



Deleted: xxx

ISO/IEC JTC 1/SC 25/WG 1 **N 1076**

Date: **03-09-15**

ISO/IEC JTC 1/SC 25  
INTERCONNECTION OF INFORMATION TECHNOLOGY EQUIPMENT  
Secretariat: Germany (DIN)

**DOC TYPE:** Draft  
**TITLE:** ISO/IEC CD 18012-2 Information technology —  
Interconnection of information technology  
equipment — Home electronic system —  
Guidelines for product interoperability — Part  
2: Taxonomy and Lexicon  
**SOURCE:**  
**PROJECT:** 25.01.07.01-02  
**STATUS:** CD 20  
**ACTION ID:** ACT  
**DUE DATE:** **YYYY-MM-DD**  
**REQUESTED:** Comment  
**ACTION**  
**MEDIUM:** Defined  
**No of Pages:** **20** (excluding cover) Size in kb: xxx (including  
cover)  
**DISTRIBUTION:** ISO/IEC JTC1 SC25 WG1

Secretary - ISO/IEC JTC 1 / SC 25 - Dr.-Ing. Walter P. von Pattay  
c/o Siemens AG, CT SR CI, D- 81730 München, Germany  
Tel.: +49/89/636-40 774, Tfx.: +49/89/636-43 891  
EM: Walter.Pattay@MchP.Siemens.de  
Ftp address: "ftp.iec.ch", login: "sc25mem", password: see SC 25 N 449  
Home page: "http://www.iec.ch/sc25"

**DRAFT**

**ISO/IEC 18012-2**

**INTERNATIONAL STANDARD**

---

---

**Information technology –  
Home Electronic System (HES) –  
Guidelines for product interoperability: Part 2:  
Taxonomy and Lexicon**

---

---



## CONTENTS

	Page
1 Scope .....	4
2 Normative References .....	4
3 Terms, definitions and abbreviations.....	5
3.1 Definitions .....	5
3.2 Abbreviations .....	6
4 Conformance clauses .....	6
4.1 Safety .....	7
4.2 Management.....	7
4.3 Additional clauses.....	7
5 Application Interoperability Model.....	7
6 HES Application Interoperability Taxonomy.....	8
6.1 Application Domains .....	9
6.1.1 Definition .....	9
6.1.2 Application Domain List .....	10
6.1.3 Additions and modifications .....	11
6.2 Functional Class.....	11
6.2.1 Definition .....	11
6.2.2 Functional Class structure .....	11
6.2.3 Functional Class List .....	12
6.2.4 Additions and modifications .....	12
6.3 Objects .....	12
6.3.1 Definition .....	12
6.3.2 Structure.....	12
6.3.3 Object Types.....	14
7 Interoperability Procedures .....	14
7.1 Application Models and Scenarios .....	14
7.2 States and state-transitions .....	15
7.3 Functional Classes and Objects.....	15
Annex A An Interoperability Application Specification (example).....	16
A.1 Lighting Application.....	16
A.1.1 Procedure .....	16
A.1.2 Generic Lighting Subsystem Application Model .....	16
A.1.3 System A .....	17
A.1.4 System B .....	17
A.1.5 Interoperable System – Functional Model .....	17
Annex B Notes on interoperability.....	18
B.1 Taxonomy definitions .....	18
B.2 Taxonomy of the existing systems .....	18

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc

## FOREWARD

ISO (the International Organisation for Standardisation) and IEC (the International Electrotechnical Commission) form the specialised system for world-wide standardisation. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organisation to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organisations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

This part of ISO/IEC 18012 was prepared by Joint Technical Committee ISO/IEC JTC 1, Information technology, Subcommittee SC 25, Interconnection of Information Technology Equipment.

ISO/IEC 18012-1 consists of the following parts, under the general title Information technology — Interconnection of information technology equipment — Home electronic system — Guidelines for product interoperability:

- Part 1: Introduction
- Part 2: Taxonomy and Lexicon
- Part 3: Application models

## INTRODUCTION

The widespread development of many national or regional standard or proprietary specifications for home and building automation necessitates a standard for ensuring interoperability among products from multiple manufacturers that may need to interwork. Where different devices need to interoperate it is desirable that they do so seamlessly to present a single, uniform network system and deliver a variety of applications over it. Some examples of such applications are lighting control, environmental control, audio/video equipment control, and home security. This standard will ensure that home system applications on the same or dissimilar networks co-exist within premises and where they are required to interoperate they will do so in a safe, reliable, predictable and consistent manner.

Interoperability in distributed systems is defined as the ability of two or more distributed entities to communicate and cooperate in predictable ways despite differences in the implementation language, execution environment, or the distributed function model abstraction. Three main levels of interoperability between components in a distributed application can be distinguished:

- Syntactic level, where the names and the signatures of the entities, their components and operations are defined;
- Protocol level, where the ordering between the exchanged messages and the blocking conditions are defined; and
- Semantic level, where the meaning of the possible interactions between the entities in the distributed system is defined, e.g. the application behaviour due to a change of the state of the variables in the interacting entities and in the system at large.

At semantic level interoperability clashes between two application entities attempting to cooperate are caused by differences in the conceptual schemas (lexicons) of the specification of the entities. The possible clashes can be classified into two main groups:

1. Lossless clashes, which can be resolved with no loss of information. Some examples in this category include entity *naming clashes*, when the same entity/information is represented by different labels; *structural clashes* when information elements are grouped in different ways in different systems, and *unit clashes*, when some scalar value (e.g. distance or temperature) is represented with different units of measure.
2. Lossy clashes, which include interoperability clashes for which any transformation available, in either direction, causes loss in the information being communicated between the two application entities. These clashes are associated with differing levels of granularity, refinement or precision of the representation of the information. Note that a lossy translation between the two application entities may be an acceptable solution, provided that it does not affect the functional safety of the application and the system as a whole.

This part of the standard defines the taxonomy and the lexicon of the distributed home electronic systems. The taxonomy consists of the classification of the applications in a framework which may be used, together with the lexicon, to instantiate an interoperable system composed of devices on the same or dissimilar networks. The lexicon is the set of common data type primitives and the associated common action primitives that, within the taxonomy framework, can be used to describe a set of application models. The taxonomy framework and the lexicon operate at the application and presentation layers of the OSI reference model, and define a common generic home system application framework to be used for syntactic translation while maintaining the semantics of the application behaviour independently of the implementation and mode of operation.

This part of the standard is based on the understanding of the difference in the primitiveness of the actions at different levels of object definition. Furthermore, it is based on the separation between the concept of the action primitives and the actual implementation of the invocation of these primitives. For example, if Specification-A uses <get>/<put> functions to implement

local or remote reading and writing of variable values, these are not considered as “action primitives” in the context of this standard, but rather a programming mechanism used to invoke the application actions; different actions are caused by <put>-ing (setting) the same variable to different values, i.e. it is the variable and the values that it contain at a given time that define the application action. These *same actions* can be invoked in Specification B by performing a (different) remote or local function call. In this case it is possible for both systems to implement the same lexicon, but with their own invocation mechanism.

This part applies to components in operation mode within networks, between networks located within dissimilar networks, and to devices located at the junction of dissimilar networks.

Two (or more) dissimilar networks within the same premises (referring to Figure 1) must conform to this standard if, when linked by some communication system, they are expected to behave as if both networks were logically the same network from an application perspective.

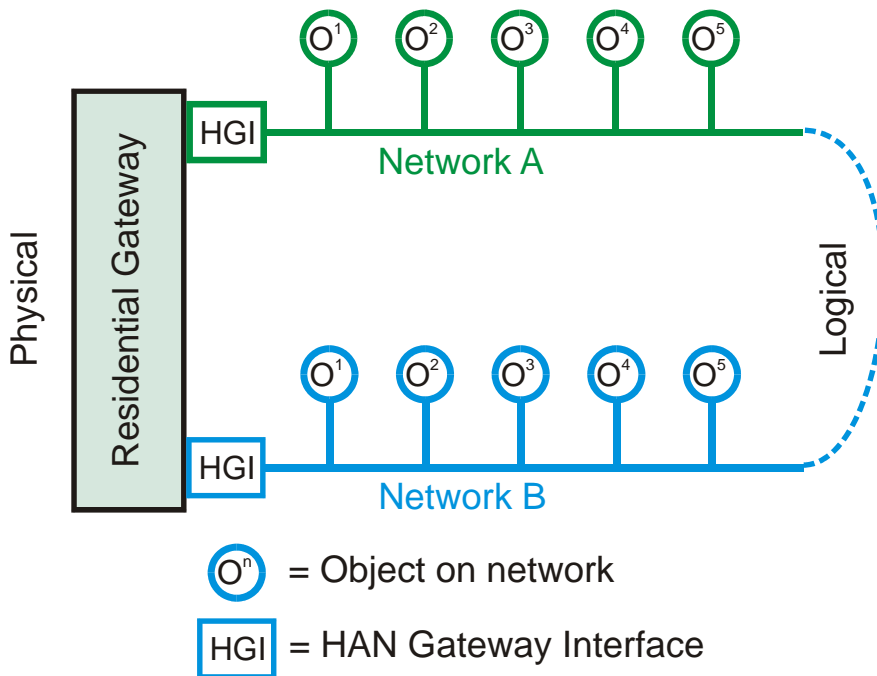


Figure 1 - Two Interoperating Networks

This document comprises the following sections:

- A section which describes the application interoperability model (clause 5)
- A section describing the taxonomy used for the inter-system and intra-system interoperability (clause 6)
- A section outlining recommended procedures for interoperable HES product specification (clauses 7)
- Annex A describes an example of an interoperable application specification... (TBA)

**Comment [EE1]:** Ron Ambrosio:  
*probably reverse the placement of the example and then the explanation, instead of the way it is now*

**Comment [EE2]:** DK (D1)  
*This explanation is not very clear. It requires (a) removing the second part of the paragraph (from <For example ...>), (b) moving into the taxonomy or lexicon introduction, or (c) a proposal for a better wording. To be discussed at the London meeting.*

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

# Information technology — Interconnection of information technology equipment — Home Electronic System — Guidelines for product interoperability — Part 2: Introduction

## 1 Scope

This International Standard specifies a taxonomy framework for product interoperability in the area of home and building automation systems by classifying applications into Application Domains, Application Functions, Application Functional Actions, and a lexicon in the form of a list of Objects, Properties, Property Data Types and Property Action Primitives. It is concerned with layers six (Presentation) and seven (Application) of the OSI Reference Model with sufficient detail needed to design interoperable Home Electronic System products. Layers one through five are not specified here; requirements regarding these layers are specified in Part 1 of this standard to the point needed to check whether the devices will be able to interoperate with one another.

Comment [EE3]: DK (D1)

This places DCTP in Layer 5 (transport); if existing systems are providing interworking functions above this level then this raises questions about the need for DCTP. Otherwise we probably will need to limit the scope to application layer (Layer 7) alone if we expect SC25WG1 to accept that DCTP will cover layer 6 (presentation). Or this may make it easier to define the translation function as a proxy mechanism, which is inline with what all the existing systems are doing (LonWorks, Echonet, HPnP/VHN, and possibly Konnex). To be discussed at the London meeting

This International Standard is applicable to:

- Standalone Local/Home networks, connected devices and applications.
- Mixed Local/Home networks, connected devices and applications.
- Automatically configured devices.
- Installer configured devices.
- Installer configured groups/clusters of devices.

This standard specifies an interoperability framework for system operation and management applied to devices connected to a single home control system or to different home control systems. Although a single uniform home electronic system would simplify operations, this standard recognises that multiple different networks may co-exist in the same house. This standard specifies the components to describe device profiles to assure that devices from multiple manufacturers work together to provide a specific application. Also, a specific device could be used for multiple applications.

Comment [EE4]: Ron: check if there is a definition of HAN and HES in N748; put one in if not: Home Area Network, and HES: Home Electronic Systems as the collection of interoperable HANs

This standard specifies a taxonomy framework based on application domain classification criteria for applications in the home electronic systems, as well as a lexicon of objects and primitive actions to effect or otherwise propagate change in the objects to provide for the specification and implementation of distributed application functions and services within the context of the home electronic system.

This document does not specify how two home control systems share a common resource or how to ensure that two home control systems used within the same premises do not interfere with each other. However, this document requires that two home control systems may share a common resource, and that they shall not interfere with one another.

Comment [EE5]: Ron: either move it or duplicate it, because it is a requirement (put it as a conformance clause)

## 2 Normative References

No:	Reference	Descriptive Title
[1]	ISO 7498 : 1984	Information processing systems - Open Systems Interconnection - Basic Reference Model
[2]	ISO/IEC TR-14543	Home Electronic System Architecture
[3]	ISO/IEC TR-14762	Guidelines for Functional Safety for Home Control Systems

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc

[4]	ISO/IEC FDIS 18012-1	ISO/IEC FDIS 18012-1 Information technology — Interconnection of information technology equipment — Home electronic system — Guidelines for product interoperability — Part 1: Introduction
...		

### 3 Terms, definitions and abbreviations

#### 3.1 Definitions

For the purposes of this International Standard, the following definitions are applicable.

##### 3.1.1 API

Application Programming Interface: The collection of invocation methods and associated parameters used by one piece of software to request actions from another piece of software

##### 3.1.2 Co-existence

Two or more networks within a premise that do not interfere with one another

##### 3.1.3 Component

This refers to a logical subunit of a larger, encompassing concept. For example, the concept of Interoperability is broken down into constituent components such as Safety, Management, and Operation. These constituent components are further broken down within their respective sections.

The term component is also used to refer to logical subunits of system architecture concepts, such as the components of a networking implementation (e.g., addressing)

##### 3.1.4 Device

A distinct physical unit on a network that performs a (set of) specific function(s) in a particular context. It can either be an end node on the network, or an intermediate node (as in the case of a network gateway device connecting two distinct physical networks).

##### 3.1.5 Functional Class

A Functional Class is a collection of objects and actions on objects that models a particular application function within an application domain

##### 3.1.6 Functional Action

Functional Actions allow for modelling of functionally composite behaviour within a Functional Class (i.e. in a specific context)

##### 3.1.7 Interoperability

Logical entities function together for applications on a network [4].

It is the ability of two or more devices to work together in one or more distributed applications, regardless of their manufacturer. The application data, their semantic and application related functionality are defined in such a way that if one device is replaced with a similar or different one, from same or different manufacturer, the distributed application in which the device participates will continue to operate as before.

##### 3.1.8 Network

A distinct interconnection of devices that share a single physical layer implementation in terms of the OSI layered network model [4].

**Comment [EE6]:** Ron to re-check and update the definitions of **interoperability, multiple implementations and intermediate implementations**. The updated ones to be ported in this document (or dropped if not referenced)

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc

### 3.1.9 Object

- a) A unit of software functionality. Used as traditionally defined in object-oriented programming. It has properties and methods for accessing these properties and/or interacting with other objects.
- b) A collection of related data (attributes) and methods (procedures) for operating on that data, with a well defined boundary (interface) and identity that encapsulates state and behaviour [OMG]. The state is represented by attributes and relationships, while behaviour is represented by operations, methods and state machines.

**Comment [EE7]:** DK3: this definition is taken from [4] (part 1 of this specification).

Either adopt one of the other definitions, or modify the single on the original FDIS to ensure consistence.

**Choose either (a) or (b).**

### 3.1.10 Product

A device or network that may be purchased to make up a Home Electronic System

### 3.1.11 Single implementation

A single, homogeneous network implementation, where interoperability is only of concern within the single network

### 3.1.12 Multiple implementation

A mixed collection of two or more network implementations. To establish interoperability, each network shall have a routing path to every other network in the system. This path may involve one or more hops through multiple intermediate networks

### 3.1.13 Intermediate implementation

A mixed collection of two or more network implementations. To establish connectivity, it provides for a common intermediate translation between any two networks, assuring a worst-case translation path of two hops (from any network to the common translation, and then from the common translation to the destination network).

### 3.1.14 Interoperability Gateway

A logical device that implements the interoperable object interfaces to and from different systems. It may be the same as the residential gateway/home gateway devices.

### 3.1.15 Interworking

The ability of two or more devices to support exchange of data and actions between them, having the same communication interface and application data types.

### 3.1.16 Interworking Function

A software module that provides syntactical translation services between a device and its standardised interoperable representation residing in an intermediary unit that connects two, possibly different, communication systems.

## 3.2 Abbreviations

API	Application Programming Interface
HAN	Home Area Network
HES	Home Electronic System
IWF	Interworking Function
RGIP	Residential Gateway Interior Protocol
HGI	Home Gateway Interface

## 4 Conformance clauses

In order to conform to this standard of Product Interoperability, HES products and networks shall meet the conformance clauses listed in [4] and the following clauses.

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

#### 4.1 Safety

It is required that particular considerations regarding safety in Interoperating Home Electronic Systems identified in [4], shall be followed. Safety is the fundamental consideration for the interoperability framework and its implementation into HES products. No action will be translated between dissimilar network systems if there is any doubt on the safe outcome of the action being carried out.

**Comment [EE8]:** Check if Part 1 clause (referenced) is adequate or not.

**Deleted:** [4]

#### 4.2 Management

Interoperability considerations regarding management processes in HES are described in [4]. This part of the standard addresses only the management aspects related to the operation mode of Interoperable HES and does not cover the management processes of individual constituent networks. It is assumed that the propagation of the necessary actions to support the operational application object binding further down the protocol stack, as well as the provisioning of the necessary conditions (e.g. establishing and maintaining connectivity between devices hosting the distributed application objects independently of the fact that the devices belong to the same system or different) are fulfilled by separate management services not described in this part of the standard. This allows for transparency to the native application management support services provided by separate, though interoperable, systems independently of the fact that these services are automatic or installer-based.

**Deleted:** (or this part of the standard – TBD).

**Comment [EE9]:** DK (D1)

*We are trying to cover only the application management, i.e. normal application object binding. It is assumed that the propagation of the necessary actions to support this binding down the protocol stack and/or the necessary conditions (connectivity between devices hosting the application objects independently if the devices belong to the same system or different) are fulfilled by separate management services not described in this standard. This allows for transparency to the native application management support services provided by separate systems independently if these are automatic or installer based. To be discussed at the London meeting.*

*Editor's note: Some examples to be given here of definition and enumeration of application management functions to guide/clarify what is not being covered.*

#### 4.3 Additional clauses

None in this version.

### 5 Application Interoperability Model

The model is based on the definition of an abstract Application Domain realm populated with abstract generic application objects that interact with each-other by invoking actions on the object properties using an Interoperable Residential Gateway Interior Protocol (RGIP).

**Comment [EE10]:** Think about some definition/enumeration of application mgmt functions → give examples, as we say we're not covering these here.

**Comment [EE11]:** DK (D1)

This clause can be left here, or moved after the Scope clause and before the Normative References clause. It is informative (covers the approach to interoperability), but cannot be left in the introduction only as it has architectural implications.

**Comment [EE12]:** DK + Ron

Leave it here!

**Comment [EE13]:** The RGIP is transparent to applications, i.e. it is internal to the gateway.

**Error! Reference source not found.** shows an instantiation of the abstract application interoperability model. Two objects (Object 1-A and Object 2-B) are installed into two different network systems, respectively Network A and Network B, and require interacting with each-other to provide an application within the home. The networks are logically linked together through a logical entity called the Interoperability Gateway (IG). The Interoperability Gateway interfaces with different network systems via Interoperability Gateway Interfaces (IGI). The interfaces contain logical functions that ensure the translation between the generic interoperable object instance (e.g. Object 1 within the IG realm) and their system-specific counterpart (Object 1-A in Network A) by using an Interworking Function (IWF). The interfaces (IGI) maintain the logical connection between the generic interoperable object instance within the IG realm and their system-specific counterpart. Note that both networks A and B may contain other devices and that, if these are not required to interwork with devices on differing networks, these will not be represented in the gateway.

**Deleted:** Error! Reference source not found.

**Deleted:** Error! Reference source not found.

**Deleted:** Figure 2

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

The Interoperability Gateway is an abstract functional entity. It can be implemented as a single separate entity, as a distributed entity, or its functionality may be integrated with other functional entities such as a Residential Gateway.

The specific Network Systems shall specify the IWFs to the Interoperability Gateway for the objects of interest.

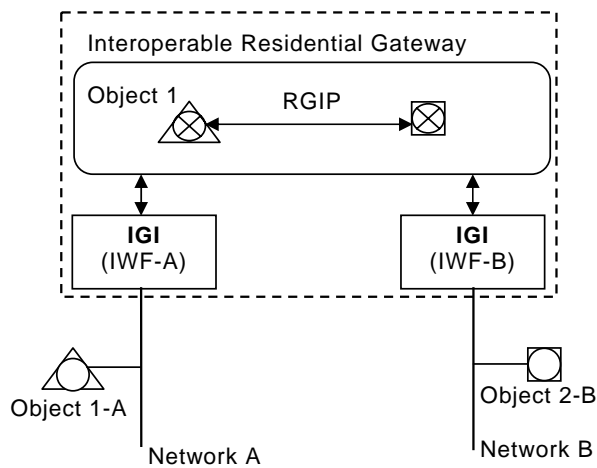


Figure 2 – Application Interoperability Model

**Comment [EE14]:** In the figure try to use the existing acronyms and terminology, unless a newer one is defined (**define it**). Replace IGI with HGI. Clarify (define) what IWF. Put it in the abbreviations table (done).

## 6 HES Application Interoperability Taxonomy

HES Application Interoperability Taxonomy is shown in [Error! Reference source not found.](#) For purposes of interoperability specification the HES domain is classified into the following categories:

1. Application Domain
2. Functional Class
3. Object
4. Functional Actions
5. Property
6. Property Primitive Actions

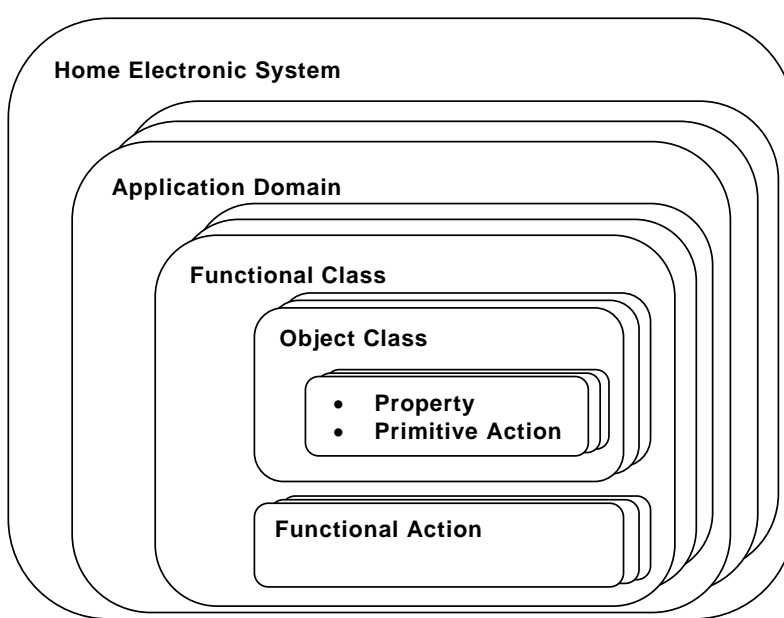
Deleted: Error! Reference source not found.

Deleted: Error! Reference source not found.

Deleted: Figure 3

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc



**Comment [EE15]:** This provides the taxonomy (structure). Show by example how this will be instantiated within the RGIP.

**Formatted:** Indent: Left: 0.05", Bulleted + Level: 1 + Aligned at: 0.25" + Tab after: 0.5" + Indent at: 0.5", Tabs: Not at 0.5"

**Deleted: 2**  
**Deleted: 3**

**Figure 3- Interoperable System Taxonomy**

Application Domains are specified in terms of Functional Classes, which represent logical grouping of some device functionality. Each Functional Class is specified as a collection of objects, which represent application atomic state and functions. This specifies the semantics of the applications and the components of the Functional Class. Objects in turn are specified as collection of properties, which define the content representation for both storage and transport (syntax) purposes.

The separation of properties and objects, as logical entities in the taxonomy network, is designed to allow for separation of the syntactic and semantic aspects of the interoperability. Properties are used to specify the syntax, i.e. the data representation of the object components. This allows for a direct mapping/translation between properties as specified in this standard and those specified in other specifications through the use of interworking functions (for example, data type translation and matching). This implicitly takes care of the data-sharing interaction model. It is expected that DCTP [include reference to SC25 WG1 DCTP project work] shall operate at this level. Objects cover the semantics of the application; they represent behaviour, which is exhibited through the interaction with other objects by invoking primitive actions on the properties. The separation between the semantics (object level) and the syntactic representation of the data (property) allows for flexibility in the definition of the interoperability interworking functions.

For example, a TemperatureSensor functional class may have an object called RoomTemperature of Temperature data type (semantic), which is measured in degree Celsius. The object in the interoperability specification is defined as having a property called "TempValue", which is of type "Float" which follows the ANSI/EEE 754 specification.

**Deleted: ¶**

Detailed definitions of the framework components are given below.

## 6.1 Application Domains

### 6.1.1 Definition

Application Domains serve as context-setting descriptions. They group together functions that interact in semantically related applications. A non-exhaustive list of the application domains is given in clause Error! Reference source not found.

ISO/IEC JTC 1/SC interop\_p2-D2v02\_N10761.doc

**Deleted: Error! Reference source not found.**

**Deleted: Error! Reference source not found.**

**Deleted: 6.1.2**

**Deleted: interop\_p2-D2v02\_N1076.doc**

**Deleted: interop\_p2(D2v02)\_N1076.doc**

Application Domains are enumerated entities. As part of the taxonomy they can be used, in their word form or represented by their enumerated value, to form part of the object name for application binding.

*Editor's note: The enumeration values are subject of further discussion. Note that, as the interoperability taxonomy is to be used as an intermediary translation reference, the actual values are of little relevance.*

**Comment [EE16]:** DK (D1)

The last sentence in the paragraph is to be discussed; I would not be against it being dropped altogether. If this taxonomy is to be used as XML templates (or DTD – Document Type Definition) for various systems to provide their translation into and from, ideally with simple value replacement. Some translations like this do exist between HPnP and LonWorks to the interoperability model.

To be discussed at the London meeting.

**A:** Leave it here.

### 6.1.2 Application Domain List

Application Domain Name	Description
General	This domain groups together functions that provide generic services to other entities in a home electronic system. E.g. Time/Date, Location, User Interface, etc.
Audio/Video	This domain groups together functions related to the control of the distribution and consumption of audio and/or visual content in a home electronic system. Examples include audio amplifier, tuner, video-display, speaker, etc.
Lighting	This domain groups together functions related to the lighting environment control in a home electronic system. Examples include Light, Light-sensor, Light-switch. etc.
Communications	This domain groups together functions related to the control of the communication session set up and distribution/transfer around a home electronic system. Examples include an Intercom, DataPoint, etc.
Heating	This domain groups together functions related to the control of heating sub-system, or parts thereof, of a home electronic system. Examples of functions include RoomController or ZoneController, TemperatureSensor, etc.
Ventilation	This domain groups together functions related to the control of environmental sub-systems (air-conditioning and ventilation). Examples include TemperatureSensor, HumiditySensor, VentilationZoneController, etc.
Utility	This domain groups together functions related to the management and/or monitoring of utilities such as electricity, water, gas, heating. Examples include UtilityMeter, LoadController, etc.
Security	This domain groups together functions related to the management and/or monitoring of the security sub-system. Examples include SecuritySensor, WindowSensor, SecurityZoneController, etc.
Appliance	This domain groups together functions related to the management and/or monitoring of domestic appliances. Examples include Washer, TumbleDryer, <u>Cooker, Oven, ...</u>
...	

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

### 6.1.3 Additions and modifications

The Application Domain List is non exhaustive. Proposals for adding new domains should be submitted to the standardisation body maintaining the list. Additions are made official upon publication of the new standard.

*Editor's note: The role of the registry, its placement, and its management procedure, including registration of application models and object profiles, updating of the lexicon, considerations if this process will be dynamic or manual, and clarification of the certification procedure are subjects of further discussion. (These processes cover the registration and update of the application model list and the objects, which form the lexicon)*

*Editor's note: It is necessary to select a language fore the model description language that can be used: of particular interest (and of similar use in other international standardisation activities) are UML and XML. The editor's recommend XML (XML Schemas), mainly due to the general acceptance that it provides more than sufficient support for the interoperability of web services framework. Although XML Schemas is a specific technology, the editors think that it currently provides the most straightforward representation for application models that matches matches the approach followed in the interoperability project. Currently it is not certain if there is enough expressiveness to guarantee a single translation from a particular UML model to a single XML schema; note that the uniqueness of translation is necessary for interoperability. UML models are to be assessed further as a possibility, time permitting, or if there is a strong comment in favour of it.*

**Comment [EE17]:** DK (D1)

The Application Domain List will be part of the overall HES Lexicon, maintained as XML DTD or Schemas. Acceptance of proposals will be reflected in an updated HES Lexicon DTD.

To be discussed at the London Meeting.

**Comment [EE18]:** DK + Ron (D2)

The registry is not expected to be DTDs. Two choices for languages to express the taxonomy and the application models are UML and XML Schemas.

The editors favour XML Schemas; it is accepted that these provide more than sufficient support for interoperability of webservices. Although XML Schemas is a specific technology, the editors think that it currently provides more the most straightforward representation for application models.

It is doubtful if there is enough expressiveness to guarantee a single translation from a particular UML model to a single XML schema (uniqueness of translation is necessary for interoperability). UML models are to be assessed further as a possibility, time permitting or if there is a strong comment in favour of it.

## 6.2 Functional Class

Classes are the basis of the classification. Classification requires that real entities are presented as objects, and objects with the same data structure (attributes) and behaviour (operations) are grouped into a class. A class is an abstraction that describes properties and behaviour relevant to a particular application, and ignores all the rest.

### 6.2.1 Definition

A Functional Class is a collection of objects and actions on objects that models a particular application function within the Application Domain. An instantiation of a Functional Class is a logical device, i.e. a software entity that provides some specific standardised control and/or information functionality in the HES in which it is installed.

### 6.2.2 Functional Class structure

A Functional Class is composed of Objects and Functional Actions. Functional Actions are specified as (possibly combined) actions on individual objects that lead to a functional operation of the logical or physical device in the system. A Functional Action has contextual meaning, and is specified together with the Function Class. The invocation of a Functional Action leads to a specified series of invocation of component objects.

A Functional Class describes in detail the application layer interface, including input and output objects, configuration properties, default and power-up behaviour required on devices for specific commonly used control functions. An instance of a Functional Class is implemented in a physical device as (part of) the application program.

An instance of a Functional Class is a logical device locally addressed via a Functional Class ID. The Functional Class ID (FCID) value can be a result of local enumeration, or a global FCID can be used. In any case the mapping between the FCID and the service access point serving this instance of the Functional Class is maintained by the application support layer.

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

### 6.2.2.1 Objects

Objects are actionable entities that contain one or more properties. They are locally addressable entities. An object models, through its properties and primitive actions, the most generic behaviour of sensors and actuators in a home electronic system. An object contains basic descriptive properties, configuration properties, and operational properties.

*Editor's note: Only the operational properties of objects are subject to this document at this stage.*

Objects are described in detail in Clause Error! Reference source not found.

Deleted: Error! Reference source not found.

Deleted: Error! Reference source not found.

Deleted: 6.3

### 6.2.2.2 Functional Actions

Functional Actions allow for modelling of functionally complex behaviour within a Functional Class (i.e. in a specific context). Functional Actions are enumerated entities within a Functional Class, and have meaning only within the class. Examples of a Functional Action is <SwitchOn> within a LightController or a LightSwitch Functional Class.

Invoking a Functional Action means to invoke at least one primitive action on at least one of the properties in at least one object within the same Functional Profile, local or remote.

### 6.2.3 Functional Class List

The Functional Class List consists of all the defined Functional Classes for each Application Domain. The Functional Class List is maintained by a internationally recognised body.

Comment [EE19]: DK + Ron Same comments as before regarding the registry location, maintenance procedure etc. For further discussion.

Deleted: <TBD>

*Editor's note: The role of the registry, its placement, and its management procedure, including registration of application models and object profiles, updating of the lexicon, considerations if this process will be dynamic or manual, and clarification of the certification procedure are subjects of further discussion. (This covers the maintenance of the lexicon, i.e. the registration and update of the application model list and the objects)*

### 6.2.4 Additions and modifications

The Functional Class List is non exhaustive. Proposals for adding new Functional Classes should be submitted to the standardisation body maintaining the list. Additions are made official upon publication of the new standard.

## 6.3 Objects

### 6.3.1 Definition

Objects are self-describing actionable entities that contain one or more properties. They are locally addressable entities. An object models via its properties the most generic behaviour of sensors and actuators in a home electronic system.

### 6.3.2 Structure

Objects are collection of one or more Properties. Each Property is an internally structured self-describing entity that encapsulates some information source or control variable.

Comment [EE20]: DK (D1)

Object description has implications in two directions: addressing (or the maintaining of a global object list with numerical identifiers) and, as a result, the mechanisms that can be used for service discovery (or capability exchange, or binding, or any similar term).

To be discussed at the London meeting.

Deleted: its

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc

Every object has a self-description property and an object instance identifier property. The definitive object structure is TBD.

The object instance identifier can be instantiated dynamically, or seeded with a (default) value specified in the generic application model.

ISO/IEC JTC 1/SC interop\_p2-D2v02\_N10761.doc

### 6.3.2.1 Properties

Properties are atomic actionable structured data entities that contain a single variable. Their structure is given below. Normally they are simple data format, and are characterised by a data type, methods (which may be implicitly specified), and direction (in/out, producer/consumer).

Property	
Description	Property_Name
	Property_Action_List
Coding	Property Data Type
	Number of elements
Value	Value
	<u>Default Value</u>
	<u>Value Range</u>
Application Semantics	Property Unit
	Support (Mandatory / Optional / Conditional)
	<u>Access (In / Out / IO ),</u>

### 6.3.2.2 Property Data Types Primitives

- Character String (ISO 646)
- Boolean
- Integer (Integer8, Integer16, Integer32)
- Unsigned Integer (UInt8, UInt16, UInt32)
- Float (ANSI/IEEE754)
- Date
- Time (and Date/Time)
- Data Octet String
- ....

Editor's note: Liaise with RIGP project on the syntax of Residential Gateway Interior Protocol project to verify and complete the list.

### 6.3.2.3 Property Unit List

These are semantic data units, for example degree Celsius for Temperature semantic data type.

*Editor's note: Need to put them in the definitions part of the document, and also to tie with the semantic vs. syntactic (representational) data types. To be taken from existing*

ISO/IEC JTC 1/SC interop\_p2-D2v02\_N10761.doc

**Comment [EE21]:** Ron + DK:

Need to differentiate earlier between the semantic data types and the syntactic data types. For example: Temperature, AngularSpeed, BinarySwitch are semantic data types (possible examples – try to find better ones); Integer, Boolean, Char, Float are syntactic data types. A mapping will be defined between the semantic and syntactic datatypes; the mapping may not necessarily be one-to-one.

Introduce this earlier in the document (**DONE**), probably with a one-liner or a paragraph at the Application Domain section to introduce the distinction between them.

**Comment [EE22]:** Do we need to list the mapping at the property granularity or at the system granularity (probably s.th. in between) both types at the table?

For further discussion.

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

*specifications (notably CEBus/CAL, EIB/KNX, and LonWorks, where these are based on ISO standards).*

#### 6.3.2.4 Property Action Primitives

The actions allowed on the properties are:

- READ / GET
- WRITE / SET
- ...

#### 6.3.3 Object Types

Objects can be classified into two groups: operational application objects and application management objects (configuration).

The Object List shall specify the type of object.

### 7 Interoperability Procedures

The interoperability specification consists of the description of:

- Application Model and Scenarios
- Functions required
- States and state transitions within the Functional Classes to ensure safe feature interaction in the home electronic system.
- Functional Classes (logical devices) and their composite objects
- Conformance requirements (mandatory, optional and conditional components).

**Comment [EE23]:** DK (D1)

Do we want to describe the interoperability procedure? And within the product interoperability do we define device interoperability, i.e. the specification of Functional Classes (equivalent to Functional Profiles in LonWorks or the Context Classes in HPnP) or application interoperability (like in EHS 1.3a or, partially, HPNP (CEBus))? Like the rest of the work, either viewpoint can be taken – any suggestions?

To be discussed at the London 2002 Meeting.

*Editor's note: We want to describe the interoperability procedure. Within the product interoperability we define application interoperability, i.e. the specification of Functional Classes, which represent the mapping of the devices into this application interoperability model. It is the responsibility of the manufacturer to make sure that the devices, which are expected to change much more often than the application in which they participate, map into this interoperable application model. The manufacturer either has to provide this mapping, or conform to an already specified application model.*

#### 7.1 Application Models and Scenarios

The scenarios establish the framework for understanding the requirements and defining functional profiles required to realise the scenarios. The analysis of the scenarios shall lead to application models with abstract components (Functional Classes and composing generic Objects) by focusing on the functional aspects of the application.

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

## 7.2 States and state-transitions

It is recommended that a graphical presentation, such as **Grafcet (IEC 848)** or an informal tabular notation is used to specify the state machine for the given application, following the style of many OSI/ISO protocol descriptions. The tabular notation consists in listings of *events* → *states* → *actions* → *transition rules* (the Mealy/Moore model).

## 7.3 Functional Classes and Objects

The baseline for this work is the functional decomposition in the application model derived from the analysis of the application scenarios. The main task is to specify exactly the Functional Classes, i.e. the composing objects and Functional Actions. Functional Actions shall be specified from the application scenarios, but it is recommended that primitive object interactions are taken into consideration so as not to build too complex aggregation of object action invocations to achieve a specified Functional Action.

**Comment [EE24]: Ron:**

Need to check what else is recommended/standardised out there (ISO / IEC). Leave it open for comment. Also Check the IEC 61499 for recommendations for graphical representation language and tool, or what UML defines.

Ron is going to check in W3C what they use for programming specifications or similar specification activities.

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

## Annex A An Interoperability Application Specification (example)

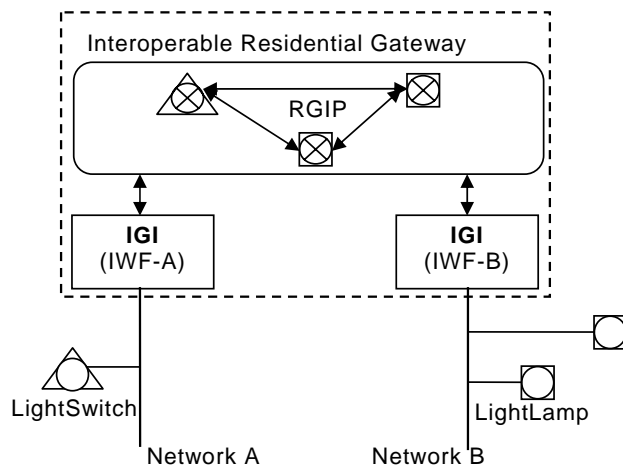
To be added based on SC25WG1 Technical Reports on HES Application Models (lighting).

### A.1 Lighting Application

This example is taken from PDTR Home Electronic System – Guidelines for Product Interoperability (ISO/IEC JTC1 SC25 N889), Lighting Subsystem section.

#### A.1.1 Procedure

The procedure followed in this example consists in the definition of a generic lighting subsystem application model (simple case). The components of the system are installed on two different network systems (namely System A (possibly LonWorks) and System B (possibly CEBus). The components of the framework are exemplified in (a) logical interoperable system, (b) respective functional/device profiles in each system, and (c) examples of the interworking functions for each case.



#### A.1.2 Generic Lighting Subsystem Application Model

##### A.1.2.1 Functional Classes

The examples will include partial XML schemas for the individual components, and exemplify the lexicon document (Generic HES Interoperability – GHESIO)

**Comment [EE25]:** This can be added based on one of the analysed application models; for example the lighting context control (technical Report type 2 (TR2 in SC25WG1), or the lighting functional profiles from LonWorks interoperability guidelines, or the energy management system from EHS1.3a, which initially Konnex was going to adopt.

To be discussed at the London 2002 meeting.

**Comment [EE26]:** Make it an annex, and exemplify with lighting control (Lon + CEBus)

**Deleted: Application models and lexicon**

**Formatted:** Bullets and Numbering

**Deleted:** interop\_p2-D2v02\_N1076.doc

**Deleted:** interop\_p2(D2v02)\_N1076.doc

- A.1.2.1.1 LightSwitch
- A.1.2.1.2 LightLamp
- A.1.2.1.3 LightSensor
- A.1.3 System A
  - A.1.3.1 System A: Light Switch Functional Profile
  - A.1.3.2 System A ↔ SysA-IOP Interworking Function
- A.1.4 System B
  - A.1.4.1 System B: Light Lamp Context
  - A.1.4.2 System B: Light Sensor Context
  - A.1.4.3 System B ↔ SysB-IOP Interworking Function
- A.1.5 Interoperable System – Functional Model

Deleted: interop\_p2-  
D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076  
.doc

## Annex B Notes on interoperability

### B.1 Taxonomy definitions

Application Domain ::= SEQ of { Application Model }

Application Model ::= SEQ of { Functional Class }

Functional Class ::= SEQ of { Object, Functional Actions }

Object ::= SEQ of { Basic Property, Operational Property, Configuration Property }

Device Profile ::= A document description of the minimum a collection of Functional Class instances that are implemented in a product to allow this product (physical device) to comply with the functionality specified in the Application Model(s).

### B.2 Taxonomy of the existing systems

The following table represents a mapping between different entity names into the interoperability taxonomy framework.

LonWorks	CEBus	EIB/KNX	EHS	Generic
Network / Configuration Variable	Instance Variable	Property	Object (Object Type)	Property
<u>Functional Block</u>	<u>Object</u>	<u>Object</u>	<u>Object</u>	<u>Object</u>
<u>Functional Profile</u>	<u>Context</u>	<u>Device</u>	<u>Device</u>	<u>Functional Class (logical device)</u>
<u>?(Application Model)</u>	<u>? (Application Model / Group)</u>	<u>Application Specification</u>	<u>Application Model</u>	<u>Application Model</u>
<u>Device Profile</u>	<u>Device Profile</u>	<u>Device Profile</u>	<u>Device Profile</u>	<u>Device Profile</u>

For example, the LonWorks Network Variable is not strictly an object, according to the definition adopted in this standard, as it does not exhibit behaviour. But it is the equivalent of the EHS::Object because it is implicitly accessed at application level through get/set actions, where allowed respectively by the access properties defined in the functional profile for the variable.

Objects may have direction (in/out, producer/consumer - inherited through their properties), access properties, an access list and notify list.

Deleted: interop\_p2-D2v02\_N1076.doc

Deleted: interop\_p2(D2v02)\_N1076.doc