PALAS – Powerline as an Alternative Local AccesS
IST-1999-11379

Deliverable D6: Software architecture requirements

Authors: Kamphuis, René (ECN),
Warmer, Cor (ECN)

Table of contents:

1 INTRODUCTION 4

2 CONTEXT AND INTERFACES 7

2.1 Network topology characterisation 7
2.2 Application data transfer characterisation 7
2.3 Roles of service providers 9
2.4 Standards for device interfacing and inter-device communication 9
2.4.1 Device access technology and installation 9
2.4.1.1 IP: the Internet Protocol 9
2.4.1.2 Jini: device drivers served by devices 9
2.4.1.3 Universal Plug and Play (UPnP) 10
2.4.1.4 LON 11
2.4.2 Local network technology 11
2.4.2.1 LONWorks 11
2.4.2.2 X10/EHS 11
2.4.2.3 IEEE-1394 11
2.4.2.4 HiperLAN-2 11
2.4.2.5 Bluetooth 12
2.4.2.6 HomePNA 12
2.4.2.7 The HomePlug association 12
2.4.2.8 SCP, the simple control protocol 12
2.4.2.9 HAVI 12
2.5 The services gateway 12
2.5.1 Existing devices. 13
2.5.1.1 CoActive connector 13
2.5.1.2 i-LON 13
2.5.1.3 E-box. 14
2.5.1.4 Enikia 14
2.5.1.5 Cisco 14
2.5.1.6 Aladn, emWare 14
2.5.2 The Open Services Gateway initiative (OSGi) 15
2.6 Integration through XML/DOM 17
2.7 Utility service end 17
2.7.1 IEC60870 19
2.7.2 UCA 19
2.8 Summary 19
2.9 References: 20

3 POWERLINE LAST MILE AND HOME NETWORK ARCHITECTURE 23

3.1 Introduction 23
3.2 Devices, architecture and value creation 26
3.3 The role of access nodes 26
3.4 Base Station 27
3.5 Service Provider 27

3.6 Gateway Keeper 27
3.7 References 27

4 SERVICES MODELLING 29

4.1 UML Modelling 29
4.2 Distributed local Control and Monitoring Systems 30
4.2.1 Use Case: Automatic Meter Reading 30
4.2.2 Expansion to Monitoring and Control 34
4.2.3 UML Object Diagram 35
4.3 Point to Point Applications 36
4.3.1 Use Case: Telecommunication 36
4.3.2 UML Object Diagram 39
4.4 Information Exchange Applications 40
4.4.1 Use Case: The Internet Service 40
4.4.2 Expansion 43
4.4.3 UML Object Diagram 43
4.5 Entertainment/Multimedia Applications 44
4.5.1 Use case: Video On Demand 44
4.5.2 UML Object Diagram 47
4.6 Automation / Control Network 48
4.6.1 Use Case: Comfort Management 48
4.7 References 51

5 ARCHITECTURAL IMPLEMENTATION ISSUES 52

5.1 Partitioning of functionality 52
5.2 Required management and configuration facilities 52
5.2.1 Object persistence 52
5.2.2 Replication mechanisms, serialisation and versioning 53
5.2.3 Multi-agent architectures 53
5.2.4 Control timing implementation and synchronisation 54
5.3 Conclusions and solutions 54
5.4 References: 56

6 USED ABBREVIATIONS 57

1 Introduction

In other deliverables of the PALAS-project business models, the current market developments and technological aspects of data transmission through the powerline are dealt with in detail. Eventually, all these aspects are reflected in functional requirements for software systems. In the design phase of the development of these software systems, as a first step, processes and business rules, behaviour and control aspects and data structures are modelled in an abstract, implementation independent way. From this view with three aspects, a fourth view is developed: the architectural view. In the architectural view, hardware and software components and their interfaces are identified. The components and objects identified in this process are mapped onto existing, standardised hard- and software components. The application then consists of a number of processes using these components glued together to the application logic.

In this document, the software architecture is discussed in terms of attributes of components and the interfaces between components. The software architecture is the bridge between the PLC business models and processes, the hardware components and the communication channels between the hardware components. Currently, in architectural terms, the emphasis is shifting from PC-centred computing to appliance-centred computing with software being the middleware gluing applications together. This implicates, that on small-scale networks, PLC has a natural role in empowering and interconnecting devices-

In traditional software development methods, as a first step in building a system, a specification, abstracted from the implementation, only looking at functionality of an application is made. In the PALAS project, a number of technological developments and existing business models form essential boundary conditions and prerequisites for the architectural work. Therefore, in this document a middle-out approach is followed: a top-down approach from the functionality point-of-view is combined with a bottom-up approach from the technical aspects of PALAS. In this way, it is tried to identify differences in PLT-based networks and variants with other transmission media in order to indicate most suitable applications using power line transmission on the last mile and to define a proper role to developments in high and low bandwidth application spectrum.

In the architectural discussion a differentiation in a number of application types for PLT is made. Furthermore, a number of service delivery frameworks are discriminated. From the point-of-view of software technology, designing applications for telemetry and telecontrol pose no insurmountable problems. However, the track record of mass-introduction of successful applications is small. The distributed nature of these applications in terms of security, guaranteed (non) accessibility, safety and (un) interruptability pose software architectural problems. Furthermore, merely designing and implementing one application on a dedicated infrastructure will prove not to be very cost-efficient. There is a large number of stakeholders for in-house and SOHO applications in residential areas. Service bundles of applications with comparable architectures and infrastructures are more likely to be implemented. To deliver properly aggregated service bundles alliances of application providers have to be formed. In this document the factors will be identified, that affect the introduction of service applications and the optimal

aggregation in terms of bandwidth and processor requirements by a service bundle provider.

A larger probability for successful introduction of large scale distributed applications comes from hardware and network innovations and from software standardisation, developments now rapidly progressing. In chapter 2 the focus is on the role of these developments on the software architecture as they form the context of applications in terms of devices, nodes and networks each with their respective interfaces. With respect to the software architecture, software standards elements relevant for PLT-application-types will be discussed briefly. The operating environment, as reflected in standards for distributed system operation, is also discussed in chapter 2. These pertain to JINI on the micro-network level, COM, CORBA, Java-RMI for inter-application interfaces, XML/DOM for interfacing to a distributed database and WWW-wide inter-application communication) and ongoing industry co-ordination activities such as the OSGi (Open Systems Gateway initiative) will also be discussed in the chapter 2. Furthermore, the software architecture requirements definition will be strongly influenced by the information systems context for PLC-based services. This context includes software already in use at the utility side. In this respect one might think of customer information and relation management systems, billing, distribution automation systems, electricity load balancing apparatus and so on. Future, renewable energy generation systems will be of a small scale. Distributed operation for optimal utility DA/DSM then will pose a large problem, if information flows across small-scale networks is not possible. Proper matching of scale and application scope of processes and optimal utilisation of information content is an essential prerequisite. A short description of the latter aspects concludes chapter 2.

On a more detailed level, last-mile service applications, from a software architectural point of view, require setting up a hierarchical architecture with a number of distribution levels where processes intercommunicate. Software systems on the service provider level have to be interfaced to software systems on ever-smaller scale systems, when the hierarchical tree is descended. This is discussed in Chapter 3.

Technically speaking, partitioning and dimensioning of components along the hierarchical tree has implications for object serialisation, persistence and replication. Obviously, no generic architectural framework for all applications can be derived. The way to define the abstract system architecture in this document is by following the industry standard UML (Unified Modelling Language) method, elaborating use cases and associated object models and business rules (object constraints) for a number of archetypal key application-types. These types cover clearly distinguishable categories of applications. This is the subject of chapter 4.

Assignment of tasks to a suitable hierarchical network level plays an important and discriminating role in possible service development schemes. E.g. adequate partitioning of tasks between a low-intelligence meter at-home and a medium voltage contractor serving a large number of households can play a key role in cost-effective implementation. Also manageability and versioning of the software and hardware modules in fine-grained networks are discussed and further features the mapping of these architectural considerations onto bundling of service types.

These subjects are treated in chapter 5. Finally a number of conclusions are stated.

2 Context and Interfaces

In this section a description of software architecture relevant attributes for PLT-applications is given and standardised software component attributes forming the context of the technological development of distributed applications are discussed. The scope is taken one segment broader than purely the last mile connection to gain a better insight in the development of applications, that not only have requirements with respect to the last-mile but also to dwellings and buildings as well as to utility company software systems.

2.1 Network topology characterisation

A number of functional entities in networks are relevant to PLT-applications.

1. The local network. At this moment, small-scale networks are rapidly emerging [1]. The local network enables local data-exchange between intelligent appliances or computers.
2. The local access node. This node is the portal (i.e a transfer-point) for exchange of selected information between the local network and the outside world. Possible local access nodes are a two-way communication, third generation (intelligent) set-top box, a PC+modem or a dedicated residential gateway.
3. The concentrator node. The concentrator node typically is placed in a transformer station in an urban district. The transformer station enables communication to the fourth level of communication between local access nodes.
4. The utility node. This node is the top-level in the network hierarchy in the scope of the discussion.

With respect to the physical data transfer medium a large number of combinations exist between powerline, RF, cable, fibre optic or a copper telephone line to interconnect individual entities.

2.2 Application data transfer characterisation

From the network topology above a number of application types can be discriminated from the point of view of data transfer frequencies and data rates and from control behaviour. In the discussion, a further discriminating point is the extent to which information has to pass all levels in the network topology. Shortly mentioned in the discussion are home automation networks. These sometimes contain devices consisting of networks themselves. These point-to-point communications are concentrated in piconets [2] or micronets, which, in their own, are part of larger networks integrated and accessible as one component. Bluetooth in this respect may be considered as a piconet replacing a USB-cable connection. Piconets together form a functional component/appliance. The distributed applications may be characterised as follows:

1. Distributely managed, local control and monitoring systems (DCMS). In the network topology these have their control processes and data distributed over the network. Required bandwidth for both directions is low. Real-time constraints are moderate. Software reliability, management and maintenance requirements however are heavy. Applications, standards and tools to produce

the applications may be compared to those for large industrial real-time DCSs. Typical required transfer rates are in the order of kB/s. Control network applications place the service delivering companies in the control-loop of critical user processes. In terms of the software architecture this means, that careful consideration has to be given to data and object persistence as function of time, replication mechanisms in case components in the network fail and serialisation mechanisms. This kind of application is most difficult to build, test and manage. Furthermore these applications are very costly, because the expenditure on a per programming statement base for control code is several times larger than for database handling code or output handling code.

2. Point to point applications (P2P). Although the signals may be sent across the powerline in a broadcast manner, one might imagine the result is a one-to-one connection. These traverse the network topology to a certain level in both directions. One might think of an in-building telephone call, a call within an urban district or a wider area normal telephone call. These types of applications are heavily real-time. Bandwidth requirements in both directions are moderate. Application logic is contained in dedicated small DSP-processors and large telephone exchange systems. Transfer rates are in the order of tens of kB/s. At the moment it is expected, that circuit-switched telephony is going to be replaced by packet-switched telephony. Touring then will become a matter of intelligent IP-socket communication and real-time data transfer.
3. Information exchange applications (IP). These applications are based on the IP-protocol used on the Internet. Data packages are interchanged with low to moderate real-time constraints as information content. Which network components are traversed in the above scheme is arbitrary. Low-bandwidth suffices upstream from the local network; moderate bandwidth is required for the opposite direction from the “content” repository. For distributing data-streams in information networks, the required volume is higher, but the reliability requirements are much lower. Performance also is dependent on connections outside the network topology. Functional requirements and manageability of IP-applications dictate a mechanism with an emphasis to server or client-side computing. Examples of the first are active server pages (ASPs) and servlets and of the second, applets. Bandwidth requirements for the first category are lower than for the second. The opposite is true for the processor-capability. Distributed systems based on remote-procedure call mechanisms have known difficulties with firewall-protection when programmed in Java.
4. Entertainment/multimedia applications (EMA). These applications require a large bandwidth to the local access node. Large volumes of information are transferred in parallel down the network hierarchy. Upstream volumes are typically very low. Entertainment networks are most demanding in terms of data transfer; the distribution of data however is a minor point because signals are analogously multiplexed.

Given functional requirements hybrid applications may be imagined. For instance, for a pay-TV application e.g. via PLT, ordering and accounting can be done using a low-speed reliable connection, e.g. via PLT, and signal transmission by high bandwidth optical fibre.

2.3 Roles of service providers

There are a number of actors engaged in the delivery of services [3] and in the operation of software systems. Deployment of services needs service developers, service brokers, partnering with a service aggregator, an internet service provider, a connection provider and/or a device vendor, and subscribers. The platform and local infrastructure provider, who delivers the hardware infrastructure and basic communication software for the local access node defines IP-address mapping from in-home devices to the gateway. A service/application provider delivers the application functionality. In the business software market, it is expected, that the proportion of applications, that are run centralised by dedicated service providers will increase significantly. The communications service provider (service enabler), that operates communications access for an ASP (application service provider). The role of utilities in this respect may vary. Consumer research [[3]] shows that utility companies are the preferred providers for bringing packaged services into the homes.

2.4 Standards for device interfacing and inter-device communication

In order to combine all these kinds of networks, transparent interfaces are to be defined to establish the common data-structures and procedures. Genericity of the interfaces is the key to interoperability of these different apparatus, applications and networks. Devices have to be connected to other nodes in the network as seamless as possible. A large number of standards for these interfaces and for small-scale network traffic have been defined by the industry.

2.4.1 Device access technology and installation

2.4.1.1 IP: the Internet Protocol

Recent miniaturisation efforts from the industry make it possible to have an IP-stack [1] on very limited hardware. In this way the large amount of money invested in Internet communications software is used effectively. Very thin-client applications are promoted in this way, leaving the main responsibility for operation to a connected server. The software to control an IP-node in such a way is connected to attribute-syntax within an HTML context.

2.4.1.2 Jini: device drivers served by devices

Jini is developed to enable spontaneous, dynamic networking for all kinds of devices [[4]]. Jini is a specification launched by Sun Microsystems. Adoption of Jini by device manufacturers relieves the task of configuring and installing devices in heterogeneous networks. Furthermore, the management and maintenance of complex networks is facilitated, if devices are installed according to this specification. Devices, in the Jini-philosophy, are responsible for dispatching their device driver code and device management parameters to applications instead of having device driver code installed and managed as part of network management. Through this mechanism, the task of building, maintaining and changing a network of users, software and devices is simplified. Devices can be plugged into any network and make their services available to any other device as well as are becoming aware of any services other devices on the network can provide.

From an architectural point of view, Jini devices, once they are installed, can be instantiated and managed as objects, to and from which messages can be received. The Jini infrastructure (the *Jini federation*) consists of

- Look-up service: the database for services available in the federation.
- Discovery protocol: allows discovery of any resource plugged into the federation and registers it's services using the look-up service.
- RMI 1.2 (a method or function call across a network): Remote Method Invocation is a part of the Java Virtual Machine and makes it possible to make use of methods located at a remote machine.
- Distributed security system: a security framework in which an ACL (Access Control List) describes which components are allowed to get access to other components.

Jini can be implemented in as little as 48 kB of base infrastructure code for the virtual Java machine and it's class libraries. Jini is tightly coupled to developments of the Java programming language. From JDK 1.2.2 on, it is supported by the Java development environment. Java does not yet meet the high expectations on delivering transparent inter-process communication primitives for distributed communications.

Although the specification is licensed to over 20.000 customers, examples of use of the Jini specification are coming up slowly. One of the early adopters is Echelon, which is the owner of LON-technology. Hard disk suppliers such as Quantum incorporate Jini into its hard disks, so that each disk drive can make itself available to all computers on a local network. Madura will launch an ERP package based on the Jini concept as a service over the Internet rather than being sold to customers. Highest pay-off would yield the application of JINI dynamically driving a printer in a PC-network, but presently no examples exist.

2.4.1.3 Universal Plug and Play (UPnP)

As for Jini, UPnP [11] is a standard, defined to make installing devices as general as possible. This mechanism is more classical, as all data and control processes are part of the device driving application. The standard is a follow-up and micro-networked self-exploring extension of the common device driver concept. A device consists of a number of configuration parameters and of a number of specific functions, which are called by applications in a generic scheme. Management and maintenance of device driving code is major focus point for the software architecture. Mechanisms as defined for JINI as an auto-discovery protocol are implemented in the standard.
Universal Plug and Play (UPnP) is an architecture for pervasive peer-to-peer network connectivity of PCs of all form factors (physical card dimensions), intelligent appliances, and wireless devices. It is a distributed, open networking architecture that leverages TCP/IP and the Web to enable seamless proximity networking in addition to control and data transfer among networked devices in the home, office, and everywhere in between. In UpnP device control protocols are developed now for a large number of devices. These device control protocols are the counterparts of the discovery protocol of Jini.

2.4.1.4 LON

Devices having LON-node hardware have a dedicated, proprietary processor handling I/O on one hand and network traffic on the other. LON-devices can form a dynamically configurable network. All LON-devices operate in an advanced peer-to-peer configuration and communication schedule. In the LONMark association a number of companies are governing standard network variable types and node objects.

2.4.2 Local network technology

Currently a large number of technological developments take place and standards are developed to interconnect devices on a small scale. The developments are spread from very low (kB/s) to very high bandwidth (400 MB/s). Especially the development of SOHO's (small offices, home offices) adds to a large increase in small local networks.

2.4.2.1 LONWorks

LONWorks [5] is a network technology built especially for small-scale networks. In most commercial building management systems today, this technology is common. LON abstracts the logical network from the physical network by a communication bus architecture and processors in such a network, which can be addressed from a central place. Connectivity of apparatus and control logic in a LON-network can be programmed and configured dynamically. Main drive for LON was the vast reduction in cabling necessary for experimental installations. LON technology is independent of the physical medium for data transfer. Frequently in-building power line communication is used. In some residential gateways, LON-technology is used extensively.

2.4.2.2 X10/EHS

X10 is the standard for low bandwidth, home automation communication. As a physical transport medium for signals the powerline is possible. EHS is the result of a number of projects conducted by the European Home Systems Association [[6]]. In architectural terms, X10 and EHS can be seen as bus-structures allowing the individual addressing of network components and appliances. EHS-networks can interface to an EHS-modem central in the network to communicate to peripheral networks.

2.4.2.3 IEEE-1394

For high band-width applications a bus architecture, IEEE 1394/Ilink, has been developed to facilitate high-bandwidth interconnectivity of entertainment and home theatre applications. The 400 Mb/s databus twisted pair and low-connection cost enables high volume data-transfers if broadband has passed the access node. At lower bandwidth the Apple advocated FireWire standard connects devices without wires. The IEEE 802.11 (Ethernet) standard has been extended to bring network technology to the homes and has a wireless version.

2.4.2.4 HiperLAN-2

These standards utilise the 4.0 GHz-band to wirelessly interconnect high-speed data-transfer devices in a computer network setting.

2.4.2.5 Bluetooth

Bluetooth is a recent development lead by Ericsson [2]. The standard uses wireless technology in 2.4 GHz-band to realise 750 kB/s point-to-point connections. In the topology instantaneous piconets are set-up with one master and up to 8 slaves within a reach of 10 meters. Problems for in-home use of Bluetooth enabled devices may come from microwave oven radiation, which may emit in the same frequency band.
From an architectural point of view, piconets present a one-point addressable subsystem in an application. As such, they can be modelled as separate entities in the system. Prime target of Bluetooth is to replace the USB-bus cabling of PC's to peripheral apparatus with wireless communication.

2.4.2.6 HomePNA

The Home phoneline network alliance uses existing copperwire to connect the local access node to the individual devices. The ADSL and XSDL standards, currently available, support high data-transfer rates; up to real-time video.

2.4.2.7 The HomePlug association

In this association research activities are set-up and co-ordinated to make the in-house powerline suitable for high-bandwidth applications. Data transfer speeds obtained are in the order of 2 Mbit/s; just below the limit for digital video.

2.4.2.8 SCP, the simple control protocol

Microsoft advocates this protocol. Simple Control Protocol (SCP) is a lightweight, royalty-free networking technology optimised for devices with very limited memory and processing power and for networks with low bandwidth, such as power-line carrier (PLC) networks. Devices that stand to benefit from SCP include lights, home security devices, home automation devices, and other small appliances that are not capable of supporting TCP/IP networking or that connect to the home network through a low speed powerline carrier medium (PLC).
Microsoft, General Electric and other industry leaders are combining their home control networking technology focus to converge existing home control initiatives into SCP, thus enabling a global standard for lightweight home control networking. Originating from Microsoft SCP is tightly coupled to uPnP. The first implementation of SCP will be embedded in an inexpensive power-line-networking chip next year. Domosys, ITRAN Communications Ltd., and Mitsubishi Electric Corp. are all actively developing SCP-enabled PLC chips which manufacturers can use in the development of smart appliances and home control products.

2.4.2.9 HAVI

For inter-device communication for audio and video devices a software standard describing the interface, message format and the protocol for audio and video devices has been defined by industry leaders in the multimedia branch. The standard is geared towards high-bandwidth transmissions.

2.5 The services gateway

The services gateway is the central point by which part of the local network is controlled and served. The services gateway may be a residential gateway but this does not necessarily have to be so. Essentially, a modem+telephone is a primitive from of a service gateway. The services gateway is attached to a wide services

network to connect service providers to internal clients. RG's terminate all networks and enable multiple home targeted services. RG's are to be discriminated on the complexity dimension of applications and on the number of networks interconnected. RG's may be only enabling one service specific service and one specific network. At the other extreme a whole-house gateway terminates all external networks and enables all services.

2.5.1 Existing devices.

The residential gateway forms the entrance point from the last-mile to the buildings or homes. From a software architectural point of view the residential gateway has a definitive role in synchronising and controlling applications implemented by several service providers. The physical implementation of the residential gateway has a number of forms. For metering applications one might imagine an intelligent meter. For streaming multimedia at the moment a third generation set-topbox with two-way communication possibilities can be imagined. More universal residential gateway concepts are under development now. Devices consist of a central processor, an amount of memory and access channels to several networks. In that sense they mimic a conventional computer. The processors mostly have an embedded real-time operating system. Operating systems range from very dedicated and functionality tailored ROM-able ones up to general purpose Linux with real time extensions. Programming the devices can be done by uploading cross-compiler generated code, a purpose designed command language or through access through an IP-stack.

2.5.1.1 CoActive connector

For control applications, CoActive systems has a gateway on the market in which the system has an Internet-address and software to communicate to the inside and outside world using the IP-protocol. The IP-messages are transmitted through 10BaseT Ethernet or modem. Internally the connection is established through an EIA-709 connection via LONWorks. The IOConnect Architecture in the Coactive connector avoids the server bottleneck at the centre of other architectures. Instead of requiring that all control data be gathered into a single server, and then forwarded across this server's network connection in a fixed way, the IOConnect Architecture allows you to connect multiple control subsystems to the IP network in a distributed fashion according to logical and physical requirements. The subsystem connection is accomplished using a compact embedded connectivity device, rather than a PC. The IOConnect Architecture makes the IP infrastructure available to all electronic devices. There are applications for automated meter reading using Sensel. The latter company uses CoActive systems to do meter reading and offering feedback using the Internet. In a field trial on the Isle of Gotland in Sweden, the Vattenfall spin-off, Sensel, positions itself as an infrastructure provider using the Co-Active systems technology. Several thousands of households are currently equipped with intelligent residential gateways to allow application service providers to install applications.

2.5.1.2 i-LON

LONWorks markets the I-LON technology. In an I-LON box a bridge between a LON- and an IP-network is established. In this way, systems may be remotely controlled from every place having an Internet connection. I-LON devices are managed through a dedicated operating system interface with a number of DOS-like commands with 1 Mbyte flashdisk as a background storage. I-LON devices

are addressable in a HTML-environment using a networkvariable type. With a special tag in the HTML syntax input and output attributes of devices may be set. Using the tag, the mapping of networkvariables to displayable fields in forms is made. Using forms these may be linked to fields in forms, that are transmitted using the get-form attribute. The iLON box has a WEBserver, that mediates the IP-traffic and makes the conversion to LON-network topology. Appropriate mapping schemes exist between logical and physical network components and to build hierarchical data and network structures and security mechanisms. Using NAT (Network Adress Translation) the individual in-home IP-addresses may be translated to one central address for communication with the provider. The mapping scheme adheres to the LONMark conventions for configuring types of devices. In this way the WWW can be safely connected to small local LON or IP peer-to-peer networks. LON-networks are not addressable across firewalls. CGI-like processing takes care of sending the whole content of a form from one place to the other. Network variables may be exchanged on a get and on a by polling base. Using this iLON technology in Italy 27 million homes are equipped with an intelligent meter by ENEL in the coming year. Services implemented include meter-reading and load management applications. In Italy especially exceeding a threshold value leads to more expensive tariffs. In the future more value-added services are foreseen.
More advanced control logic for applications may be obtained by Java applet or servlet programming. The latter will be the method of choice for large centralised control applications. Currently however the iLON-box does not support the Java because the JVM is not supported.

2.5.1.3 E-box.

Ericsson targets its strategy not only on home networks but also on home applications. Essentially, the added value is the extension of the context of their wireless devices to a central interconnectivity box in the home. As a part of this, an E-box was developed as a solution for a residential gateway. From a software architectural point of view, the accessibility and interfaces of this kind of apparatus are most important. The E-box software is based on a multi-processing environment in the Unix operating system. An E-box can be connected to a WEBPad, which allows a portable in-home screen for establishing an Internet connection.

2.5.1.4 Enikia

In the Enikia [8] perspective in-home powerline communication is combined with last-mile access to transparently enable Internet access directly to the device/appliance level. Via pervasive computing an information appliance network based on the Internet a platform to exploit services is set-up.

2.5.1.5 Cisco

Cisco Internet Home Gateway 2000 series is a family of residential gateway devices that enable multi-services delivery to the home over high-speed, always-on broadband connections using DSL-technology. The product family features easy home networking and self-configuration technology. The product family is open to service providers.

2.5.1.6 Aladn, emWare

These gateway-types are typically home automation centred control servers with outside connections and proprietary operating system architectures. For

communication standard low-speed protocols are used and the primary application is in home control systems.

2.5.2 The Open Services Gateway initiative (OSGi)

This initiative has been started by a number of companies including IBM. The Initiative is now gaining broad support from the industry. OSGi is creating open specifications for the delivery of multiple services over wide-area networks to local networks and devices.

In the Services Gateway, OSGi [9] is the platform for communication based service applications. The SG can enable, consolidate and manage voice, data, Internet and multimedia communications to and from the home or office. It also functions as a service enabler for a range of high value services such as energy management, home automation and security, device control and maintenance, etc. The scope of services includes

- Communication Services: point to point communication for customer, inter-PC networks and appliances.
- Energy Services: Automated Meter Reading, load management and comfort management.
- Home Automation Services: more flexibility, bundling with other residential services.
- Security Services: id.
- Remote Home Healthcare Services: special services for elderly and disabled people.

Aim is to integrate parts of the OSGi software specification in products and applications and map these to set-top boxes, cable modems, routers, residential gateways, alarm systems, energy management systems, consumer electronics, PCs, and so on.

As a software specification, OSGi is platform and communication medium independent. It supports multiple local network technologies whether wired or wireless. It also supports multiple device access technologies such as UPnP and Jini. OSGi is strongly supporting Java as an implementation platform. At the moment a completely documented Java object model is available.

From an architectural point of view, OSGi has an embedded server, attached to the wide-area network, that connects external service providers to internal clients. OSGi includes APIs for service life cycle management, internal service dependency management, data management, device access and management, client access, resource management and security. In May 2000 release 1 of the OSGi-specification has been issued. Objects in OSGi are described and implemented in the Java programming language. Thereby OSGi anticipates on expected developments in the Java arena like JavaOS's for appliances, embedded Java, personal Java and Jini. On the other hand, HTML, XML and other Internet-related technologies are supported. The transmission medium for program code and data for services is the Java .jar archive format. In the OSGi model the service provider provides access to services by downloading software on the gateway as a .jar-file, the deployment item in Java. In the OSGi-specification a service aggregator can also be discriminated, that provides a set of

bundled - compatible - services. Furthermore a role is attributed to a gateway operator, that manages and maintains service gateways and its services. In many cases the gateway operator will also be the gateway owner, retailer, installer and/or hardware maintainer. A provider comparable to an “Internet Service Provider” provides the necessary communications over a wide-area network.

Entities in an OSGi scope are:

- Service management. OSGi defines API’s for creating and running services in a multiprocessor, heterogeneous network. A multiple provider environment is also defined.
- Device-access management.
- Logging of events.

Follow-on activities within OSGi include HTTP Services, Client Access Device interaction schemes, configuration data utilities and persistent data services linking to large database systems and legacy software through transparent interfaces.
The service framework provides a convenient context for developers to design applications. Services may be contained in bundles. OSGi provides primitives for live-insertion of services into aggregates called service bundles, life-cycle support of services and containment of devices in device bundles. In the following architectural discussion OSGi is one of the key mapping mechanisms

With respect to the device access specification, OSGi is a software layer complementary to Jini, in the sense that it can use the Jini infrastructure to define the device interfaces. OSGi has primitives for automatic detection of attached and detached devices. A mechanism has been designed for device driver search in a network. Devices can be configured dynamically and may be plugged in an executing set of applications. Furthermore, the object interaction vehicle is the Java language. The packaging concepts and expressiveness of this language are also used to define the detailed specification [[9]]. Native programming code, code not executable by a virtual machine, is encapsulated into JNI, the Java Native Interface. The way native programming code, different from Java-code, is integrated and deployed in the different components of the environment is defined in the standard.

In abstraction level, the OSGi specification resembles the CORBA and DCOM-standards for object interaction in distributed networks. A casting to the operational environment with distributed intelligence and data in heterogeneous size and processing power processors and different speed networks is made.
The key entities in the Framework are:

- Services - Classes that perform certain functionality written the interface and the implementation separated. Service providers have to write software to implement the interface functions. Essential for the design and implementation of the service functions are their live-update, reliability and stability attributes. Services have a well-defined level of operation as reflected in various states of service-processes.
- Bundles - The functional and deployment unit for shipping services. A bundle consists of a number of programs deployed as a Java-archive, .jar, file. Bundles may interact with each other through marshalling via a registration and publishing mechanism similar to common operating systems environments.

- Bundle contexts - The execution environments of the bundles in the Framework. The bundle context governs the execution of bundles and services in a heterogeneous system environment.

OSGi gives a number of concepts to discuss service gateway applications in a common way, be it from the hardware or from the software point of view. The further developments within the OSGi consortium are of utmost importance to companies developing hardware and software for applications. In the further discussion connection is sought.

2.6 Integration through XML/DOM

Several software vendors are strongly advocating XML at the moment to tackle problems with distributed data management. In the Microsoft .NET-architecture [12] the standard plays an important role for application development. All various components are integrated to the XML meta-language. XML features data management aspects for very large-scale applications, XML has self-describing mechanisms for device access. Inter-application mapping using XML is facilitated through the Document Object Model standard. In this standard, distributed objects are made understandable in a transparent way across a network.
SOAP (Simple Object Access Protocol) is a lightweight communication protocol designed to let COM or CORBA objects communicate. SOAP is a lightweight protocol for exchange of information in a decentralised, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data-types, and a convention for representing remote procedure calls and responses. SOAP can potentially be used in combination with a variety of other protocols (COM, CORBA). With the SOAP protocol methods can be invoked through the Internet. SOAP codifies the existing practice of using XML and HTTP as a method invocation mechanism. The SOAP specification mandates a small number of HTTP headers that facilitate firewall/proxy filtering. The SOAP specification also mandates an XML vocabulary that is used for representing method parameters and signature, return values, and exceptions. SOAP has been submitted to the W3C for the formation of a working group.

2.7 Utility service end

Utilities are trying to discriminate themselves by getting into the E-commerce business. For instance, in the USA, Kansas City Power and Light [13] gives customers the possibility to access customised information. ENRON, together with IBM and America-On-Line recently established the New Power Company [14] to deliver services partly as described above to residential and small customers.

For accounting, billing and customer relationship management, systems operate batch-wise using standard third or fourth generation business software. These systems are loosely coupled to systems governing the primary process, delivery of electricity, gas or water, which operate in a real-time environment. Small local intelligent systems near electricity producers or consumers operating in concert with above two types increase the complexity of interaction.

With energy saving and renewable generation in mind the near-future trend is from large, centralised power generation units with a easy production forecast to small,

distributed generation units, the production of which is more difficult to predict. An "energy-farm" may be thought of as to consist of a cluster of mini- and micro-CHP (Combined Heat Power) units, solar energy and wind-turbines. These "energy-farms" do not utilise a large distribution infrastructure but can pose net-balancing problems on several time-scales (from milliseconds to hours). Management of these small individual energy generating and optimally using components requires the design of advanced distributed information systems and new algorithms for distribution automation.

Components in such next generation systems perform:

- Data-acquisition van essential parameters with several sampling intervals.
- Storage of operational parameters in a distributed database. In the database the history, the long and short-term use of data, is an important attribute.

With the advent of small scale, distributed computing, the following new service applications are possible for utility systems:

- (A)DSM. Advanced, fine-grained, demand side management specific to energy farms.
- AMR. Automated meter reading and trending on a small scale. Aggregation of production figures and determination of forecasts to optimise performance. Small sizes of production units require solid procedures for accountable registration.
- Tampering and outage detection. Algorithms can be implemented on small processor hardware to detect temporal anomalies in consumption patterns. This may give valuable information about the misuse or defects of the distribution network.
- Monitoring of the quality of power (harmonics, dips, spikes, probability of failure) delivered. Depending on the use of power, a customer may require different qualities.
- Predictive maintenance. Viewing the behaviour of selected, sampled parameters on different time-scales may allow the detection of upcoming defects in equipment before these defects cause damage.
- (A)DA. Advanced Distribution Automation. Systems now are under human supervised real time control via SCADA-software. The future network topology will change yielding several clusters of individually optimised production islands. Given a set of distributed computers in a suitable network topology, distribution resource planning with real-time pricing constraints can become a reality.

The topology of the distribution and demand side management gives utilities a competitive advantage to set up value added services. For exerting these services a connection to local home/building automation networks is necessary. Utilities in this way may be in the position to build up a more extensive customer relation as energy knowledge experts. Derived information of metering and seasonal trends may lead to advise for improving efficiency. Especially if the readings of electricity, gas and heat delivered are analysed, efficient partitioning between these carriers may improve overall efficiency. Preventive monitoring techniques used in the distribution process may also be delivered to customers. For larger customers, in real-time tariff situations, an increase in information is an essential prerequisite for contract management, strategy implementation and cost-effective prediction of future trends in generation and consumption.

For large customers in a liberalised market, next day tariffs are to be given on a quarter of an hour basis from the demand and the supply side. This stresses the need for increasing metering and control intelligence.

For tele-metering and tele-control a number of standards are or are being defined by the IEC. Some notable examples are discussed below.

2.7.1 IEC60870

The IEC 60870 standard describes a protocol for remote metering and control applications. It incorporates an object model and the interfaces between communicating parties on both sides of the powerline. The communication mechanism defined in the standard, however, has a poor performance. Especially setting up a large number of connections takes a lot of time.

2.7.2 UCA

The Utility Communication Architecture (UCA) is an ensemble of open protocols and standards that helps eliminate the extra costs, redundancy and inconvenience associated with using system-specific, or proprietary, communications interfaces each time new equipment is connected to automation systems. UCA makes use of a selection of international data communication standards for the complete range of electric utility needs. It facilitates distribution automation and has interfaces to Supervisory Control and Data Acquisition (SCADA). In the standard an attempt is made to seamlessly connect all of their key facilities, from the control centre to the customer's meter, along the electronic communications highway. UCA is based upon open standards. UCA offers interconnectivity between equipment from different manufacturers and interoperability between the databases used to exchange high-speed, real-time data in utility operations. In other words, it allows utilities to "plug and play" equipment from different vendors over the same data network.

2.8 Summary

In essence the following scheme illustrates the standardisation efforts of the application types, that are further discussed in this chapter:

Type	Local network	Local access node	Concentrator/ router node	Utility node
DCMS	LON, X10, EHS, HomePNA, HomePlug, SCP, IEC60870	OSGi, DCOM, CORBA, RMI		IEC80870, XML/DOM, UCA
PP	DECT, Blue Tooth	Home PNA		
Internet	IP, HTTP	IP	IP	IP, XML/DOM, SOAP
EMA	HAVI, IEEE1394, HiperLAN2, IEEE802.11	OSGi		

Table 1 Hardware and software standards

Essentially, at the moment the number of technical hardware and software standards is sufficient to build powerful applications. With the OSGi-standard a first conceptual standard is becoming accessible. There have to be more of these conceptual standards and implementations of existing standards like COM and CORBA to facilitate large scale layered application development. Furthermore, standardisation further on the hierarchy, especially on the concentrator node is essentially lacking. This concentrator node would have a discrete advantage for PLT-applications in allowing RG's to be relatively cheap, "thin" clients served by a "thick" concentrator node server. This not being the case, the concentrator node only has a transparent routing and data transfer function with emphasis in application logic on the utility node and RG processing power.

2.9 References:

[1] D. Estrin, R. Govindan, J Heidemann, "Embedding the Internet", Communications of the ACM, May 2000.

[2] Bluetooth specification can be found on www.bluetooth.com.

[3] Proceedings the second Home Networks European congress. 23-24 May London, 2000. Especially the articles of Parks Associates contained therein.

[4] www.jini.org. The central place for obtaining the specification.

[5] LON is extensively described at www.echelon.com.

[6] EHS can be reached at www.ehs.org. EHS-devices are the results of a number of EU-projects conducted in the ESPRIT-framework.

[7] Sensel is a firm installing residential gateway technology from Co-Active systems. The role of Sensel is providing the infrastructure. References are www.coactive.com and www.sensel.com.

[8] Enikia is a firm recently established for operating services through gateways. On www.enikia.com detailed information can be found.

[9] The Open Service Gateway initiative is described on www.osgi.org.

[10] SOAP (http://www.w3.org/TR/SOAP) gives a description of the proposed standard
[11] UPNP2000. www.unpn.org The main source of information Universal plug-and-play forum.
[12] Various authors. Microsoft Developer Network Journal. September 2000. Microsoft Press.
[13] This service can be found at http://www.kcpl.com.
[14] The company presents itself at http:/www.newpower.com.

3 Powerline Last Mile and Home Network architecture

3.1 Introduction

As we saw earlier, the powerline service infrastructure consists of at least four levels,: the devices at the end-nodes, a local access node through which the home is coupled to the outside world, a concentrator or base station, which connects the powerline to the communication backbone, and the utility node, which delivers the services from the utility or service provider.

A number of requirements can be summed up for smart devices in a last-mile<> home network. A device should have a unique id to allow addressing in large hierarchical networks. A physical and logical mapping mechanism should exist to enable proper context definition. Furthermore a device should be reasonably priced and standards used in the device hard- and software architecture should not be proprietary and have the support of many vendors; it should be self-configuring, live-insertable in the network and have self-diagnostic features and have non-volatile memory. In this way, smart devices can be seen as appliances performing services on a small scale: a smart device has responsibility for a special task. Other components in the network can call upon the device to perform this task.

Traditionally, the local access node to networks is dedicated to one application. At this moment, in-home networking and automation is emerging at the market, in which the access node is implemented using a residential gateway [1]. This gateway provides local intelligence, is able to connect different Internet access networks to multiple types of in-home networks and can deliver different e-services over one single connection. Although this gateway can serve as an 'all-purpose' access node we will have to take into account that a home will have more than one local access node.

The base station, also called in literature the concentrator, for PLT-applications is normally located at the transformer substation. A main task for the base station will be to act as a multiplexer for requests from the top end of the network and as an intermediate for information transfer from the lower part of the network. The way the multiplexer station is embedded in the power distribution network, determines the functionality of powerline value-added services. In the USA, a very limited number of households are connected to one base station. In Europe, the situation differs per country, but the number of connections generally is very much larger. This poses a definite advantage for local applications with residential area scope.

From the software architecture point of view, the OSGi separates utility node in two roles, the service provider role and the gateway keeper role. The first party is responsible for the content of the services, the second party is responsible for the way these services are set up and rolled out in the network.

We therefore enhance the four levels mentioned above to five levels, as outlined in figure 3.2, and which will be elaborated further in the following paragraphs:

- devices,

- local access node,
- base station,
- service provider,
- gateway keeper.

These infrastructure levels must communicate using standardised interfacing such as described in chapter 2. Powerline services then can be deployed on the above infrastructure in such a way that decisions can be taken at a logical level, thus relieving the rest of the network. The main responsibility of a software architecture is to deploy tasks at the right level and to assure that information needed to fulfil these tasks is communicated to this level. The infrastructure structure requirement, as stated above, complies with the OSGi model. In some applications two or more levels will coincide with one network node.

In the context, mentioned above, the following elements have to be considered for our model:

1. The user or customer. Application types have been discussed in Chapter 2. It should become clear at which level the user can operate on the network. The user interfaces should be simple yet adequate. Both browsers and touch screens can deliver this functionality.
2. The legacy systems at the level of the service provider and/or the gateway keeper. For obvious reasons the powerline service systems should be linked with BackOffice systems in areas as finance or accounting, customer relationship management, marketing and sales, etc. These systems can be loosely coupled.

The user should not be aware of the underlying technical details and network infrastructure. All objects and components should be real world and be transparent.

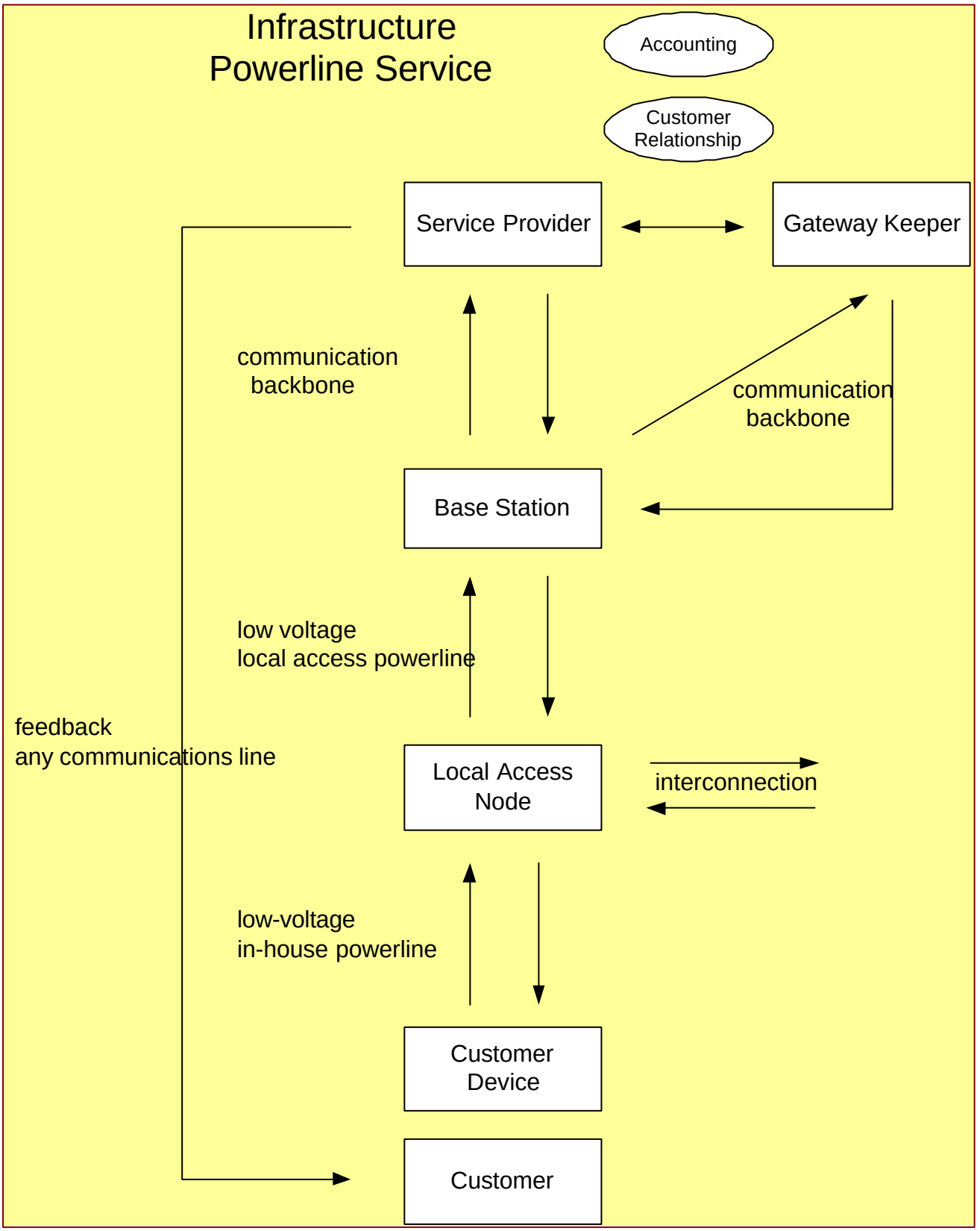

Figure 3.1 Infrastructure Powerline Services

3.2 Devices, architecture and value creation

Devices are the basic elements in the information structure, as they either provide information to be handled in the service or they receive information, which will be handled internally by the device.

Enikia [2] differentiates three types of devices:

1. An **appliance** creates value primarily through its physical processing features. Therefore, the embedded intelligence features will likely focus on device control and feedback data relating to the device's mechanical functions such as power consumption, maintenance status, or performance data. Examples are a commodity meter, which provides usage statistics, and a washing machine, which washes clothes.
2. The fundamental value of **computers** is using embedded intelligence to process digital content.
3. In between lie **electronic devices** that rely on both embedded intelligence and mechanical functions to add value (for example, CD players or printers).

Appliances, electronics, and computers will all have embedded intelligence and communications abilities. These can be seen as the service(s) of a device, which are available for other nodes in the network. Note that the term 'service' in this context is not quite the same as the 'service' in PALAS terms. On request a device can deliver information, or it can perform a task.

3.3 The role of access nodes

The access node is a first level of aggregation, which acts as a buffer between in-house information and applications and the outside world. This access node might provides local intelligence, might be able to connect different Internet access networks to multiple types of in-home networks and might deliver different e-services over one single connection. Note that a home may be supplied with more than one access node, each used for different applications. One can regard even a modem as an access node. Its only intelligence is to transfer data from a computer to the telecommunication network and vice versa. An intelligent access node, such as the residential gateway, can be used as a layer where control is based at a low point in the network. If we can deploy decisions to be taken at this level, we might prevent extensive data traffic to and from the base station and the service provider. Also the service provider will not be extensibly busy for individual customers, but can concentrate on integrating tasks.

An important discriminating point for applications utilising the intelligence contained in access nodes is the extent to which control actions may be performed autonomously. As an example consider an entrance security application. Many home automation applications can be working autonomously at the gateway. This also holds for energy management in the house based on e.g. fixed pricing and / or availability of local energy supply and energy buffers.

3.4 Base Station

Powerline is especially equipped to serve as a medium for communication in-house and at the last mile. The concentrator serves at the buffer between the powerline last mile services and the (telecommunications) backbone, thus opening up the Internet-based services for the in-house applications. Communication between the concentrator and the gateway will be provided using the powerline. Communication between the concentrator and the service provider may use any communications infrastructure, such as ADSL, cable, wireless, etc.

Example: direct telecommunications applications can use the concentrator level to set up direct connection to another party without involving a central host of the service provider. Connection information can be gathered real time either at the gateway level or at the concentrator level.

3.5 Service Provider

The service provider is responsible for the content of the service. Each provider will also be interested it in the usage of its services. Therefore this information has to be collected at the service host and stored for later handling. Typically, processes at this level are accounting and billing and customer relationship management.

3.6 Gateway Keeper

The gateway keeper is primarily concerned with operation and maintenance of the hardware infrastructure and basic communication software. For powerline applications it is assumed that the utility serves as the gateway keeper. Note that the gateway keeper can also act as a service provider. However, conceptually we keep these two tasks separated.
All operational, management and maintenance issues around services and gateway configuration will be dealt with by the utility.

3.7 References

[1] Coactive Networks - *The business case for residential gateway deployments - delivering a new world of Internet services*. Coactive Networks Brochure, 2000 - http://www.coactive.com/media/busmodbro.pdf.

[2] Enikia - *The Information Economy Derivative Markets Model: A Technology Value Chain for the Digital Economy* - http://www.enikia.com/download/iedmm.pdf.

4 Services Modelling

4.1 UML Modelling

In chapter 2 we have differentiated between the following application types for powerline services, each with its own characteristics and standards:

- DCMS: Distributed local Control and Monitoring Systems
- P2P: Point to Point applications
- IP: Information exchange applications
- EMA: Entertainment and Multimedia Applications

In this chapter we will describe some typical service examples for these application types and elaborate these into object diagrams according to the Unified Modelling Language (UML) standard. UML is an industry standard for developing object models and is described thoroughly in Rumbaugh [1], Jacobson [2] and Booch [3].

UML starts with describing a system by so called use cases, in which, in clear language, examples of operational parts of the system are worked out.
The next task is to identify the object classes in the model from the description of these use cases. A class can be described by its attributes and operations. Attributes are properties of each class object, e.g. the name of a person. Operations are the responsibilities of the class objects. It will be clear that the meter is responsible for supplying some kind of usage figures.

Typical classes, which will be identified, are:

- Information carriers, such as: Meter, Electric Appliance, ...
- Information handlers, such as: Registry, Billing or Accounting, ...
- User interface devices: Screen, Audio, Telephone, ...
- Involved Parties: Customer, Utility, Service Provider, ...
- Topology: relations between involved parties.

In the UML object model the basic data and functions of a system are determined. In a detailed UML design also the complete behavioural model is determined. In this document we will only discuss this aspect for a number of typical applications as far as it reveals the main characteristics of these applications.

Since PLC services will be working in a distributed environment special attention has to be paid to the deployment of the responsibilities (operations) on the network topology. Deployment can be described as the configuration of all nodes in a system and the distribution of all components, objects and operations on these nodes. In chapter 4 we have already indicated a likely network configuration for powerline services. From here we can deploy the object model on this network.
References for deployment are Orfali [4] and CORBA (see Ben-Natan [5]). The latter is a standard for object oriented component system interfacing and is closely related to the standards described in chapter 2.

4.2 Distributed local Control and Monitoring Systems

In the DCMS network the control processes and data are distributed over the network. The system is interested in a service delivered by any of these nodes in the network. Examples of this type of application are automatic meter reading and monitoring of appliances. In this paragraph we will elaborate on the first example.
The communication network uses infrastructure such as BlueTooth, USB, PLC on a narrow bandwidth up to 100 kBps.

4.2.1 Use Case: Automatic Meter Reading

Automatic Meter Reading (AMR) can be described as the remote collection of consumption data from customers' utility meters. AMR provides electric, gas and water utilities with the opportunity to streamline metering, billing, and collection activities and to enhance service to customers and gain a competitive advantage (AMRA definition). AMR also provides detection of tampering and/or energy theft.

Step 1: Description of the use case

The metering service can be described as follows:

Each customer has one or more meters which provide information on the usage of any commodity (electricity, gas, and water). These usage data have to be *registered* such that they reflect the price fluctuations over time of the commodity. Pricing can be based on fixed prices for fixed time intervals (e.g. day vs. night tariff) or it can be market dependent, in which case prices can fluctuate every hour or so. The information needed in automatic meter reading is the amount of usage in every price period.
At certain intervals the registered usage data will be *read out* by the relevant utility company. The read information is *stored* for later reference.
At certain periods in time the utility company *sends out* a bill to the customer, who *pays* either electronically or by normal bank transfer. The bill is based on the commodity usage and the prevailing prices at the time of use.
The customer can *review* his usage pattern by visualisation of the usage history. Also the cost of usage can be *visualised* on the basis of prevailing prices.
The utility can *analyse* the data on tampering and/or energy theft. How this is done is not subject of study in this case.
Another part of the AMR service is maintenance, including adding and deleting new meters, etc. Standards such as Jini, UPnP and OSGi can be used to automate this task.

Step 2: Identifying objects and attributes

From the description we can derive the following objects in AMR:

–Meter	A meter measures usage, e.g. of a commodity. A meter usually is a kind of counting device.
–Utility	The commodity provider.
–Customer	Has an agreement with the utility on delivery of one or more commodities.

–Commodity	Electricity, gas and water are mentioned as commodities, which can be metered. A commodity is characterised by a certain amount of usage over a time period.
–Tariff	The cost per unit associated with the commodity; tariff is time-dependent.
–Clock	Recognition of intervals and periods requires data to be read on a regular time-base. A clock will provide us with timing.
–Usage data	Usage data consist of periodic readouts of the meter counter. The readout period depends on the tariff changes of the commodity. Each change in tariff determines a new readout period.
–Accounting System	Metering information is used for billing. Therefore we need an accounting system to handle this billing. In the accounting system the billing frequency and banking information is included. The usage data are the input in the accounting system. Output is an account view to be sent to the customer.
–Information System	The user should be able to view the metering information. Therefore we need an information system to handle this user feedback. The usage data are the input in the information system system. Output is a usage view, to be shown to the customer.

NB. We will not consider the preparation of the bill as a part of the service system. This will be handled by the accounting system. However, on-line viewing of the account can provide added value. We can enhance this to a special service for "e-Billing".

Step 3: Identifying operations

The following operations can be identified in the above description of AMR:

–register usage	Commodity usage over time has to be registered by the meter. The residential gateway can be used as a buffer to store these data for a longer period.
–read out meters	Either the provider periodically reads meter usage data from its customers' gateways or the gateways send usage data to the provider host. The frequency depends on the buffer capacity of the gateway. In order to reduce communication traffic the concentrator may act as an intermediate by assembling the data from its gateways and sending these to the provider host.
–store usage	The collected usage data, including the time of usage or the valued price, have to be stored into the accounting system of the service provider.

–visualise usage	A customer should be able to view his or her own usage statistics, including prices. Standard Internet facilities can be used.
–get price	The price of a commodity within a given period has to be retrieved.
–prepare bill	On the basis of usage and prices a bill has to be prepared for a user, either on request by the user or periodically by the service provider.
–send out bill	The prepared bill has to be sent out to the customer. This can be done electronically or traditionally by postal service.
–pay bill	The user should pay the received bill.
–detect failure	The residential gateway can detect any meter deviation and notify either the service provider or the gateway keeper.
–analyse usage	The usage patterns may be analysed in order to detect any misbehaviour or to inform and advise the customer or to compare usage with standard patterns, etc.

When we think about the remote metering service operational and maintenance tasks must not be forgotten. We must be able to add or remove customers, meters and appliances, detect meter failure, handle operational errors, synchronise the service system, etc. Also we must be able to set commodity price levels or cost rises.

Step 4: deployment of tasks

A typical way of deployment of the metering service is to provide the homes with a local access node, e.g. residential gateway. Information is then gathered by the utility by addressing this access node for each customer through last mile access, using the base station as an intermediate. The access node takes care of the in-home information using the home network.

The access node and the base station are used to buffer usage data. For visualisation of these usage data information will be gathered not only from the service provider (where the history data are archived), but also from the access node or the base station, since the most recent usage data will not always be known at the service provider.

In figure 4.1 the AMR schedule and deployment of tasks is outlined.

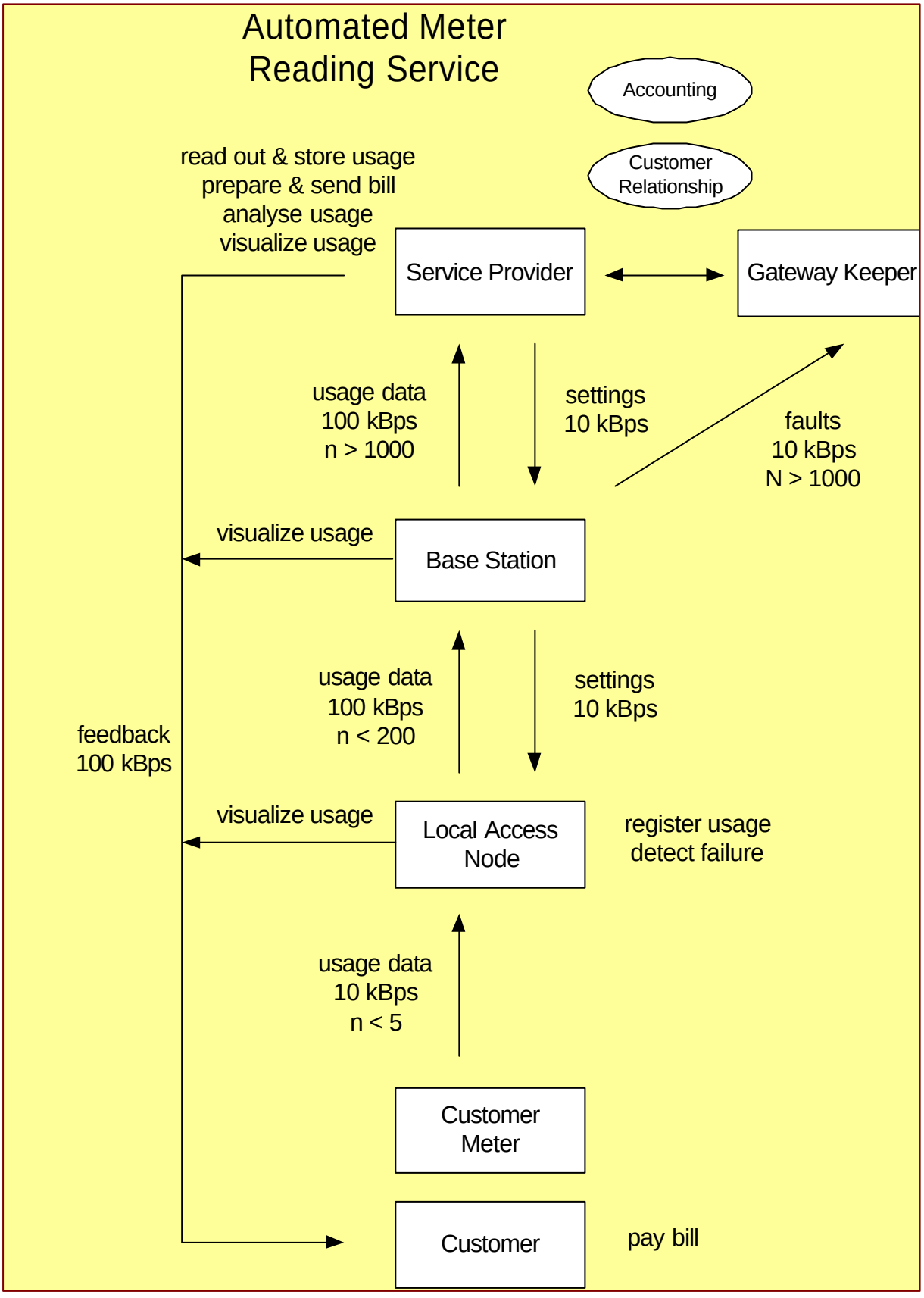

Figure 4.1. Powerline Automatic Meter Reading Schedule

4.2.2 Expansion to Monitoring and Control

Meter reading as described above is concerned with meters and usage figures, needed for billing. We can also see meter reading as a special case of monitoring and control, in which a device can exchange any kind of information with the outside world and its state influenced by the system.
We then arrive at the following object generalisation:

- Device — A device delivers information the user or utility is interested in. A device has embedded intelligence. Communication ability enables the device to share information with the outside world. This outside world can also influence / control the state / operation of the device.

Devices are not strictly reserved to in-home usage, but might also be placed at outer sources (e.g. current regional climate data, Internet based market data). Contrary to in-home devices, which are usually owned by a customer, these devices share information among a group of customers. Therefore we introduce the SharedDevice class:

- SharedDevice — A shared device is defined as a device, which is shared by more customers.

Devices, by their embedded intelligence, can be seen as services in the sense of Jini / OSGi. A meter is a service, which delivers usage data on request; a device also delivers e.g. state or value-based information and will listen to commands to control it. Together these services can form a service bundle for monitoring and control.

In paragraph 5.6 an example has been worked out of a comfort management system. This example has been taken from the TRAFO project [6].

4.2.3 UML Object Diagram

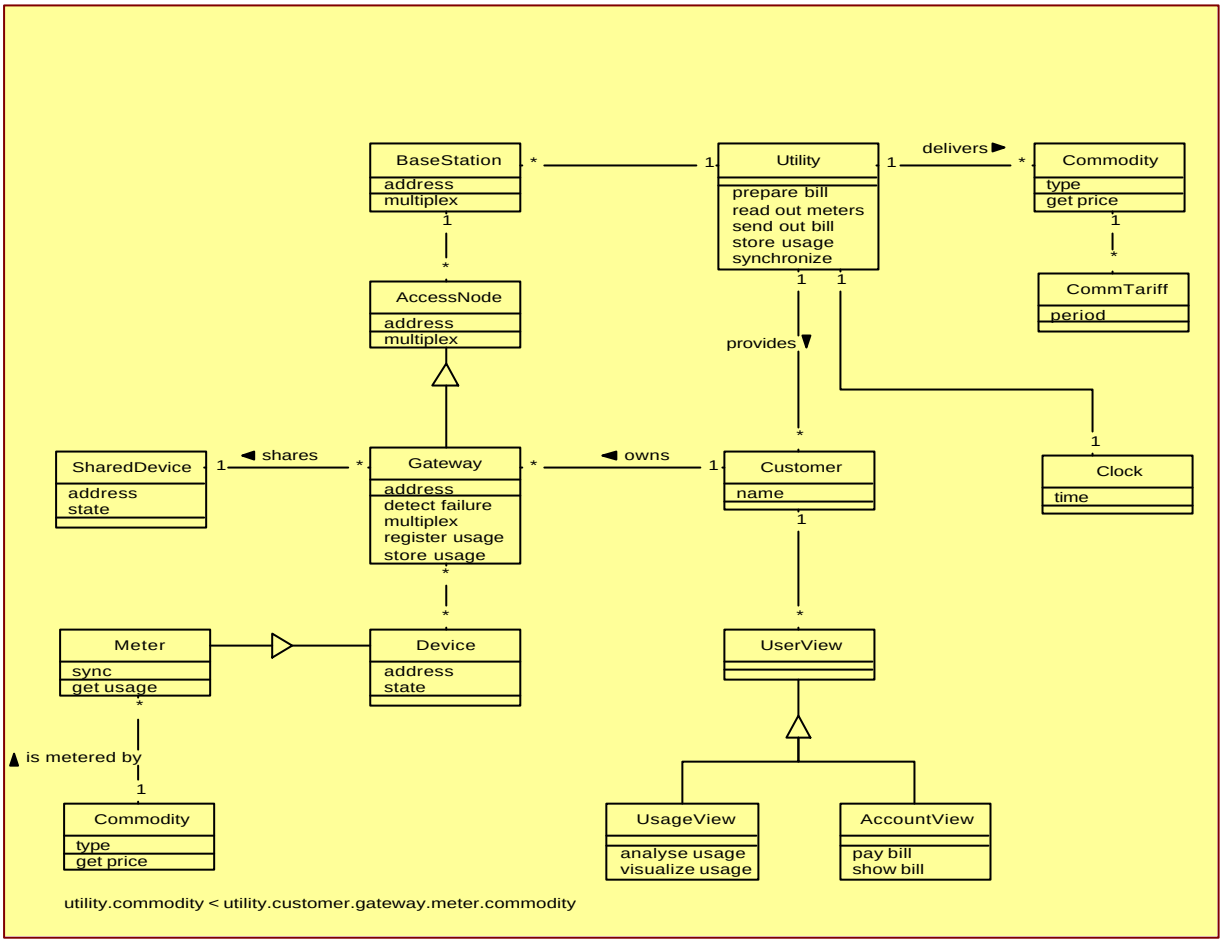

Figure 4.2. UML Object Diagram Automatic Meter Reading

Although Automatic Meter Reading seemingly is a simple application, a lot of stress is laid upon the communication channels and the network nodes. System failure at any level should not lead to loss of data. Therefore metering information objects have to be persistent by data replication at all levels in the infrastructure. This complicates software maintenance.
Since timing is essential in AMR (especially since commodity pricing is time dependent) synchronisation also is essential on the network.
In a control network less emphasis is laid on communication. Local intelligence is important, and therefore more processor capacity and larger memory requirements are required. Since control networks often need a lot of information, the control granularity of the (local) network becomes finer.

4.3 Point to Point Applications

4.3.1 Use Case: Telecommunication

Step 1: Description of the use case

The telecommunication service can be described as follows:

A customer uses a device (normally a telephone, but other devices are possible) to *set up* a connection with an other communication device. Identification of this device traditionally is done by number, but can be any other unique addressing on the network (IP addresses). The required connection will be set up on the network such that both sides can *communicate*. In some cases further *identification* (e.g. username / password) will be necessary to set up a connection or to start communication.
Telephone connections communicate by voice data. The connection usage is continuous during the time of communication. Payment is done on the basis of the length of a connection and the location of both sides.
At some time the connection will be *closed* on request by either side.
The telecom company *registers* the connection and the period that the connection stays open.
At certain periods in time the Telecom Company *sends out* a bill to the customer, who *pays* either electronically or by normal bank transfer. The bill is based on the number of connections the customer has set up, the length of time of these connections and the location of both connection sides. This information is also *visualised* to the customer.

Step 2: Identifying objects and attributes

From the description we can derive the following objects in telecommunication:

–Telecom company	The communication provider.
–Customer	Has an agreement with the telecom company on delivery of the communication service. Identification information are attributes of the customer.
–Connection	A connection is established between two customers. The connection period is characterised by the starting time and the length of the connection. Also the location of both connection sides is important and the communication medium.
–Telephone	A telephone device delivers the means to communicate.
–Voice data	The voice data do not play a role in the telecommunication service. The only functionality is that they are transferred during a connection.
–Price	Price can be a complex issue, depending on the type of connection, the starting time and the location of each connection side.

–Accounting System	Connection information is used for billing. Therefore we need an accounting system to handle this billing. In the accounting system the billing frequency and banking information is included. The connection data are the input in the accounting system. Output is an account view to be sent to the customer.
–Information System	The user should be able to view the connection information. Therefore we need an information system to handle this user feedback.

Several of these objects can be seen as services in the sense of Jini / OSGi. A telephone device is a service, which transfers information to another telephone device. The telecom company service accepts a connection and monitors it. Together these services can form a service bundle for telecommunication.

NB. We will not consider the preparation of the bill as a part of the service system. This will be handled by the accounting system. However, on-line viewing of the account can provide added value. We can enhance this to a special service for "e-Billing".

Step 3: Identifying operations

The following operations can be identified in the above description of telecommunication:

–set up connection	
–identification	The customer must identify him/herself.
–communicate	Data has to be transferred from one side to another and vice versa.
–close connection	On request the connection should be closed.
–register connection	The connection data have to be stored into the accounting system.
–visualise	A customer should be able to view his or her own connection statistics, including prices. Standard Internet facilities can be used.
–send out bill	On the basis of connection prices a bill has to be prepared for a user.
–pay bill	The user should pay the received bill.

When we think about the communication service, operational and maintenance tasks must not be forgotten. We must be able to add or remove customers, handle operational errors, etc. Also we must be able to set connection price levels or cost rises.

Step 4: Deployment of tasks

Traditionally, a telephone connection is set up by establishing a direct line. We could use the residential gateway mentioned earlier to take over this role.

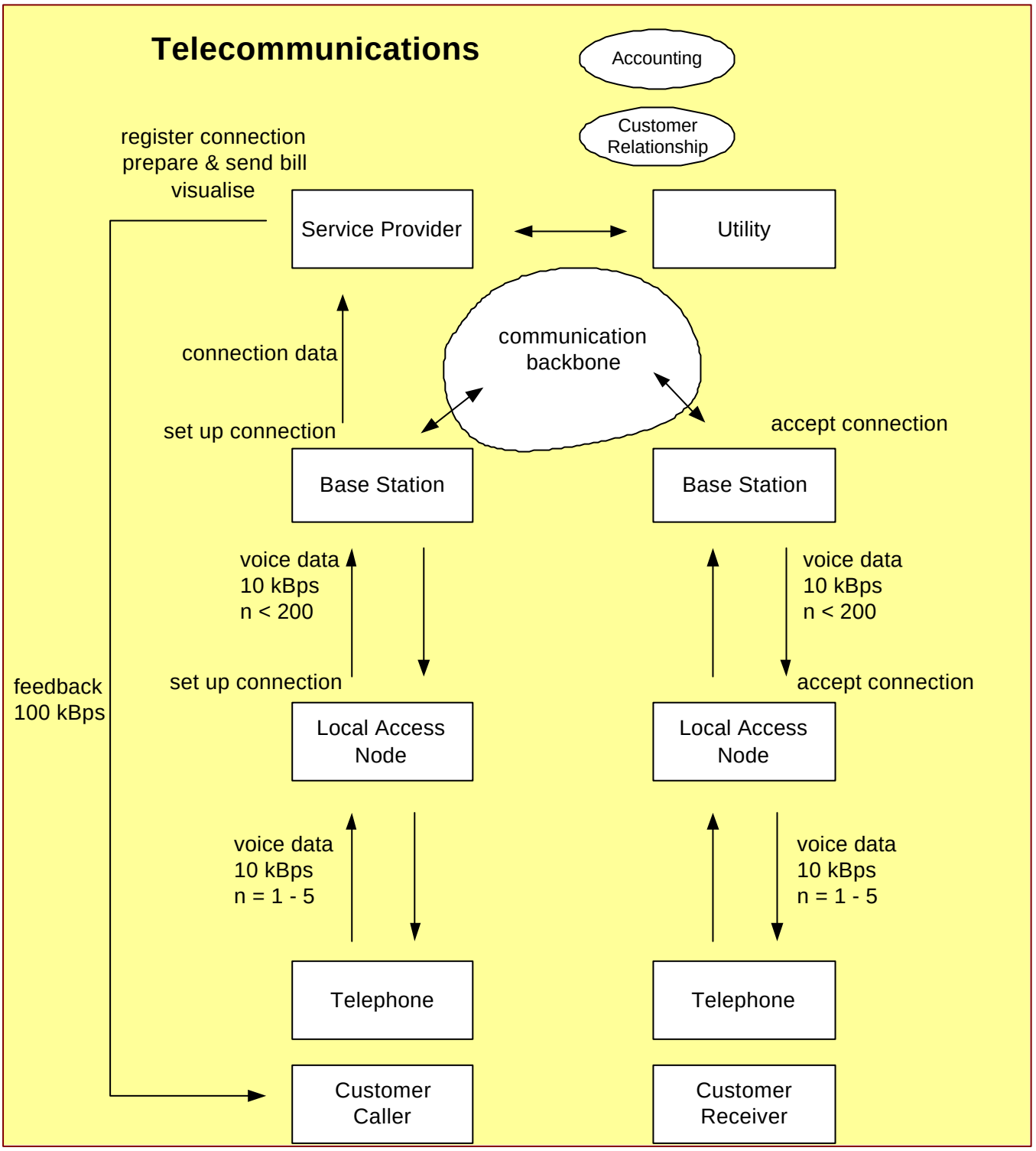

Figure 4.3. Powerline Telecommunications Schedule

4.3.2 UML Object Diagram

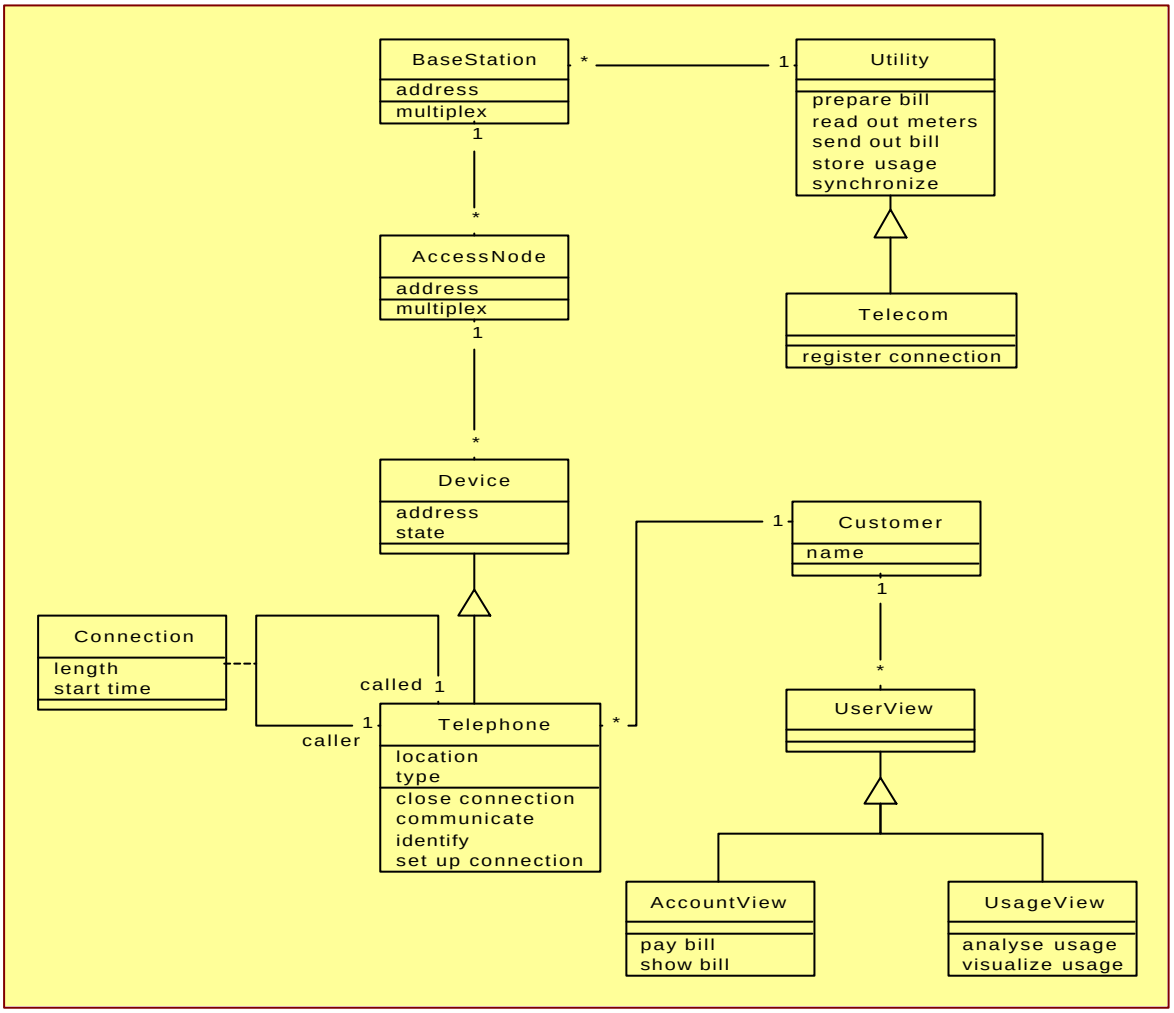

Figure 4.4. UML Object Diagram Telecommunication

In essence telecommunication has a much simpler control structure than DCMS applications. The service is less sensitive to failure, since a connection is only characterised by existence or non-existence. Registration of a connection only takes place at the central (provider) level. Therefore also timing is not essential in telecommunication applications.

4.4 Information Exchange Applications

4.4.1 Use Case: The Internet Service

Step 1: Description of the use case

The Internet service can be described as follows:

> The customer *requests connection* to a provider, using its address and an identification procedure (username, password). After a *verification* procedure a connection or session is opened.
> A customer uses a device (normally a PC modem, but other devices are possible to *set up* a connection with an Internet provider. Identification of the provider traditionally can be done by number or any other unique addressing on an existing network (IP addresses). Identification of the customer usually is done by username and password. After a *verification* procedure the connection is opened and the customer can *communicate* on the network.
> Internet connections communicate by data transfer using standardised protocols. The connection usage is characterised by periods of continual packet transfer alternated with periods of inactivity. Payment is done on the basis of the length of the connection and / or the total amount of data transfer.
> At some time the connection will be *closed* on request by either side.
> The Internet provider *registers* the connection. It also *registers* the amount of data that is transferred during the connection period. Note that, if a telephone line is used, the telecom company registers the period that the connection stays open (see the telecommunications use case).
> At certain periods in time the Internet provider *sends out* a bill to the customer, who *pays* either electronically or by normal bank transfer. The bill can be based on the number of connections the customer has set up, the length of time of these connections and the amount of data transferred. This information is also *visualised* to the customer.

Step 2: Identifying objects and attributes

> From the description we can derive the following objects in Internet service:

–Provider	The Internet provider.
–Customer	Has an agreement with the Telecom Company on delivery of the communication service. Identification information are attributes of the customer.
–Session	A connection is established between the customer and the provider. The session period is characterised by the starting time and the length of the connection. And also by the amount of data transferred during a session. Note that the Telecommunication service can be used to set up a session.
–Modem	A modem device delivers the means to communicate.

–Data	Data are sent in packages. The total amount of data sent during a session should be registered by the provider.
–Accounting System	Session information is used for billing. Therefore we need an accounting system to handle this billing. In the accounting system the billing frequency and banking information is included. The session data are the input in the accounting system. Output is an account view to be sent to the customer.
–Information System	The user should be able to view the session information. Therefore we need an information system to handle this user feedback.

Several of these objects can be seen as services in the sense of Jini / OSGi. A modem device is a service which transfers information to the provider. The provider service accepts a session and monitors it. Together these services can form a service bundle for Internet service.

NB. We will not consider the preparation of the bill as a part of the service system. This will be handled by the accounting system. However, on-line viewing of the account can provide added value. We can enhance this to a special service for "e-Billing".

Step 3: Identifying operations

The following actions can be identified in the above description of Internet service:

–set up session	
–identification	The customer must identify him/herself.
–communicate	Data has to be transferred from one side to another and vice versa.
–close session	On request the connection should be closed.
–register session	The connection data have to be stored into the accounting system.
–send out bill	On the basis of connection prices a bill has to be prepared for a user.
–pay bill	The user should pay the received bill.

Step 4: Deployment of tasks

Traditionally the Internet provider is the gateway to the World-Wide-Web. This is also reflected in the infrastructure diagram. As an alternative the WWW-service or part of it can be placed on the Base Station or even the Local Access Node, creating an Intranet. In this case more intelligence is needed at these nodes.

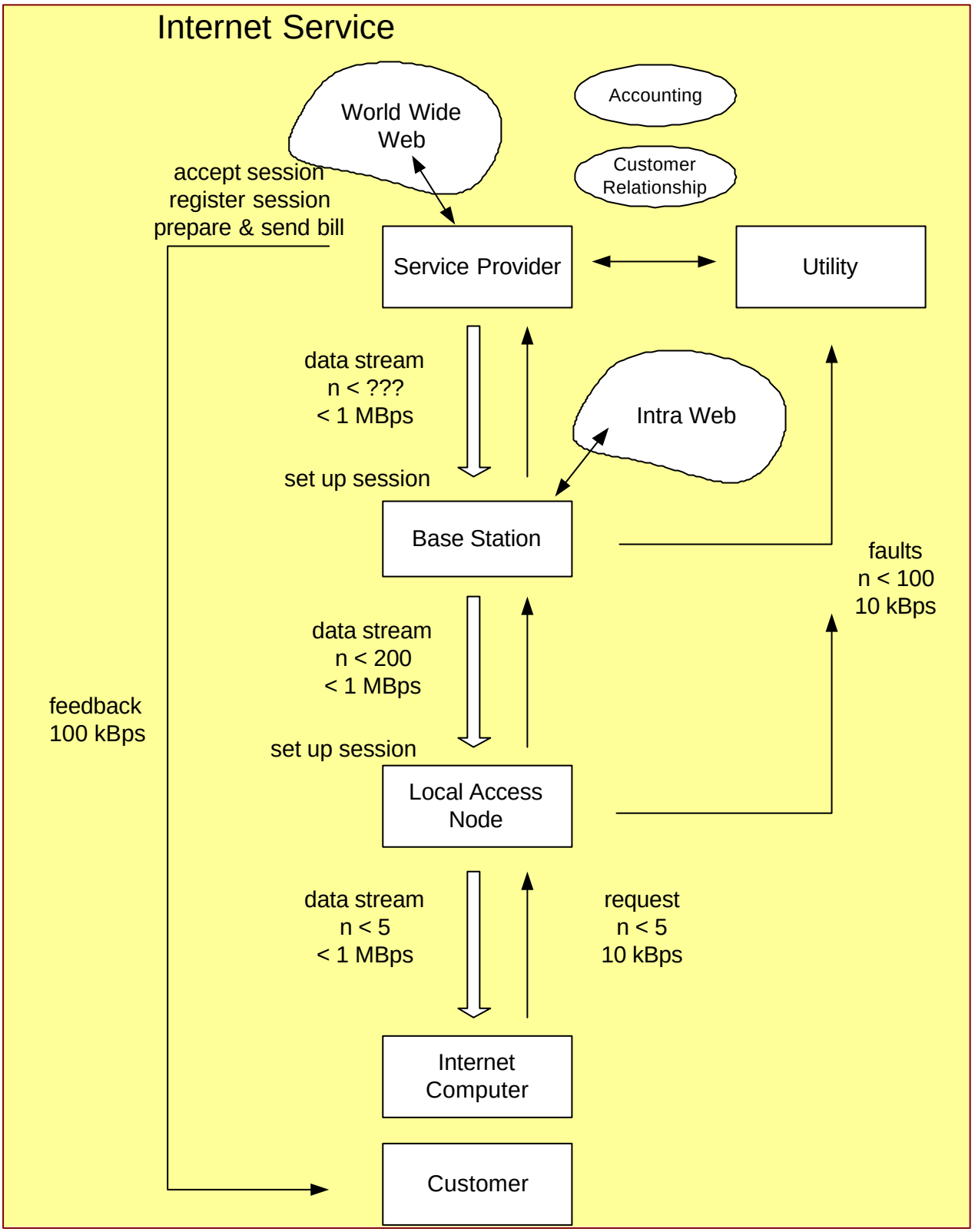

Figure 4.5. Powerline Internet Service Schedule

4.4.2 Expansion

Note that Telecommunications, worked out in the previous paragraph as a point to point connection, can also be performed using real-time package switching as described in this paragraph.

4.4.3 UML Object Diagram

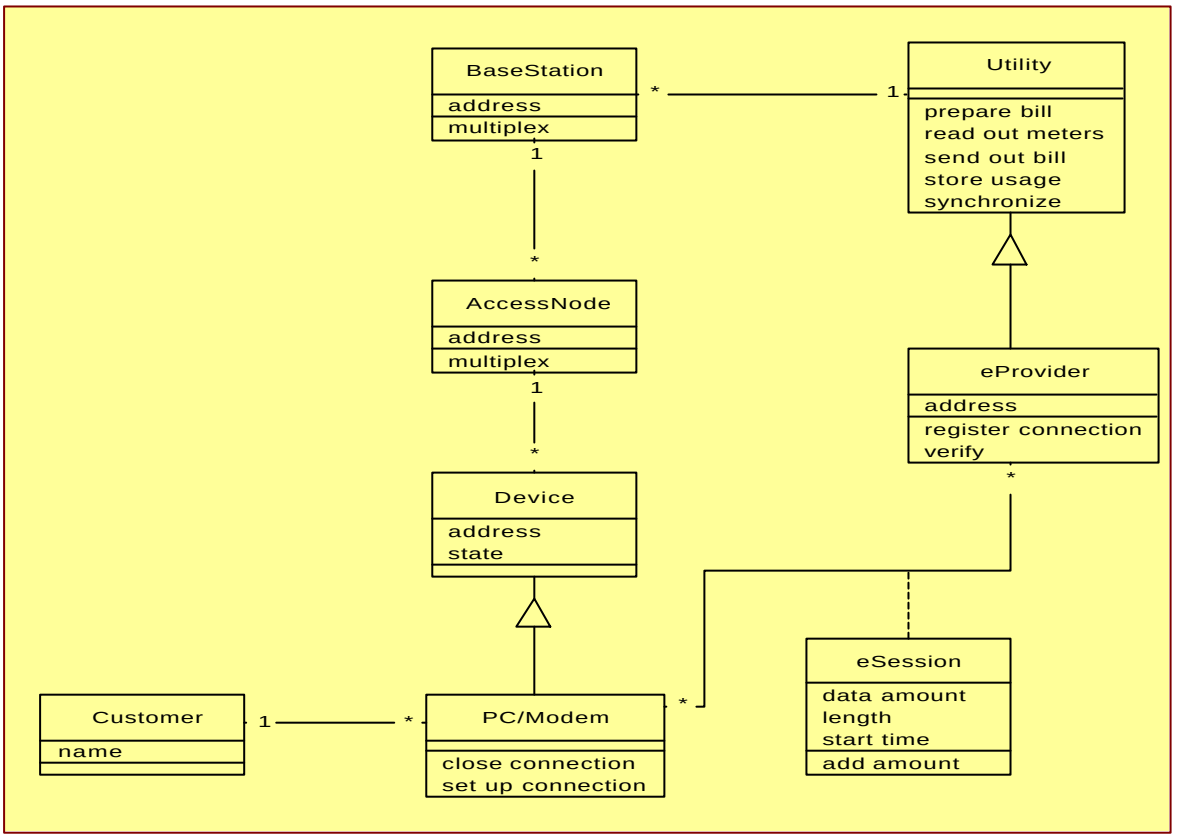

Figure 4.6. UML Object Diagram Internet Service

As telecommunications also Internet service has much simpler control structure than DCMS applications. The service is less sensitive to failure, since a session is mainly characterised by existence or non-existence. Registration of session information only takes place at the central (provider) level. Therefore also timing is not essential in Internet service. Finally WEB-applications are output/browse-driven with no major consequences, when errors in output coding occur.

4.5 Entertainment/Multimedia Applications

4.5.1 Use case: Video On Demand

Step 1: Description of the use case

The video on demand service can be described as follows:

A customer uses a device to *request* video service. The video distributor lets the user *choose* one of the available videos. The customer may *require a sample* clip from a video. Price information should be *mentioned* on time of choice.
The customer *accepts* the video after which he *starts* the video play. In some cases further *identification* (e.g. username / password) will be necessary to receive a video.
The distributor *registers* the chosen video for the customer and *starts* the video transfer on request. It also *registers* the amount of data that is transferred during the playing period.
At some time the video will be *paused* during a short period on request by the customer. The distributor registers this. At the end of the video the connection is *closed*.
At certain periods in time the distributor *sends out* a bill to the customer, who *pays* either electronically or by normal bank transfer. The bill is based on the number of videos the customer has ordered. Monitored information on the video service is also *visualised* to the customer.

Step 2: Identifying objects and attributes

From the description we can derive the following objects in VoD:

–Video distributor	The video on demand provider.
–Customer	Has an agreement with the video distributor on delivery of the VoD service.
–Video	The stream data that is transferred from the distributor to the customer.
–Video Play	The actual visualisation on the customer side.
–Monitored Information	The list of videos the user has watched, the time of watching, the video prices, etc.
–Price	Price is normally based on the video price, which could be time dependent (either time of day or difference between e.g. premiere videos or older ones).
–Accounting System	Connection information is used for billing. Therefore we need an accounting system to handle this billing.
–Information System	The user should be able to view the connection information. Therefore we need an information system to handle this user feedback.

Several of these objects can be seen as services in the sense of Jini / OSGi. A device is a service which transfers video images to the customer screen.

The VoD service accepts a video connection and monitors it. Together these services can form a service bundle for Video on Demand.

NB. We will not consider the preparation of the bill as a part of the service system. This will be handled by the accounting system. However, added value can be provided by on-line viewing of the account. We can enhance this to a special service for "e-Billing".

Step 3: Identifying operations

The following actions can be identified in the above description of VoD:

–request video service	
–show video list	
–choose video	
–identification	The customer must identify him/herself.
–start video	
–play video	Data has to be transferred from one side to another and vice versa.
–pause video	
–close connection	On request the connection should be closed.
–register	The video play data have to be stored into the accounting system.
–send out bill	On the basis of video prices a bill has to be prepared for a user.
–pay bill	The user should pay the received bill.

When we think about the VoD service operational and maintenance tasks must not be forgotten. We must be able to add or remove customers, handle operational errors, etc. Also we must be able to set connection price levels or cost rises.

A user authentication might be necessary before a connection is opened to communication.

Step 4: Deployment of tasks

Traditionally telephone or Internet connection is set up by direct line. We could use the residential gateway mentioned earlier to take over this role.

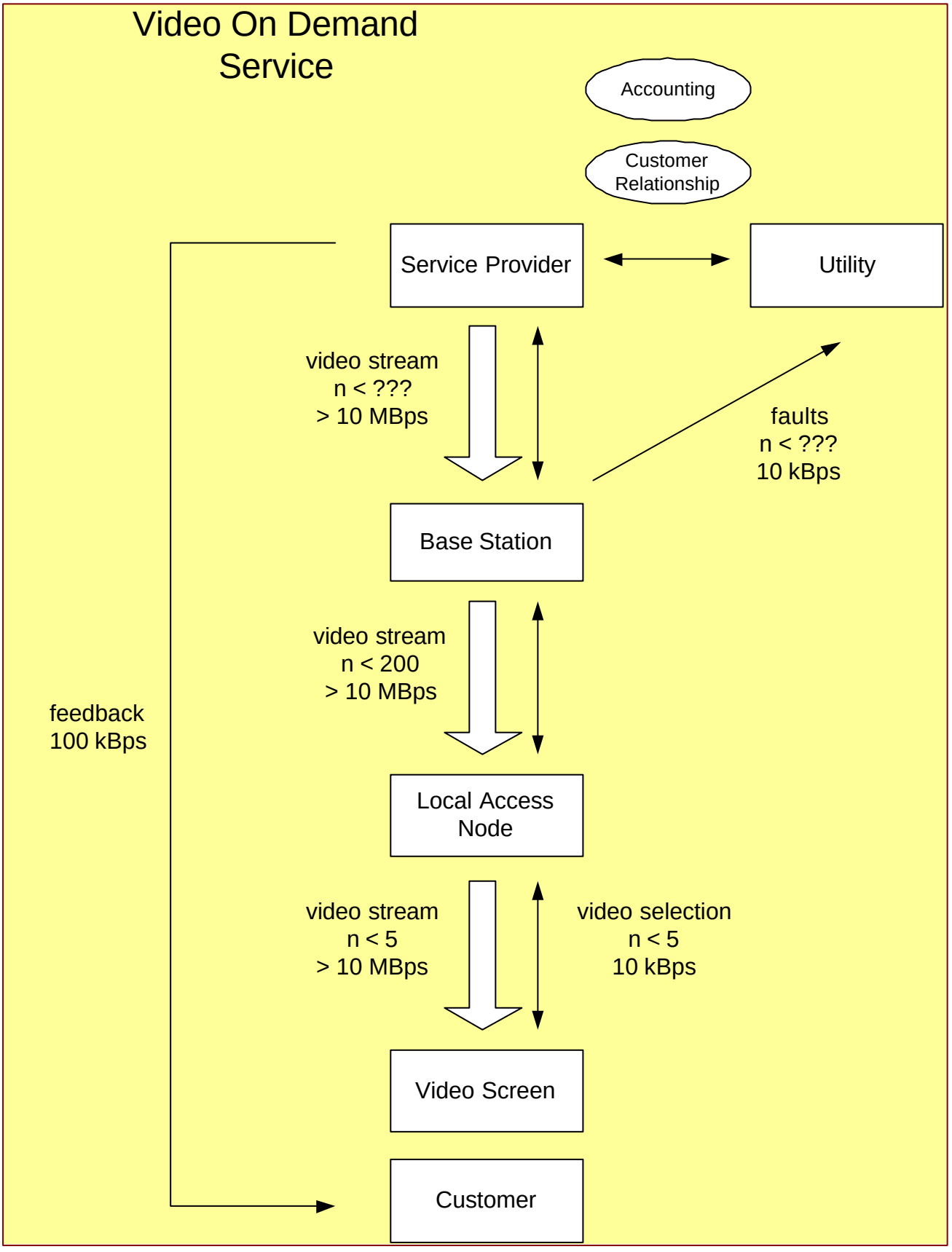

Figure 4.7. Powerline Video on Demand Schedule

4.5.2 UML Object Diagram

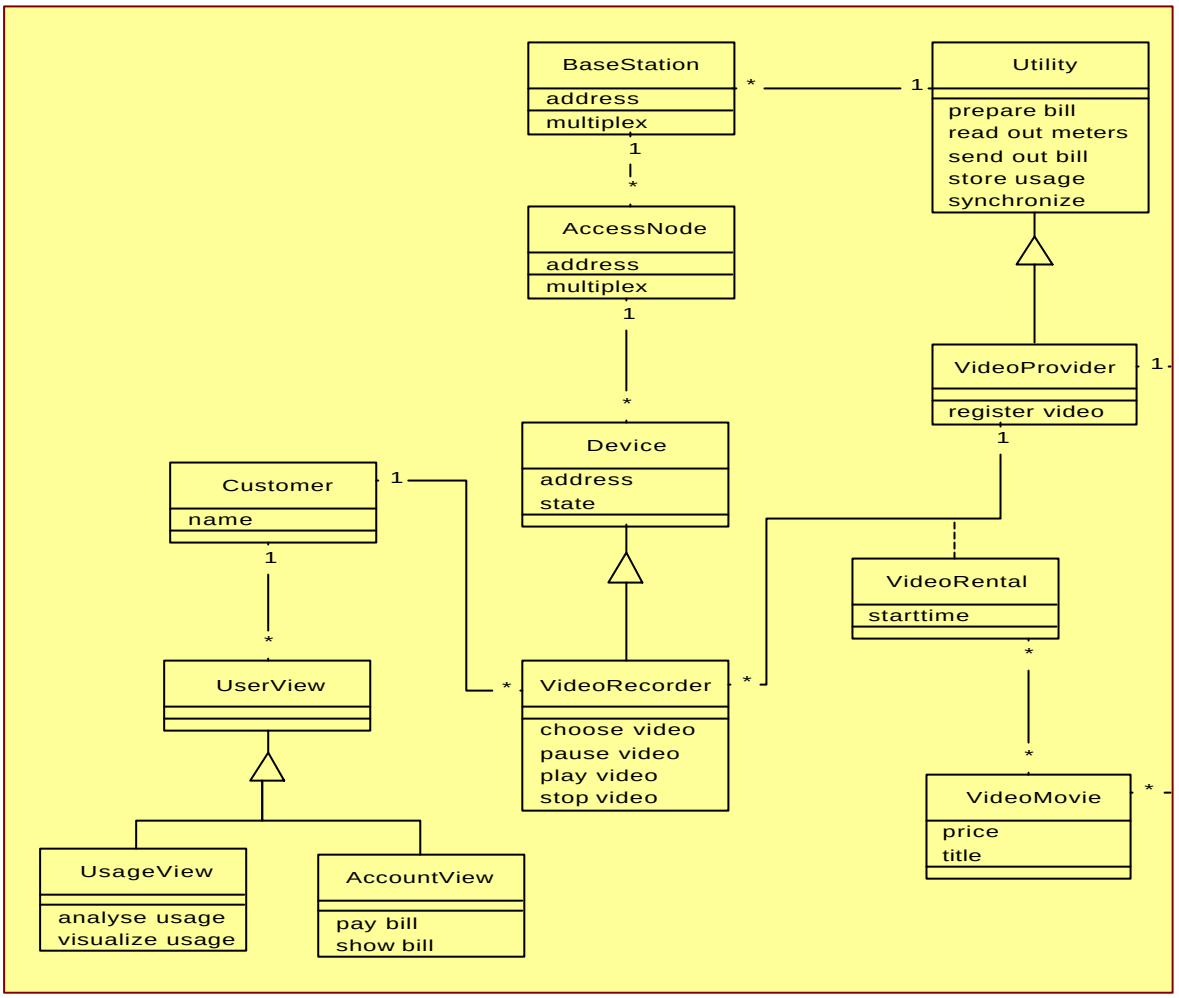

Figure 4.8. UML Object Diagram Video on Demand

Video on Demand is basically comparable with telecommunications and Internet service. However, a significant difference exists in the control of service.
When a video is ordered, but system failure prevents the customer from seeing the complete video, one can argue that no service is given. Therefore failure control is an important service issue. Timing is not essential in video on demand services.

4.6 Automation / Control Network

In a control network we have different types of devices working together to perform a defined task. The control network is based on the following procedure:

- collecting information from a number of sources,
- making any decision based on this information,
- communicate the decision to the involved devices and sources.

This procedure can have a semi-continuous character, in which the control is continuous over time, or it can be a single decision in time, potentially followed by some action elsewhere in the network. Also control can function as an autonomous process or it can be triggered by an event in the network.

Part of these control systems can function locally. Added value for powerline services is the connection to the outside world. A main issue for a good software architecture is the distribution of the control logic in the network.

Examples of such control networks are:

- Presence detection: a sensor can be used to registrate the event. The control system acts by e.g. switching on the lights or turning on the heat control.
  Presence detection can function purely in-house. If it is extended to security, a warning can be issued to the outside world, e.g. the security service provider.
- Performance monitoring: power quality control, outage detection, appliance diagnostics, etc. The control system monitors the network for any special event. On occurrence either an action can be taken or a warning can be issued at some level in the network.
  Performance monitoring can function mainly in-house. In case of an event it can transcend to the service provider.
- Heat control: on the basis of presence in rooms, expected climate conditions (temperature, wind, sunshine) and even expected energy price a heat control system can continuously optimise the indoor climate in a room or building.
  Comfort management can function in-house or near-house. However, information on fluctuation of energy prices, which will be retrieved form the outside market, enhance the value of the system since it can be used to cut costs.
- Load management: given a number of tasks a load control system can distribute these tasks over time such that within certain constraints peak energy usage can be avoided.
  Load management in-house does not add much value. Peak avoidance might be a main topic for the utility if cumulative effects over a large number of households can be taken into account.

4.6.1 Use Case: Comfort Management

Step 1: Description of the use case

The following example is taken from [6]. In this paper a complete comfort control system has been worked out. At this level we are not concerned with the logics of the control itself, but we will concentrate on the main tasks in the control network.

Comfort management can be described as follows:

A customer can *define* personal profile for thermal comfort and air quality in the rooms of his/her house, based on expected presence, type of activity, etc.
The comfort management system *controls* a number of devices, which can be used to influence the indoor climate. These or other devices can also be used to *measure* the different aspects of the indoor climate and the environment, which influences the indoor climate.
Each setting of a control device during a certain period in time requires an amount of energy, which can be obtained from local sources (e.g. solar power, solar radiation, heat buffer) or has to be obtained from an energy market. Energy usage from the market is coupled with a market price for each period.
The task of the comfort management system is to provide in each room comfort within the customer-set ranges at a minimum price from the customer point of view. In this task information is used from expected market prices and weather forecasts.

Step 2: Identifying objects and attributes

From the description we can derive the following objects in comfort management:

–Customer	The communication provider.
–Customer profile	Time-dependent settings of required comfort by the user, given either as a time-series of values or as a time-series of ranges on comfort aspects.
–Indoor climate	A possibly multi-dimensional vector giving the quality of the comfort in a room. Thermal comfort and air quality are both aspects of the indoor climate.
–Room / House	We have to model the customer premise in which each room might have different characteristics and different possibilities for comfort control.
–Device	In the above description we have two types of devices: control devices can be instructed to perform a task; measurement devices give information needed by the control system.
–Period	
–Energy source	Different energy sources can be used to actuate different control devices. Energy sources can have limited capacity and time-dependent pricing.
–Market	The market is the place where energy is supplied. In many cases the utility serves as the market place for the customer. Also the power exchange might serve as the energy market.
–Market price	At the energy market energy will be supplied at a price per period principle. Note that the energy market can be considered at the same level as a device: on request the market can deliver information about price.

Step 3: Identifying operations

The following operations can be identified in the above description of comfort management:

–set user profile	the user can define a required comfort profile and permitted deviations over a period in time (typically 24 hours).
–get value	from a number of in-house and shared devices information on current climate must be gathered which serves as input for the comfort control:
–control comfort	given the current settings of a number of comfort devices and the expected market energy prices and the weather forecasts a scenario is determined to deliver comfort within given profile bounds at optimum (minimal) cost.
–set device	on the basis of the output of the comfort control the setting of each comfort influencing device must be adapted.
–get prices	expected market prices for energy over the decision period (24 hours) must be retrieved.
–get forecast	expected climate conditions over the decision period (24 hours) must be retrieved.

Step 4: Deployment of tasks

Comfort management is a service, which is focussed on one house. Most of the information needed, can be obtained from the house or its direct environment. Optimisation by price requests pricing information from the market to be available.
The logical level for comfort control therefore is the access node to the home. A weather 'station' can be set up at a base station to obtain climate information. The connection to the outside world is used to obtain weather forecasts and market price info. If prices are fixed by contract with a utility these prices might be made available at the access node.

A detailed design is described in [6]. Characteristic in this design is the high level of control needed at a low level in the infrastructure, the Local Access Node. Therefore the access node requires larger processor capacity and memory. Communication to the service provider is less essential. Most information is gathered locally in the house or near the house. Pricing information and weather forecasts have to be retrieved from a central level. In the comfort management system example multiple autonomous agents achieve this.

4.7 References

[3] James Rumbaugh, Grady Booch, Ivar Jacobson - *Unified Modeling Language Reference Manual*. Addison-Wesley, 1998.

[4] Ivar Jacobson, Grady Booch, James Rumbaugh - *The Objectory Software Development Process*. Addison-Wesley, 1998.

[5] Grady Booch, James Rumbaugh, Ivar Jacobson - *Unified Modeling Language User Guide*. Addison-Wesley, 1998.

[6] Robert Orfali, Dan Harkey, Jeri Edwards - *The Essential Distributed Objects Survival Guide*. John Wiley & Sons, 1996.

[7] Ron Ben-Natan - *CORBA, a Guide to Common Object Request Broker Architecture*. McGraw-Hill, 1995.

[8] Boertjes, E., J.M. Akkermans, R. Gustavsson, R. Kamphuis. *Agents to Achieve Customer Satisfaction: The COMFY Comfort Management System.* Proceedings of the Practical Application of Intelligent Agents and Multi-Agents Conferences (PAAM), April 10-12, 2000.

5 Architectural implementation issues.

5.1 Partitioning of functionality

In the previous chapter a number of application types enabled by PLT were modelled. Next step is deriving a partitioning scheme for the objects discussed there. Classically, in the hierarchical model three partitions are to be discriminated:

- Peripheral equipment. Sensor, actuator.
- Communication. LAN, MAN, WAN.
- Data processing

When viewing in this way, the largest number of interfaces is generated in communication. A customer communication interface has to interact with the power line carrier, the bcal, metropolitan and wide area network, the utility side telecommunications interface and the data processing packages through several layers. In order to enable two-side communication of control signals, very complex mechanisms have to be designed. Therefore delegation of control and distribution tasks to suitable concentrator subsystems in the chain are necessary.

Apart from the large number of different interests of parties involved in realising applications and the lack of standardisation to “open” hard- and software and the cost-consequences thereof, these issues form the largest problem for realising applications. Current developments in software and hardware technology lead to an increasing processing power and bandwidth for small scale, powerful dedicated systems. In this respect there is less a need for **big** bandwidth, but for **smart** bandwidth. The last statement stresses the fact that, for most powerline applications, efficient and reliable usage of limited bandwidth between powerful processors is more important than a high bandwidth between “dumb” processors. Important in this respect is the point at which data is converted into information and control directives.

5.2 Required management and configuration facilities

5.2.1 Object persistence

In current software design methods nowadays, data (attributes) and procedures and behavioural aspects are contained in objects. Depending on the functionality of applications and for security reasons, the attributes of objects and the state information have to be saved at the proper level. In case of outage of the application or system, procedures have to be designed to restore the object’s state to the one before the crash.

For a “simple application” as meter reading this already poses problems. A crash can occur during read-out using the sensors, during transfer to the gateway or during operations further on to the legacy, utility accounting system. How higher hierarchically the metering “intelligence” is in the network, the more complicated is the restoring algorithm. Therefore metering information objects should be persistent at the lowest hierarchical level. From the software maintenance point of view, metering applications should be as much as possible thin-client. Embedded processors should have high MTBF-figures and should be able to store status

information with high reliability. Metering read-out should be preferably realised using “servlets” wrapped to legacy metering and billing applications and databases.
In more complex control applications like multi-parameter optimising comfort management, more complex software is implemented on larger processor capacities. Memory requirements will be larger. Information contained in objects can be downloaded at regular intervals from a larger computer system in the network. Again the emphasis in reliability and computing power is on the RG. Loosely coupled feedback (analysis of measured data and adaptation of control parameters) on the performance of the algorithms from larger computer resources higher in the network hierarchy will be a major benefit but not essential prerequisite. Sustained operation in the absence of larger computing systems is essential.
For the other application types like P2P, IP and EMA the emphasis for secured object persistence clearly is on the server side. The control complexity as can be derived from the UMLK-models is lower.

5.2.2 Replication mechanisms, serialisation and versioning

If object information is needed at other levels in the application hierarchy, data and state information have to be transferred and replicated across the network. For successful execution of remote procedure calls between objects residing at different nodes of the network, a serialisation mechanism transfers all necessary information to the correct nodes. A number of standards facilitate this mechanism. Currently COM and CORBA are the mainstream standards for remote procedure calls, but the needed resources are heavy and are geared to larger processor-capacities than present in current small-scale apparatus. In SOAP and DOM, a WWW based document object model, a connection to these standards is made also incorporating XML. In Java, a mechanism for serialisation using the RMI standard is also contained. This technology however, especially concerning real-time Java, is in the development phase and has a lack of support of the leaders in the field of real-time software development. Furthermore, firewalls pose large difficulties for transparent Java-RMI.
From a software maintenance point of view, the home network and gateway appliances will have a long usage cycle. Maintenance of hardware and software versioning therefore is important for proper operation of applications. Current software standards have a versioning mechanism to check if objects interact with each other are generated with the same compiler products and support libraries. This mechanism will be mandatory for DCMS applications.

5.2.3 Multi-agent architectures

Applications, in which a large number of information-sources have to be monitored or inspected, benefit from multi-agent architectures and technology. The scope of applications for agents therefore is geared to surveillance systems, scheduling of events and meetings in residential areas and for calendar applications. Agents have the advantage of an auto-replication mechanism. Furthermore, they are autonomous in bargaining with other agents for a resource using market algorithms [1]. In this way optimisation problems are easier to solve in a computer network environment. Especially for large distributed systems the agent abstraction has its benefits. The control behaviour of these systems is hardly codeable in conventional ways because the large number of possible states of the

system as a whole. The optimisation of the system operating with distributed processors in a concerted way, e.g. for energy management, is hardly solvable by central, analytical algorithms and procedures as well. Market and auction algorithms implemented in a large number of autonomous processes in a distributed processor network are the only way for successful implementation.

5.2.4 Control timing implementation and synchronisation

A large layered control-network has delicate timing. Frequent time synchronisation and propagation mechanisms are necessary to guarantee secure parallel operation. Again, the DCMS-type of applications is most important in this respect. For the other types of applications the responsibility can be laid on the server side.

5.3 Conclusions

In view of the above architecture requirements the perspective of single residential gateway apparatus covering the four application types discussed above is difficult to imagine. Distributed control and management applications do stand too far apart in architecture requirements. The prime combination potential in gateway functionality combination lies in telephony and Internet application types especially for packet-switched telephony and in telephony and multimedia applications, if the DSL-bandwidth can be extended to support real-time video. However, the three order of magnitude difference between telecommunication company and cable communication fares remains a large barrier. Therefore, hybrid solutions, in which several gateway apparatus are involved in combinations, are most likely. Integration in the application then is on a higher level on the network application software. This conclusion follows from the software architecture perspective, where only reduction of the complexity of interfaces and control behaviour and transparency counts. Many stakeholders being involved, a multi-gateway, multi-service software architecture, from a business point of view, imposes a constraint on the business modelling process.
The possibility of attributing a role to the transformer station hard- and software as a residential **area** gateway is an important difference compared to other last-mile access technologies. In case of control applications, if such a gateway could replicate the functionality to a large number of small size, cheap DCMS-gateways in dwellings a definite advantage can be imagined. To a lesser extent this also holds for IP and telephony applications. However, the number of standards and software components for residential area gateways is scarce.

The table below gives an indication of the importance of certain aspects of application types on residential gateways discussed before:

	DCMS	Point-to-point	Internet	Multimedia (3rd generation Set-top box)
Preferred messaging	Client/thin server, asynchronous	Peer-to-peer, synchronous	Thin client/server, asynchronous	Thin client/server, synchronous
Software deployment, maintenance	PROM, Download of applications	PROM	Download	Once
Local Processing power	High	Low	Medium	Low
Local Memory	Low	Low	High	Low
Central processing power	Low	Medium	High	Medium
Control complexity	Large	Medium	Medium	Low
Autonomous Intelligence	High	Low	Medium	Low
Upstream bandwidth	Low	Moderate	Moderate	Low
Downstream bandwidth	Low	Moderate	High	Very High
Response time	Low	High	Moderate	High
Always-on connection of communication channel	Low	Low	High	Moderate

Table 2. Comparison of application type attributes.

DCMS need to be very reliable and should perform autonomously. Control is fine-grained, exception handling is very delicate and processing power requirements need to be laid out on handling complex transition state logic and real-time handling of combinations of events. Most of the application logic resides on a locally processing RG, whose bandwidth requirements are very limited.

Point-to-point communications are based on synchronous peer-to-peer communication. Processing power requirements are limited; real-time software with dedicated DSP's (Digital Signal Processors) will take care of coding and decoding.

Internet traffic uses its processing power mainly for dynamic graphical visualisation and for bulk data transfer. Control is coarse-grained and exception handling not complicated.

For broadband, digital multimedia applications third generation set-top boxes with local intelligence have very limited, but very fast local processing power in FPGA's or ASIC's.

In an architectural sense imagining an RG handling all four kinds of applications cannot be foreseen. The diverse requirements for residential gateway technology

will most likely lead parallel application driven implementation of several gateways. As stated earlier, hybrid solutions, from the resemblance in requirements, are only between Internet applications and DCMS-applications.

5.4 References:

[1] P. Novais, L. Brito, J. Neves. Agreement in Virtual Marketplaces with CBR supported negotiation. Proceedings of the Fifth International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology. Manchester, 2000.

6 Used abbreviations

ADSL	Asymmetric Digital Subscriber Line
ASIC	Application Specific Integrated Circuit
ASP	Application Service Provider
ASP	Active Server Page
COM	Common Object Model
CORBA	Common Object Request Broker Architecture
DA	Distribution Automation
DCMS	Distributed Control and Measurement system
DCS	Distributed Control System
DOM	Document Object Model
DSM	Demand Side Management
EHS	European Home Systems
IEC	International Electrotechnical commission
IP	Internet Protocol
LON	Local Operations Network
OSGi	Open Services Gateway initiative
PALAS	Powerline as an Alternative Local AccesS
PLC/T	Power Line Communication/Transmission
RG	Residential Gateway
RMI	Remote Method Invocation
SCP	Simple Control Protocol
SOHO	Small Offices Home Offices
UCA	Utility Communication Architecture
UPNP	Universal plug-and-play
USB	Universal Serial Bus
XML	eXtended Markup Language