KNX System Specifications

3

Architecture

1

Summary

This document contains the description of the KNX system architecture. It is intended for information before reading the entire specification.

Document Updates

Version	Date	Modifications
v1.0	2001.08.27	Incorporation of CSG comments and proposals – 17th CSG Released by CSG to be submitted to CTB for RfV
v1.1	2001.09.17	Integration of LTR and LTS according to KTB decision Draft Document for RfV
v2.0	2001.10.24	Integration of comments from RfV
v2.0	2002.03.06	Preparation of the Approved version/
V 3.0	2009.06	Updating in view of the publication of V2.0 of the KNX Standard

Filename: 03_01_01 Architecture v3.0.DOC
Version: 3.0
Status: --
Savedate: 2009.06.24
Number of pages: 26

Contents

1 Introducing the KNX Network 4

2 Elements of the KNX Architecture 5
2.1 Applications, Interworking and Binding 6
2.2 Basic Configuration Schemes 6
2.3 Network Management and Resources 6
2.4 Communication: Physical Layers 7
2.5 Communication: Common Kernel and Message Protocol 8
2.6 Resources 8
2.7 Device Models 9
2.8 Device Identification 9

3 System Capabilities, Communication and Addressing Models 10
3.1 Logical Topology and Individual Address Space 10
3.2 Network & Resource Management with Broadcast and Unicast “Point-to-point” Services 11
3.3 Multicast “Group Addressing” for Run-time Efficiency 11
3.4 Frame Overview 11

4 Application Models, Datapoints and Binding 13
4.1 Datapoints and Distributed Applications 13
4.2 Group objects 13
4.3 Properties of Interface Objects as Datapoints 14
4.4 Free or Structured Binding 14
4.5 Tagged Binding 15

5 Interworking Model 17
5.1 The Application: Datapoint Types and Functional Blocks 17
5.2 Parameter Datapoints 17
5.3 Good Citizenship and Multi-mode Integration 18

6 Configuration Modes 19
6.1 General 19
6.2 System Mode 19
6.3 Controller Mode 20
6.4 Push-button Mode 20
6.5 Logical Tag Extended Mode 21

7 Profiles 22
7.1 Definition and Use 22
7.2 Profiles description 22
7.3 Profiles as Guideline to this Specification 22

8 ETS™, eteC, KNXnet/IP 24
8.1 The ETS Tool Family 24
8.2 The eteC Components and API’s 24
8.3 KNX Broadband, Intranet, Internet and Integration Services 25

9 Certification 26

1 Introducing the KNX Network

This chapter outlines the main elements of the KNX system, and the concepts behind it. It should be useful as a guideline for newcomers to the system in finding their way around the KNX specification, for product managers and development engineers looking for suitable implementation options within the system, as well as for those with experience from KNX' "parent systems" to get acquainted with some new terminology and challenging new possibilities.

Building Control technology as provided by KNX is a specialized form of automated process control, dedicated to the needs of home and building applications. One premise for KNX is to furnish a radically decentralized, distributed approach; hence the term *network*.

The KNX Device Network results from the formal merger of the 3 leading systems for Home and Building Automation (EIB, EHS, BatiBus) into the specification of the new KNX Association. The common specification of the "KNX" system provides, besides powerful runtime characteristics, an enhanced "toolkit" of services and mechanisms for network management.

On the KNX Device Network, all the devices come to life to form distributed applications in the true sense of the word. Even on the level of the applications themselves, tight interaction is possible, wherever there is a need or benefit. All march to the beat of powerful Interworking models with standardized Datapoint Types and "Functional Block" objects, modelling logical device channels.

The mainstay of S-("System")Mode is the centralized free binding and parameterisation (typically with the PC-based ETS tool). It is joined by E ("Easy")-mode device profiles, which can be configured according to a structured binding principle, through simple manipulations – without the need for a PC tool. These configuration modes share common run-time Interworking, allowing the creation of a comprehensive and multi-domain home and building communication system.

The available Twisted Pair and Powerline communication media are completed with Radio Frequency (868 MHz band).

KNX explicitly encompasses a methodology and PC tools for Project Engineering, i.e. for linking a series of individual devices into a functioning installation, and integrating different KNX media and configuration modes. This is embodied in the vendor-independent Engineering Tool Software (ETS) suites for Windows.

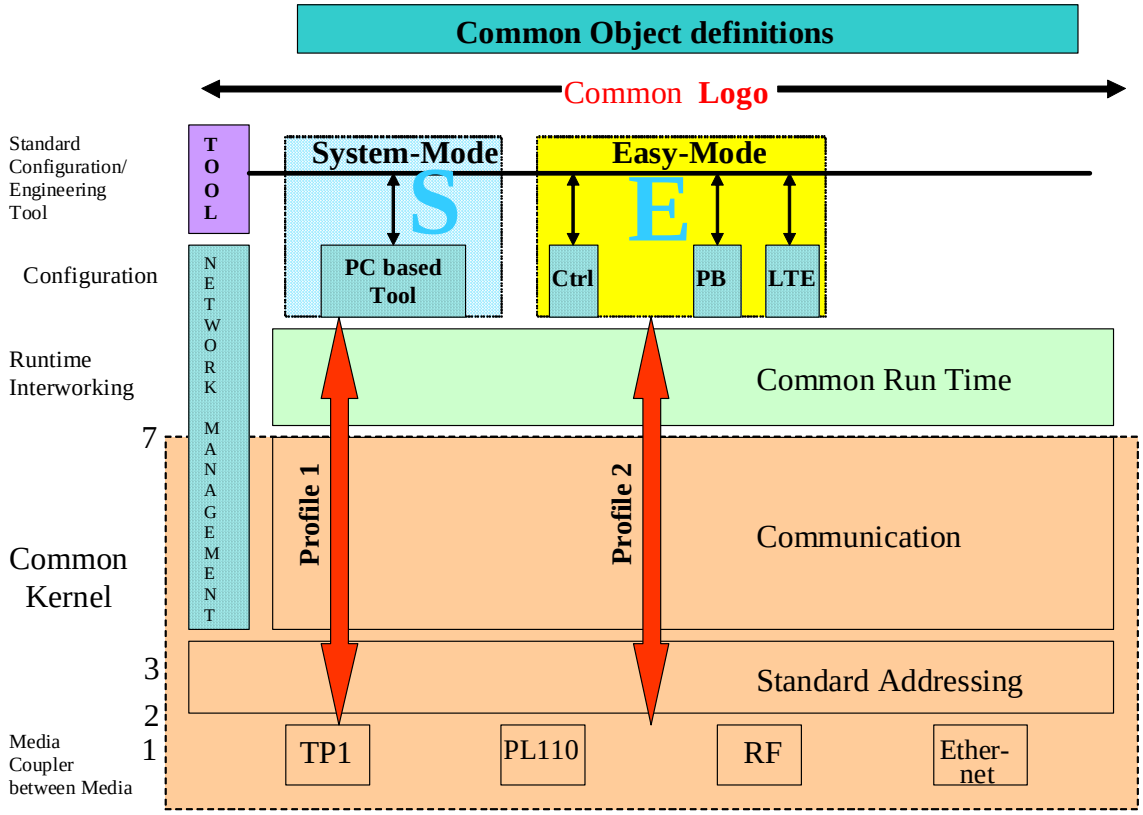
In contrast to the "one size fits all" creed, the KNX system is entirely independent of any specific microprocessor platform or even architecture. Depending on the profile chosen by the manufacturer, he can select any suitable industry-standard chip, or opt for available KNX OEM solutions like Bus Coupling Units, BIM's, chip sets etc. Some KNX profiles allow a tiny system footprint (say < 5 kb), and easily run on an 8-bit processor. Other implementations use 16- or 32 bit processors, or even PC's in the full sense of the word.

Through all of the above, KNX Device Networks may be flexibly adapted to present an optimal solution for each application domain and installation. Furthermore, they have also the capability to be inserted in a "Service Network" environment (usually based on broadband networks running IP, the Internet Protocol), to further amplify and leverage the benefits of our intelligent home, office or business environment.

Joining all these requirements into one common, streamlined system – fulfilling stringent compatibility requirements with a large installed base – is no mean feat. The next section summarizes the essential bricks KNX uses to accomplish all this, while further sections zoom in more closely on some distinctive features and characteristics of the KNX system.

2 Elements of the KNX Architecture

KNX specifies many mechanisms and ingredients to bring the network into operation, while enabling manufacturers to choose the most adapted configuration for their market. Figure 1 below shows an overview of the KNX model, bringing the emphasis on the various open choices. Rather than a formal protocol description, the following details the components or bricks that may be chosen to implement in the devices and other components a full operational system.



Ctrl = Controller Approach) **PB** = Push Button approach **LTE**= Logical Tag extended

Figure 1 - The KNX Model

As essential ingredients of KNX, we find in a rather top-down view.

- Interworking and (Distributed) Application Models for the various tasks of Home and Building Automation; this is after all the main purpose of the system.
- Schemes for Configuration and Management, to properly manage all resources on the network, and to permit the logical linking or binding of parts of a distributed application, which run in different nodes. KNX structures these in a comprehensive set of Configuration Modes.
- a Communication System, with a set of physical communication media, a message protocol and corresponding models for the communication stack in each node; this Communication System has to support all network communication requirements for the Configuration and Management of an installation, as well as to host Distributed Applications on it. This is typified by the KNX Common Kernel.
- Concrete Device Models, summarized in Profiles for the effective realization and combination of the elements above when developing actual products or devices, which will be mounted and linked in an installation.

Below, let's have a closer look on how KNX deals with all of this.

2.1 Applications, Interworking and Binding

Central to KNX' application concepts is the idea of Datapoints: they represent the process and control variables in the system, as explained in the section Application Models. These Datapoints may be inputs, outputs, parameters, diagnostic data,…The standardized containers for these Datapoints are Group Objects and Interface Object Properties.

The Communication System and Protocol are expected to offer a reduced instruction set to read and write (set and get) Datapoint values: any further application semantics is mapped to the data format and the bindings, making KNX primarily "data driven".

In order to achieve Interworking, the Datapoints have to implement Standardized Datapoint Types, themselves grouped into Functional Blocks. These Functional Blocks and Datapoint Types are related to applications fields, but some of them are of general use and named functions of common interest (such as date and time).

Datapoints may be accessed through unicast or multicast mechanisms, which decouple communication and application aspects and permits a smooth integration between implementation alternatives.

The Interworking section below zooms in on these aspects.

To logically link (the Datapoints of) applications across the network, KNX has three underlying binding schemes: one for free, one for structured and one for tagged binding. How these may be combined with various addressing mechanisms is described below.

2.2 Basic Configuration Schemes

Roughly speaking, there are two levels at which an installation has to be configured. First of all, there is the level of the network topology and the individual nodes or devices.

In a way, this first level is a precondition or "bootstrap" phase, prior to the configuration of the Distributed Applications, i.e. binding and parameter setting.

Configuration may be achieved through a combination of local manipulations on the devices (e.g. pushing a button, setting a codewheel, or using a locally connected configuration tool), and active Network Management communication over the bus (peer-to-peer as well as more centralized master-slave schemes are defined).

As described in the corresponding section below, a KNX Configuration Mode:

- picks out a certain **scheme for configuration** and binding
- maps it to a particular **choice of address scheme**
- completes all this with a **choice of management procedures** and matching resource realizations.

Some modes require more active management over the bus, whereas some others are mainly oriented towards local configuration.

2.3 Network Management and Resources

To accommodate all active configuration needs of the system, and maintain unity in diversity, KNX is equipped with a powerful toolkit for network management. One can put these instruments to good use throughout the lifecycle of an installation: for initial set-up, for integration of multi-mode installations, for subsequent diagnostics and maintenance, as well as for later extension and reconfiguration.

Network Management in KNX specifies a set of mechanisms to discover, set or retrieve configuration data actively via the network. It proposes Procedures (i.e. message sequences) to access values of the different network resources within the devices, as well as identifiers and formats for these resources – all of this in order to enable a proper Interworking of all KNX network devices. These resources may be addresses, communication parameters, application parameters, or complex sets of data like binding tables or even the entire executable application program.

The network management basically makes use of the services offered by the application layer. Each device implementing a given configuration mode (see below) has to implement the services and resources specified in the relevant "profile" (set of specifications, see below).

For managing the devices, these services are used within procedures. The different configuration modes make use of an identified set of procedures, which are described in the "configuration management" part. As indicated above, and further demonstrated in the Configuration Modes section below, KNX supports a broad spectrum of solutions here, ranging from centralized and semi-centralised "master-slave" versions, over entirely peer-to-peer to strictly local configuration styles.

However, mechanisms and Resources are not enough. Solid Network Management has to abide by a set of consistency rules, global ones as well as within and among profiles, and general Good Citizenship. For example, some of these rules govern the selection of the (numerical value of) the address when binding Datapoints.

But now, we first turn our attention to how the Communication System's messaging solutions for applications as well as management, beginning with the physical transmission media.

2.4 Communication: Physical Layers

The KNX system offers the choice for the manufacturers, depending on his market requirements and habits, to choose between several physical layers, or to combine them. With the availability of routers, and combined with the powerful Interworking, multi-media, and also multi-vendor configurations can be built.

The different media are :

- TP 1 (basic medium inherited from EIB) providing a solution for twisted pair cabling, using a SELV network and supply system. Main characteristics are: data and power transmission with one pair (devices with limited power consumption may be fed by the bus), and asynchronous character oriented data transfer and half duplex bi-directional communication. TP 1 transmission rate is 9600 bit/s.
  TP1 implements a CSMA/CA collision avoidance. All topologies may be used and mixed (line, star, tree, ….)
- PL 110 (also inherited from EIB) enables communication over the mains supply network. Main characteristics are: spread frequency shift keying signalling, asynchronous transmission of data packets and half duplex bi-directional communication. PL 110 uses the central frequency 110 kHZ and has a data rate of 1200 bit/s.
  PL110 implements CSMA and is compliant to EN 50065-1 (in the frequency band without standard access medium protocol).
- RF enables communication via radio signals in the 868,3 MHz band for Short Range Devices. Main characteristics are: Frequency Shift Keying, maximum duty cycle of 1%, 32 768 cps, Manchester data encoding.
- Beyond these Device Network media, KNX has unified service- and integration solutions for IP-enabled [1] media like Ethernet (IEEE 802.2), Bluetooth, WiFi/Wireless LAN (IEEE 802.11), "FireWire" (IEEE 1394) etc., as explained in the KNXnet/IP section below.

1 IP = Internet Protocol

2.5 Communication: Common Kernel and Message Protocol

The Communication System must tend to the needs of the Application Models, Configuration and Network Management. On top of the Physical Layers and their particular Data Link Layer, a Common Kernel model is shared by all the devices of the KNX Network; in order to answer all requirements, it includes a 7 Layers OSI model compliant communication system.

- Data Link Layer General, above Data Link Layer per medium, provides the medium access control and the logical link control.
- Network Layer provides a segment wise acknowledged telegram; it also controls the hop count of a frame. Network Layer is of interest mainly for nodes with routing functionality.
- Transport Layer (TL) enables 4 types communication relationship between communication points: one-to-many connectionless (multicast), one-to-all connectionless (broadcast), one-to-one connectionless, one-to-one connection-oriented. For freely bound models (see below), TL also separates (“indirects”) the network multicast address from the internal representation.
- Session and presentation Layers are empty.
- Application Layer offers a large “toolkit” variety of application services to the application process. These services are different depending on the type of communication used at transport layer. Services related to point-to-point communication and broadcast mainly serve to the network management, whereas services related to multicast are intended for runtime operation.

Remember KNX does not fix the choice of microprocessor. Since in addition, KNX covers an extensive range of configuration and device models, the precise requirements governing a particular implementation are established in detailed Profiles, in line with the Configuration Modes. Within these boundaries, the KNX developer is encouraged to find the optimal solution to accommodate his implementation requirements! This is expounded in later sections.

As we shall also find out later, the KNX message frame or telegram format also reflects this communication structure.

2.6 Resources

We have seen that Network Management consists of Procedures for manipulating Resources, and that the Common Kernel provides a toolkit of services for this purpose. Given central importance for the system, these Resources merit some further consideration.

Remember that they can be:

- “System” (configuration) resources, with address, lookup and parameter information to help the layers of the communication system carry out their task.

  By way of example, we mention the address and indirection tables for free Group Communication or the Individual Address of the node as such. Still among the system resources, we also find “discovery” information, which allows a partner on the network to find out about the capabilities of some other node or application. (One bonus of this is certainly that a very rich interaction becomes possible between Configuration Controllers or PC-based tools such as ETS, and the network.)

- Parameters controlling the application.

The present specification gives detailed descriptions not just of the role, but also of the identifiers, formats and encoding of each resource, which is after all a (set of) data element(s).

Some different formats may be defined for a given abstract resource, allowing for simpler or more sophisticated realizations, and perhaps depending on the Configuration Mode. For complex resources (like binding tables), realizations which are more white-box allow more management responsibility to be shifted to the Configuration Master.

It is worth noting that KNX' Interface Objects, provide a powerful, implementation-independent framework for realizing resources, the individual elements of which can be modelled as Datapoints. Interface Objects and their relationship to Datapoint are explained below.

At any rate, all identifiers and formats given in the specification shall be understood as a network interface, i.e. not necessarily an internal memory map of the device.

2.7 Device Models

Eventually, a KNX installation always consists of a set of devices connected to the bus or network. Every facet we have discussed so far is ultimately realized in and through the devices. All of these adhere to a number of *logical node architectures* for devices harbouring resources and implementing the protocol. Models vary according to node capabilities, management features and configuration modes; and not to forget, according to its role in the network, e.g. typical "application (end) device", configuration master, router, gateway etc.

KNX also standardizes certain general-purpose device models, such as for Bus Coupling Units (BCUs) or Bus Interface Modules (BIMs), mainly used in combination with ETS and downloadable application programs. Specifically for platforms like these, supplementary "hosting" API's [2] are defined, such as the Communication Object model (see below), and the External Message Interface (EMI).

Together with the characteristics of the Configuration Modes, these device models are all laid down in the Profiles.

2.8 Device Identification

In complement to the basic operational system, a set of identification mechanisms is provided:

- Devices may be identified and subsequently accessed throughout the network either by their individual address, or by their unique serial number, depending on the configuration mode.
- Installation's extendibility and maintenance is considerably eased through product identification (i.e. a manufacturer specific reference) and functional identification (manufacturer independent) information retrievable from devices.

Mechanisms are defined around the unique serial number feature to

- get individual address of a device with a given serial number (so providing further access)
- set individual address of a device with a given serial number
- retrieve serial number of a device at a given individual address

The uniqueness of the serial numbers is ensured through controlled allocation of number ranges by the KNX Association Certification Department

2 API = Application Programming Interface

3 System Capabilities, Communication and Addressing Models

Before tackling the Configuration Modes and the Application and Interworking Models, it is good to have a better understanding of the Addressing Schemes and the related Communication Modes of KNX.

The encoding space for addressing fixes some fundamental capabilities of the system in terms of size (maximum number of addressable devices and Datapoints). The addressing is reflected in the encoding format of the message frame or telegram, as is the "shadow" of the communication stack and kernel. According to which principles addresses are used to identify Datapoints (binding) will be discussed in section 4.

3.1 Logical Topology and Individual Address Space

KNX is a fully distributed network, which accommodates up to 65'536 devices in a 16 bit Individual Address space. The logical topology or *subnetwork* structure allows 256 devices on one *line*. As shown in Figure 2 lines may be grouped together with a *main line* into an *area*. An entire domain is formed by 15 areas together with a *backbone line*.

Note that KNX KNXnet/IP optionally allows the integration of KNX subnetworks via IP.

As shown in Figure 2, this topology is reflected in the numerical structure of the individual addresses, which (with few exceptions) uniquely identify each node on the network.

On Powerline, nearby domains are logically separated with a 16-bit Domain Address. Without the addresses reserved for couplers, (255 x 16) x 15 + 255 = 61'455 end devices may be joined by a KNX network. Installation restrictions may depend on implementation (medium, transceiver types, power supply capacity) and environmental (electromagnetic noise, …) factors. Installation and product guidelines shall be taken into account.

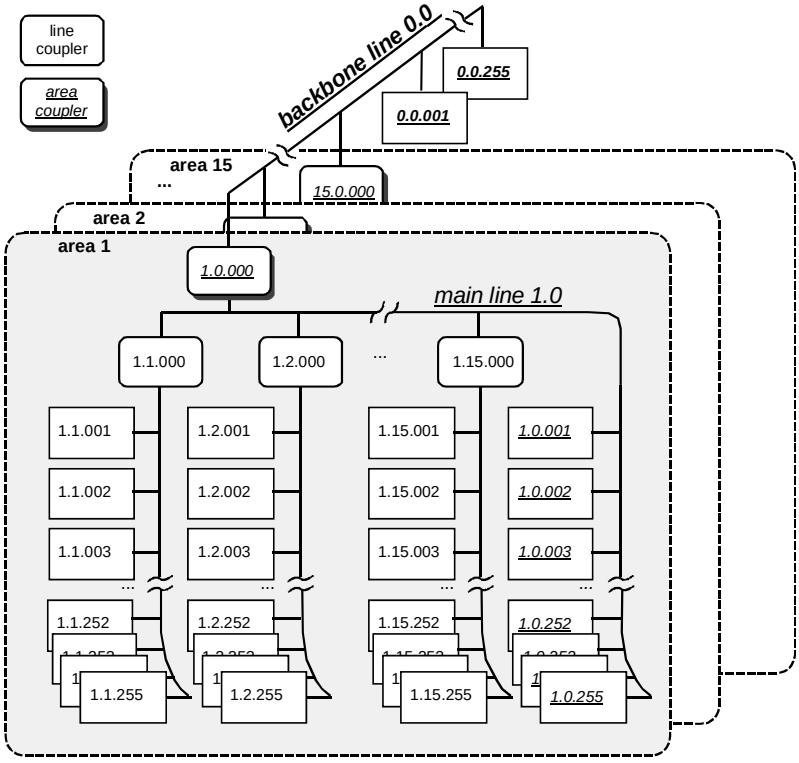


Figure 2 - The logical topology of KNX

Couplers connect lines or segments, e.g. within the Twisted Pair (TP) medium, or different media; their functionality may be (some combination of) repeater, bridge, router, package filter (for traffic optimisation), firewall protection etc. KNX defines various standard coupler profiles.

3.2 Network & Resource Management with Broadcast and Unicast “Point-to-point” Services

To manage network and device resources (e.g. when configuring an installation), KNX uses a combination of broadcast and point-to-point communication.

Most often, each device in the installation is assigned a unique Individual Address via broadcast (optionally using a device’s unique serial number), which is used from then on for further point-to-point communication.

- A connection (optionally with access authorisation) may be built up, for example to download the complete ‘applet’ binary image of an application program.
- Some resources may also be accessed in connectionless point-to-point communication.

3.3 Multicast “Group Addressing” for Run-time Efficiency

KNX supports full multicast (“group”) addressing, which provides the mainstay of KNX run-time communication. *Full* means that:

1. KNX is not limited to grouping devices: each device may publish several Datapoints (known as “(Group) Communication Objects”) individually, which can be grouped independently from one another into network-wide shared variables. As a bonus, properties of Interface Objects (see 4.3 "Properties of Interface Objects as Datapoints") may be published as shared variables as well.
2. As explained above in the description of the group-oriented KNX communication stack, a shared variable can be fully read/write bi-directional. In this way, all devices can also send unsolicited multicast frames.
3. KNX makes a 16 bit address space available for these shared variables. This signifies that one installation may have up to 64k shared variables (or “group addresses”), each with any number of local instances.

In this way as well, KNX goes some distance towards reducing the need for redundant automation hierarchy levels (and bandwidth!) through appropriate addressing and device modelling schemes.

Later paragraphs explain how Group Addresses may be used for free as well as for tagged binding schemes.

3.4 Frame Overview

Now’s the time to have a look at the actual KNX message format, as serially encoded in the frames or telegrams which are sent on the bus.

Depending on the modulation technique or access and collision control of any specific medium, some preamble or envelope sequence may be defined, which we ignore here. The following example format actually corresponds to the interface above Layer2. Special acknowledge frames etc. are all described at length in the actual specification.

octet 0	1	2	3	4	5	6		7	8	..	N - 1	N ≤ 22
Control Field	Source Address		Destination Address		Address Type; NPCI; length	TPCI	APCI	data/ APCI	data			FrameCheck

Figure 3 - KNX LPDU standard frame structure (long frames allow N < 255)

First of all, the Control Field determines the frame Priority and distinguishes between the Standard and Extended Frame. In each case, there is an individual Source Address and individual (unicast) or group (multicast) Destination Address; the Destination Address type is determined by a special field.

A frame's Hop Count is decremented by routers to avoid looping messages; when it becomes zero, the frame is discarded from the network.

The TPCI [3] controls the Transport Layer communication relationships, e.g. to build up and maintain a point-to-point connection. In turn, the APCI [4] can tap into the full toolkit of Application Layer services (Read, Write, Response, …) which are available for the relevant addressing scheme and communication relationship.

Depending on the addressing scheme and APCI, the standard frame can carry up to 14 octets of data. Segmentation for bulk transfer, like the download of an entire application program, is the responsibility of the management client, e.g. the ETS tool.

The standard frame ensures direct upward compatibility from EIB. The extended frame can harbour up to 248 octets of data. Its usage is mainly defined for LTE Mode.

Finally, the Frame Check helps ensure data consistency and reliable transmission.

3 Transport Layer Protocol Control Information

4 Application Layer Protocol Control Information

4 Application Models, Datapoints and Binding

Ultimately, all elements of the KNX architecture we have met so far just serve as infrastructure and means for getting application for lighting, HVAC, security, … to run on the system. In this section, we investigate the central role in KNX application modelling of Datapoints and how they are linked ("bound"). Once we have investigated their role and appearance, we are ready proceed to the Interworking section, which lies at the heart of KNX.

4.1 Datapoints and Distributed Applications

KNX models an application on the Device Network as a collection of sending and receiving Datapoints, distributed over a number of devices.

The system comes to life when Datapoints in different devices are linked via a common identifier, in other words *bound*, as exemplified by the multicast Group Address. Accordingly, data can be transferred between different devices, each with its local application, which is of course the whole purpose of having a network.

When a local application in a device, a sensor, say, writes a value to a sending Datapoint, this device sends a ("write") message with the corresponding address and the new value. A receiving Datapoint with this same address will receive this value, and inform its local application. In turn, this receiving application can now act upon this value update if it wishes to do so. This action can be an internal state change or updating one of its own sending Datapoints (like in a controller), or modifying some physical output status (for example in an actuator device); or indeed any combination of these.

In this way, local applications in a number of devices, with linked Datapoints, combine to form a Distributed Application. Underscoring their quintessential role in KNX, Datapoints will figure prominently in the Interworking Models.

Next, we turn to different approaches, which KNX permits for such logical linking or *binding* of communication partners in the application, and for the realization of a Datapoint.

KNX encompasses three underlying schemes for linking Datapoints, according to whether the value of the address carries semantic information or not, and whether the binding is precisely predefined, or merely follows some loose rules; this leads to the following classification:

- free binding;
- structured binding;
- tagged binding.

To achieve a thorough understanding of the Configuration Modes later on, we first proceed to investigate the ways to realize a Datapoint, and then look at each of these individually.

4.2 Group objects

KNX's principal realization form for Datapoints, is given by the Group objects; as their name suggests, they are accessed via standard, multicast "group" addressing. The links are precisely the Group Addresses. Combined with the standard group message format, they make up the foundation of the system's cross-discipline Interworking and multi-mode integration facilities.

As we shall see below, Group objects can be used with free, structured or tagged binding alike – always assuming standard Group Addressing, of course. Depending on the application and the configuration mode, the awareness of the local application program may vary, as to its communication partners on the network or the actual address values used on the bus.

In this respect, KNX's Group objects permit an interesting (if optional) encapsulation pattern, which provides a local application with a systematic, binding-independent interface to the bus. Applying this, applications using Group Addressing may effectively achieve the most radical decoupling from the network communication. This encourages reuse of the same application code for different Configuration Modes.

Here, the (local) application sees the bus as a limited set of Group objects; these correspond to those Datapoints, which are of direct relevance to it. Put simply, each Communication Object appears to its application as a local variable with supporting attributes. Not surprisingly, the variable holds the value received from, or to be sent to the bus. Via the attributes, a default handler on top of the node's communication stack can inform the application that the corresponding value has been updated; vice versa, the application can request the stack to send a value. Clearly, this assumes a cyclic polling on both sides of the interface; more sophisticated implementations may map this interface to a custom call-back handler.

Some Group Communication stack versions support this purpose with 2 levels of indirection. Transport layer converts a received Group Address to a purely local "internal reference" (using the Group Address Table resource). Now it's up to Application Layer to map this reduced internal message, with possible multiplexing, to one ore more Communication Object Number identifiers (by means of the Association Table resource). The converse happens for sending (without local multiplexing). S-Mode exploits this n-to-m flexibility to the furthest.

4.3 Properties of Interface Objects as Datapoints

To accommodate additional requirements, KNX also provides a more intrinsic notion of Datapoint, in the form of a Property belonging to an Interface Object. The Interface Object simply groups a set of *property* Datapoints into a common interface structure or object.

Whereas a node's Group objects constitute a flat set of Datapoints, which are each directly addressed, each property of an Interface Object is *referenced relative to this Object*, according to the <ObjectReference>.<Property> pattern. In this respect, an Interface Object is well suited to model a Functional Block (see clause 5.1 "The Application: Datapoint Types and Functional Blocks") from the Application Models.

Clearly, Interface Objects are not limited to application Datapoints; they also allow a Datapoint-style modelling of management resources in the devices.

The Interface Object itself is referenced relative to the node, in standard addressing with an Index and on point-to-point communication. In this fashion, they are used for configuration and parameter setting.

LTE Mode uses extended addressing and references the Interface Object through the combination <ObjectType>.<InstanceNumber>.

In each case, the <ObjectReference>.<Property> combination forms an explicit tag in the message, to identify a specific Datapoint. What this means is explained in the paragraph on tagged binding below.

4.4 Free or Structured Binding

We have already met the KNX Communication & Addressing models, distinguishing between broadcast, multicast and unicast. Obviously, addressing comes into play when different local applications running in different nodes have to be bound (linked). In the present and next section, we focus on the *binding* principle underlying the selection and assignment of the address, regardless how the message is distributed. We will see that, from a network management point of view, binding is a distinguishing characteristic of the Configuration Modes.

Indeed, in order to establish a link on one given communication partner, we have to go through two steps:

1. to select the numerical value of the address;
2. to assign the address to the Datapoints to be linked.

Both the free and structured binding schemes discussed immediately below, assume *free addressing*. This means that the numerical value of the address carries absolutely no application semantics. The only assumption is that all Datapoints who wish to communicate directly with each other, be assigned the same address. This concept stands in contrast with tagged addressing, which we will come across in the next section.

The next aspect is the assignment of the chosen address, where we distinguish the following categories:

- *free binding:* there is no *a priori* prescription on which Datapoints may be linked to one another, apart from some very general "equal Datapoint type" consistency rules; in combination with free addressing, this supports customized multicast grouping at the level of individual Datapoints – i.e. not just at the level of Functional Blocks or even devices. Free binding is central to S-Mode.
- *structured binding:* the application model in the KNX specification, stipulates a precise pattern for linking a whole set of Datapoints, usually corresponding to a Functional Block or Channel; but remember: the address value as such is free. Controller and Push-button Modes follow this concept.

4.5 Tagged Binding

Several modes rely on a *tagged binding* approach. Their binding is also pre-structured by the application models, but here, the numerical value of address is *never* don't care; instead, part of its value, the *semantic (Datapoint) identifier*, directly implies the target Datapoint(s) in the communication partner.

For the "tagged E-Mode" (LT-E) using this principle, tagged binding goes hand in hand with zoning. The *logical tag* or *zoning* part of the address selects the intended communication partner(s) at the level of the device. So for a given Datapoint, its semantic ID is fixed by the application model; the zone is set – often locally on the device – for a particular instance in a given installation. This concept caters for simple binding between devices or Functional Blocks, but of course according to a predetermined Application Model. When Datapoints have been assigned the same zone, they clearly form a group, so zoning also adheres to the multicast principle.

KNX foresees 3 possibilities for this tagging to be realized.

1. Tag mapped to standard Group Addressing

   Some E-Configuration Modes map the semantic ID to a Connection Code, in a fixed range of Group Address space; the remainder of the Group Address is then the zoning tag; schematically: <GroupAddress> = <Zone>.<ConnectionCode>. In standard KNX Group Addressing, special ranges are reserved for tagged addressing.

2. Properties on Individual Addressing

   To accommodate complementary application and configuration requirements, KNX also supports Datapoints where the tag is not encoded in the group address.

   These implementations exploit the implementation-independent structure of KNX Interface Objects, which allows one individual Datapoints to be considered (and addressed) as a property of such an object. In this case, this takes on the explicit form <Target>.<SemanticID>, with <SemanticID> = <ObjectReference>.<PropertyID>.

This mechanism can be used on individual addressing, in this case, <Target> = <IndividualAddress> of the node. To this end, the Application Controller keeps a persistent copy of the Individual Address of the devices it has enrolled. In this case, there is of course no zoning concept.

3. Properties on Extended Addressing

The same Principle as in (B) may also used on Interface Objects via extended group addressing, as is done in LTE mode. The group address space of LTE mode is designed especially with this purpose in mind and is only used for the zone information. The connection code (or semantic ID) is build from ObjectType + Property ID.

In the upshot, the application models for tagging have strong and detailed linking semantics already fixed in the model itself, which is in turn embedded in the devices, via the semantic identifiers. With free binding in contrast, much of the application can be (or of course: has to be) designed and tailored explicitly to the needs of each individual installation; this is also what designing a project in ETS ultimately amounts to.

5 Interworking Model

Having set the scene in the preceding section, time has come to discover the concept of KNX Interworking. Holding out the promise to a world of open, multi-vendor installations for intelligent homes and buildings, Interworking is, so to speak, the icing on the KNX cake.

For installers and integrators – or indeed, the consumer! – the Interworking Models are definitely the most valuable and valued among the numerous assets KNX has to offer. Interworking guarantees them the possibility to achieve the richest possible integration between devices within any application, as well as between various application domains – especially when combined with the ETS project tools.

When the Interworking concepts were laid down, great care was taken to ensure continuity with the parent systems of KNX, and to consolidate and extend the multi-vendor setting.

5.1 The Application: Datapoint Types and Functional Blocks

KNX Interworking principles clearly rest upon the Application Model from the previous section.

Essentially, they describe how each local application looks, when seen from the network; in other words: what is its Datapoint interface? Completed, to the relevant extent, with its intended behaviour in terms of internal state machines, message and physical I/O, this constitutes the so-called Functional Block description of each local part of the distributed application.

Inside the Functional Block specification, each Datapoint is assigned an explanatory name, together with its required Datapoint Type; the type fixes the format of the data, which the Datapoint sends to or receives from the bus. The core set of the KNX data type family comprises types for:

- Binary Value (Boolean)
- Relative Control (“%”)
- Analogue Value (long and short float)
- Counter Value (signed and unsigned integer)
- Date & Time
- Status (bit field)
- …

For implementation in tagged binding models, the Functional Block description also has to specify the (standardized!) semantic identifier of its Datapoints.

5.2 Parameter Datapoints

“Application variable” Datapoints exhibiting such general purpose types cover most of the elementary communication needs for run-time operation. For a particular application, its variables in this sense enable it to perform its essential functionality. Complementary to these we find Parameters: specialized Datapoints, sometimes requiring more specific types, to give more subtle and sophisticated control over the basic conduct of the application.

Using System Mode, parameter Datapoints may be implemented in a more private manner, without impairing the fundamental task of the application. “More private” means e.g. that they can only be accessed by a client with a certain knowledge about the implementation. On the other hand, this assumes that this knowledge itself is available in a neutral way, as is the case with the ETS project design tools (with the help of detailed information from the manufacturer’s product database entry for the relevant product and application, possibly even describing a detailed memory map), or an appropriately configured Building Control Station.

5.3 Good Citizenship and Multi-mode Integration

Proper Interworking implies some additional prerequisites. Surely, run-time application Interworking as portrayed in the previous paragraph becomes worthwhile only as soon as the different devices and subsystems involved, can be linked and configured among one another to begin with. The broad spectrum of Configuration Modes, which KNX allows, of course adds to this challenge.

An important step towards coming to grips with this diversity is for a minimum set of *Discovery* prerequisites to be fulfilled. Each device and subsystem is expected to be able to furnish essential information about itself to interested partners on the KNX Device Network, for example the KNX Profile it implements. To this end, KNX defines a concise set of Descriptor fields and objects (resources!), with corresponding discovery procedures.

In addition, this specification prescribes for each Configuration Mode, via the Profiles, minimum management requirements for its devices, to allow for flexible integration.

Vice versa, all KNX Configuration Masters have to master (!) sufficient “search” capabilities to find their way around in a multi-mode KNX network, and are of course expected to manage the part of the network within their responsibility as good citizens with respect to the rest of the system, and in compliance with the specification.

In all of this, the ETS tools have a special role to play: to allow rich and meaningful integration between different Configuration Modes, in one single installation. In the hands of the system integrator, ETS becomes the final arbiter. The KNX Association is mounting a special development effort, to make the requisite tools available in the shortest possible time.

6 Configuration Modes

6.1 General

Let's return now to the issue of Configuration. To address many diverse needs, the KNX specification includes a *toolkit* of management features enabling the choice between several configuration modes, each adapted to different markets, local habits, level of training needed or application environment.

The specification ensures some degree of freedom for the manufacturer, whilst guaranteeing consistency and Interworking in one mode, even in a multivendor context. All configuration modes include provisions in order to extend or modify the installation using the same mode.

By use of ETS, extension and Interworking in multimode installations is made possible. The main reasons for this are the use of consistent management procedures, and for the run-time, exchange of data via the group objects, accessed through group addressing and multicast communication. ETS uses the device descriptor feature of the devices to know the type and mode used in the device. It then uses the management procedures corresponding to the information retrieved.

Devices compliant to one configuration mode shall implement the network management and runtime profiles, as stated in the relevant Volume 6. Specifications to the Network Management are given in Volume 3 Part 5.

6.2 System Mode

Devices implementing System Mode offer the most versatile and multi-usage configuration process, while permitting a compact implementation: the complexities of binding and application configuration are shifted to a powerful configuration master. Traditionally this role is taken on by a set of PC based tools from the ETS family, supplied by the KNX Association. With the aid of the ETS project tools, one can configure these devices and set them into operation.

For the special information this requires about the devices, ETS makes use of a database representing all possible functionalities of the devices (or products) it supports; this "product template" information is created and maintained by each manufacturer for his own products, using dedicated ETS tools.

Usually, the manufacturer supplies the resulting database to his customers. The trained installer is now able to incorporate (import) into the database the product templates originating from several manufacturers, thus allowing him to build also complex and multi-vendor KNX Network installations. In this way, he may choose his functionalities among the broad offer from the various manufacturers.

The ETS tools for KNX project design support the configuration of the following features:

- Binding: setting the group addresses in order to enable group object communication between the Functional Blocks. Group objects may be set into relationship if they share the same Datapoint type. The possibly complex address and indirection tables are constructed by the configuration master (like ETS), and downloaded into the device.
- Parameterisation: setting the parameters of the devices according to the documentation of the manufacturer. Some parameters are standard for the considered Functional Blocks, some others may be manufacturer-specific.
- Download of application program is also possible for multi-purpose devices exhibiting this special feature. In particular devices consisting in two physical parts (e.g. flush mounted BCU + interchangeable application module) may offer different functions depending on the chosen and downloaded application program.

All of this adds up to the KNX installer himself designing and tailoring the desired Distributed Application functionality, to fit the precise needs of each individual installation – with the help of ETS, and using the building block libraries provided by the manufacturers in the form of their Product Databases. This relies entirely on the concept of free binding.

Mini-Profile : S-Mode uses the standard frame format. It needs active Network Management (as performed by ETS) with Broadcast and Individual Addressing. Run-time links by default employ the Group Address range for free binding; by design, tagged or binding links can be added to achieve run-time Interworking with the E-modes based on Connection Codes.

6.3 Controller Mode

The Controller Mode is defined to support installation of a limited number of devices on one logical segment of a physical medium. An installation using the controller mode will comprise one special device called controller that is in charge of supporting the configuration process. The controller supports one or more applications (e.g. lighting). It is not needed, but recommended that the controller remains present in the installation at runtime.

The KNX network devices implementing this mode exhibit with limited parameterisation the various functionalities as described in the corresponding application specifications. They are provided with the ability to be dynamically configured by setting the needed individual and group addresses and parameters.

At configuration time, the role of the controller is to establish the links between the so-called “channels”, which represent the set of group objects for the given functionality. The links and parameters are identified by an action of the installer, which may be different from one controller manufacturer to another. However, the set of channels supported by the KNX specification is uniquely specified, and therefore any device from any manufacturer will be taken into account by any controller from another manufacturer. The controller doesn’t need to have any knowledge of the functionalities supported by the devices. This functionality can be read out of the devices by the controller and is “hard coded” in the so called Device Descriptor #2 implemented by each device.

Following the instructions of the installer, the controller assigns individual addresses to the devices, calculates the links at Datapoint level using known rules which are part of the specification, and sets the parameters which are available for the considered devices. The runtime operation is independent from the configuration controller.

Mini-Profile : Ctrl-Mode uses Group communication with the standard frame format. It needs active Network Management (as performed by ETS) with Broadcast and Individual Addressing. Run-time links by default employ the Group Address range for structured binding.

6.4 Push-button Mode

The Push Button mode, as the controller mode is defined to support installation of a limited number of devices on one logical segment of a physical medium. There is however no need for a specialised device for configuration, therefore each device implementing the push button mode shall include the means of configuration related to its application. The devices implement fixed pametrizable functionalities as described in the corresponding application specifications.

Each device is provided with the ability to be dynamically configured by setting the needed individual and group addresses and parameters. Exchange of parameters is also possible, but will be mainly local on the devices.

At configuration time, the installer successively designates the devices the functions of which will be linked (therefore the name push button), the way he does it is manufacturer dependent. The exchange of configuration data occurring between devices, typically sensors and actuators, is made trough a single application layer service. Each sending Datapoint device acquires itself its unique group address. This address will be given to the receiving Datapoints using the pushbutton procedure. The configuration obeys to the rules of the channels and of the Functional Block descriptors given by the KNX specification.

Mini-Profile: For configuration, PB mode relies only on active management from one device directly by another. It uses group addressing with the standard frame, with structured binding.

6.5 Logical Tag Extended Mode

Device configuration is made using tags set locally by physical means.

For the time being, Logical Tag extended is limited to HVAC applications, which need longer set of structured data. These data are exchanged via interface objects using the extended frame of the KNX protocol. Exploiting the extended address space, the tags represent powerful zoning information, which is essential in modular structured applications (e.g. heating of big buildings).

Data points of general interest may be shared with other applications, and are defined in the specification. These Datapoints may be accessed by group objects as usual and linked on base of the tag settings.

Mini-Profile: LTE defines extended frames for its “native” run-time communication, with tagged binding on Interface Object properties. The applicable profile requires LTE devices to support a supplementary interface with freely bound, standard Group Address communication. They implement standard management for Individual Address assignment and the free binding.

7 Profiles

7.1 Definition and Use

As it has been stated repeatedly in this document, the KNX specification is a kind of “toolkit” where one has to extract a set of features which will enable a device to interwork within a given configuration mode and within the whole network.

In order to maintain a coherent system, and to help the design and enable certification, these sets of features have been grouped in so called “profiles”. Profiles are in limited number and have been designed to cover the needs and habits within the KNX Association community. Following a given profile (or a combination of them) will enable to build devices and systems which easily integrate into the KNX System.

Profiles are available for:

- System mode devices
- Easy mode devices
- Management clients for every mode

At certification time, the manufacturer shall declare to which profile(s) his device(s) conforms. Certification testing is then made accordingly.

7.2 Profiles description

Any profile may be based on the available media. The set of requirements to these media are given first in Volume 6 “Profiles”. The common kernel and network management features are then configuration-mode dependent.

Profiles for system mode devices are either generic (Sytem1 and System2), and ensure full compatibility with ETS, or encompass also standardized components features (BCU1, BCU2, CU) available in OEM.

These last profiles are mainly the available implementations inherited from EIB, but also newer ones, and provide supplementary to the generic features also extra hardware and embedded software features that lighten application design. Special profiles are needed for routers and bridges.

Easy mode profiles are only generic ones, but include also inheritance from previous system implementations: Controller mode DMA1 and DMA2, Push Button and Logical Tag Extended.

Profiles for management clients are laid down to enable the realization of the management devices corresponding to one configuration mode. The specification of these profiles guarantees the fact that within one mode, every management client is able to take any device into account within its given application field.

7.3 Profiles as Guideline to this Specification

The present chapter should give you a pretty good overview of the strengths and possibilities of the KNX system. It is recommended that you compare this carefully with your company’s application, product and market intentions.

Which Configuration Mode(s) best correspond to these intentions? Will I begin my development from scratch, or do I take a jump-start by using available OEM implementations? These are definitely among the first questions you have to answer, when contemplating KNX development.

To assess these options in more detail, as well as to take your development from there, the Profiles provide a very practical compass to guide you through this very elaborate specifications document. You may consider each single profile as a Table of Contents, giving a top-down view of consistent set of requirements for any corresponding implementation.

Finally, the Test Specifications in Volume 8 of this Specification, constitute the conclusive interpretation of the requirements given in Volume 3. For this reason, it is often advantageous to read the requirements and tests more or less in parallel. Proceeding in this way will answer many questions, and rule out any unpleasant surprises at a late stage of your development.

8 ETS™, eteC, KNXnet/IP

8.1 The ETS Tool Family

Throughout the preceding sections, we repeatedly encountered ETS, the suite of PC (Windows) software tools from the KNX Association.

The most famous members of the ETS family are the tools for the design and configuration of KNX installations or projects; these deal mainly with 2 tasks:

- design and configuration (management) of S-mode installations;
- integration of multi-mode KNX Device Networks.

As described in the paragraph on S-mode, these ETS modules rely on a "product database" with detailed information about each S-mode product, provided by the manufacturer. This description also allows ETS to show the product, its available (downloadable) application programs plus parameters, and the corresponding possibilities for Group Address binding in a graphical way to the user. This person can now effectively design the whole system off-line, and download this result subsequently into the system with all components mounted.

A stored ETS project in fact constitutes an off-line representation of the thus configured installation. This project database or repository contains rich descriptive meta-information about the installed system. This can readily be used for troubleshooting and diagnostics access, also remote. Another application is as reference for the configuration of Building Control Stations, Service Gateways, Intranet Couplers etc. The entire ETS repository model is XML [5]-enabled.

When connected to the bus, ETS behaves as an extremely powerful Configuration Master – be it always under the direction of the ETS user, say the installer or integrator. For integration between modes, ETS can scan the installation and "discover" the various elements present in the installation. This information can now be used to define cross-links or adjust parameters.

Combined with the iETS communication component for IP, the resulting internet ETS ensures access to KNX installations via any appropriate IP link. Apart from system interventions through the local LAN, iETS also caters for remote maintenance functionality.

For manufacturers, a specialized set of editors is available to create the visual *plug-in* representation of their products for the product database – without a single line of programming. As a result, ETS encourages a harmonized look-and-feel for all product entries from all manufacturers, and combines them all into a common environment for project engineering. Again, this improves project design efficiency and permits the smooth realization of multi-vendor installations. Various tools for network analysis and diagnostics further complete ETS.

8.2 The eteC Components and API's

ETS is built on top of a framework of DCOM [6] software-engineering components for PC/Windows platforms, called the *eTool Environment* – Component Architecture (eteC). eteC provides an abstract API and Object Model for on-line and off-line access to KNX resources. Viewed as a set of set of API's, eteC forms part of the KNX standard.

One practical – and very useful – consequence of mapping the KNX logic to DCOM objects is that the resulting eteC interfaces may be used with almost any programming language: C, C++, Java, Visual Basic, etc. Even macro and scripting languages, like JavaScript and VBscript are supported; this may seem uninteresting at first, but it dramatically lowers the threshold for using these potent building blocks, while at the same time increasing flexibility significantly.

5 XML = Extensible Markup Language, a universal data-description language.

6 Distributed Component Object Model

The key eteC components commercially available for 3rd party use is the Falcon, which encapsulates physical access to the network, in the" form of a method-level interface to the KNX Device Network protocol, and

The commercially unavailable Eagle component maps the relevant physical database structure of the ETS repository to an abstract model of persistent "KNX Domain Objects".

8.3 KNX Broadband, Intranet, Internet and Integration Services

In KNX a coherent blend of protocols, programming interfaces, models and tools are available, which the KNX manufacturer and integrator / installer alike, may exploit to realize solutions which integrate a KNX Device Network installation in a LAN or WAN environment.

The KNX specifications amongst others describe a compact and flexible IP (Internet Protocol) tunnelling protocol, which (roughly) carries a KNX frame over an IP stretch. Implementations of this protocol exist, in the form of the iETS (Internet ETS) communicators, which allow remote maintenance of KNX installations.

Appropriate connectivity to strategic partner systems or environments or applications can in this way be realised:

- SCADA (Supervision, Control and Automated Data Acquisition), e.g. via OPC (OLE for Process Control);
- remote service gateways, e.g. via OSGi;
- local or remote network environments, say intranet, extranet and internet applications, systematically via IP-based protocols (IP = Internet Protocol);
- specific established standards in building automation, such as BACnet;
- etc.

As it does for the KNX Device Network, ETS extends its role as universal configuration, integration and repository platform for such solutions.

9 Certification

KNX not only provides its members with a powerful and versatile specification, but also with organisation of certification services. Among these, product certification is certainly one of the pillars of the KNX Association. By marketing certified products, the KNX Association members claim for the belonging of these products to a strong and widely shared standard, but also for the verified performance of these products to the Interworking Rules laid down in the specification.

This is a guarantee for all the customers involved in the usage of HBES systems, who are sure to find the right function, in a multi-vendor context, which fits without problem into a global home management system.

Certification means full third party assessment, and granting the use of the KNX logo to show compliance of devices to this specification.
It covers :

- independent testing and assessment of compliance to the KNX specification,
- conformance to the hardware requirements contained in the EN 50090 series,
- quality management of manufacturing to the ISO 9000 series,
- guarantee of full multi-vendor Interworking at runtime and configuration within one mode
- possible smooth integration in a global home management system with use of the software tools, on base of the certified database.

Certification rules are laid down in Volume 5 of the KNX specification, and testing in Volume 8.

Development engineers are strongly encouraged to integrate the KNX Association Certification process into their development management. This early integration enables to get the KNX logo “at no extra cost”, and to take advantage of the powerful certification tool software and procedures for the product validation phase.

