural language descriptions and tables provided by the assets, rather than bottom-up, by first automatically generating OWL from XML, and then having to edit the result in order to make a proper OWL ontology out of it. Notice that this choice still allows future

extensions in terms of mappings from the OWL ontologies to XML for the interested stakeholders.

In this way, for some assets we have reused existing work created by other renowned bodies and organizations, rather than creating new ontologies from scratch, according to best practices in ontology engineering. When translating the other 16 assets into OWL, we took care of expressing the intended semantics correctly so that others, both machines and humans, could properly understand and reuse the ontologies being created. Particularly, we have created the ontologies in this document with the following best practices in mind:

1. include basic metadata that allow others to correctly understand and properly reuse the ontology being built, such as *creator*, *date of issue*, *title*, *description* and *source* of the ontology;
2. make the ontology self-descriptive by using labels, definitions and comments for each class or property;
3. provide proper documentation not only using label and comments, but creating a human-readable description that explains the main classes and properties;
4. make the ontology accessible for a long period by providing some guarantee of maintenance;
5. publish the ontology at a stable URL to guarantee persistent access, and facilitate reusability.

According to these principles, the second step of our approach was to create a human-readable description that reflects the content of the OWL files, in order to support the reader in understanding the main concepts and navigating the ontologies. These descriptions are presented in Chapter 4, together with important observations concerning the development of the ontologies and suggestions for future extensions.

The third step was to publish the ontologies at a stable URL[17] in order to guarantee persistent access and facilitate their (re)usability in the smart appliances community. The smart appliances website provides a page for each of the 20 assets in the short list, with the URL to download the corresponding ontology and a human-readable explanation to describe the main classes and properties. In order to guarantee transparency during the process and take into account the feedback of the stakeholders, especially the "owners" of the considered assets, each page includes a tab for posting comments (available when logged on to the website with a Google-account).

In parallel to the steps described above, the fourth step was to extend the long list of 43 assets considered in the *DS-1 Interim Study Report* with assets that resulted to be missing after the discussions that took place at the 1st stakeholders' workshop. Therefore, we contacted the authors/owners of these assets to obtain relevant material, and, if existing, to acquire the original OWL files. In the next phase (task 3) we will assess the possible contribution of these additional assets to our study and eventually consider them for creating the reference ontology for the smart appliances domain.

[17] https://sites.google.com/site/smartappliancesproject/ontologies

4. Ontologies

The following sections present the ontologies that we have created[18] providing their *title*, the *source* document used as a main reference to create the ontology, a *description* of the main classes and properties, and, eventually, *observations* necessary to better understand the choices underlying the ontology design and suggestions for its future extension. Notice that the project does not have the resources to elaborate every ontology in OWL in all possible detail. However, we think this is not necessary as, to achieve the final goal of the project, we only need to find the commonalities between the various ontologies. Moreover, having learned from our approach, every stakeholder can now do the work himself and improve/extend the ontology to his liking given the open character of our results. These ontologies have to be considered as an intermediate result that allows us to achieve the final goal of the project, namely provide a reference ontology for the smart appliances domain to create semantic interoperability among assets from different stakeholders, but they are not the ultimate result of this project themselves.

These ontologies and their descriptions are also published online, where the "owners" of the corresponding assets can check them to validate whether the meaning they originally intended for their assets is actually reflected in our ontologies. Moreover, the content of this document will be presented at the 2nd stakeholders' workshop, which provides an additional opportunity to discuss the ontologies described here with the stakeholders of the smart appliances domain. Any eventual change after the workshop and until the end of the project in March 2015 will be covered in the online version of the ontologies, and major changes will be addressed in the *D-S4 Final Study report*, which will be officially passed to ETSI Smart M2M for further development into, as is currently foreseen, a Technical Specification.

Table 2. Ontologies overview

Ontology title	Source Language	URL	Number of classes	Number of Object properties	Number of Datatype properties
Dectule	Natural Language + Tables + XML	https://sites.google.com/site/smartappliancesproject/ontologies/dect_ule-ontology	76	6	4
Echonet	Natural Language + Tables	https://sites.google.com/site/smartappliancesproject/ontologies/echonet-ontology	187	27	2
Ediana	OWL	https://sites.google.com/site/smartappliancesproject/ontologies/ediana-ontology	70	25	12
Enocean	Natural Language + Tables	https://sites.google.com/site/smartappliancesproject/ontologies/enocean-ontology	240	4	3
Fanfpai	Natural	https://sites.google.com/site/smartapplianc	31	16	17

[18] These ontologies are expressed in OWL-DL and serialized in Turtle (therefore, their file extension is ".ttl"), which is a compact syntax alternative to RDF/XML. These ontologies can be opened with any ontology editor, such as TopBraid Composer, Protégé and NeOn.

	Language + Javadoc	esproject/ontologies/fan-ontology			
Fiemser	Natural Language + UML	https://sites.google.com/site/smartapplianc esproject/ontologies/fiemser-ontology	47	5	34
Fipa	Natural Language + Tables + UML	https://sites.google.com/site/smartapplianc esproject/ontologies/fipa-ontology	14	14	18
Hydra device	OWL	https://sites.google.com/site/smartapplianc esproject/ontologies/hydra-ontology	66	9	14
Knx	Natural Language + Tables	https://sites.google.com/site/smartapplianc esproject/ontologies/knx-ontology	20	1	3
Mirabel	Natural Language + Tables + UML	https://sites.google.com/site/smartapplianc esproject/ontologies/mirabel-ontology	24	16	0
Omalwm2m	Natural Language + Tables + XML	https://sites.google.com/site/smartapplianc esproject/ontologies/oma-lightweight_m2m-ontology	30	4	7
Oms	Natural Language + Tables	https://sites.google.com/site/smartapplianc esproject/ontologies/oms-ontology	67	0	1
Osgidal	Natural Language + Javadoc	https://sites.google.com/site/smartapplianc esproject/ontologies/osgi_dal-ontology	17	9	19
Poweront	OWL	https://sites.google.com/site/smartapplianc esproject/ontologies/dogpower-ontology	945	41	60
Seempubs	Natural Language + Tables	https://sites.google.com/site/smartapplianc esproject/ontologies/seempubs-ontology	44	9	4
Sep2	Natural Language + Tables +UML + XSD	https://sites.google.com/site/smartapplianc esproject/ontologies/sep2-ontology	39	5	12
Smartcode	Natural Language + Tables	https://sites.google.com/site/smartapplianc esproject/ontologies/smartcode-ontology	32	3	4
Upnp	Natural Language + XSD + XML	https://sites.google.com/site/smartapplianc esproject/ontologies/upnp-ontology	8	11	23
W3C SSN	OWL	https://sites.google.com/site/smartapplianc esproject/ontologies/w3c_ssn-ontology	116	137	6
Z-Wave	Natural Language + Code lists	https://sites.google.com/site/smartapplianc esproject/ontologies/z-wave-ontology	77	6	0

4.1 DECT ULE

Ontology title

Dectule: Digital Enhanced Cordless Telecommunications (DECT) Ultra-Low Energy (ULE)

Source

'HF-Protocol', 'HF-Service', 'HF-Interface', 'HF-Profile', version1.0, 23 January 2014, available at (for free after registration) http://www.ulealliance.org/registration.aspx?f=11

Ontology description

The DECT ULE HF standard is based on a star network topology of network entities represented by the `HFNetworkEntity` class in the DECT ULE ontology. A `HFNetworkEntity` can be a `HFConcentrator`, which is the network's master device, or a `HFDevice`. There are up to thousands of devices supported by the concentrator and connected to it. The HF protocol supports several types of HF messages exchanged between network entities (i.e., commands, requests, responses), and each of these messages has a message type code. A `HFMessage` is structured in 3 fields (i.e., network, transport and application layers).

A `HFNetworkEntity` implements one or more services. A `HFService` is either fundamental for the correct operation of a HF network, or provides advanced network features that may be useful on certain applications. Services from the latter category are optional to implement, while fundamental services are mandatory. Mandatory services for the `HFConcentrator` are the `AttributeReporting`, `BindManagement`, `DeviceInformation` and `DeviceManagement` services. Mandatory services for a `HFDevice` are the `AttributeReporting`, `DeviceInformation` and `DeviceManagement` services.

An `HFUnit` is a conceptual entity inside a `HFDevice` that instantiates the functionality of a specific type. There are several unit types, namely `HomeControlUnitType` (Simple On-Off Switchable, Simple On-Off Switch, Simple Level Controllable, Simple Level Control, Simple Level Controllable Switchable, Simple Level Control switch, AC Outlet, AC Outlet with Simple Power Metering, Simple Light, Dimmable Light, Dimmer Switch, Simple Door Lock, Simple Door Bell, Simple Power Meter), `SecurityUnitType` (Simple Detector, Door Open Close Detector, Window Open Close Detector, Motion Detector, Smoke Detector, Gas Detector, Flood Detector, Glass Break Detector, Vibration Detector, Siren), `HomecareUnitType` (Simple Pendant), `ApplicationUnitType` (User Interface, Generic Application Logic), and `ProprietaryUnitType`. Each `HFUnit` has a unique identifier.

An `HFInterface` is a conceptual entity inside a `HFUnit` that defines a collection of commands and attributes, allowing for units to understand each another. Interfaces can be mandatory or optional to implement by a unit, and they have a role – client or server – associated with them. An `HFInterface` has attributes.

Observations

– The attributes of the `HFInterface` class are not defined in this version of the ontology because their level of granularity is too fine. However, this ontology can be extended to cover also the

attributes by adding them under the `Attribute` class, according to the HF-Interface specification.

4.2 ECHONET

Ontology title

Echonet: Energy Conservation and HOmecare NETwork (ECHONET) for Device Objects

Source

ECHONET Specifications Appendix 'Detailed Requirements for ECHONET Device Objects' Release C, 31 May 2013, available at
http://www.echonet.gr.jp/english/spec/pdf_spec_app_c_e/SpecAppendixC_e.pdf

Description

The Echonet ontology represents Echonet device objects and their properties. A `Device` defines one or more `DeviceObject`. Device objects represent mechanical functions of a device and aim at facilitating controls and status verification through communications between devices. There are general properties applicable to any device object, such as `hasOperationStatus`. These general properties are defined as subproperties of the `hasDeviceObjectProperty` property.

There are 7 groups of device objects, namely `AirConditionerRelatedDevice`, `AudiovisualRelatedDevice`, `CookingHouseholdRelatedDevice`, `HealthRelatedDevice`, `HousingFacilitiesRelatedDevice`, `ManagementOperationRelatedDevice` and `SensorRelatedDevice`. Each group has a corresponding code (`hasGroupCode` property) and is characterized by the `hasOperationStatus` property, which indicates whether the function native to this group of objects is operating or not (ON/OFF). The `CodeList` class defines enumerations used to represent admitted values for some properties, for example, the `OperationStatus` class defines the instances `On` and `Off` as admitted values for the `hasOperationStatus` property.

Each of the 7 groups mentioned above is further refined in device object subclasses with a specific code (`hasClassCode` property). For example, the `AirConditionerRelatedDevice` group includes the `AirCleaner`, `Dehumidifier`, `ElectricHeater`, and `HomeAirConditioner` classes, among others. Specific properties that characterize a certain device class, but not any device object, are defined as subproperties of the `hasClassSpecificProperty` property. For example, the `hasOperationModeSetting` property characterizes the `HomeAirConditioner` class.

Observations

- The general properties applicable to any device object, such as `hasOperationStatus` are defined globally under the `hasDeviceObjectProperty` property. The amount of `DeviceObject` classes was such that it was not possible to restrict (the cardinality of) these properties for all classes. For the future it is advised to restrict these properties locally at the level of each `DeviceObject` subclass.
- The amount of `hasClassSpecificProperty` properties specific to device classes, but not applicable to any device object, was such that it was not possible to define them all. However, we have defined three properties (`hasDetectionThresholdLevel`, `hasOpenCloseSetting`,

and hasOperationModeSetting) that should be used as example to further populate the hasClassSpecificProperty according to the ECHONET Device Objects specification.

4.3 eDIANA

Ontology title

Ediana: Embedded Systems for Energy Efficient Buildings (eDIANA) ontology for Device Awareness

Source

'D2.2-A Ontology for Device Awareness', 30 November 2009, available at http://s15723044.onlinehome-server.info/artemise/documents/D22A_Ontology_for_Device_Awareness_m10_IMSML.pdf

Ontology description

The eDIANA ontology defines the universe of concepts and their relations in the domain of eDIANA Platform Architecture, related to device awareness. The ontology defines three main classes, namely the Information, Service and Device classes. The Information class contains the different categories of information that will be referenced by the elements defined in the Service and Device classes. It includes Direction_Information, Comfort_Variable_Information (such as Humidity_Information, Luminosity_Information, Noise_Information, and Temperature_Information), and Smart_Actuator_Command_Information (such as Change_Configuration_Command_Information, Delayed_Turn_Off_Command_Information, Delayed_Turn_On_Command_Information, Turn_Off_Command_Information, and Turn_On_Command_Information).

The Service class specifies the different interfaces at a very high level,. They are divided in External_Services and Internal_Services. The concrete definition of these interfaces is recommended as future work in the document used as source of the ontology.

The Device class contains different categories of devices that compose the eDIANA platform to enable device awareness services and plug-and-play services, by characterizing the devices, their properties and their interfaces. Devices include Concentrator, Actuator, Appliance (including Generator, Load, Storage), Sensor (including Video_Camera, Airflow_Sensor, Gas_Sensor, Humidity_Sensor, Light_Sensor, Power_Sensor, Sound_Sensor, Sun_Radiation_Sensor, Temperature_Sensor, Fire_Sensor, Movement_Sensor and Smoke_Sensor), and User Interface.

Observations

– We did not create the eDIANA ontology. We are reusing the OWL version that was provided to us by the authors of the 'D2.2-A Ontology for Device Awareness' document.

4.4 EnOcean

Ontology title

Enocean: EnOcean Alliance Equipment Profile (EEP)

Source
'EnOcean Equipment Profiles (EEP) ', Version 2.6, 17 December 2013, available at http://www.enocean-alliance.org/eep/

Ontology description
The Enocean ontology specifies the user data embedded in the structure of a radio telegram as defined by the EnOcean Equipment Profile (EEP). Therefore, the ontology defines an `EEP_profile` class. Through the `hasElement` property, the `EEP_profile` class is characterized by 3 elements:

- the `RORG` class, which represents the ERP radio telegram type using a code, for example, the value `F6` represents an RPS telegram type;
- the `FUNC` class, which represents the basic functionality of the data contained in a radio telegram, for example, `TemperatureSensor`, `AutomatedMeterReading`, `Detector`, and `HVAC_component`; and
- the `TYPE` class, which represents the specific characteristics of a device type, for example, a temperature sensor with range between -10°C and 30°C (`TemperatureSensor_range-10CtoC30C` class).

The ontology defines 4 types of telegrams according to the EEP profile, namely RPS, 1BS, 4BS and VLD, which are represented by the corresponding classes `TelegramRPS`, `Telegram1BS`, `Telegram4BS`, and `TelegramVLD`, respectively. Each telegram has a RORG (`hasRORG` property), and can have several device functions (`hasFUNC` property) and types (`hasTYPE` property).
Each `RORG` class, `FUNC` class and `TYPE` class has a code (`hasRorgCode` property, `hasFuncCode` property and `hasTypeCode` property, respectively). These codes are used to assemble the 3 field code that characterizes a specific telegram. For example, the code `A5_02_04` characterizes a 4BS telegram (`hasRorgCode` with value `A5`) with a temperature sensor function (`hasFuncCode` with value `02`) and a temperature sensor type with range between -10°C and 30°C (`hasTypeCode` property with value `04`).

Observations

- The TYPES are defined completely for the `TelegramRPS` and `Telegram1BS` classes. For the `Telegram4BS` class the TYPES are defined until and including the `A5_10` subclass. For the `TelegramVLD` class the TYPES are not defined at all. For completeness, it is advised to add the remaining TYPES in the future.
- The EEP document defines enumerations that are used to further characterize the specific TYPE of telegrams. These enumerations are too many and too detailed to be included in the current version of the ontology. However, the ontology could be extended in the future to cover also this aspect of the EnOcean Equipment Profile.
- The source used to create the ontology is a secured pdf from which the information could not be automatically copied. As a consequence, comments that could better explain the telegrams are missing in the ontology.

4.5 FAN

Ontology title

Fanfpai: Flexible Power Alliance Network (FAN) Flexible Power Application Infrastructure (FPAI) ontology

Source
'Interface description: Interface report', Version 1.0 (final), 7th January, 2014, available at (for free after registration) http://www.flexiblepower.org/downloads/

Ontology description
The Fanfpai ontology describes the resources (appliances) used in the Flexible Power Application Infrastructure (FPAI) . These resources are defined in the Resource Abstraction Interface (`RAI` class), which is used to express the energetic flexibility that appliances can offer and how this flexibility should be exploited. The RAI is an interface layer between:

- the Resource Abstraction Layer (`RAL` class) that monitors and controls the appliances and knows how much flexibility they can offer. The RAL consists of two main components: the resource manager (`ResourceManager` class) and the resource driver (not considered in this ontology);
- the energy apps (`EnergyApp` class) that are typically provided by a third party and exploit the flexibility that appliances have to offer. An energy app is only interested in exploiting energetic flexibility and not in the details of a specific appliance, such as a washing machine, for instance.

A `Resource` represents an appliance within a household or a building that can provide flexibility with regard to consumption, storage and production of energy. There are several type of resources defined in FPAI:

- `TimeShifter` resources, which are a category of appliances that produce or consume energy according to a predetermined energy profile and whose flexibility comes from their ability to shift the start time of this profile. Typical examples of time shifting appliances are `WashingMachine`, `DishWasher`, `AutomaticVacuumCleaner`;
- `Buffer` resources, which are a category of appliances that can provide electrical flexibility. With a buffer appliance one can choose to consume/produce more energy now (within certain operational constraints) so that it consumes/produces less energy later, or the other way around. Most buffers are thermal. Examples of such appliances are `Refrigerator`, `Freezer`, `HeatingSystem`;
- `EnergyStorage`, which is category of appliances similar to buffers, but with the main difference that with buffers the electrical energy only flows in one direction: it is either consumption or generation, while with a storage appliance the electrical energy flow is bidirectional. The storage category includes self-discharging batteries (e.g. Li-In/NiMH batteries), chemical storage batteries with conversion loss (e.g. flow batteries), and mobile storage in electrical vehicles;
- `UncontrolledLoadOrGeneration`, which is a category reserved for appliances that cannot be controlled and, as a consequence, cannot offer flexibility. It is however important to know how these appliances behave energetically to make informed decisions about the usage of flexibility in the other three categories. A `SolarPanel` is an example of an uncontrolled generation appliance, which generates energy that cannot be controlled since it depends on external natural conditions (i.e. the weather), whereas `Lighting` represents an uncontrolled load.

Important concepts in the ontology are `ControlSpace`, which is used to describe the energetic flexibility of a particular resource/appliance, and `Allocation`, which indicates how this flexibility

should be used. The ResourceManager constructs and communicates a ControlSpace. In response to a ControlSpace communication, a ResourceManager can receive an Allocation, which contains a precise EnergyProfile that the ResourceManager should try to follow as closely as possible. There are several control spaces that correspond to the different categories of resources, namely TimeShifterControlSpace, BufferControlSpace, EnergyStorageControlSpace, and UncontrolledControlSpace. Each control space is characterized by specific properties. The ontology also defines the Energy, Power, Duration classes and their corresponding units of measure.

Observations
None.

4.6 FIEMSER

Ontology title
Fiemser: Friendly Intelligent Energy Management Systems in Residential Buildings (FIEMSER) ontology

Source
'D5 FIEMSER Data Model', February 2011, available at http://www.fiemser.eu/wp-content/uploads/2011/12/D5_FIEMSER-data-model_m9_CSTmb_REVIEW.pdf

Ontology description
The Fiemser ontology describes the main classes of the Energy-focused BIM model and WSN-related data that are part of the FIEMSER data model. The ontology describes the building space organization in terms of the Building, BuildingPartition, BuildingSpace and BuildingZone classes. A building partition defines a part of a building managed by either a dweller (e.g., a flat) or a facility manager (e.g., a common building area). A building space defines the physical spaces of the building. A building zone defines a functional area in the building that will be controlled as a unique zone. A building consistsOf some building partitions, a building partition consistsOf some building spaces, a building zone consistsOf some building spaces. The Fiemser ontology also describes the devices (Device class) used in the building in terms of HomeEquipment and ControlledDevice.

A HomeEquipment is any home appliance or mechanism to increase building energy efficiency, such as Generator, Load, Mechanism and Storage. Generators represent devices that provide part of the energy required by the building, for example, PV (of type ElectricalGenerator) and Boiler (of type ThermalGenerator). Loads represent devices that consume energy and offer a service to the user, for example, TV (of type ElectricalLoad) and Radiator (of type ThermalLoad). Mechanisms represent devices that are installed in the home to increase its energy efficiency, but don not generate or consume energy by themselves, for example, a Blinder. Storage devices represent devices that store energy and can be used to provide convenient energy management strategy, for example, Battery (of type ElectricalStorage) and Tank (of type ThermalStorage).

A ControlDevice represents a device directly connected to the FIEMSER control infrastructure and used to monitor and/or control the environment and its appliances. A control device consistsOf

some ControlComponent that can be a hardware component (Sensor or Actuator or CommDevice) and a software component. An Actuator is any actuating hardware installed in a control device, such as a Dimmer, Switch and Controller. A Sensor can be a MeasurementSensor (e.g., thermostat) or StateSensor (e.g., presence). A communication device (CommComponent) identifies the communication devices used for data exchange and uses a specific Network protocol (NetProtocol class).

Observations

- The Fiemser ontology describes the main classes of the Energy-focused BIM model and WSN-related data that are part of the FIEMSER data model. Although the other 6 models of the FIEMSER data model contain relevant information also, we decided not to include them in the current version to keep the size of this ontology balanced in comparison with the other ontologies. It is therefore advised to do so as part of future work.
- The source used to create the ontology is a secured pdf from which the information could not be automatically copied. As a consequence, comments that could better explain the ontology may be missing.

4.7 FIPA

Ontology title

Fipa: Foundation for Intelligent Physical Agents (FIPA) Device Ontology

Source

'FIPA Device Ontology Specification', document number SC00091E , 3 December 2002, available at http://www.fipa.org/specs/fipa00091/SI00091E.pdf

Ontology description

The Fipa ontology describes a device ontology that aims at enabling interoperability between software agents, as defined by the FIPA Device Ontology Specification. This ontology can be used by agents when communicating about devices: when agents pass profiles of devices to each other, these profiles can be validated using the information contained in this ontology.

The main class of the ontology is the Device class, which defines a device and its general properties. A device has some InfoDescription, such as the name, vendor and version of the product under consideration, and has some hardware and software properties. Software properties include the details of the device's operating system (hasOperatingSystem), such as its name, vendor and version. Hardware properties are the type of connection that the device uses (hasConnection), the amount of memory that it requires (hasMemory), the user interfaces offered by the device (hasUserInterface), and the type of central processing unit (hasCPU). The connection type is expressed in terms of name, vendor and version of the connection provider (hasConnectionInfo). The MemoryTypeDescription class defines the unit of measure of the memory (hasMemoryUnit), and its usage type, namely application, storage, or both application and storage (hasMemoryUsageType). The UIDescription class defines the information that characterize the screen of the device (hasScreen), such as its width (hasWidth), height (hasHeight), resolution (hasResolution), and the measurement units (hasWidhtHeightUnit). The ontology also defines

the RequestDeviceInfo function that can be used in the FIPA framework by an agent to make a query to request the device information contained in the ontology.

Observations

- We have created an OWL version of the FIPA ontology according to the FIPA device ontology specification. This specification refers to some classes defined in other FIPA ontologies, namely the FIPA-Nomadic-Application and FIPA-Agent-Management ontologies. These ontologies are out of the scope of this study and, therefore, have not been translated to OWL. However, our Fipa ontology can be extend to consider the FIPA-Nomadic-Application by using the AgentPlatform class, and the FIPA-Agent-Management ontologies by using the QoS class.

4.8 HYDRA

Ontology Title

Hydra: Heterogeneous physical devices in a distributed architecture (HYDRA) ontology

Source

'Deliverable D6.6 Updated MDA Design Document', version 1.0, 20 August 2009, http://www.hydramiddleware.eu/hydra_documents/D6.6_Updated_MDA_Design_Document.pdf

Ontology description

There are several ontologies developed in the Hydra project, but for the purpose of this study we are mainly interested in the Device ontology, which consists of the following modules:

- Basic Device Information
- Device Services
- Device Events
- Device Malfunctions
- Device Capabilities and State Machine

The Basic Device Information module represents general device information. The HydraDevice is the main ontology class, which is further divided in the PhysicalDevice and the SemanticDevice classes. Physical and semantic devices share common device properties, such as deviceId or inLocation, but have different semantic interpretation and behaviour. The HydraDevice class refers to the InfoDescription class using the info property. The InfoDescription class contains basic information about device friendlyName, manufacturer data, i.e., manufacturerName and manufacturerURL, and device model data, i.e., modelName, modelDescription and modelNumber. An important part of the basic device information is the representation of device type modelled as sub classes of the PhysicalDevice concept, such as SensorDevice, ActuatorDevice, MediaDevice and MobileDevice. Further, the hasEmbeddedDevice property of the SemanticDevice class recursively refers to HydraDevice concept. This property enables the creation of models of composite devices, such as in case of the HeatingSystem device, which can be, for example, composed of Thermometer and Pump devices.

Observations

– We did not create the Hydra ontology. We are reusing the description of the ontology in the 'Deliverable D6.6 Updated MDA Design Document' and the OWL files provided by the authors of this document.

4.9 KNX

Ontology Title

Knx: Konnex (KNX) ontology

Source

- 'KNX System Specifications Interworking Datapoint Types', Version 1.07.00, 26 April 2010
- 'KNX Advanced Course- Interworking_E1209b'

Ontology description

The Knx ontology represents the several types of data point defined in the KNX specification, namely Datapoint types for common use (`DatapointType4CommonUse` class), Datapoint types for HVAC (`DatapointType4HVAC` class), Datapoint types for Load Management (`DatapointType4LoadManagement` class), Datapoint types for Lighting (`DatapointType4Lighting` class), and Datapoint types for Systems (`DatapointType4System` class). Examples of `DatapointType4CommonUse` are `DPT_ControlDimming`, `DPT_Scaling`, `DPT_Step`, `DPT_Switch` and `DPT_UpDown`. Each data point type has an identifier with an allowed value range.

Combinations of data point types in a device are called functional blocks (`FunctionalBlock` class). Many functional blocks have been defined by KNX, but only two of them have been standardized: the `DimmerActuatorBasic` and `SunblindActuatorBasic` , which are represented in the Knx ontology. The other functional blocks are not standardized since they are not tested and certified as such. A KNX certification means that all the relevant data point types have been implemented correctly.

The `DimmerActuatorBasic` functional block combines the `DPT_ControlDimming`, `DPT_Scaling` and `DPT_Switch` data point types. This functional block can be in one of the 3 states `{"On", "Off", or "Dimming"}`, which are defined under the `DimmingActuatorBasicState` class. The change from one state to another is triggered by the so-called Events with one of the values listed under the `DimmingActuatorBasicEvent` class.
The `SunblindActuatorBasic` functional block combines the `DPT_Step` and `DPT_UpDown` data point types. This functional block can be in one of the 4 states `{" Stopped", "InMotion", or "StepUp", "StepDown"}`, which are defined under the `SunblindActuatorBasicState` class. The change from one state to another is triggered by events with one of the values listed under the `SunblindActuatorBasicEvent` class.

Observations

– A very large amount of Datapoint types are defined. Unfortunately, we could not represent them all in this initial version of the ontology. However, we have defined the `DatapointType4CommonUse` that were relevant to define the `DimmingActuatorBasic` and `SunblindActuatorBasic` functional blocks, which are the only two functional block

standardized and certified. The other data points need to be added eventually according to the KNX specification used as source of this ontology.

4.10 MIRABEL

Ontology Title

Mirabel: Micro-Request-Based Aggregation, Forecasting and Scheduling of Energy Demand, Supply and Distribution (MIRABEL) ontology

Source

'D7.5 MIRABEL-ONE: Initial draft of the MIRABEL Standard', version1.0, 22 December 2011, available at http://wwwdb.inf.tu-dresden.de/miracle/publications/D7.5.pdf

Ontology description

The Mirabel ontology defines how actors can express in the form of user preferences their energy flexibility for a specific device with respect to amount, time and price. Each device has an energy profile that describes the amount of energy consumed and/or produced over a certain time span. A flex offer is issued by an actor and combines the user preferences with the corresponding device energy profile.

The `User` class represents the person using the device. This person may own the device or not. Depending on the device, the user can take the role of a consumer, producer or prosumer. A user expresses his user preferences with respect to a device. A user `uses` a device and `specifies` some preferences. A `Device` is an electricity consuming and/or producing appliance. Three types of devices can be identified, namely energy production, consumption and storage devices (`ProducingDevice`, `ConsumingDevice` and `StorageDevice` classes, respectively). A `Preference` describes the minimum demand by a user for the electrical consumption/production of a device. A `Preference` is expressed with respect to time, price and amount constraints (consists of `exactly 1 Amount`, `exactly 1 Price`, `min 1 TimePoint`, `max 2 TimePoint`). Time can be expressed as one point in time (`TimePoint`) or as several points in time (`TimeInterval`). `TimePoint` is either expressed as `CalendarTime` or as a `RelativeTimePoint`. The `Price` represents the minimum/maximum price that the user is willing to pay for energy production/consumption. The `Amount` can be an absolute amount (`AbsoluteAmount`) or a percentage of this amount (`RelativeAmount`).

An `EnergyProfile` describes the energy load production and/or consumption of a device over a time span. It specifies the profile in terms of time (`consistsOf exactly 1 TimeInterval`) and indicates whether or not there can be breaks/ interruptions between the end of one interval and the start of the next interval. The energy profile also specifies power and/or energy (`consistsOf max 1 Power`, `max 1 Energy`) in terms of an `Amount`.

An `Offer` combines the user `Preference` and the `EnergyProfile` of a device. It can either be a `SingleOffer` or `CompositeOffer`. A single offer is a `FlexOffer` that consists of a unary expression with only one operand. A composite offer is a `FlexOffer` that consists of a binary expression with 2 operands connected using the conditional elements "AND, OR". Through a `FlexOffer` the flexibility in energy supply and demand can be offered on a marketplace. The

FlexOffer is issued by a FlexEnergyIssuer (this can be the same person as the user) and submitted to a FlexEnergyAcquirer.

Observations
None

4.11 OMA Lightweight M2M

Ontology Title
Omalwm2m: Open Mobile Alliance (OMA) Lightweight (LW) Machine-to-Machine (M2M) ontology

Source
'OMA Lightweight Machine-to-Machine Technical Specification', candidate version 1.0, 10 December 2013, available at
http://technical.openmobilealliance.org/Technical/release_program/docs/LightweightM2M/V1_0-20131210-C/OMA-TS-LightweightM2M-V1_0-20131210-C.pdf

Ontology description
The OMA LWM2M architecture is based on a client component, which resides in the LWM2M Device, and a server component, which resides within the M2M Service Provider or the Network Service Provider. Each piece of information made available by the client is a resource. A client may have any number of resources and these resources are organized into objects. Each resource supports one or more operations. The Omalwm2m ontology describes the resources, objects and operations supported by the OMA LWM2M architecture.

An object (Object class) consists of one or more resources (Resource class). Different resources are organized into an object. Each Object has a unique identifier (hasObjectID property), and each Resource has a unique identifier within the object it belongs to (hasResourceID property). Both objects and resources can be mandatory or optional (isMandatory property). Both objects and resources can have a single instance (hasInstance min 0 and hasInstance max 1 constraints) or multiple instances (hasInstance min 0 constraint), represented by the ObjectInstance and ResourceInstance classes, respectively. Each object instance and resource instance have a unique identifier (hasObjectInstanceID and hasResourceInstanceID properties). A Resource supports one or more operations (hasOperation property). Examples of operations are Read, Write, and Execute.

According to the OMA Lightweight Machine-to-Machine Technical Specification, the Omalwm2m ontology defines several type of objects, namely the LWM2MSecurity, LWM2MServer, LWM2MAccessControl, LWM2MDevice, LWM2MConnectivityMonitoring, LWM2MFirmware, LWM2MLocation, and LWM2MConnectivityStatistics objects. Each object has a unique identifier (hasObjectID property). The ontology further details the LWM2MDevice object and its corresponding resources, such as, for example, the Manufacturer, PowerSourceCurrent, and AvailablePowerSource resources, among others. The LWM2MDevice object hasObjectID with value "3", supports a single object instance, and is a mandatory object. The Manufacturer resource hasResourceID with value "0", supports a single resource instance, supports the "Read" operation, and is optional for the LWM2MDevice object.

Observations

- Not all the OMA LWM2M objects as defined in the 'OMA Lightweight Machine-to-Machine Technical Specification' are in the scope of our study. Therefore, we have chosen to include details in our ontology of only the `LWM2MDevice` object and its corresponding resources. Following the same approach, the ontology can be extended with the details of the other objects (`LWM2MSecurity`, `LWM2MServer`, `LWM2MAccessControl`, `LWM2MConnectivityMonitoring`, `LWM2MFirmware`, `LWM2MLocation`, and `LWM2MConnectivityStatistics`), which are currently "open" classes, i.e., these classes are defined in the ontology, but they may be further detailed.

4.12 OMS

Ontology Title

Oms: Open Metering System (OMS) ontology.

Source

'Open Metering System Specification Vol.2 – Primary Communication Issue 4.0.2', 27 January 2014, available at http://oms-group.org/fileadmin/pdf/OMS-Spec_Vol2_Primary_v402.pdf

Ontology description

The Oms ontology is a taxonomy of the devices supported by the Open Metering System (OMS) specification. The `Device` class represents these devices. Each device has a corresponding code (`hasCode` property). There are two type of devices: the devices that are actually supported (`SupportedDevice` class) and the devices that are not certifiable (`NotCertifiableDevice` class). The OMS specification covers the devices under the `SupportedDevice` class, which are further classified in `OMSMeter` and `OMSDevice`. The `OMSMeter` class covers the meter type of devices, such as `ElectricityMeter`, `GasMeter`, `HeatMeter` and `WaterMeter`, among others. The `OMSDevice` class covers other types of devices, such as `Breaker`, `BidirectionalRepeater` and `CommunicationController`, among others. The devices under the `NotCertifiableDevice` class may also be integrated in the Open Metering System. However these devices cannot be approved by the OMS-Compliance Test. Therefore the interoperability for the devices under the `NotCertifiableDevice` class cannot be guaranteed.

The Oms ontology also provides a taxonomy of the DataPoints according to the OMS-Data Point List (OMS-DPL), such as, for example, the current in Ampere (`CA` class), the energy in Joule or Watt hour (`EJ` and `EW` classes, respectively), the temperature in °C (`TC` class), the volume in cubic meters (`VM` class) and the voltage in Volt (`VV` class).

Observations

- The Oms ontology focuses on the type of metering devices and does not consider the architecture of the Open Meter System, such as the Multi Utility Communication Controller (MUC), which is the hardware system used to readout different metering devices and to transfer subsets of this data to AMM back office systems for billing, servicing or other purposes.

– The DataPoint classes are “open” classes, i.e., these classes are defined in the ontology, but they may be further detailed. As future work, the ontology can be extended to take into account the separation of OMS-Datapoints in M-Bus tags and VIB-type lists.

4.13 OSGi DAL

Ontology Title

Osgidal: OSGi Device Abstraction Layer (DAL) ontology

Source

'RFC-196 OSGi Alliance Device Abstraction Layer, Draft', 30 January 2014, available at https://raw.githubusercontent.com/osgi/design/a71f2871f4ed0b97c4da79cf756a15876a61a347/rfcs/rfc0196/rfc-0196-DeviceAbstractionLayer.pdf.

Ontology description

The Osgi_dal ontology focuses on the concepts of ‘device’ and ‘function’ that are central in the OSGi architecture. A device (Device class) represents a physical device or a functional part of it in the OSGi service registry. A device is characterized by a mandatory unique identifier (hasDeviceUID property) and a set of properties, most of which are optional, namely the device type, such as DVD or TV (hasDeviceType property), model (hasModel property), serial number (hasSerialNumber property), driver (hasDriver property), firmware and hardware vendor and version (hasFirmwareVendor, hasFirmwareVersion, hasHardwareVendor and hasHardwareVendor properties). Moreover, a device has a status (hasStatus mandatory property) that can assume one of the values “Removed”, “Offline”, “Online”, “Processing”, “NotInitialized”, or “NotConfigured”. Optionally, the reason of the current device status can be defined using the hasStatusDetail property, which can assume fixed values, such as “Connecting”, “Initializing”, or “DeviceBroken”, among others. A device can support zero or more functions, which are described by the Function class.

A function is an atomic functional entity that characterizes a device. A function is registered in the OSGi service registry. There are 8 functions defined by OSGi, namely Alarm, BooleanControl, BooleanSensor, KeyPad, Meter, MultiLevelControl, MultiLevelSensor and WakeUp. The ontology also defines several function types (FunctionType class), such as Light, Occupancy and Temperature, which further specifies a certain function. For example, one can have a temperature sensor or an occupancy sensor represented by a BooleanSensor function with Temperature or Occupancy function type, respectively. Each function provides a set of operations and properties (hasPropertyName and hasOperationName properties). For example, the Meter function has an operation “resetTotal” and two properties, namely “current”, which contains the current consumption, and “total”, which contains the total consumption measured since the last call of the “resetTotal” operation or the device initial run. Finally, the ontology defines units of measure in the UnitOfMeasure class.

Observations

– The ontology defines the status of a device (Status class), but not the status transitions to go from one state to another, namely the dynamic behaviour of the device, which is represented by state diagrams in the OSGi specification.

- Operations and properties that correspond to a certain function are currently defined as `hasPropertyName` and `hasOperationName` properties. Eventually, one could extend the ontology by defining the `Operation` and `Property` classes under which those can be further detailed.
- The ontology includes some example units under the `UnitOfMeasure` class, but more units should to be added for completeness according to the OSGi DAL specification (see page 72 of the 'OSGi Alliance Device Abstraction Layer').
- Events are also part of the OSGi specification, but they are out of scope for our study. However, the ontology could be extended by defining a `FunctionEvent` class and its corresponding properties.

4.14 PowerOnt (previously SEIPF)

Ontology Title

PowerOnt: Power Profiling for Intelligent Domotic Environments (imports DogOnt: Ontology Modeling for Intelligent Domotic Environments)

Source

- Dario Bonino, Fulvio Corno, 'DogOnt - Ontology Modeling for Intelligent Domotic Environments', 7th International Semantic Web Conference. October 26-30, 2008. Ed. Springer-Verlag, Lecture Notes on Computer Science, pp. 790-803, available at http://www.cad.polito.it/db/iswc08.pdf
- DogOnt website, Politecnico di Torino, available at http://elite.polito.it/dogont
- PowerOnt ontology, available at http://elite.polito.it/ontologies/poweront/poweront.html

Ontology description

The PowerOnt ontology provides energy consumption information for different appliances in the house using the underlying DogOnt ontology, which models the domotic system of a house supporting intelligent operations. The DogOnt ontology consists of the following main classes: `BuildingThing`, which models available things, either controllable or not; `BuildingEnvironment`, which models the place where things are located; `State`, which models the stable configurations that controllable things can assume; `Functionality`, which models what controllable things can do; and `Command`, which models the way a given device property can be modified (e.g., light intensity) and the values it can assume.

The `BuildingEnvironment` class supports a coarse representation of domestic environments, as whole architectural units, including several types of `Room`, the `Garage` and the `Garden`. The `BuildingThing` concept represents all the elements that can be located or that can take part in the definition of a `BuildingEnvironment`. DogOnt defines a clear separation between objects that can be controlled by a domotic system (`Controllable` class) and all the other objects that can be found in a home (`UnControllable` class). Controllable objects can be appliances (`Appliance` class) or can belong to house plants such as the HVAC3 plant. Appliances and are further subdivided in `WhiteGoods` and `BrownGoods`, according to the EHS taxonomy. House plants include `HVACSystems`, `ElectricSystems` and `SecuritySystems`. Uncontrollable objects are all the home components that cannot be directly controlled by a domotic system. They are mainly subdivided in `Furniture` and `Architectural` elements. Furniture models all the elements usually

adopted as furniture like chairs, cupboards, desks. Architectural elements model all the elements that define a living environment such as `Walls` and `Floors`. All the objects that are usually referred to as "device", in the DogOnt ontology are objects belonging to the `Controllable` class. Each device class is associated to a set of different functionalities, by means of the `hasFunctionality` relationship. Each functionality defines the `Commands` to modify a given device property (e.g., light intensity) and the values they can assume. Functionalities are divided in different classes on the basis of their goals: `ControlFunctionality` models the ability to control a device or a part of it. `NotificationFunctionality` represents the ability of a device to autonomously advertise its internal state and in particular the ability of detecting and signalling state changes. `QueryFunctionality` encompasses the capabilities of a device to be queried, or polled, about its condition, e.g., failure and internal state values. `States` are classified according to the kind of values they can assume: continuously changing qualities are modelled as `ContinuousStates`, while qualities that can only assume discrete values (e.g., On/Off, Up/Down, etc.) are classified as `DiscreteStates`.

The PowerOnt ontology adds the `PowerConsumption` class, which encodes the power consumed by the appliances defined in DogOnt in a given state (`StateValue` class).

Observations

– We did not create the DogOnt and PowerOnt ontologies. We are reusing the OWL version that was provided to us by the authors of the 'DogOnt - Ontology Modeling for Intelligent Domotic Environments' article.

4.15 SEEMPubs

Ontology Title

Seempubs: Smart Energy Efficient Middleware for Public Space (SEEMPubS) ontolology

Source

'Deliverable D5.1, Data Format Definition', version 1.0, 30 September 2012, available at http://seempubs.polito.it/images/stories/documents/WP5/D.5.1.pdf

Ontology description

The Seempubs ontology describes the sensors and data that have been used in the use cases of the SEEMPubS project to control the building services, and monitor the indoor conditions and energy consumptions in some rooms of the Politecnico di Torino Campus and the Valentino Castle in Italy. The `Sensor` class represents the different type of sensors that have been used, namely `Controller`, `IndoorTemperatureHumiditySensor`, `IndoorTemperatureSensor`, `LightSensor`, `OccupancySensor`, `OutdoorTemperatureSensor`, `PowerMeter4Lightingsystem`, `PowerMeter4Appliance`, `SuppliedAirTemperatureSensor`, `Switch` and `Thermostat`. The data recorded in the use cases can be classified as related to indoor comfort conditions, related to energy consumption (electrical or thermal consumption), and related to the use of spaces and building services. Therefore, each sensor belongs to one of the categories: `ComfortCondition`, `EnergyConsumption` or `UseOfSpaceAndBuildingService`. Moreover, a sensor has a certain position in the room (`hasPosition` property) in which a number of sensors is positioned (`hasSensorNumber` property), is associated to a certain protocol, such as, for example,

"EnOcean" or "BACnet" (hasProtocol property), and requires a certain communication time to transmits its data (hasCommunicationTime property).

Each sensor measures some quantity (MeasuredQuantity class). For example the IndoorTemperatureSensor measures Temperature , the OccupancySensor measures Presence or Absence, and the Controller measures FanCoilStatus1, FanCoilStatus2, or FanCoilStatus3. The MeasuredQuantity class can have a unit (UnitOfMeasure class), a mode (OperationMode class) and a status (Status class). For example, Temperature has unit CelsiusDegree, Presence has mode Present, and FanCoilStatus1 has status FanVelocity1. Each sensor processes data in certain terms, such as, for example, average values or individual single data, which are enumerated in the DataProcessing class. Each sensor also represents data in a certain manner, for example, daily, monthly, according to an annual trend or a cumulative frequency, as enumerated in the DataRepresentation class.

The LightSensor class provides an example of how all the classes and properties mentioned above can be instantiated for a specific sensor.

Observations

- Only the LightSensor class is fully detailed with the values provided in the source of the ontology ('Deliverable D5.1, Data Format Definition'). The other type of sensors need to be detailed analogously (i.e., Controller, IndoorTemperatureHumiditySensor, IndoorTemperatureSensor, OccupancySensor, OutdoorTemperatureSensor, PowerMeter4Lightingsystem, PowerMeter4Appliance, SuppliedAirTemperatureSensor, Switch and Thermostat).

4.16 SEP2

Ontology Title

Sep2: ZigBee Smart Energy Profile 2.0 (SEP2) ontology

Source

' Zigbee Alliance/HomePlug Alliance Smart Energy Profile 2 Application Protocol Standard, ZigBee Public Document 13-0200-00', April 2013, available at http://www.zigbee.org/Standards/ZigBeeSmartEnergy/ZigBeeSmartEnergy20Standard.aspx

Ontology description

TheZigBee SEP-2 ontology is a taxonomy that represents the SEP-2 resources and function sets. Resources are classified in resources that provide operational information or services to manage and support the end devices of an SEP-2 network (SupportResource class), resources that provide general purpose and non-domain specific functionality (CommonResource class), and resources that are specific to the domain of Smart Energy (SmartEnergyResource class). Examples of support resources are represented by the EndDeviceResource and DeviceStatusResource classes, common resources by the DeviceInformationResource, PowerStatusResource and TimeResource classes, and Smart energy domain resources by the MeterReadingResource class. Each resource can be further detailed with its specific properties. The ontology further details the

DeviceStatusResource and the DeviceInformationResource class. For example, the DeviceStatusResource is characterized by the time at which the reported values were recorded (hasChangedTime property), the number of times that the device has been turned on (hasOnCount property), the device operational state (hasOpState property), and the total time device has operated (hasOpTime property).

A function set (FunctionSet class) is a logical grouping of resources that cooperate to implement SEP-2 features, such as, for example, metering (MeteringFunctionSet class). Therefore, a function set groups a number of resources (groups property), while a resource is grouped in a certain function set (isGroupedIn inverse property). For example, the EndDeviceResourceFunctionSet class groups the EndDeviceListResource, EndDeviceResource, RegistrationResource and DeviceStatusResource classes.

Under the TypesPackage class, the ontology represents some data types that are relevant to describe the considered resources, such as DeviceCategoryType (e.g., Water Heater, Sauna, Hot tub, Smart Appliance, Irrigation Pump, etc.), PowerSourceType (e.g., battery, local generation, emergency, etc.) and UnitType (e.g., kWh, kW, Cubic Meters, etc.).

Observations

- The Sep2 ontology presents examples of resources, function sets and package types that are in the scope of our study. The ontology also describes in detail the DeviceStatusResource and theDeviceInformationResource classes, which can be used as example to further detail other resources defined in the SEP 2 specification. In fact, the SEP 2 specification contains a large number of resources, function sets and package types that are not considered here, but can be eventually added to extend the current version of the ontology.

4.17 SmartCoDE

Ontology Title

Smartcode: Smart Control of Demand for Consumption and Supply to enable balanced, energy-positive buildings and neighbourhoods (SmartCoDE) ontology

Source

'Deliverable D1.1.2 -Model of local energy resource cluster', 31 December 2012, available at https://www.fp7-smartcode.eu/system/files/page/d-1.1.2.pdf

Ontology description

The Smartcode ontology presents a classification of Energy using Products (EuPs) into seven categories, namely variable services (VARSVC class), thermal services (THMSVC class), schedulable services (SCDSVC class), event-timeout services (ETOSVC class), charge control (CHACON class), complete control (COMCON class), and custom control (CUSCON class). These products have some parameters, such as Configuration, OnlineInput and SensorInput. Each product is characterized by an energy management strategy (hasEnergyManagementStrategy property) and its cost profile can be of interest of not for energy management purposes (isCostProfileInteresting property).

The VARSVC class includes appliances that provide a user-variable service that is balanced with sensor input. For example, Blind, DimmableLighting and LightingIlluminaceControlled are variable services included in this class.

The THMSVC class includes appliances that provide an inert, thermal service that can serve as a virtual storage. For example, Freezer, Heating and WaterBoiler are thermal services included in this class.

The SCDSVC class includes appliances that provide a service that can be scheduled within a certain time-frame. For example, BakingMachine, Dryer and WashingMachine are schedulable services included in this class.

The ETOSVC class includes appliances that are controlled by sensor events and time-outs. For example, LightingPresenceControlled is an event-timeout service included in this class.

The CHACON class includes appliances that charge a possibly removable device. For example, BatteryCharger, Emergency and HandHeldVacuum are charge controls included in this class.

The COMCON class includes appliances that charge a possibly removable device, like CHACON, but the usage of the charged power can also be controlled. For example, RobotVacuum is a charge control included in this class.

The CUSCON class includes appliances that do not fit into other classes or have too high user interaction to be controllable. For example, Hifi, Oven and PC are appliances included in this class.

Observations

- The enumerations under the Parameter class contain the values defined in the source of the ontology ('Deliverable D1.1.2 -Model of local energy resource cluster'). However, these enumerations can be extended with new values, if necessary.

4.18 UPnP

Ontology Title

Upnp: Universal Plug and Play (UPnP) ontology

Source

'UPnP Device Architecture 1.1,', 15 October 2008, http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

Ontology description

The UPnP ontology represents the devices and services defined by the UpnP device architecture specification. A device (Device class) represents a logical device and is a container that embeds one or more services (Service class)and may embed other logical devices (Device class). A device must have a description (DeviceDescription class), which contains all relevant information about the device filled in by the vendor, such as manufacturer name, model name, model number, serial number, and URLs for control, eventing, and presentation, among others. The device description also includes the services corresponding to that specific device (hasService property). A service exposes some actions (Action class), namely the commands supported by the service, and state variables that characterise the status of the service (StateVariable class). Actions and state variables are

specified in the service description (`ServiceDescription` class). An action has arguments (`Argument` class), which are parameters that can be input or output of a service, and may have a return value. A state variable can trigger events (`Event` class) as notification of one or more changes in the state variables exposed by a service.

The ontology instantiates examples for the `SolarBlindProtection`, `HVAC_System` and `HVAC_ZoneThermostat` devices with their corresponding services. Consider the `SolarProtectionBlind` instance of the `Device` class, which embeds the `TwoWayMotionMotor` instance of the `Service` class. Some manufacturer details of the `SolarProtectionBlind` device instance are mandatory and should be filled in, but they are left out from this version of the ontology for the sake of simplicity. The `TwoWayMotionMotor` service instance contains a number of corresponding actions, such as `Close`, `Lock`, `Open`, `Stop`, `SetPosition`, etc., and some state variables, such as `OperationMode`, `Position` and `ServiceLocked`.

Observations

- The UPnP ontology presents instance of devices, services, actions, arguments and state variables that should be used as an example to further extend the ontology according to the source, namely the 'UPnP Device Architecture 1.1' document.

4.19 W3C SSN

Ontology Title

Ssn: Semantic Sensor Network (SSN) Ontology

Source

Semantic Sensor Network Ontology, available at http://www.w3.org/2005/Incubator/ssn/ssnx/ssn.

Ontology description

The SSN ontology is an OWL ontology that provides a framework to describe sensors, observations and related concepts. The official description of the ontology from W3C is available at http://www.w3.org/2005/Incubator/ssn/ssnx/ssn. The SSN ontology does not describe domain concepts, such as time and locations, since these concepts are intended to be included from other ontologies via OWL imports. A sensor is a specific device whose purpose is to report measurements and observation real world phenomena. A sensor is different in nature from other types of devices such as actuators, because of its event based behaviour and the temporal relationships that need to be considered. The SSN ontology is a basis for reasoning about the measurements that can ease the development of advanced applications. For instance, when reasoning about sensors, constraints such as power restriction, limited memory, variable data quality need to be taken into account. It is possible to reason either about individual sensors as well as about the connection of a number of sensors.

The SSN ontology is composed by several modules. Some modules in the scope of our study are:

- Skeleton module, which represents
  - `FeatureOfInterest`, i.e., an abstraction of real world phenomena, such as thing, person, event;

    - Observation, i.e., a Situation in which a Sensing method has been used to estimate or calculate a value of a Property of a Feature Of Interest;
    - Property, i.e., an aspect of an entity that is intrinsic to and cannot exist without the entity and is observable by a sensor;
    - Sensing, i.e., a process that results in the estimation, or calculation, of the value of a phenomenon);
    - Sensor, i.e., any entity that can follow a sensing method and thus observe some Property of a Feature Of Interest. Sensors may be physical devices, computational methods, a laboratory setup with a person following a method, or any other thing that can follow a Sensing Method to observe a Property;
    - SensorInput, i.e., an Event in the real world that triggers the sensor;
    - SensorOutput, i.e., a sensor outputs a piece of information (an observed value), the value itself being represented by an Observation Value),
    - Stimulus (an Event in the real world that 'triggers' the sensor. The properties associated to the stimulus may be different to eventual observed property. It is the event, not the object that triggers the sensor)
- Measuring module, which represents SensingDevice, SensorDataSheet;
- Measuring Capability module, which represents Accuracy, DetectionLimit, Drift, Frequency, Latency, MeasurementCapability, MeasurementProperty, MeasurementRange, Precision, Resolution, ResponseTime, Selectivity, Sensitivity;
- Data module, which represents ObservationValue;
- Time module, which represents end Time and startTime;
- Constraint Block module, which represents Condition;
- Device module , which represents Device;
- Energy Restriction module, which represents BatteryLifetime, OperatingPowerRange.

Observations

- We did not create the SSN ontology. We are reusing the OWL version that is available on the W3C website.

4.20 Z-Wave

Ontology Title

Zwave: Z-Wave Application Layer ontology

Source

' Z-Wave Technical Basics - Chapter 4: Application Layer', 1 June 2011, available at http://www.domotiga.nl/attachments/download/1075/Z-Wave%20Technical%20Basics-small.pdf

Ontology description

The Z-Wave ontology covers the application layer of the 3-layer general model of wireless communication defined by Z-Wave. This application layer defines the messages to be exchanged, such as switching a light or increasing the temperature of a heating device. The Z-Wave ontology is a taxonomy of the supported type of devices (i.e., basic, generic or specific), the product categories to

which these devices belongs to, and the type of functions, or commands, supported by these devices. Each `Device` class belongs to a `ProductCategory` class, such as `ElectricalDimmer`, `ElectricalSwitch`, `ThermostatControl`, `MotorControl` and `Sensor`. Moreover, devices can be classified in basic devices (`BasicDevice` class), namely the basic category to which every device must belong, generic devices (`GenericDevice` class), which allows to specify the general function common to a certain type of devices, and specific devices (`SpecificDevice` class), which allows to further specialize the functions of a certain generic device. For example, each basic device must be a `Controller`, `Slave` or `RoutingSlave`. Examples of generic devices are a thermostat, meter, and alarm sensor, which are represented by the `ThermostatGeneric`, `MeterGeneric` and `AlarmSensorGeneric` classes, respectively. The generic device thermostat can be further specialized in the `ThermostatGeneralV2`, `SetbackScheduleThermostat`, and SetbackThermostatclasses, which are examples of specific devices.

The ontology further represent the commands supported by the Z-Wave devices under the `Command` class. This class enumerates the commands supported by the standard according to the Annex A of the source used to create our ontology (namely, the ' Z-Wave Technical Basics' document). In case the Z-Wave device is assigned to a `SpecificDevice` class, it must support a set of mandatory commands as functions of this specific device class, (`supportsMandatoryCommand` property). Besides the mandatory commands, Z-Wave devices can further support further optional commands (`supportsMandatoryCommand` property), which may be useful, but the standard does not enforce the implementation of these commands.

Observations

- The ontology is a rather simple taxonomy of devices and commands that was derived from the only publicly available document that we could find, which is not the official protocol specification (not available for free). Therefore, this ontology is intended as an initial representation of the main concepts defined by Z-wave and should be more accurately extended according to the original specification.

5. Mappings

The goal of the reference ontology that will result from task 3 is to explicitly specify recurring core concepts in the smart appliances domain, the relationships between these concepts, and mappings to other concepts used by different assets/standards/models. These mappings allow translation from the reference ontology to specific assets, reducing the effort of translating from one asset to another, since the reference ontology requires one set of mappings to each asset, instead of a dedicated set of mappings for each pair of assets. Figure 1 shows the role of the reference ontology in the mapping.

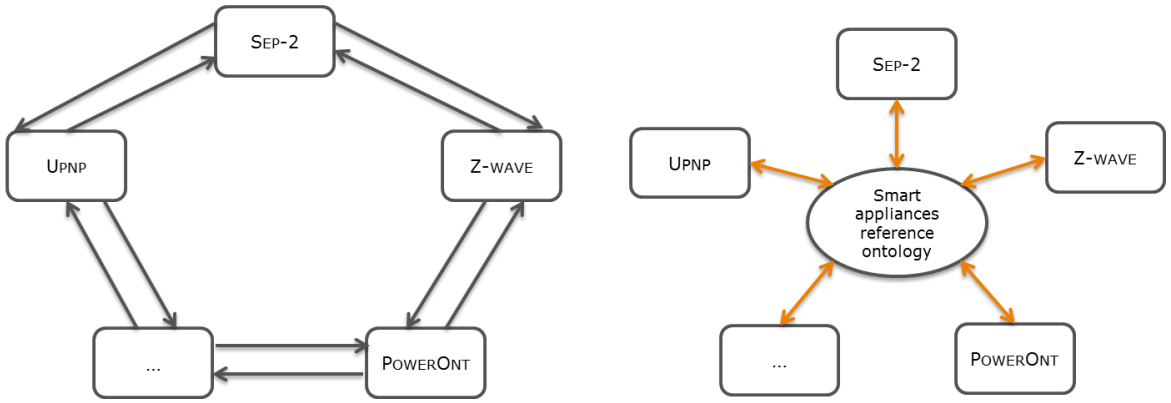

Figure 1- The role of the smart appliances reference ontology in the mapping among different assets

Although developing the reference ontology is part of task 3, we have started its development in parallel with task 2 in order to build the reference ontology incrementally while creating individual ontologies for the specific assets. In this way, not only we were able to include relevant concepts as soon as they turned out to be relevant for (several of) the specific assets, but we could also have a way to involve the expert group and stakeholders for validation in an early phase of the development of the reference ontology, instead of presenting our results only at the end of task 3. Therefore , when creating the ontologies for the specific assets in task 2, we identified relevant concepts in the scope of this study that could be part of the reference ontology. An initial proposal is to consider the following concepts:

- Device
- Device category
- Function
- Function category
- Service
- Command
- Parameter
- Mode/Status
- Energy profile
- Energy
- Power
- Time/Duration
- Building
- Sensor
- Actuator
- Meter
- Load
- Storage
- Generator
- Unit of Measure

These concepts have not been organized in any hierarchical relationship, nor relationships among them have been defined yet. Moreover, we did not yet work out their explicit definitions. These are all activities out of the scope for task 2 and therefore for this document. These activitieswill be

carried out in task 3. These concepts should be considered as a means to present the mappings shown in this document (see mapping table in the next page), and as an input for early discussion with the expert group and stakeholders about what could be included in the reference ontology. The criteria used to select these concepts were:

1) whether a concept was recurring among several of the assets for which we created ontologies, showing therefore a certain (shared) degree of relevance for the smart appliances domain;
2) whether the same concept was in the scope of our study, as laid out in section 2.1. To facilitate this task, we used the recurring concepts initially proposed in task 1.

Notice that some important concepts may be missing in this document, but they can be added later in the *D-S3 Third Interim study report*, which will cover the definition of the smart appliances reference ontology and a description of this ontology within the ETSI M2M architecture. The creation of the reference ontology is an ongoing work and a detailed analysis of its core concepts will be done in task 3.

The following table presents an initial mapping of these concepts onto the ontologies described in this document. The table shows only the presence or absence of a certain concept and it is intended to give an overview to the reader in a visual and intuitive manner. The Mappings.xlsx file attached to this document presents, the same mappings in more detail.

	DEVICE	SERVICE	DEVICE CATEGORY	FUNCTION CATEGORY	FUNCTION	COMMAND	PARAMETER	MODE/STATUS	ENERGY PROFILE	ENERGY	POWER	TIME/DURATION	BUILDING	SENSOR	ACTUATOR	METER	LOAD	STORAGE	GENERATOR	UNIT Of MEASURE
Dect_ule	x	x		x	x	x	x				x	x		x	x	x				
DogOnt/Power	x			x	x	x		x			x		x							x
Echonet	x		x		x			x					x	x						
eDiana	x	x				x								x	x		x	x	x	
Enocean	x		x	x	x									x		x				
Fan_Fpai	x		x	x	x				x	x	x	x					x	x	x	x
Fiemser	x							x					x	x	x		x	x	x	
Fipa	x																			
Hydra	x													x	x					
Knx	x			x		x		x							x					
Mirabel	x								x	x	x	x					x	x	x	x
Omalwm2m	x					x	x													
Oms	x									x	x	x				x				x
Osgi_dal	x			x	x	x	x	x						x		x				x
Seempubs	x		x					x		x	x		x	x						x
Sep2	x	x			x			x			x	x				x				x
Smartcode	x		x				x										x	x	x	
Upnp	x	x				x	x	x												
W3C_ssn	x											x		x						x
Zwave	x		x	x	x	x								x	x					

6. Conclusions

This deliverable presented the work that has been carried out in task 2 concerning the translation of the semantic assets in the short list to corresponding OWL ontologies, and the creation of an initial mapping among these ontologies. These ontologies should be considered as an intermediate result that allows us to achieve the final goal of the project, namely provide a reference ontology for the smart appliances domain, but they are not the ultimate result of this project themselves. The purpose of the mappings is to relate the 20 assets in the short list using their most recurring concepts, which will become the core concepts of the reference ontology in task 3. In order to perform the translation and mapping tasks, we have followed a systematic approach that allowed us to deal with the quantity of ontologies to be created and their complexity.

Out of the 20 assets considered in this deliverable:

- 4 assets were already expressed in OWL, namely eDIANA, Hydra, PowerOnt and SSN. We contacted the authors of these ontologies in order to obtain the original OWL files.
- 16 assets have been translated into OWL from scratch. For each of these assets we have created an ontology expressed in OWL-DL and serialized in Turtle, therefore, they have a file extension 'ttl'.

All the ontologies are published online at https://sites.google.com/site/smartappliancesproject. They can be opened with any ontology editor, such as TopBraid Composer[19], Protégé[20] and NeOn[21]. The website provides a page for each of the 20 semantic assets in the short list, with the URL to download the corresponding ontology and a human-readable explanation to describe the main classes and properties. The "owners" of the considered assets should validate whether the meaning they originally intended for their assets is actually reflected in our ontologies. To commit their feedback, they can post comments using the tab available when logged on to the website with a Google-account. The review of the ontologies by the "owners" has been actively solicited on the smart appliances workshop of stakeholders group on LinkedIn[22]. We will further solicit the review by contacting the "owners" individually by e-mail. Moreover, a 2nd stakeholders' workshop will take place at the ETSI premises in Sophia Antipolis on October 15, 2014. This workshop will provide an additional opportunity to discuss the ontologies described here with the stakeholders of the smart appliances domain. Any eventual change after the workshop and until the end of the project in March 2015 will be covered in the online version of the ontologies, and major changes will be addressed in the *D-S4 Final Study report*, which will be officially passed to ETSI Smart M2M.

This deliverable also listed some additional assets that were brought to our attention during the 1st stakeholders' workshop. The long list presented in the *DS-1 Interim Study Report* will be therefore extended in the *D-S4 Final Study report* with these additional assets. These assets have not been translated into OWL ontologies, but in task 3 we will assess their possible contribution to the study and eventually consider them for taking part in creating the reference ontology for the smart appliances domain as the final output of this project.

[19] http://www.topquadrant.com/downloads/
[20] http://protege.stanford.edu/
[21] http://www.neon-project.org/
[22] https://www.linkedin.com/groups/Workshop-Stakeholders-on-Smart-Appliances-7450648

This deliverable proposed an initial mapping of the ontologies by means of a number of concepts that we have identified as most relevant in the smart appliances domain and in the scope of this study. These concepts have not been organized in any hierarchical relationship, nor relationships among them have been defined yet. Moreover, we still did not work out their explicit definitions. These concepts should be considered as a means to present the mappings shown in this document (see mapping table in the next page), and as an input for early discussion with the expert group and stakeholders about what could be included in the reference ontology. These concepts provide the basis for creating the reference ontology in task 3. Therefore, task 3 will start from these concepts, eventually adding new or different concepts, if necessary. We will further provide explicit and precise definitions for these concepts, and add proper relationships and axioms to constrain their intended meaning. Task 3 will result in the *D-S3 Third Interim study report*, which will cover the definition of the smart appliances reference ontology and a description of this ontology within the ETSI M2M architecture.

As a limitation of the work that has been done in task 2, the project did not have the resources to elaborate every ontology in all possible detail. However, we think that this is not necessary as, to achieve the final goal of the project, we only need to find the commonalities between the various ontologies. Moreover, having learned from our approach, every stakeholder can now do the work himself, improving and/or extend the ontology to his liking given the open character of our results.

Finally, an important observation concerns the maintenance of the ontologies created in task 2 for the specific assets and the reference ontology that will be created in task 3. The specific ontologies are an initial step to capture the semantics of the assets in an explicit and formal way using OWL-DL. We acknowledge that our interpretation of the assets may not be always as intended by the "owners", therefore we will improve and update the (online version of the) ontologies according to the stakeholders' feedback until the end of the project in March 2015. However, the specific ontologies are a means to create the reference ontology, but they are not the final result of this project. Therefore, changes or extensions on these ontologies, together with new mappings to the reference ontology that may emerge in the future, should be realized by the interested stakeholders once the project is ended. Concerning the reference ontology, its development is incremental and we will gradually improve it taking into account the reviews of the project's expert group and the stakeholders until March 2015, when the *D-S4 Final Study report* will be officially passed to ETSI Smart M2M. Afterwards, the maintenance of the reference ontology will be possibly guaranteed by ETSI Smart M2M.

References

[1] "Manual for statistics on energy consumption in households", ISSN 2315 - 0 815, Eurostat (2013).

[2] Frank den Hartog, "Consumer Networking Standardization: trends and research opportunities", IEEE Webcast Tutorial, http://www.comsoc.org/webcasts/view/consumer-networking-standardization-trends-and-research-opportunities , IEEE (2011).

[3] Invitation to Tender for a Study on "Available Semantics Assets for the Interoperability of Smart Appliances. Mapping into a Common Ontology as a M2M Application Layer Semantics" – SMART 2013/0077, http://ec.europa.eu/digital-agenda/en/news/invitation-tender-study-available-semantics-assets-interoperability-smart-appliances-mapping, EC (2013).

[4] Daniele, L.M. and Ferreira Pires, L. (2013) An ontological approach to logistics. In: Enterprise Interoperability, Research and Applications in the Service-oriented Ecosystem, IWEI'13 Proceedings, 26 Mar 2013, pp. 199-213. ISTE Ltd, John Wiley & Sons, Inc.

[5] Guarino, N., Oberle, D., Staan, S., "What is an Ontology". In: Handbook on Ontologies, pp. 1-17. Springer Verlag (2009)

[6] Gruber, T. R., "Ontology". In: Encyclopedia of Database Systems. Springer Verlag (2009)

[7] Guizzardi, G., "Ontological Foundations for Structural Conceptual Models". PhD Thesis, University of Twente, The Netherlands (2005)

[8] Uschold, M., Gruninger, M., "Ontologies: Principles, Methods and Applications". In: the Knowledge Engineering Review, vol. 11(2), pp. 93-136. Cambridge University Press (1996)

[9] Bohring, H., Auer, S., "Mapping XML to OWL Ontologies". In: Proceedings of the 13th Leipziger Informatik-Tage (LIT 2005), Lecture Notes in Informatics (2005).

Acknowledgements

This study is commissioned by the European Commission. The Project Officers Rogelio Segovia (until his retirement in September 2014) and Svetoslav Mihaylov (from oktober 2014) from the EC's DGCNECT department has been indispensable in creating the support of and attention from the smart appliances ecosystem needed to perform this study successfully and to embed it in the industry. The authors would also like to thank the following members of the project's Expert Group for the interest shown in our work, their helpful guidance and review comments:, Jerome Euzenat, Frank van Harmelen, Patricia Martigne, Susan Schwarze and Marten van Sinderen. We would further like to acknowledge the ETSI Smart M2M TC and other representatives of ETSI for their enthusiastic support of this study. Finally we would like to thank the following representatives of projects and organizations for actively providing us with the material and information we needed: Josef Baumeister for the CENELEC-CEM Technical Specifications, Dario Bonino for the DogOnt and PowerOnt ontologies, Lorenzo Sommaruga for the DomoML-env ontology, Adrian Noguero Mucientes for the eDIANA ontologies, Gregorio López for the ENERsip ontology, Markus Eisenhauer for the Hydra ontologies, and Martín Serrano for the OpenIoT ontology.

European Commission

Study on semantic assets for smart appliances interoperability

Luxembourg, Publications Office of the European Union

2014 –55

Digital Agenda for Europe

Digital Agenda for Europe