VAN

FP6/2004/IST/NMP/2 - 016969 VAN

Virtual Automation Networks

Work Package 7

Cooperation of private and public networks

Task 7.4

Selection and Integration Guidelines

Deliverable D07.4-1

Integration Guideline and

Conformance Test

Document type	**:** Deliverable
Document version	**:** final
Document Preparation Date	**:** 09.09.2009
Classification	**:** confidential
Contract Start Date	**:** 01.09.2005
Duration	**:** 31.08.2009

Information Society Technologies

Project funded by the European Community under the "Information Society Technology" Programme (2002-2006)

Rev.	Content	Resp. Partner	Date
0.1	Document structure	Müller, Siemens	08/10/10
0.2	Document structure changes from WP7 Kick-off Meeting (Brno) and responsibilities discussed	Müller, Siemens	08/10/28
0.3	Changes from Telephone conference included	Müller, Siemens	08/11/06
0.4	Comments to chapters added	Müller, Siemens	08/12/10
0.5	Input added Siemens, Ifak and CVS. Missing parts marked.	Müller, Siemens	09/07/06
0.6	Input added from: Aucoteam. Missing and not completed parts marked	Müller, Siemens	09/07/10
0.7	New version from WP7 Meeting.	Müller, Siemens	09/07/17
0.8	Updates of chapters	Müller, Siemens	09/07/29
0.9	Version for review	Müller, Siemens	09/08/18
0.10	Review Comments added and worked out	Müller, Siemens	09/09/03
1.0	Final Version	Müller, Siemens	09/09/09

Final approval	Name	Partner
Review Task Level	Müller	Siemens
Review WP Level	Werner	ifak
Review Board Level	Klostermeyer	Siemens

Executive summary

The deliverable on hand D07.4-1 “Integration Guideline and Conformance Test” of the VAN project reports the outcome of the work within work package 7, Task 7.4 “Selection and Integration Guidelines and Conformance Test” and was established in the project months 39 to 48.

The documentation specifies the outcome of the implementation work basically defined in the previous deliverables of WP7:

- D07.1-1 “Report on analysed public network technologies”
- D07.2-1 “Integration Concept - Architecture Specification”
- D07.2-2 “Integration Concept - VAN QoS over public networks specification; Network Management and common signaling specification”
- D07.3-1 “Common Interface Specification, Gateway Specification”

The aim of WP7 task 7.4 is the prototypically implementation of the functionalities described in foregone deliverables as well as providing support to integrate the results from the implementations into the Process and Factory IES.

To describe the implementation, the addressed functionalities like VAN Telecontrol Profile, VAN QoS Monitoring and the VAN Provider Switching were separated in the general component specification which also provides information about changes to previous deliverables and the installation and user guideline.

Therefore the document is structured into the following main chapters:

- Chapter 1: Introduction
- Chapter 2: System environment regarding IES
- Chapter 3: Components of VAN WP7 Software
- Chapter 4: Installation and User Guideline
- Chapter 5: Fulfilment of Requirements from D01.2-1
- Chapter 6: Recommendation for a Conformance Testing
- Chapter 7: Conclusion

The document starts with an Introduction which describes more detailed the content of the chapters and their relation to the previous deliverables.

The chapter System environment regarding IES contains the final status and information about the Factory and Process Industrial Experimental Setups.

Based on the component structure of the WP7 software chapter 3 discusses and explains each functionality provided by this work package. This includes also the description of the relations and requirements to other functionalities.

Chapter 4 comprises an operating manual for the previous described Telecontrol, Switching and QoS Monitoring application.

Chapter 5 finalises the requirements collected by work package 1 and chapter 6 gives a rough recommendation for a conformance testing.

The document is a result of a co-operation between the parties Siemens, ifak, CVS, Aucoteam and TSA. Task Leader of the task was Siemens. The preparation of the document structure and the compilation of the contributions were also carried out by Siemens.

Contents

List of figures

List of Tables

1 Introduction

This deliverable describes the results of the implementation efforts of specifications of previous tasks in order to support the Industrial Experimental Setups for process and manufactory automation. It points out the final status of previous specifications:

- VAN Telecontrol Profile specified in VAN Telecontrol Profile Mapping on [IEC 61158] Type 10 [D07.3-1] and previous deliverables [D07.2-1] and [D07.2-2],
- VAN QoS Monitoring described in [D07.2-2] and [D07.3-1],
- VAN Provider Switching contained in [D07.2-1] and [D07.3-1].

The final statuses of the previous specifications are described in the chapters of this deliverable. The components implemented within this task form the general basis for the Industrial Experimental Setups. Thus the relations of the Factory and Process IES are worked out in the chapter *System environment regarding IES*. Also the requirements and structure of the two setups with main focus on functionality regarding WP7 are described. With the aid of this deliverable the integration of the mentioned components was carried out.

The final status of the components of WP7 Software is specified to describe the updates to previous specification which occurred during implementing and integrating the components in the industrial experimental setups:

- The *VAN Telecontrol Profile* shows the overview of the software divided in PROFINET CBA Server, Telecontrol Profile and communication interface. Also the changes to the previous definition of the VAN Telecontrol Config Class as well as the interfaces to parameterise and access the profile are presented. The differences based on the operating system behaviours of Linux and Windows are determined and documented.
- The used mechanism for the *VAN QoS Monitoring* to synchronise the clock with GPS and NTP are presented. Following this, the software overview and details about the measurement mechanism are explained. The IOC (Inter Object Communication) communication is defined to allow communication with the VAN Provider Switching, which configures and starts the run of a QoS measurement and accesses the results via this interface.
- The *VAN Provider Switching* implementation defined in [D07.2-1] and [D07.3-1] is explained and provides a software overview and also the necessary interfaces are described to exchange information with the corresponding QoS Monitoring.
- The *Integration Aspects for WP7 related Functionalities* handles the interaction and functionality of the Telecontrol Profile. The experiences collected while integrating the Telecontrol Profile in the Factory IES are explained as well as the used solutions.

The main topic of the deliverable is the *Installation and User Guideline* which provides information how to use each component:

- The *Telecontrol Profile* description starts with the package of the Telecontrol Profile which can be downloaded from the VAN groupware and information about the folder options with necessary previous settings. The application operating mode is divided into *Starting the application*, *Telecontrol Profile Parameters* - which are optional parameters handed over at the start-up to configure the handling of the application - and *Closing the application*. Finally information how to parameterise the *Telecontrol Config Class* with the optional and mandatory attributes explained with examples are presented.
- The installation and use of the *VAN Provider Switching* is described and also the pre-requisites to run the switching application.

- The *VAN QoS Monitoring* needs some pre-requisites which are explained as well as starting the GPS and NTP daemon. Also the IOC Communication which is used to allow communication between the QoS Monitoring and VAN Switching is shown.

In [D01.2-1] requirements for every work package were collected. Within the final task of work package 7 these requirements were worked out. They were checked based on if and how the requirements were fulfilled with the WP7 software components.

A general framework for conformance testing was already treated by [D02.4-3]. This task extends the information by important aspects for functionalities developed by work package 7.

2 System environment regarding IES

This chapter contains an overview of the system environment regarding the industrial experimental setup (IES) for factory and process industry:

- Testing the VAN-tools by using the real hardware platform and a simulated technological plant
- Description of the future testing that works in the real technological plant

For the description of the system environment it is necessary to consider the specific properties of the factory and the process industry which are different from each other.

The main objects of Factory IES are:

- Real-time communication
- Safety over public network (Centralised safety signals management)
- Secure communication over public network

Process Industry:

- Continuous processes
- Control tasks are defined as compensation of correctable and uncorrectable disturbances (Stabilisation, optimisation, safety control)
- Control activities happens in the case of disturbances only

The next chapter will explain the system environments for the Factory IES and chapter 2.2 handles the Process IES.

2.1 Factory IES

A detailed explanation about the Factory Automation Industrial Experimental Setup is in chapter 2 of [D09.2]. From work package 7 only the Telecontrol Profile release 1 is used, thus this chapter handles only the information necessary for the Telecontrol Profile.

The following figure gives an overview of the machines used in the Factory IES:

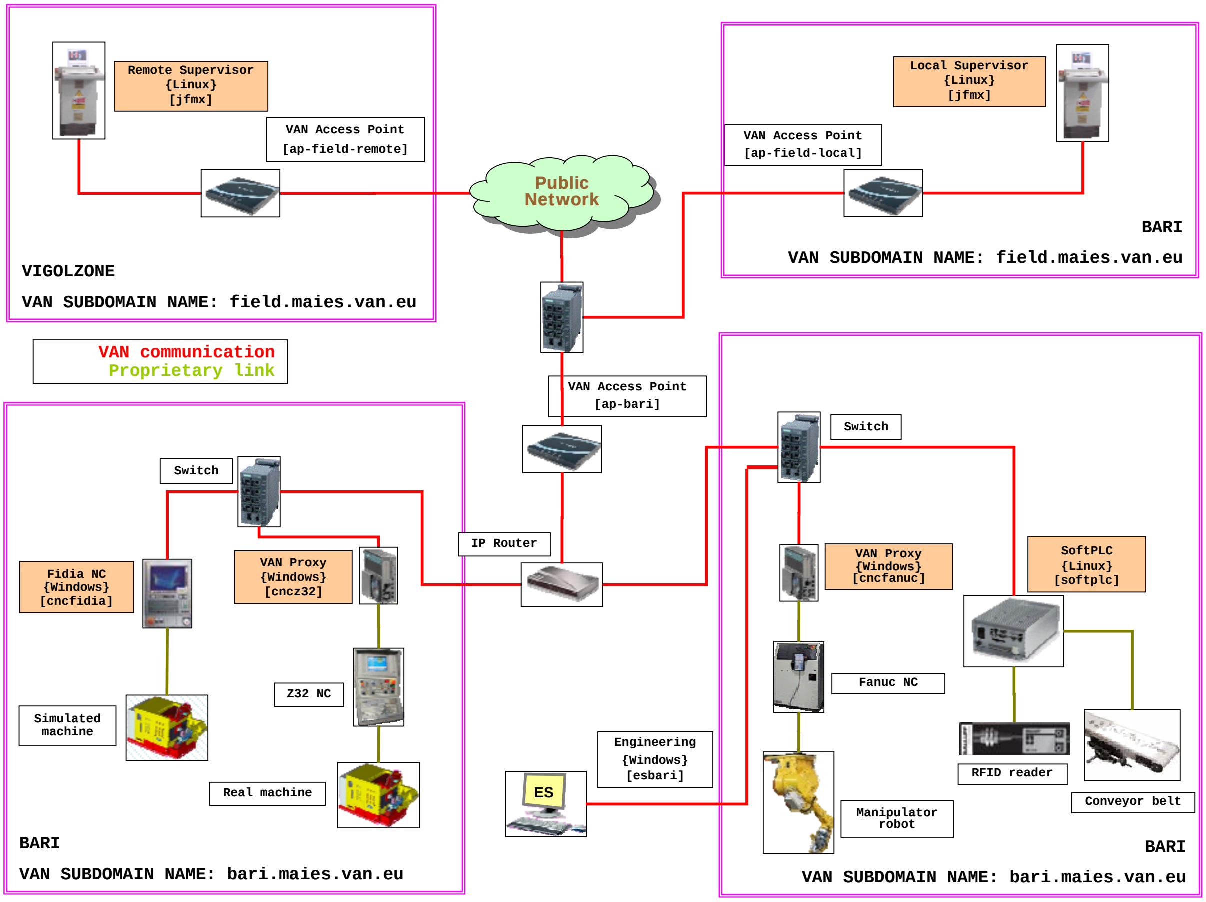

Figure 2-1 Factory IES Plant Overview

The VAN machines can be differentiated in:

- VAN Access Point (AP)
- VAN Engineering System (ES)
- VAN Automation Device (AD)
- VAN Proxy Device (PD).

For the Factory IES the *Access Point* and *Engineering System* are not in scope because they will not use any software of WP7. The *Automation Device* and *Proxy Device* are using the Telecontrol Profile release 1 (see Figure 2-1, orange marked parts), the following ADs exist:

- JFMX (Windows)
- Fidia numerical control of simulated Machine (Windows)
- Soft PLC, Control of Conveyor Belt (Ubuntu).

The following PDs exist:

- Z32 numerical control of MCM Machine (Windows)
- Fanuc Robot (Windows)

The next chapter gives an overview of the requirements for the Manufacturing IES.

2.1.1 Requirements

The general requirements to a properly functioning machining line are described in [D09.2] chapter 2:

- Fast and robust (real-time) communication between the devices in order to properly coordinate them and avoid any collision risks while performing various operations,
- Safety management: a great number of safety devices are integral part of the typical machining line, the signals coming out from these devices must be properly gathered and an adequate answer has to be provided in the shortest time,
- Security management: the data exchanged between the devices is often very sensible and covered by copyright; any interception risk must be avoided.
- Remote management and maintenance of the real plant.

2.2 Process IES

For the IES PA a Bio Power Plant application was selected. The Bio Power stations are geographical distributed centres that produce bio-energy and optional Bio fuels (Ethanol), based on selected biogenic waste materials and raw materials containing sugar and starch.

In the case of the VAN project partner, the WABIO technology company in Saxony, we have a structure of existing bio power plants and new power plants. The target is the concentration of all control and operator task to the operator centre in Gera (see Figure 2-2).

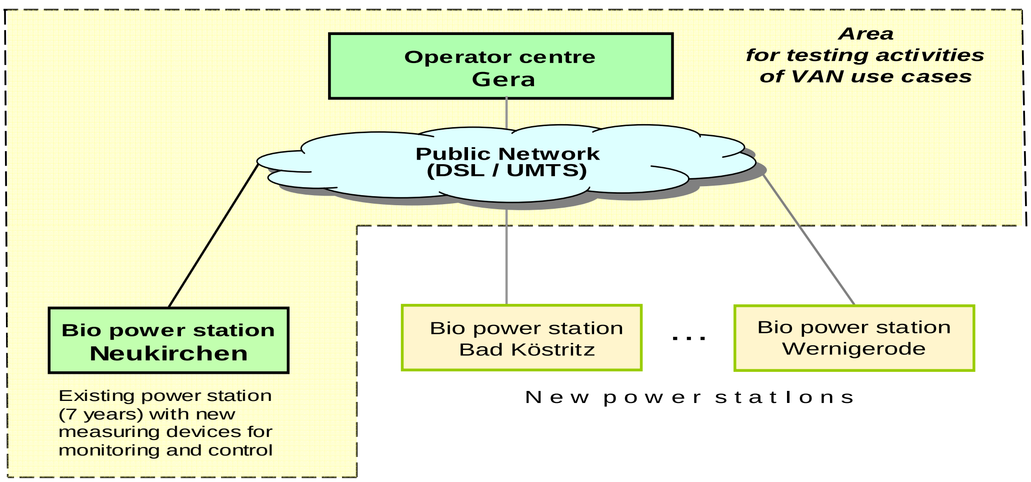

Figure 2-2 Planned structure of WABIO bio power stations

Every Bio Power plant and station is normally controlled by using a complete local control system, based on the components control system, instrumentation applied to the process. The new VAN based solution shifts some parts of the control system into a new, centralised location – the operator centre. (see Figure 2-3) and provides cost effective operation, combined with a high level automation and control, for Biogas Plants with several local Biogas Power stations.

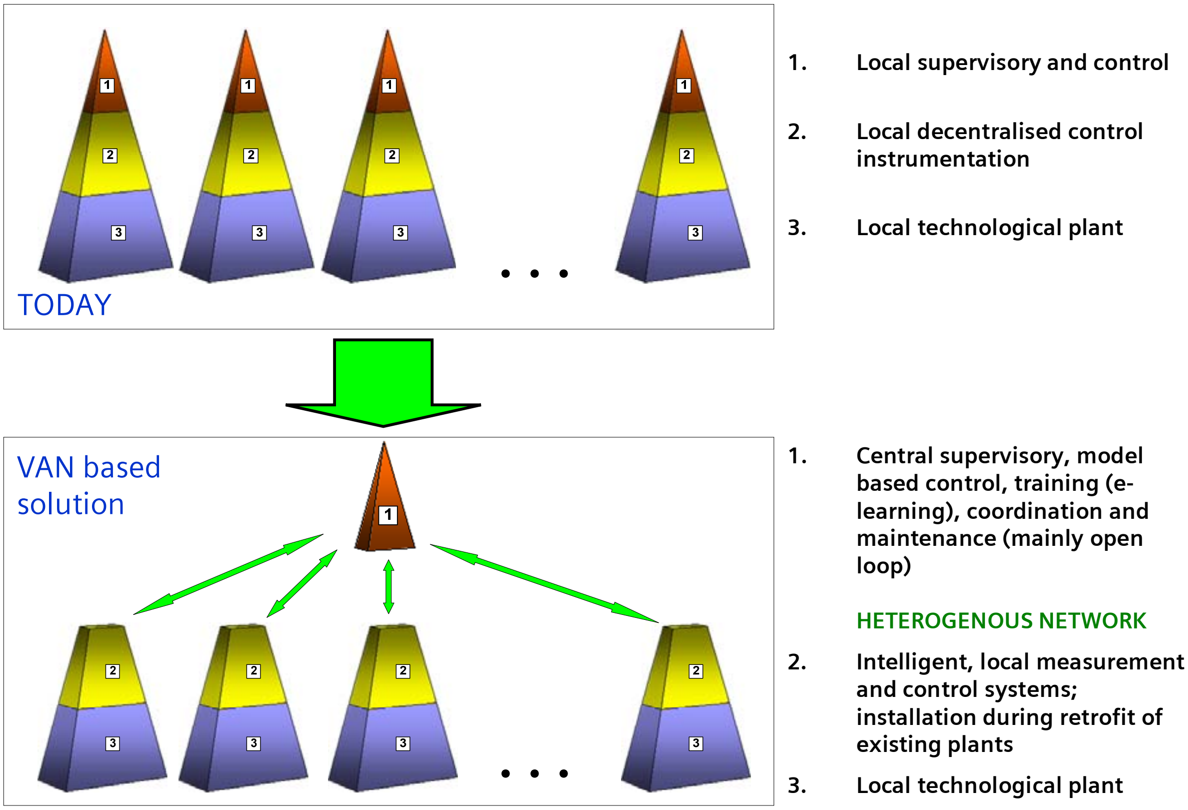

Figure 2-3 VAN based control system structures

The data between the Operator centre and the local Biogas power stations will be transferred via “heterogeneous public network” using the VAN technology.

For the demonstration of the VAN advantages were selected 2 technological use cases. The resulting VAN structure to handle these use cases and the required VAN devices is shown in Figure 2-4.

Regarding to the tasks of WP7 the following VAN-features are integrated in this structure:

- Cascaded tunnel architecture with communication paths using public networks
- QoS-monitoring
- provider switching, using of alternative public network connections (DSL and UMTS)
- using of the telecontrol profile with buffering of data in the case of temporarily broken communication paths
- VAN maintenance, realised by using the SUI system

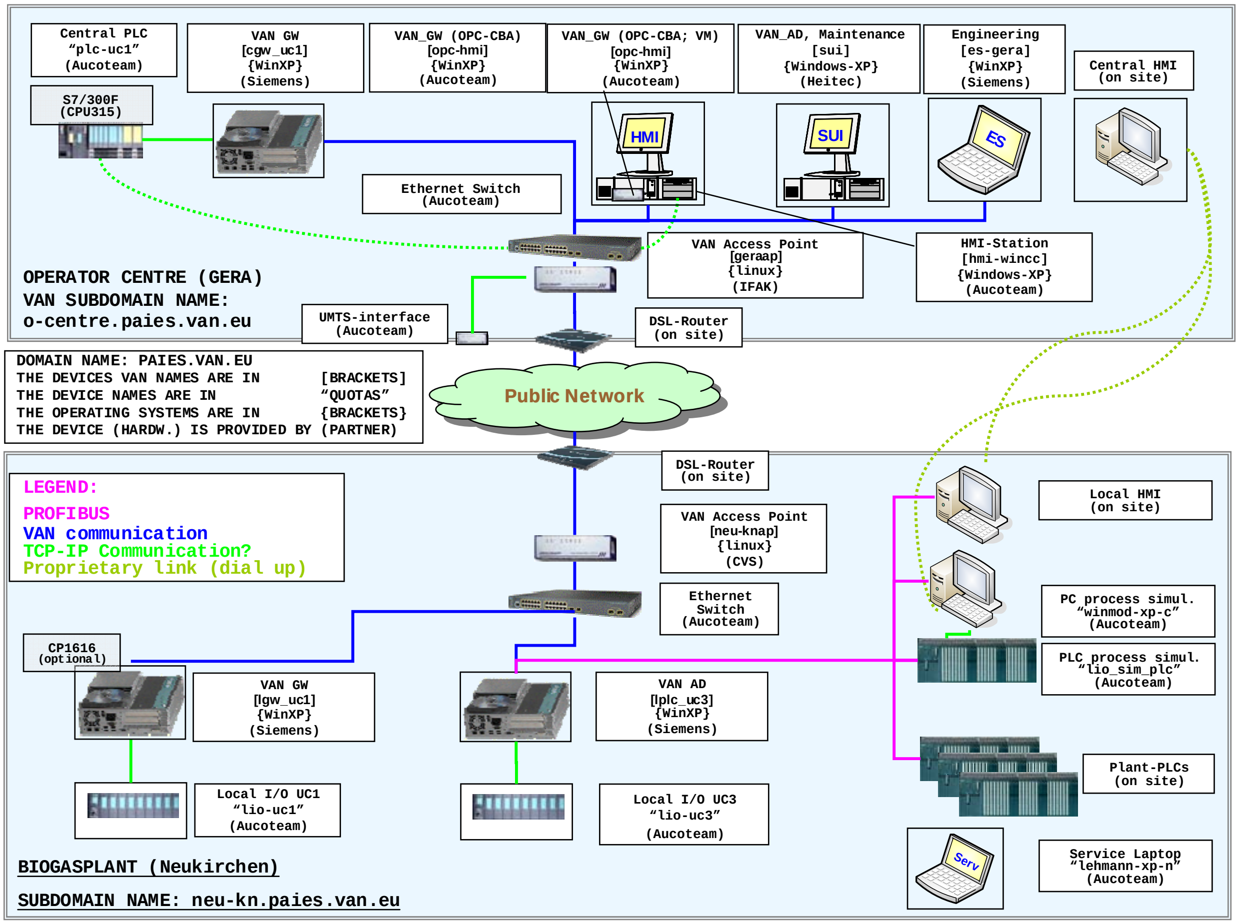

Figure 2-4 Structure and devices of the IES PA

2.2.1 Requirements

The general system features of VAN based communication connections for central control systems with distributed decentralised plants have to be evaluated from the process automation point of view:

- Real-time requirements
- Security requirements
- Safety requirements, if necessary
- QoS- monitoring and provider switching
- Mixed use of wireless and wired public communication
- Scalability of security and real-time features
- Guaranteed availability

Special requirements to the VAN solutions in the process industry for Biogas power plant regarding the VAN features and devices and the dynamic behaviour of communication systems between the central control system in the operator centre and the decentralised local systems are shown in Table 2-1.

Table 2-1 Required VAN Features and Devices for control task in the process industry

Device/ Scenario		VAN Software stacks for	Device Type	Access Mode	Max. Delay of communications system
Central control system ➔	Bio reactor	VAN GW VAN AP	No preference	cyclic scan & transfer on change (wired or wireless)	1 to 3 s
	Gas storage	VAN GW VAN AP	No preference	cyclic scan & transfer on change (wired or wireless)	3 to 4 s
	power stations (CHP)	VAN GW VAN AD	No preference	Acyclic transfer, (wired or wireless)	1 minute

2.2.2 Functional structure of VAN-based control system

Functional structure of the VAN-based control system for the biogas power plant is shown in Figure 2-5 according to Table 2-1.

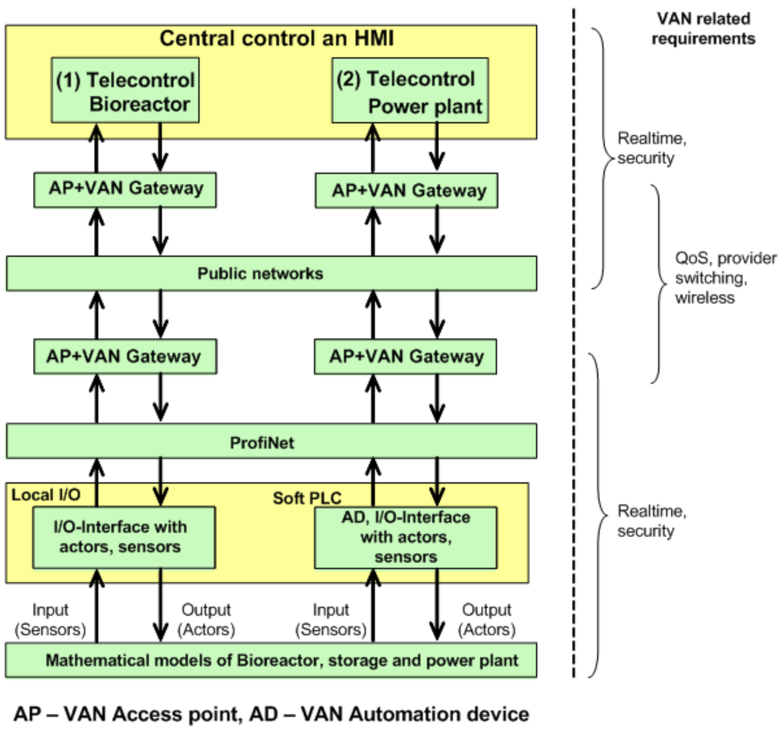

Figure 2-5 Functional structure of a VAN-based control system in the process industry

The required equipment for the interfaces with integrated VAN-tools, separated to the 2 selected technological use cases, is:

(1) Bio reactor → Closed loop control

Central control

- S7/300 process controller (hardware based PLC)
  – This PLC controls the decentralised process.
- VAN-Access Point + VAN-Gateway

Decentralised I/O-devices

- VAN-Access Point + VAN-Gateway/Proxy-device to ProfiNet
- SIMATIC DP, INTERFACEMODUL IM151-3→PROFINET (ProfiNet I/O)
- Local I/O units ET200S

(2) Power station (CHP)→ Open loop control

Central control

- WinCC-HMI-System
- VAN-Access Point + VAN-Gateway to ProfiNet

Decentralized control

- VAN-Access Point + VAN-Gateway to ProfiNet
- WinAC process controller (Soft PLC) This PLC controls the decentralised process.
- SIMATIC DP, INTERFACEMODUL IM151-3 → PROFINET (ProfiNet CBA)
- Local I/O units ET200S

Remark: The local I/O-devices are extended by a “slave PLC (S7/400)” and a software simulation system for the simulation of the technological processes for all of the use cases

The IES PA integrates the technological use cases and several VAN-devices:

- VAN Gateways/Proxy devices
- VAN Automation devices
- VAN Access Points to public networks (wired, wireless)
- VAN Maintenance system
- VAN Engineering station

The general structure of the IES PA shown in Figure 2-4 and demonstrates the VAN technologies:

- VAN routing
- VAN cascaded tunnel
- QoS and provider switching
- VAN Telecontrol including time stamp transmission
- PKI – Public Key Infrastructure
- VAN engineering
- VAN maintenance

3 Components of VAN WP7 Software

Several applications had been developed in work package 7. Their main focus is to demonstrate the results from previous deliverables. The following table gives an overview of the components of the VAN WP7 software with the corresponding partner and where information are written down in other deliverables of WP7:

Table 3-1 WP7 Components

Component	Partner	References
Telecontrol Profile	Siemens	D07.2-2, chapter 5 D07.3-1, chapter 2.7 & 3
QoS Monitoring	Ifak	D07.2-1, chapter 4 D07.2-2, chapter 2 D07.3-1, chapter 2.4
Provider Switching	CVS	D07.2-1, chapter 3 D07.3-1, chapter 2.4

The components are logically combined into two packages according to their functionality. One package is the Telecontrol Profile because it does not cooperate with the other components. The second package contains the QoS Monitoring and the Provider Switching. These two components are working close together.

The next chapter provide more information to the integration aspects of the different components. Also updates to the information provided in previous deliverables are show.

The following information provides information how to integrate and use the applications. To get more information about the structures of VAN software packages more information can be found in [D02.4-1] and [D02.4-2].

3.1 VAN Telecontrol Profile

The VAN Telecontrol Profile is available in two different versions, one for the Factory IES and one for the Process IES. The first version represents a shortened application which does not provide the real Telecontrol functionality; only the variable service is available. The purpose was to test the variable exchange within the Factory IES using tunnels. The final version of the Telecontrol Profile provides the real telecontrol functionality; the buffer and filter mechanism.

The version for the Manufacturing IES is called Telecontrol Profile release 1, while the Process IES will use the Telecontrol Profile release 2. The differences and equalities are described in the following chapter as well as changes to specifications of previous deliverables.

3.1.1 Software Overview

The Telecontrol Profile comprises three different components:

- PROFINET CBA Server with Telecontrol adaption
- Telecontrol Profile Application
- Communication Interface

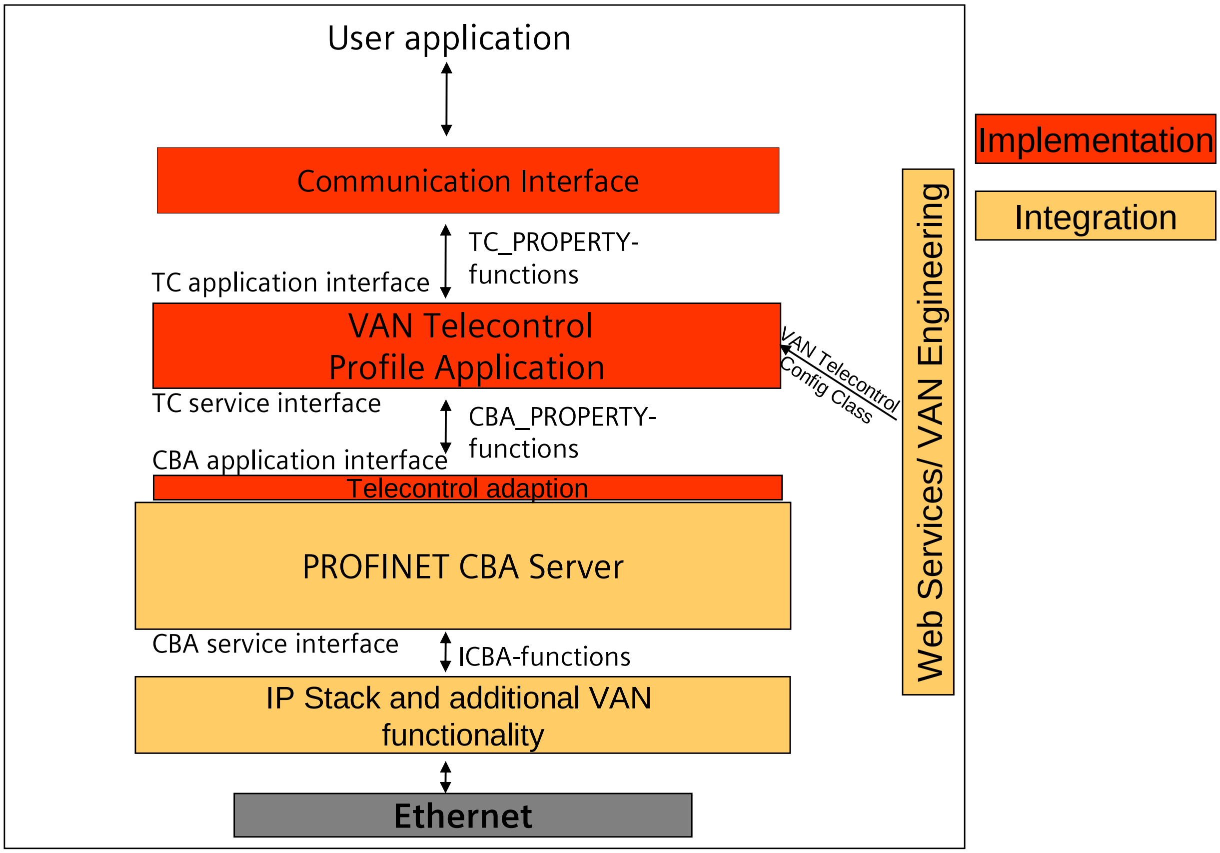

Figure 3-1 Telecontrol Profile Stack Overview

The PROFINET CBA Server was already described in chapter 2.7.3 of [D07.3-1]. Different Runtime versions were used for Windows and Linux. The Windows version is using the PROFINET CBA Runtime Version 2.01 and the Linux version is using the PROFINET CBA Runtime Version 2.20. This has the effect that the runtimes in conjunction with the different operating systems have slightly different behaviours. But the user of the Telecontrol Profile will not notice most of the differences because they are caught by the Telecontrol Profile Application which provides the same application interface for Windows and Linux. Starting the CBA Server is different according to the operating system. More Information can be found in chapter 3.1.2 and 3.4.1.

The Telecontrol Profile Application is the heart of the Telecontrol Profile. Figure 3-2 shows the Telecontrol Profile with the difference between the two releases.

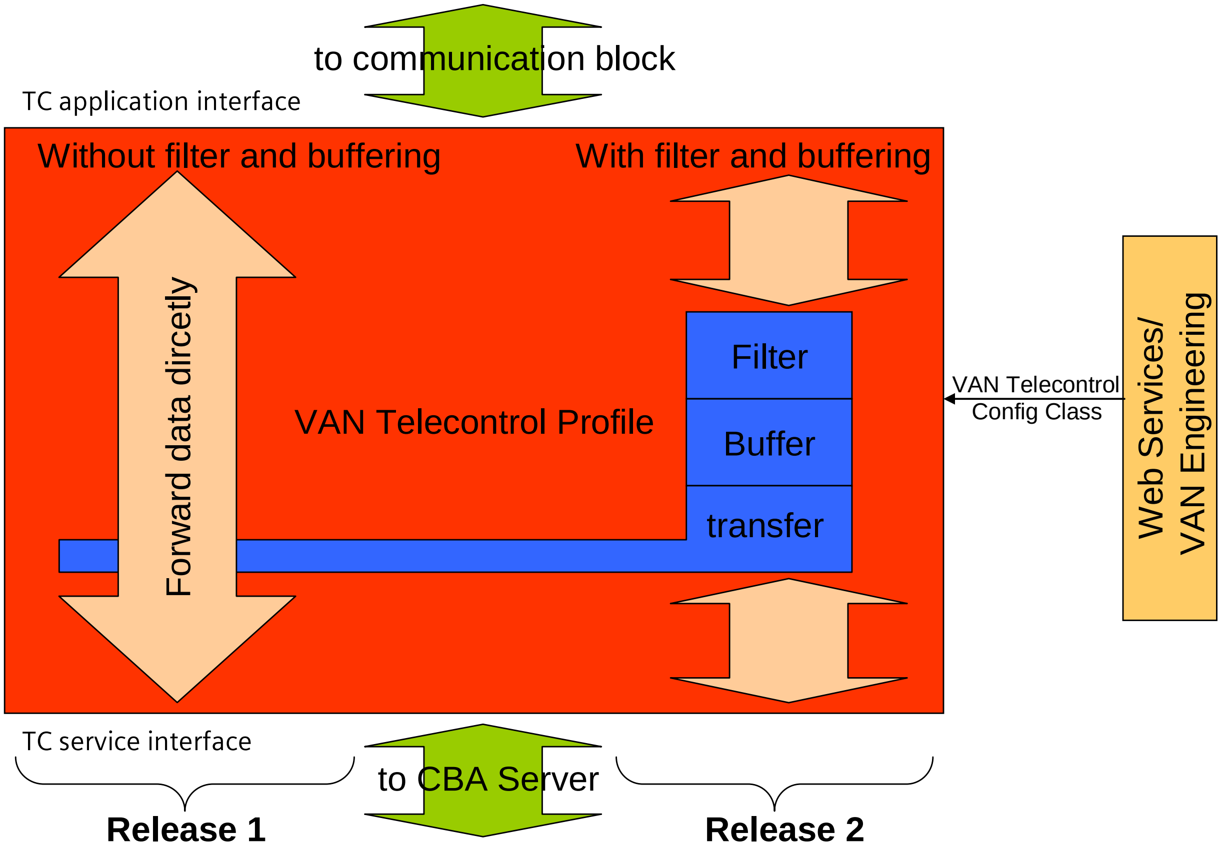

Figure 3-2 Telecontrol Profile releases

Release 1 supports the variable services. This means that the data will be directly forwarded from the Communication Interface to the CBA Server without buffering and filtering. The Telecontrol Config Class of the Device Config ASE [ASEMAN] configures the application. Basically this means handing over the property information which are used by the Telecontrol Profile to create the properties at the CBA Server. The Telecontrol Profile release 1 is available for Linux and Windows.

The Telecontrol Profile release 2 extends the previous version by the real Telecontrol functionality, the buffer and filter described in chapter 2.7.2 of [D07.3-1]. They are categorised to mandatory and optional to specify which buffer und filter are available in the Telecontrol Profile release 2. They are configured by the Telecontrol Config Class. The important attributes are explained in chapter 4.1. The Telecontrol Profile release 2 is available for Windows.

The structure of the Telecontrol Config Class is equal for both releases. But the release 1 ignores the parameters in the attributes concerning the buffer and filter settings. Thus the VAN Engineering will not differ between release 1 and release 2.

The Communication Interface provides the application interface to the Telecontrol Profile. To access the properties this interface has to be used. For Windows a DLL and for Linux a LIB are provided. In chapter 3.1.2 the interfaces are presented.

3.1.2 Interfaces

3.1.2.1 Telecontrol Config Class

The VAN Telecontrol Config Class is part of the VAN Device Config ASE [ASEMAN] and provides the Telecontrol Profile with all information set by the VAN Engineering. Because of experiences during the implementation it was necessary to make some changes to the Telecontrol Config Class defined in chapter 5.1 of [D07.2-1]. New attributes were introduced and the structure was slightly overworked to fit to the requirements of the VAN Engineering. The following new attributes were introduced:

Table 3-2 Telecontrol Config Class, new attributes

Attribute	M-Mandatory O-Optional	Data Type	Description
property-name	M	String	This mandatory attribute describes the property name which specifies the property at the CBA Server
physical-device-address	O	String	This attribute specifies the physical device address used by the Telecontrol profile and the CBA Server
provider-information	O	*element*	This element contains the information of the provider object. If this information is available, the consumer tries to build up a connection to the provider object
• provider-pdev	M	String	This attribute specifies the physical device name of the provider
• provider-ldev	M	String	This attribute contains the logical device name of the provider
• provider-rtauto	M	String	This attribute specifies the RTAuto of the provider
• provider-property-name	M	String	This attribute contains the property name of the provider application object

The following change was made according the structure of the Telecontrol Config Class: The *list-of-telecontrol-objects* element was added above all other attributes. One *list-of-telecontrol-objects* element contains one Telecontrol property; more than one element is possible. This means that for e.g. n-Telecontrol properties n-*list-of-telecontrol-objects* elements exist. The VAN ASE Manual shows the complete VAN Telecontrol Config Class.

An overview of the necessary attributes of the Telecontrol Config Class can be found in chapter 4.1.5.

3.1.2.2 Communication Interface

The Communication Interface allows the user application to access the properties of the Telecontrol Profile. The following functions are available:

- RegisterCPCallback
- RegisterNewProperty
- CPWriteValue,

and callbacks for the following events notify the user about changes:

- New value of a property
- Property disconnected

A detailed explanation of the functions and examples can be found in the Telecontrol Tech-Note on the [VAN Groupware].

3.1.3 Differences between Windows and Linux Version

The difference between Windows and Linux regarding the functionality of the Telecontrol Profile, CBA Server and Communication Interface is described in this chapter.

CBA Server

The available porting of the CBA Server for Linux and Windows are based on different Runtime Versions. The Windows porting is using the PROFINET CBA Runtime Software Version 2.01, whereas the Linux porting uses the PROFINET CBA Runtime Software Version 2.20. The basic functionality between these two Runtime versions is identically, but the porting was realised for different operating system which causes slightly different behaviours.

Because the Telecontrol Profile provides the CBA Server handling by offering a common interface, the user is only affected by some differences according the operating system. Differences according the CBA Server are mostly handled by the Telecontrol Profile, which causes different outputs at the console.

For Windows drivers and settings are needed before starting the CBA Server, whereas no settings are needed for the Linux version except that a srt driver has to be started each time before the CBA Server is started.

Communication Interface

The parameters of the callback functions are different for the Windows and the Linux version. Apart from this the functionality is the same.

Operating System

Due to the operating system, the directories for the applications are different. These locations are described in chapter 4.1.2. Also the steps to start the applications differ.

Behaviour & Handling

The behaviour and handling differs extremely between the Windows and Linux version. While the Windows applications are all running correctly without any errors, the Linux version of the CBA Server caused some problems which could not be fixed within the VAN project. More information can be found in chapter 3.4.1.

3.2 VAN QoS Monitoring

This chapter is based on former documents of WP7 especially [D07.3-1] chapter 2.3 VAN Peer-To-Peer ASE - QoS Monitoring.

3.2.1 Clock Synchronisation

The clock synchronisation is precondition for reasonable results of the QoS monitoring measurement and post processing calculations.

NTP – ***N****etwork* ***T****ime* ***P****rotocol* – is defined in [RFC 958] respectively for NTP v3, in [RFC 1305]. The NTP-process delivers the time as UTC (***U****niversal* ***T****ime* ***C****oordinated*) and synchronises the local clock. The principle of the local time synchronisation is shown in Figure 3-3.

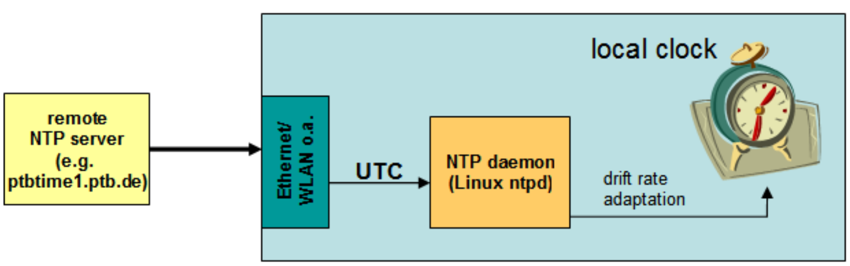

Figure 3-3 Principle of time synchronisation via NTP

The accuracy of NTP is given by the UTC-seconds. Accuracy up to 10 ms is achievable. The access to a (remote) NTP-Server is necessary; UDP-Port 123 must be open.

GPS – **G**lobal **P**ositioning **S**ystem

Beside the generally well known GPS-position data the GPS signal also contains UTC data. Furthermore there are GPS receivers providing a PPS-signal (Pulse Per Second). Depending on the receiver the accuracy of the PPS-signal can be in the µs range.

GPSD (GPS-daemon - Linux) is a service daemon that monitors one or more GPS sensors attached to a host computer through serial or USB ports, making all data on the location/course/velocity of the sensors available to be queried on TCP port 2947 of the host computer. With gpsd, multiple GPS client applications can share access to GPS information without stepping on each other.

Here we use the functionality of GPSD to extract the time-signal and PPS-signal from the GPS-data and to use it as local time source for the NTP-service (NTP daemon). The principle of the local time-synchronisation is shown in Figure 3-4.

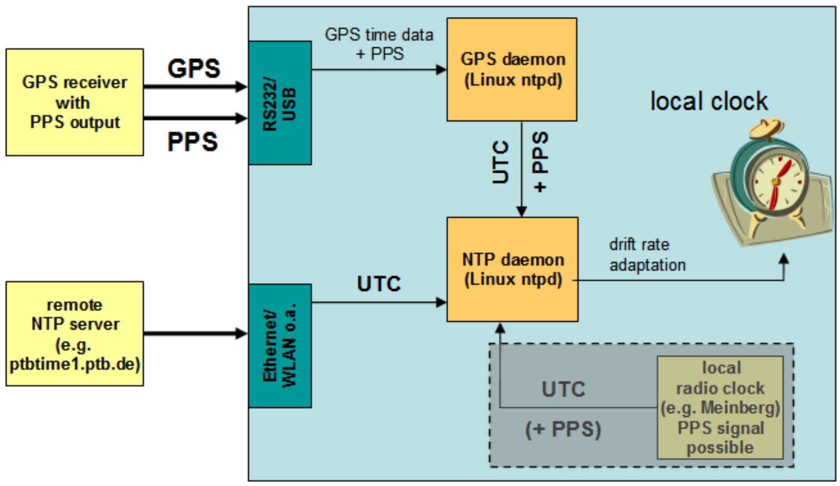

Figure 3-4 Principle of time synchronisation via GPS and NTP

Used Hardware

Figure 3-5 Garmin GPS 25-LVC

In Europe it is not easy to get a GPS receiver with a PPS output. We use the embedded device Garmin GPS 25-LVC.

3.2.2 Software Overview

Operating System

The VAN QoS monitoring prototype implementation was implemented on the Linux-Distribution Ubuntu 8.04.

Software start: as all ASE objects this object should be started after the object registry is running.

The measurement process (not the measurement object!) itself will be started by the VAN switching process by setting the respective attribute (see Interface with switching).

ASE object definition

The valid status of ASE object definition for the prototype implementation is written in the [ASEMAN]. This version replaces the former description written in [D07.2-2] chapter 6.3.

The main difference compared to [D07.2-2] is that the described structure has changed, now 2 classes are defined: the QoS Monitoring and the QoS Measurement. The QoS Monitoring stores the needed general information for measurements related to one line, mainly the configuration of the measurement client and server. The QoS Measurement defines the details for the measurement (also related to one line) and also keeps the attributes that store the measurement results.

QoS-Monitoring software structure

Figure 3-6 shows the structure of the software. The process *eu.ifak.van.wp7.p2p.qos.app* is the application for starting the VAN Monitoring- and Measurement-ASE objects. Thus with its parameterised invocation it establishes instances of the QoS-Monitoring class (*QoSMonInfo*) and the QoS measurement class (*QoSMeasInfo)*. These instances are embedded in a separate process the *eu.ifak.van.wp7.p2p.qos.* Additionally this process also contains a *QoSIOCThread* object for the inter object communication (IOC) with other VAN objects at the same device (especially for the VAN switching).

The *eu.ifak.van.wp7.p2p.qos.model* process contains the data model of the QoS-Monitoring and stores the attribute sets. According the valid definition (see above subchapter ASE object definition) a set of attributes consist of:

- server/client
- line-id
- line-name
- endpoint-address
- receiving-data-port
- time-source
- time-accuracy.

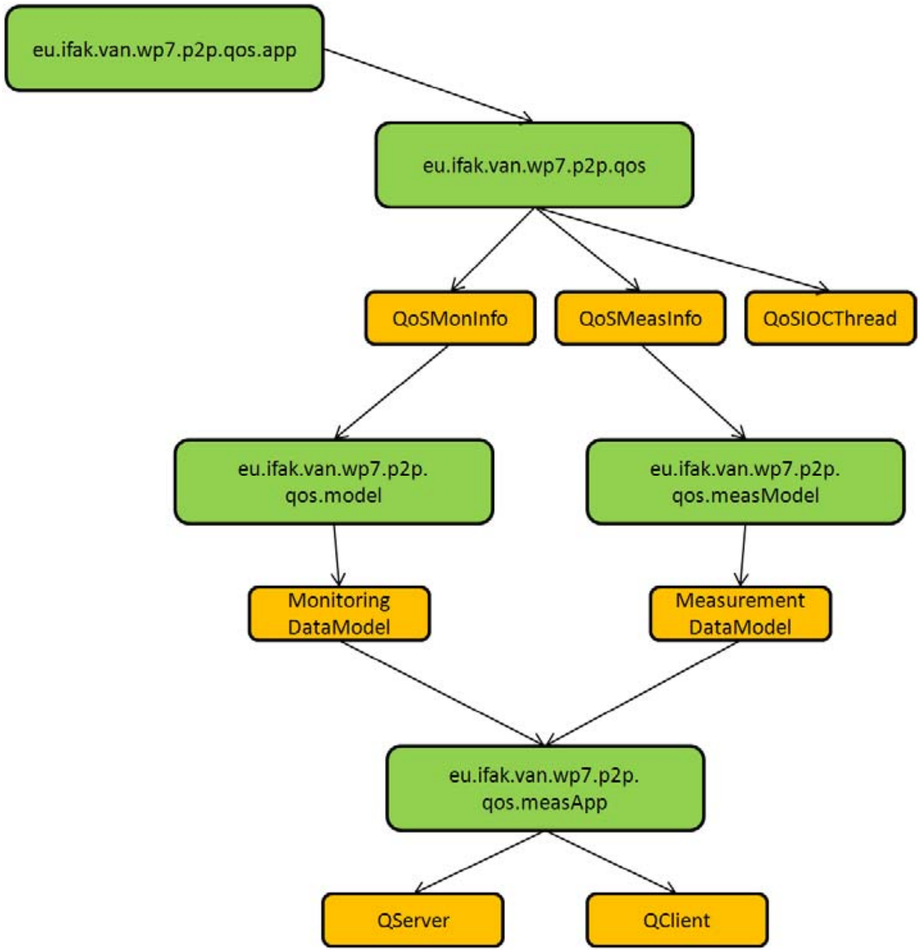

Figure 3-6 QoS-Monitoring software structure (green: process, yellow: class)

In the same way the *eu.ifak.van.wp7.p2p.qos.measModel* process contains the data model of the QoS-Measurement and stores the attribute sets. According the valid definition (see above subchapter ASE object definition) a set of attributes consist of:

- test-stream-parameters
    - operational
    - line-id
    - measurement-direction
    - measurement-mode
    - number-of-packets
    - planned-bandwidth
    - packet-size
    - inter-frame-gap
    - inter-burst-gap
- used-bandwidth
- overall-bandwidth
- measurement-results:
    - time-of-last-measurement

o current-measurement-number

o mean-free-bandwidth

o current-free-bandwidth

o mean-latency

o current-latency

o mean-latency-jitter

o current-latency-jitter

o mean-packet-loss-rate

o current-packet-loss-rate

o mean-duplication-rate

o current-duplication-rate

o mean-wrong-packet-sequence-rate

o current-wrong-packet-sequence-rate

o mean-bit-error-rate

o current-bit-error-rate

The application process *eu.ifak.van.wp7.p2p.qos.measApp* covers the actual measurement functionality. Thus – according its configuration - it is realised by the client (QClient) or the server (QServer) of a measurement.

3.2.3 Measurement details

Sequence of one QoS-Monitoring-Measurement (see also Figure 3-7**)**

1.) Pre-consumption a Server instance with the respective configuration on an VAN-AP and a Client-instance at another VAN-AP is started.

2.) Pre-configuration via Engineering.

3.) Final configuring and activating of the servers via WS-Request by VAN Switching. The server starts and waits for an incoming request at the configured port.

4.) Final configuring and activating of the Client by VAN Switching (using IOC). The Client sends an initial request to its server.

If the initial request is answered correctly the measurement starts according its configuration in initial or online measurement-mode and with the configured measurement direction (client/server or server/client).

5.) In case of an initial measurement-mode the measurement stops until the latency starts to exceed. In case of an online measurement-mode the measurement must be stopped by setting the operational attribute to zero (can be done by an VAN engineering or by the VAN switching).

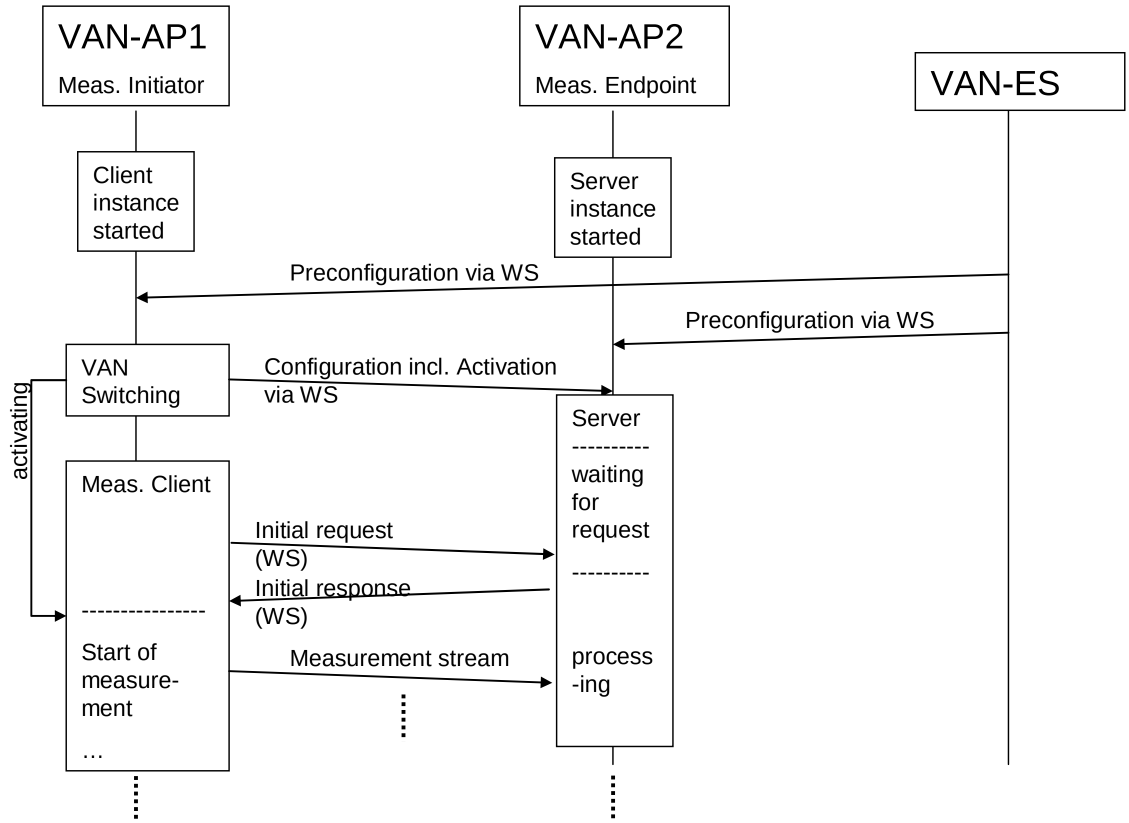

Figure 3-7 Sequence example for a start of a QoS measurement

Measurement data stream

The measurement consists of bursts and a defined inter-burst-gap – the latter is the time between the end of a burst and the start of the following next burst. A burst itself consists of a defined number n of packets with a defined size.

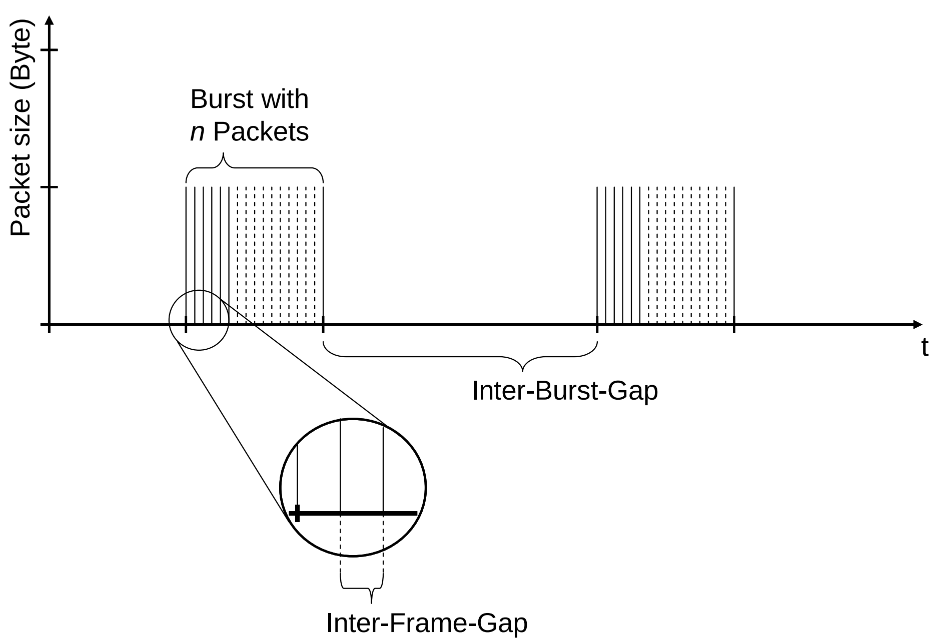

Figure 3-8 Principle of the burst based QoS monitoring measurement

The size of packets within one burst is always constant.

In case of the initial-measurement the size of the packets will be enlarged with each burst until the latency starts rising up. Then the measurement will be finished. And the measurement results will be stored in the defined attributes.

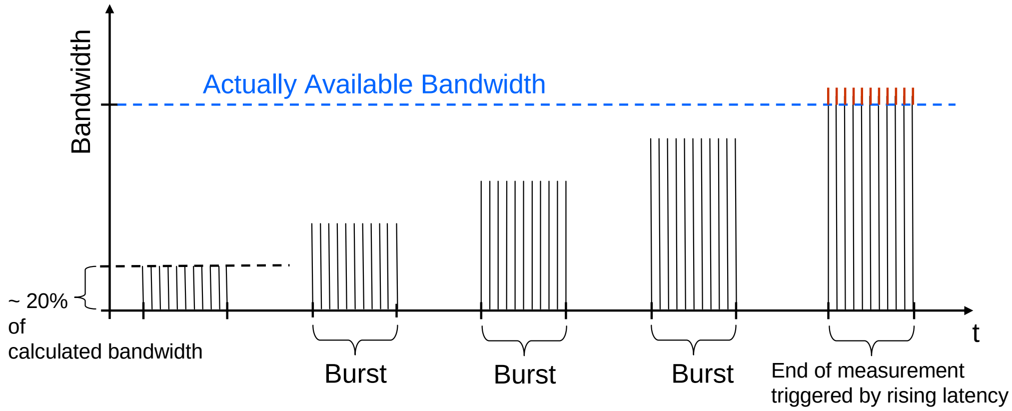

Figure 3-9 Principle of an initial-measurement

The online-measurement is running together with the process data stream. Thus its size depends on the size of the entire process data stream. A reserve of about 10% to the maximum bandwidth is recommended.

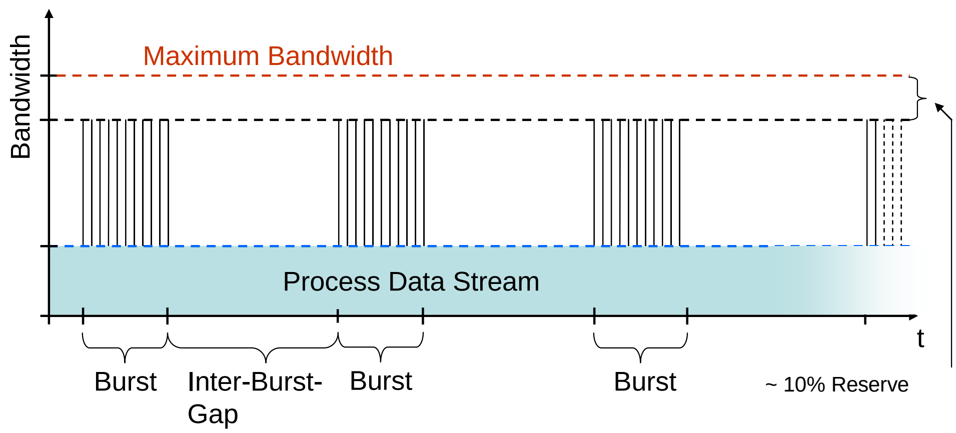

Figure 3-10 Principle of an online-measurement

3.2.4 IOC communication

The IOC communication provides an interface especially for the VAN switching:

As already stated above the access for the VAN switching (as for any other process at the same device) to the attributes of the VAN QoS measurement is handled via Inter Object Communication (IOC). For this the IOC port of the QoS monitoring must be known, it is requested by a WS request.

All messages for the IOC are defined the same as for (external) WS communication.

The allocation - which measurement object belongs to which monitoring object and to which switching object - is realised by setting the Line-ID attribute (see Figure 3-11). The switching has to poll all QoS measurement objects.

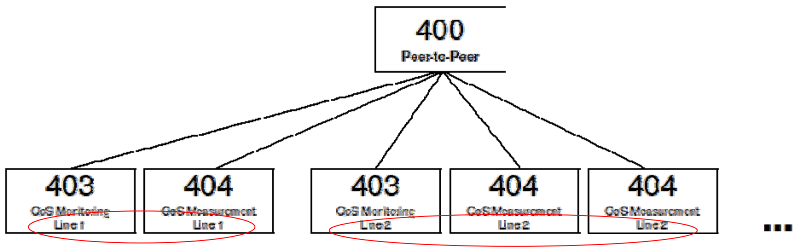

Figure 3-11 The allocation of measurement to monitoring objects is done by their Line-ID

The IOC communication is restricted to QoS measurement objects only, QoS monitoring objects can not be accessed via IOC.

For easier access to the attributes there are blocks defined which can be requested separately. There are 4 blocks (see Figure 3-12):

- test-stream-parameters block
- used-bandwidth
- overall-bandwidth, and
- measurement-results

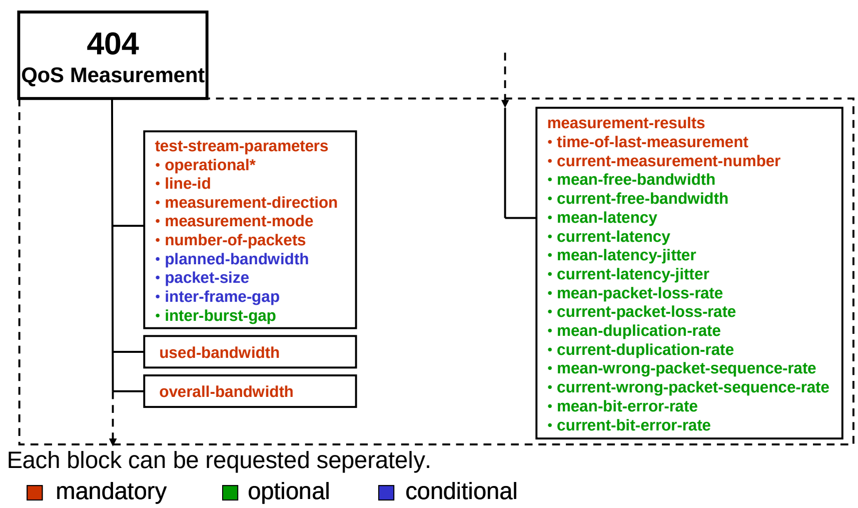

Figure 3-12 Definition of blocks for an optimised data access

Below there is an IOC example for a request and a response message for the test-stream-parameters.

```
<ns2:test-stream-parameters xmlns:ns2="http://van-eu.eu/peertopeerqos_measurement"
xmlns:ns3="http://van-eu.eu/peertopeerqos

_measurement" xmlns:ns3="http://van-eu.eu" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:nil="true"/>
```

Figure 3-13 Example: IOC request for the test-stream-parameters block

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<ns3:test-stream-parameters xmlns:ns2="http://van-eu.eu/peertopeerqos" xmlns:ns3="http://van-eu.eu/peertopeerqos_measurement" xmlns:ns4="http://van-eu.eu"> inter-burst-gap="10" inter-frame-gap="10" packet-size="100" planned-bandwidth="300" number-of-packets="100" measurement-mode="1" measurement-direction="server-client" line-id="lineID_0"/>
```


Figure 3-14 Example: IOC response for the test-stream-parameters block

The following attributes of a measurement object must be set by the switching and will not be configured by the engineering:

- Operational
- Measurement-mode
- Used-bandwidth

All others must be set by the VAN engineering. If there is no overall-bandwidth value it must be set to the default value of zero. If it is set to zero than an initial-measurement will overwrite this value after finishing its measurement. If the engineering configures a value other than zero this value will never be overwritten by an initial measurement.

3.3 VAN Provider Switching

This chapter describes the functionality of the switching application which is the responsible unit to switch to an alternative communication connection if the required QoS is not longer acceptable. The switching process has a strong relation to the QoS Monitoring and uses the IOC (Inter Object Communication) for cross communication to corresponding ASEs.

3.3.1 Switching Software Functionality

The switching application is responsible to change the tunnel connection over different communication channels if a switching event is given. The event is triggered if the current QoS values are beneath the pre-defined QoS parameters on the VAN device. The source for switching information is normally the VAN Quality of Service (VAN QoS) application as part of the Peer-To-Peer ASE. Other triggers, like manual switching, are also possible and supported. The mentioned process takes place only at VAN AP (VAN Access Point). The reason for this is that here the VAN network gets in touch with the public network. From this point the communication services become unpredictable because of the loss of direct influence to the QoS. The relation of the switching application to other systems on an access point is shown in Figure 3-15.

The Switching ASE defines the trigger values in regard to different connections. The switching application observes via the QoS software the actual running connection. If the quality of service values under-run one trigger value, a switching process will be started. This process checks the possible alternative connections with the QoS software. If the conditions of the quality of service are met, several methods to realise a switching are possible. Within this work package the advantages and disadvantages of possible solutions were discussed and a practicable one is chosen.

As described in the Deliverables [D06.4-1], [D06.5-1] and [D07.2-2] on a VAN Access Point an openVPN server and client for every tunnel segment of the cascaded tunnel is connected by a bridge. This bridge includes the TAP devices of both openVPN applications. This allows transferring the data in a secure way from one tunnel segment to the next tunnel segment. At this point switching is integrated. The switching application uses the operating system command "*route*" to change the routing tables of the OS. The old route from TAP to the NIC of the obsolete connection will be deleted and a new route from TAP to the NIC of the new connection will be added. On the receiving side of the tunnel the openVPN configuration has to be set to "*float*". This allows the change of the connection without influence to the tunnel. To change the VAN address resolution a VAN DNS server for the VAN Access Points is set. To update the addresses an encrypted *nsupdate* is used. This process is described on Figure 3-15.

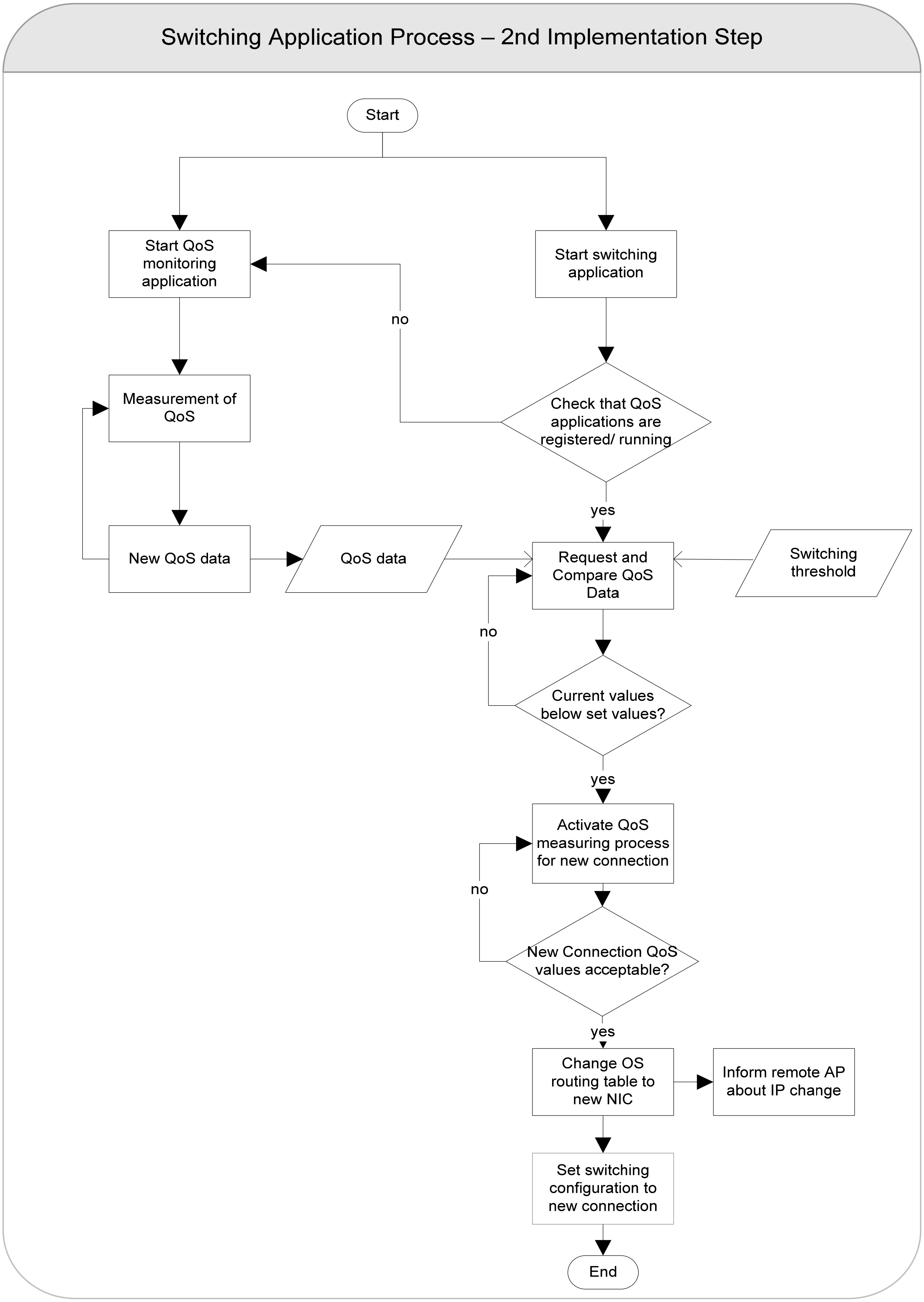

Figure 3-15 Switching Application Process

3.3.2 Software Overview

The switching software runs in different threads. One of the threads is the ASE thread. It is connected to the object registry of the web server. If the data of the ASE are changed, for instance via the engineering tool, another thread - the ASE data class - will be adapted to the new data. The Switching class, running on an independent thread will periodically pull actual information from the QoS software and compare these to the switching ASE data. If the QoS value leaves the switching threshold a switching event occurs. Then the QoS software for the alternative connection will be activated. If the according QoS values are in order, the route of the tunnel will be changed.

In order to this in a generic way the software consists of the following primary packages.

- The *manager* package contains the necessary managing function and mainly serves as start routine.
- The *ase* package is responsible for the access to the local VAN web service server. This implementation is based on the WP2 software especially the registry packages. Asedata and jaxbDataTypes handle the access to data of the Switching ASE.
- The package *switching* contains the decision logic for the switching process.
- The other packages support this main package and different testing purposes.

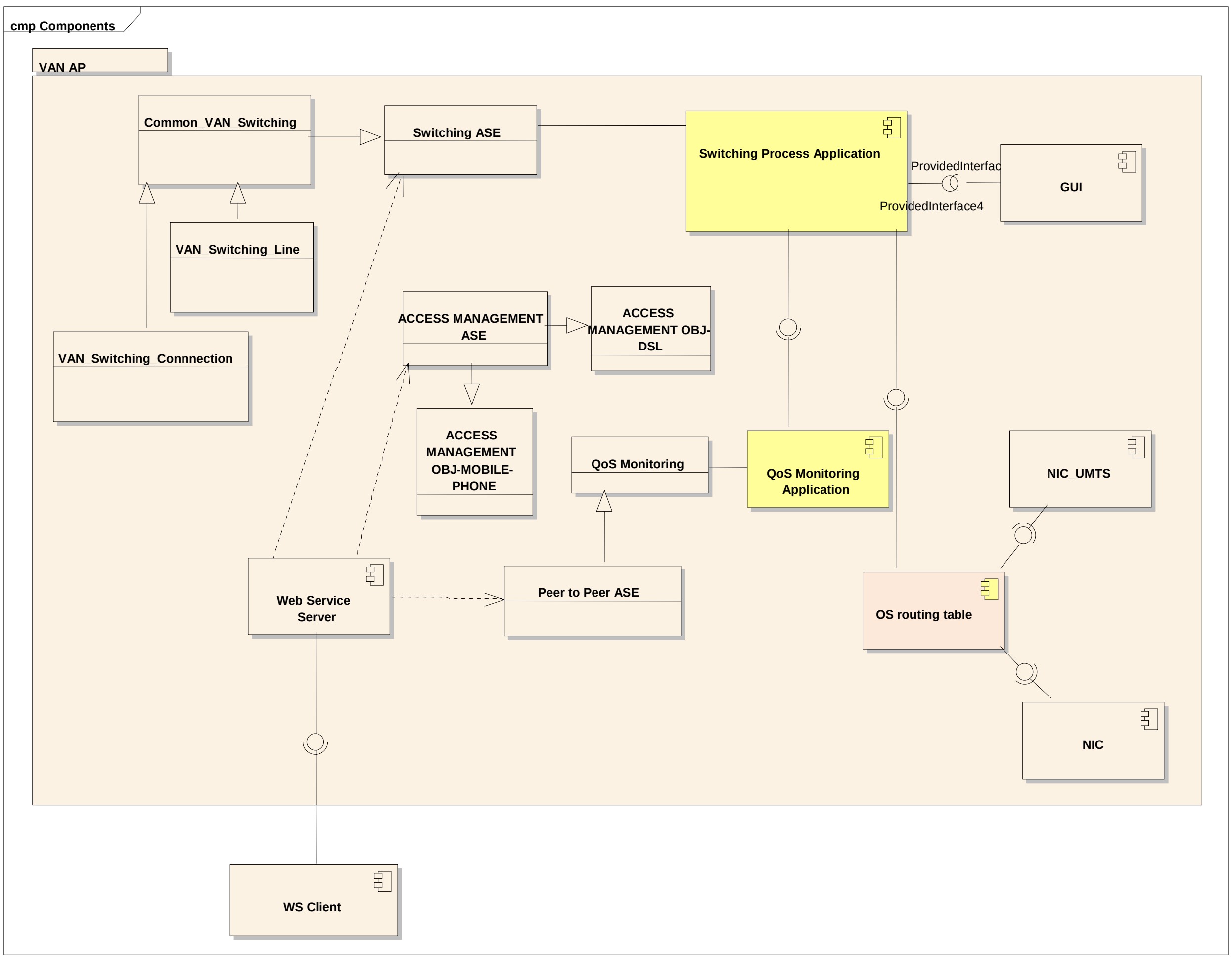

Figure 3-16 VAN Class Relationship

Operating System

The VAN Switching prototype is implemented for Java 1.6 on the LINUX Distribution Ubuntu 8.04.

The switching application starts together with all other ASE objects after the registry is running. Then the switching starts the measurement process as part of the registered VAN QoS monitoring object.

ASE object definition

For the prototypical implementation the following Switching ASE attributes of the object definition written in the [ASEMAN] are used:

- Delay
- Bandwidth
- Threshold

Software Structure

The Figure 3-16 shows the class structure of the software. The *SwitchingASEApp* is responsible for registration with the object registry and administration of the ASE objects. With this class the ASE objects will be registered and access to these objects is handled.

The *GUI* serves as an interface for testing purposes. Here the available data of the ASEs and the switching processes are shown.

The Task of the *Manager* is to function as a starting point for the whole provider switching application. Further on all processes of this application which are independent of another are started and managed here.

The *Switching* realises the logic for the switching process. It starts an independent thread and requests and handles the input of the different thresholds. If necessary it triggers a switching event.

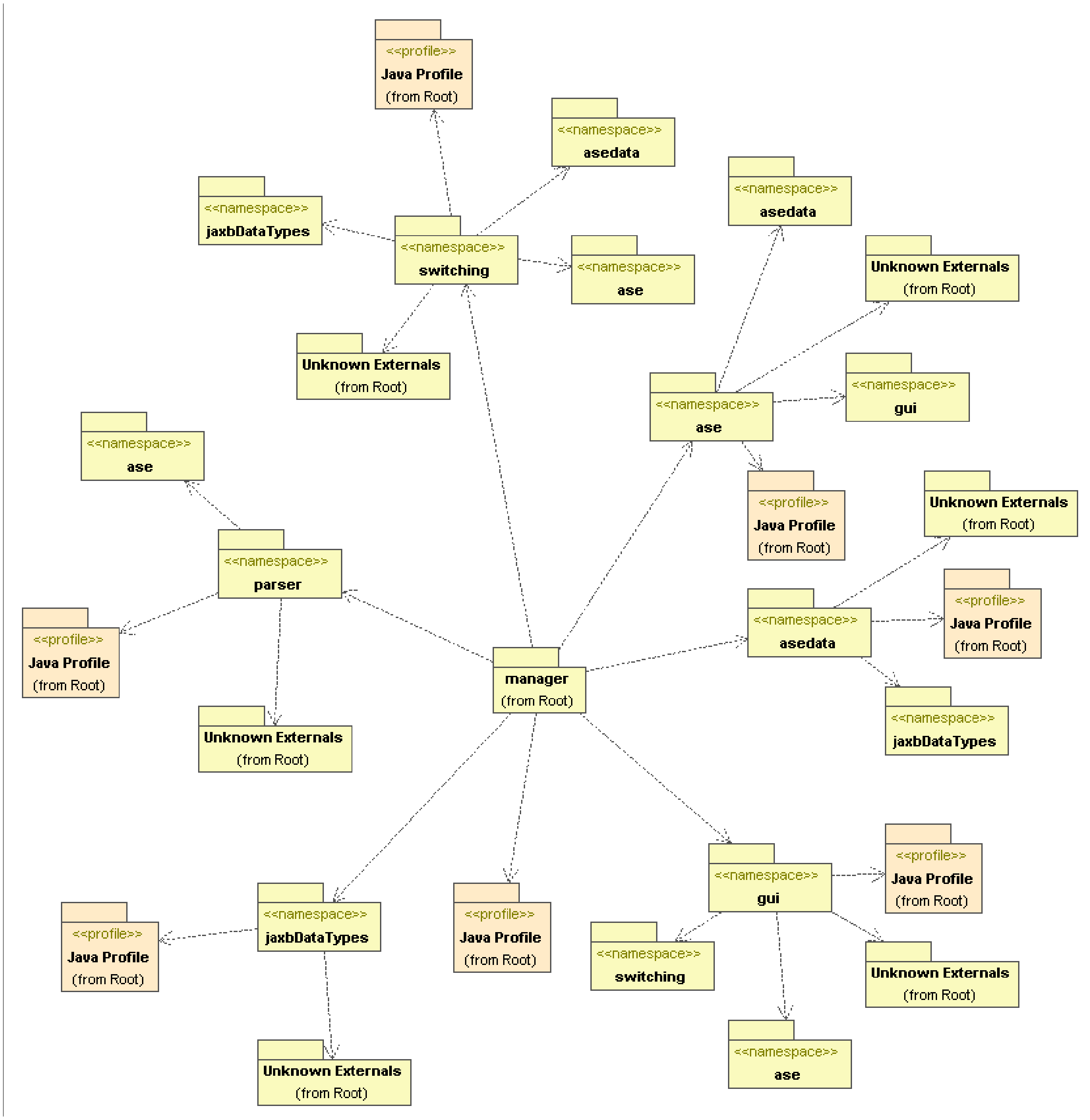

Figure 3-17 Switching Packages

3.3.3 Interfaces

Primary interface is the webservice interface. This is provided with the WP2 implementation. The description of the software is done in Deliverable [D02.2-1]. All communications with other VAN application are realised via this webservice interface, especially the inter object communication (IOC).

As soon as the IOC is established by the switching application, a request for a socket port will be send. By the means of this port all further data exchange will be performed. This will be done by the switching application in regard to QoS software.

Another interface is the use of an operation system command to change the routing tables of the used NICs. This will be done via the *route* command.

3.4 Integration Aspects for WP7 related Functionality

Some software parts of work package 7 are more tightly integrated as the pure VAN context dictates and all use non-VAN-dependant applications to fulfil there functions. These aspects will be further looked at in this chapter and all related problems and the solution to this will be recorded here.

All VAN applications have to be integrated to the ASE object registry structure and functions. The Telecontrol Profile uses the CBA server and the Quality of Service software uses clock synchronisation mechanism of NTP and GPS. The provider switching depends on the VAN QoS and uses OS commands.

3.4.1 Telecontrol Profile

The functionality of the Telecontrol Profile is already described in chapter 3.1 and an installation and user guideline can be found in chapter 4.1. This chapter will give information about the experience and problems which occurred during integrating the Telecontrol Profile release 1 on the machines used for the Manufacturer Industrial Experimental Setup.

The Telecontrol Profile is running on three VAN Automation Devices and two Proxy Devices. Both Windows and Linux operating systems are used which results in different approaches.

Windows

The Windows versions of the PROFINET CBA Server and the Telecontrol profile worked as expected and no problems occurred during the integration into the Factory IES.

Linux

The following problems occurred during working with the Linux version of the CBA Server:

- Linux does not support DCOM and only the internal DCOM of the PROFINET CBA Server was realised, which means that some functions can not be used locally on a Linux machine. This was notified during implementing the functionality to build up the connection between a consumer and provider property which is always built up by the consumer. But these functions are not available under Linux which means that the [PNTestTool] on a Windows machine has to be used for all Linux consumer properties to build up their connection to a provider.
- When the connection between a consumer and provider property is interrupted, the CBA Server tries to re-establish it. This does not always work properly for the Linux version of the CBA Server. To avoid the problem a workaround has to be used. For the Factory IES an additional Virtual Box [VB] with Windows is running on the Linux machine. On this machine, the Telecontrol Profile with the CBA Server is running as a Proxy (Figure 3-18).

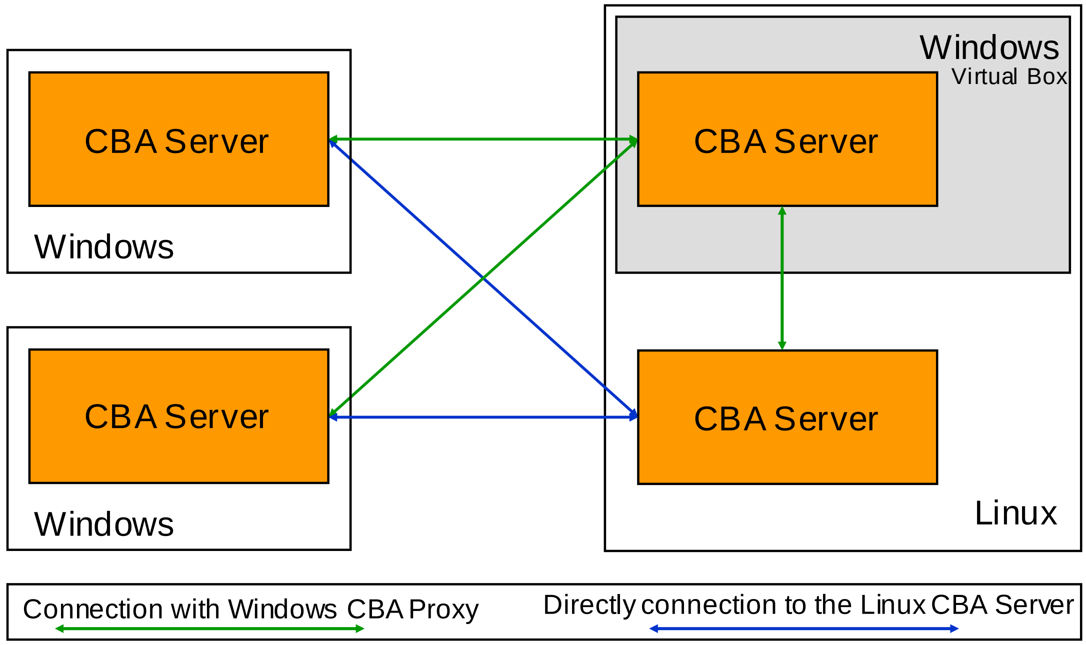

Figure 3-18 Windows CBA Server Proxy

The Telecontrol Profile on the Proxy machine is used to configure the CBA Server. The properties are all configured as CONSUMER, but they will also work as PROVIDER. The following figure describes this:

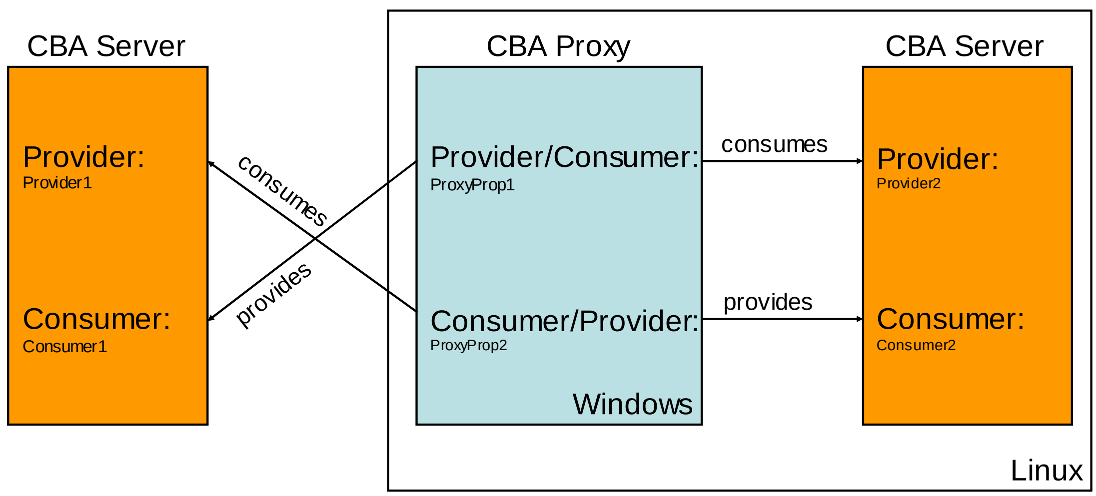

Figure 3-19 Proxy CBA Properties

The Proxy CBA Server (P-CBA) has to provide and consume the properties of the Linux CBA Server (L-CBA). On the other side, the P-CBA also has to consume and provide the properties of the remote Windows CBA Server (W-CBA). To clarify this according to Figure 3-19:

- *ProxyProp1* of P-CBA consumes *Provider2* of L-CBA and provides this property to *Consumer1* of W-CBA.
- *ProxyProp2* of P-CBA consumes *Consumer1* of W-CBA and provides this property to *Consumer2* of L-CBA.

Now the connection between W-CBA and L-CBA is no longer affected by the network and in case of any disconnection always the connection to the P-CBA Server is concerned.

These behaviours were caused be the prototypically system adaption of the PROFINET CBA Server (no product!) to Linux. To improve the server was not in scope of the VAN project, thus these solutions were chosen.

4 Installation and User Guideline

The required steps to install the previous described applications prototypically implemented by WP7 are explained in this chapter as well as the guidelines to use the applications. The regarding components are the VAN Telecontrol Profile, the VAN Provider Switching and the VAN QoS Monitoring.

4.1 VAN Telecontrol Profile

This chapter explains the steps to use the Telecontrol Profile on Windows and Linux. The following prerequisites exist:

- Operating System: Windows XP or Linux (tested with Ubuntu 8.04)
- VAN SW Stack has to be installed and has to run
- VAN Telecontrol Profile settings (see chapter 4.1.3)

The next chapter explains the structure of the Telecontrol package which can be downloaded from the VAN Groupware [VAN Groupware] under "/HOME/WP7/Telecontrol profile". The package is structured as followed.

4.1.1 Telecontrol Package

Windows

- Telecontrol profile:
  - Filename: TCprofile.exe
  - File location: [CommonDataFolder]/VAN/Telecontrol
- CBA Server:
  - Filename: CBAServer.exe
  - File location: [CommonDataFolder]/VAN/Telecontrol/CBA
  - driver:
    - profinetrtpst contains the Proxy/Stub Dll
    - physicalDevice: server for the physical device interface
    - pcSRTdriver: PN SRT driver
- Communication Interface:
  - Filename: ApplicationInterface.dll
  - File location: [CommonDataFolder]/VAN/Telecontrol/data
  - Comment: A project which is using the functions of the Communication interface is as an example included.
- XML files with Telecontrol Config Class information:
  - Filename: TelecontrolConfig.xml
  - File location: [CommonDataFolder]/VAN/Telecontrol
  - Comment: The Telecontrol profile always uses the xml file which is named TelecontrolConfig.xml. If this file isn't available, the Telecontrol profile won't start correctly.

Linux

- Telecontrol Profile:
  - Filename: TCprofile
  - File location: /var/opt/VAN/Telecontrol/
- CBA Server:
  - Filename: pn4l_srtdriver & pn4l_srtserver_deb
  - File location: /var/opt/VAN/Telecontrol/CBA/
- Communication Interface:
  - Filename: libApplicationInterface.so

    - File location: /lib
    - Comment: The ApplicationInterface.h needs to be included when another .cpp file wants to use functions of the libApplicationInterface.so. Thus, the ApplicationInterface.h is also part of the package.
      A project which is using the functions of the Communication interface is as an example included.
- o XML files with Telecontrol Config Class information:
    - Filename: TelecontrolConfig.xml
    - File location: /var/opt/VAN/Telecontrol
    - Comment: The Telecontrol profile always uses the xml file which is named TelecontrolConfig.xml. If this file isn't available, the Telecontrol profile won't start correctly.

4.1.2 Folder options

The following folder is defined to be used under Windows for the Telecontrol profile:

[CommonDataFolder]/VAN/Telecontrol

This is equal to the following folder of an English version of Windows:

C:/Program Files/VAN/Telecontrol

For Linux the /var/opt/Telecontrol folder is used for the Telecontrol files (Telecontrol Profile, CBA Server and additional data). The Communication Interface (Shared Object) must be part of the /lib folder.

4.1.3 Default Settings

In order to install the PROFINET CBA Server in a WINDOWS XP environment, the following steps need to be performed.

Install Drivers

The driver from the path [CommonDataFolder]/VAN/Telecontrol/CBA/driver/pcSRTdriver has to be installed. An installation manual is included in this folder.

In order to run PROFINETWin32 on a Win32 computer, the separate server for the Physical Device interfaces has to be installed and registered before a PROFINET server can be started. This service can be found at [CommonDataFolder]/VAN/Telecontrol/CBA/driver/physicalDevice/regpdev.bat.

Finally, the Proxy/Stub DLL *profinertpst.dll* must be registered on the client computer. This is accomplished by running the batch file *RegProfinetrtDll.bat* in the folder [CommonDataFolder]/VAN/Telecontrol/CBA/driverprofinetrtpst.

Now it is possible to start the PROFINET CBA Server by executing the CBAServer.exe in folder [CommonDataFolder]/VAN/Telecontrol/CBA.

Note:

- *Skipping one of the steps above can result in nasty error messages.*
- *For Windows SP2 and SP3 special DCOM settings are necessary. The information can be found in the Appendix of [D07.3-1].*

For Linux no special preparations are necessary for the CBA Server. Only the applications should be copied to the according directories.

4.1.4 Application Operating Mode

4.1.4.1 Starting the applications

The sequence of starting up the applications is specified:

1. parameterise the TelecontrolConfig.xml file
2. start the Telecontrol profile (Windows: TCprofile.exe or Linux: TCprofile, parameter specified in chapter 4.1.5)
3. start the CBA Server:
    - Windows: CBAServer.exe
    - Linux: pn4l_srtdriver & pn4l_srtserver_deb
4. connect with the Communication Partner to the Telecontrol profile (e.g. SoftPLC, User Application)

4.1.4.2 Telecontrol Profile Parameter

The Telecontrol Profile can be started up with arguments to configure it. These arguments are optional and if they are not used the Telecontrol Profile starts up with the standard configuration. The following arguments are supported:

- **-o** value define the output messages. The value specifies the different options and has the following syntax:
    - o **value** 8 bit value with every bit defining a special functionality

Bit	7	6	5	4	3	2	1	0
Meaning	Log	Buffer	Filter	ASE	IDL	Socket	Main	Connection

  To enable the output message of the different functionalities the related bit has to be set. Furthermore if the Log-bit is set the output will also be stored to the local file named *Telecontrol.log*.
  The following example shows how to use the arguments:

    - TCprofile.exe –o 159 **10011111** activates all output functionality except the Buffer and Filter output. This provides all available output for the Telecontrol profile release 1.
    - TCprofile.exe –o 143 **10001111** this is the standard setting for the Telecontrol Profile release 1 and activates everything accept the ASE functionality.
    - TCprofile.exe –o 255 **11111111** activates all output functionality available for the Telecontrol profile release 2.
    - TCprofile.exe –o 0 **00000000** no output

- **-c** this command will establish the connections between the consumer and provider properties. This functionality is only available for the windows version of the Telecontrol Profile. The PNTestTool hast to be used for Linux.
  When this parameter is not set, the Telecontrol Profile will not establish the connection (standard).
- **-s** this command will start the CBA Server from the Telecontrol Profile.

When the arguments are available, the Telecontrol Profile will start up according the setting. If any of the parameters isn't set, the Telecontrol Profile will use the standard settings.

Notice: Not all messages will be filtered when deactivating different bits because of some system or CBA functions printing messages to the console.

4.1.4.3 Closing the applications

The Telecontrol Profile is closed by typing **s** to the console. Then it automatically closes all internal connections to the CBA Server and the Communication Interface. For Windows, the CBA Server is also closed, for Linux the Server must separately be close by **Strg + C**.

Because the Telecontrol Profile and CBA Server are addicted to each other, if one of the applications is closed or restarted, the other one also needs to be closed or restarted.

When the Telecontrol Profile closes the connection to the Communication Interface, the callback registered for receiving messages when a provider property is disconnecting is called with the telecontrol-id -1.

The Communication Interface is independent from the Telecontrol Profile. This means that the Telecontrol Profile does not need to be restarted if the user application which uses the Communication Interface is closed. If a new partner is connecting with the Communication Interface all old connections are deleted, even if there is still a partner connected.

4.1.5 Telecontrol Config Class

The Telecontrol Config Class of the Device Config ASE provides many attributes to configure the Telecontrol Profile. Some attributes are mandatory and some are only optional. This chapter will handle the different attributes necessary to configure the Telecontrol Profile differentiated by release 1 and 2.

Telecontrol Profile release 1

The following table describes the necessary attributes to configure the Telecontrol Profile release 1 properly:

Table 4-1 Telecontrol Profile release 1 attributes

attribute	**value**	**description**	**example**
property-name	string	Name of the property	prop1
object-role	string	Object role of the property	CONSUMER or PROVIDER
data-type	string	Data type of the property	unsigned short
physical-device-address	string	IP address of the device. This attribute must be available only at the first object	192.168.1.21
If the object-role is CONSUMER			
provider-pdev	string	IP Address of the provider device	192.168.3.51
provider-ldev	string	Logical device name of the provider device	LDev_Telecontrol
provider-rtauto	string	RT-Auto name of the provider device	RTAuto_Telecontrol
provider-property-name	string	Name of the provider property	prop2

Telecontrol Profile release 2

The Telecontrol Profile release 2 also needs all the attributes already explained for the release 1, but furthermore additional attributes are necessary to configure the buffer and filter:

Table 4-2 Telecontrol Profile release 2 attributes

attribute	value	description	example
timestamp-included	bool	Defines whether a timestamp has to be generated by the Telecontrol Profile. In case, an additional property will be created at the CBA Server which value is the timestamp.	TRUE or FALSE
If the object-role is PROVIDER			
peer-object-reference	string	Object reference of the Consumer object	/dev/telecontrol
buffer-depth	unsigned8	Specifies how many values are buffered in case of connection lost	5
transfer-mode	unsigned8	Defines whether the data has to be buffered in case of communication lost	0 or 1
optional attributes			
edc-any-change-filter	unsigned8	specifies an event-driven filter on any value change to be applied or not	0 or 1
edc-threshold-filter	unsigned8	specifies a threshold filter to be applied or not. Application data is transferred as soon as a threshold condition is met	0 or 1
threshold-value-constant	same as data-type	specifies the threshold monitoring value	1234
time-driven-filter	unsigned8	This attribute specifies a time-driven filter to be applied or not	0 or 1
interval	unsigned16	specifies the time interval in ms in which application data is acquired	1234
start-time	Time	specifies the start time of the application data acquisition interval	2009-07-28T12:25:21
transfer-condition	unsigned8	Data is transmitted immediately (unconditional) or at a later point in time (conditional). The decision for immediate or later transmission is only relevant for dial-up networks. In other networks with permanent connectivity, data is always transferred immediately even if the transfer-condition is set to conditional	0 or 1

4.2 VAN Provider Switching

This chapter explains the steps to use the Provider Switching implemented on Linux platform. The following prerequisites exist:

- Operating System: Linux (only applicable OS for an VAN access point)
- VAN SW Stack has to be installed and has to run
- nsupdate tool for Linux

The next chapter presents the structure of the Provider Switching package which can be downloaded from the VAN Groupware [VAN Groupware] from the folder "/HOME/WP7/ProviderSwitching.

One base of the Switching installation is the VAN WS software stack. This means all necessary webservice applications have to be installed. The VAN WS stack consists of the VAN Service and the Routing ASE application. Another basic VAN application necessary for the function of the VAN switching is the QoS Monitoring. More details in chapter 3.3. The last base of the switching software is the operation system. In case of the access point this is LINUX. All VAN applications have to be running with the start of the switching software.

For the switching configuration the following groups of parameters are necessary:

- **Switching connection** is primary defined by the endpoints of this connection. The endpoints have to be the same as the endpoints of the according tunnel segment.
- Every switching connection has a number of switching lines. The lines are identified by **Line-IDs**. These IDs are delivered by the QoS application.
- Further on there are the **quality of service** parameters to configure. The only necessary values are these of the guaranteed QoS parameters. They specify the switching threshold. The current values are delivered by the QoS software.
- The **possible-switching-events-list** configures behaviour in case of the value under-run of the QoS parameters.

Installation

The switching application is a Java .jar file and can easily be copied to the VAN application directory.
A default configuration for this software must be available. The packet will be provided under /HOME/WP7/.

LINUX Distribution

- o ProviderSwitching:
  - Filename: VAN_switching.jar
  - File location: /var/opt/VAN/ProviderSwitching
  - Necessary prerequisites: /conf Directory with configuration data

The command to startup the application is: *java –jar VAN_switching.jar*

4.3 VAN QoS Monitoring

For using the QoS Monitoring the following prerequisites should be considered:

- Java Runtime Environment JRE version 6.0
- Operating System: Linux Ubuntu 8.04 (LTS)(or higher)
- VAN SW Stack has to be installed and has to run
- System clocks must be synchronised (GPS + NTP)

4.3.1 Startup of the GPS and NTP daemon:

1.) Make sure to have the necessary settings in /etc/ntp.conf:

- *# NMEA - for getting the time from GPS*
  # Use this only! when no remote NTP server is available!
  server 127.127.20.0 minpoll 4 maxpoll 4
  fudge 127.127.20.0 time1 0.420 flag3 1 refid GPS0

- *# PPS - for PPS signal from GPS*
  server 127.127.20.1 minpoll 4 maxpoll 4 prefer
  fudge 127.127.20.1 time1 0.560 refid PPS

- *# Remote NTP servers for adjusting the seconds*
  # Use at least one of the remote servers
  # The maxpoll/minpoll option fixes the intervall of synchronization
  server ptbtime1.ptb.de minpoll 4 maxpoll 4
  server ptbtime2.ptb.de minpoll 4 maxpoll 4
  server ntps1-0.cs.tu-berlin.de minpoll 4 maxpoll 4

2.) Create a symbolic link to the serial/USB device:

- *# For serial device and use of PPS (regular case):*
  sudo ln -s /dev/ttyS0 /dev/gps1

- *# For USB device and use of NMEA seconds:*
  sudo ln -s /dev/ttyUSB0 /dev/gps0

3.) Start the GPS daemon:

- *# For information about the options see the gpsd manpage*
  gpsd /dev/ttyS0 -n -D1

4.) Start/restart the NTP daemon:

- sudo /etc/init.d/ntp (re)start

5.) See if the GPS daemon is running by:

- watch -n1 ntpq -p

Result should look like this:

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+GPS_NMEA(1)     .PPS.            0 l   -   16   77     0.000   11.886  152.854
*ntps1-1.cs.tu-b .PPS.            1 u   -   16   77    17.877   -0.347    0.157
```

4.3.2 QoS Monitoring software packet

The QoS Monitoring is provided in the packet

QoS.tar.

This is realising the entire measurement process (including the ASE object functionality of QoS Monitoring and QoS Measurement).

A further packet is the

QoSStart.tar.

This provides a self declaring graphical user interface for starting, stopping and visualising of results of measurements. This functionality/packet is only thought for the test phase - its entire functionality will be replaced by the VAN Engineering and the VAN Switching, which functionalities it is simulating.

Both packets will be provided under /HOME/WP7/.

5 Fulfilment of Requirements from D01.2-1

The following requirements were collected by WP1 in are described in [D01.2-1]:

- R7.1 Ability of network transitions to provide information

  Status: Can not be guaranteed, therefore own VAN Quality of Service monitoring was defined and implemented.

- R7.2 Guarantee for Quality of Service

  Status: Possibility is given by the combination of VAN QoS Monitoring and VAN Provider Switching, the defined QoS Priority Mapping of automation specific communication services [D07.2-1] as well as the implementation of this mapping for OpenVPN which was realised by work package 4.

- R7.3 Secure transfer of automation data

  Status: Use of encrypted tunnel for communication including public networks.

- R7.4 Service level agreement for automation purposes

  Status: SLA template and examples provided within [D07.2-2]

- R7.5 Switching between network providers

  Status: Realised by VAN Provider Switching.

- R7.6 Recovery and backup mechanism

  Status: Line-backup by VAN Provider Switching and data backup by VAN Telecontrol Profile.

- R7.8 Guaranteed availability

  Status: Guaranteed availability can only be achieved by using private networks or much expensive connections offered by internet providers. In this case the SLA template based on definitions of WP 7 can be used. Because of the QoS Measurement, VAN is able to detect network violations and the VAN Switching can be invoked.

- R7.9 Information if redundancy is supported

  Status: Information provided by VAN Provider Switching.

- R7.10 Virtual plant wide connectivity

  Status: Given by the VAN architecture.

- R7.11 Plant wide resource sharing

  Status: Given by the VAN architecture.

- R7.12 Distributed applications

  Status: Even via heterogeneous networks (including public networks).

- R7.13 Efficient data transfer

  Status: Realised by VAN Telecontrol Profile and especially because of the usage of the VAN defined QoS mappings.

- R7.14 Reuse of standard WAN and internet technologies

  Status: Given by the VAN architecture.

- R7.15 Realisation of safety applications

  Status: See WP5.

- R7.16 Isochronous behaviour

Status: Not influenced for local networks. Not needed for public networks (see WP4).

- R7.17 Calculable cost models

  Status: Integrated by metrics attributes in VAN Routing and VAN Provider Switching. Cost models developed in D07.2-2.

- R7.18 Logging and recovery functionality

  Status: Possibility is given by the combination of VAN QoS Monitoring, VAN Provider Switching and VAN Telecontrol Profile.

- R7.19 Network reliability

  Status: Because of the QoS Measurement, VAN is able to detect network violations. So the VAN Switching can be started or the device application can be informed and is able to react.

- R7.20 Scalable real-time and determinism

  Status: Realised by VAN Priority Mapping of WP4.

- R7.21 Location awareness

  Status: Topic taken over by WP3.

- R7.RQ-1 Reasonable cost for public connection

  Status: Provided by Filter of VAN Telecontrol Profile and Service Level Agreement.

- R7.RQ-2 Network availability

  Status: Possibility is given by the combination of VAN QoS Monitoring and VAN Provider Switching.

- R7.RQ-3 Transmission of real-time data

  Status: Subject of WP4.

- R7.RQ-4 Reserved Bandwidth

  Status: Provided by SLA and DSCP Mapping of WP4.

- R7.RQ-5 Data security

  Status: Subject of WP6.

- R7.RQ-6 Provider service (Hotline, Notification Service)

  Status: Determined in SLA template

- R7.RQ-8 Existing WAN interface

  Status: Use of legal WAN interfaces in VAN architecture.

- R7.RQ-9 Well-defined responsibilities if WAN subnets belonging to different operators

  Status: Is considered in SLA template.

- R7.RQ-10 Selective distribution of data: To one destination and To multiple or all destinations

  Status: A VAN domain is not a broadcast domain and so selected distribution is generally given. Multiple distribution of a single data is realised by VAN Telecontrol functionality.

- R7.RQ-11 Selective generation of data

  Status: Filtering of data within the VAN Telecontrol Profile.

- R7.RQ-12 Protection against data loss

  Status: Buffering of data within the VAN Telecontrol Profile.

- R7.RQ-13 Reconfiguration

  Status: Supported by the VAN Engineering System.

- R7.RQ-14 Support of different data types

  Status: Realised by VAN Telecontrol Profile. VAN communication is independent from the transported data type.

- R7.RQ-15 Support of data consistency

  Status: Addressed and considered by VAN Telecontrol Profile.

- R7.RQ-16 Interoperability

  Status: Subject of WP2.

- R7.RQ-17 WEB-Services

  Status: Basic communication of VAN devices. Subject of WP2.

- R7.RQ-18 Mobility

  Status: Integration of wireless technologies in VAN architecture. Subject of WP2 and WP3.

6 Recommendation for a Conformance Testing

Within [D02.4-3], the organisational aspect for establishing a VAN Conformance testing is described. In the following roughly the most important (technical) aspects for a conformance testing concerning the functionality developed within WP7 are concluded.

WP7 requirements to general VAN Conformance prerequisites:

- Devices need a FQDN
- VAN communication Stack must be integrated:
  - Web Service Server must be able to communicate with each other
  - Tunnel establishment must be integrated

The following items and refinements which had been worked out during the project period are important and relevant for later conformance testing:

- Telecontrol Profile
  - Runtime object model where the Telecontrol Profile is adapted to (as prototypically implemented with the PROFINET CBA Server) has to support event oriented communication.
  - Acquired data should be time stamped as early as possible and as close as possible to its origin.
- Provider Switching
  - Is based on a running QoS Monitoring.
    - Switching must initialise the start of measurement of single measurements within the QoS Monitoring (IOC is needed)
  - Alternative physical communication links must be available (e.g. DSL or UMTS)
- QoS Monitoring
  - System Clocks must be synchronised
  - Online measurement mode must be able to consider existing process data stream.

7 Conclusion

This document provides the integration guidelines for the application developed by WP7 with focus on supporting the two Industrial Experimental Setups of the VAN project, the Process IES and the Factory IES. This includes the prototypically implementations of the following applications:

- VAN Telecontrol Profile
- VAN QoS Monitoring
- VAN Provider Switching

In previous deliverables these processes where specified and based on these information the applications were implemented and documented within this deliverable. Furthermore the Process and Factory IES utilise them in different show cases and in cooperation with software of other work packages.

The Telecontrol Profile implementation provides the filter and buffer mechanism to support public networks with low deterministic behaviours and the variable service extending the PROFINET CBA Server. Moreover weaknesses of the specification were detected during the implementation and commissioning of the Telecontrol Profile. Updates and changes are described as well as solutions for runtime problems.

The QoS Monitoring is used to monitor the current status of a link between two VAN access points using public networks. An important prerequisite for the QoS Monitoring is a system clock synchronisation which is realised by the use of GPS and NTP deamons. The structure of the implemented QoS software was described. Furthermore details of the measurement sequence and the burst based measurement data stream for the two modes - initial-measurement and online-measurement – are given. The data collected by the monitoring are analysed by the VAN switching. Therefore an interface for their communication at the same device was necessary, realised with the inter object communication (IOC).

The VAN Switching interacts with the QoS application and reroutes the tunnel data to another connection if this is necessary. This necessity occurs if the measured QoS parameter does not fulfil the demands. This new functionality allows the customer to monitor the bought provider connection for compliance of the contract and to react fast if the connection gets lost or a parameter is below the required values.

The VAN Switching and QoS Monitoring are independent from the used communication technology over public network. For the testing within the Industrial Experimental Setups UMTS and DSL connections were chosen.

While integrating the different application new experiences and problems were notified which required solutions and changes to the described behaviours which was solved during Task 7.4. Consequently the fulfilment of the requirements collected by WP1 was adapted to the final version of the implementations.

The most important technical aspects for a conformance testing concerning the functionality in work package 7 has been listed.

Glossary

AD	Automation Device
AP	Access Point
ASE	Application Service Element
CBA	Component based Automation
CHP	Combined Heat and Power
COM	Component Object Model
DLL	Dynamic Link Labrary
DP	Decentralised Periphery
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
ES	Engineering System
FQDN	Fully Qualified Domain Name
GPS	Global Positioning System
GPSD	GPS-Daemon
GUI	Graphical User Interface
GW	Gateway
HMI	Human Machine Interface
ID	Identifier
IDL	Interface Definition Language
IES	Industrial Experimental Setup
IOC	Inter Object Communication
IP	Internet Protocol
JFMX	Supervising Software for the Management, Control and Integration
JRE	Java Runtime Environment
LDev	Logical Device
LTS	Long Term Support
Lib	Library
MA	Manufacturer Automation
NC	Numerical Control
NIC	Network Interface Card
NMEA	National Marine Electronics Association
NTP	Network Time Protocol
PA	Process Automation
PD	Proxy Device
PDev	Physical Device
PKI	Public Key Infrastructure

PLC	Programmable Logic Controller
PPS	Pulse Per Second
PN	PROFINET
QoS	Quality of Service
RFID	Radio Frequency Identification
RTAuto	Runtime Automation Object
SLA	Service Level Agreement
SRT	Soft Real-Time
SUI	knowledge based maintenance system (Störung- Ursachen Informationssystem)
TAP	TAP is a virtual Ethernet network device
TC	Telecontrol
TCP	Transmission Control Protocol
Ubuntu	Community developed, Linux-based Operating System
UC	Use Case
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
UTC	Unicersal Time Coordinated
VAN	Virtual Automation Networks
VPN	Virtual Private Network
WAN	Wide Area Network
WP	Work Package
WS	Web-service

References

[ASEMAN]	VAN Deliverable of work package 2, “Application Service Elements – Specification and Manual”
[D01.2-1]	VAN Deliverable of work package 1 task 2, “Requirements and unresolved issues for Automation - Update of Requirements and Technological Roadmap”. The VAN Consortium
[D02.2-1]	VAN Deliverable of work package 2 task 1, “Specification of the Open Platform for Automation Infrastructure - Topology Architecture for the VAN virtual Automation Domain”. The VAN Consortium
[D02.2-2]	VAN Deliverable of work package 2 task 2, "Specification of the Open Platform for Automation Infrastructure - VAN Open Platform API-Specification”. The VAN Consortium
[D02.4-1]	VAN Deliverable of work package 2 task 2, “Specification of the Open Platform for Automation Infrastructure – Software Architecture and Interface Specification“. The VAN Consortium.
[D02.4-2]	VAN Deliverable of work package 2 task 2, “Specification of the Open Platform for Automation Infrastructure – Prototype Software for VAN Open Platform“. The VAN Consortium.
[D02.4-3]	VAN Deliverable of work package 2 task 4, “Prototype Implementation Integration - Setup of testing laboratory for VAN-products”. The VAN Consortium.
[D06.3-1]	VAN Deliverable of work package 6 task 1, "Specification of a Security Layer - Service definition and protocol (functional) specification of a security layer”. The VAN Consortium
[D06.4-1]	VAN Deliverable of work package 6 task 4, “Security mechanisms prototype implementation – Software Architecture and Interface Description“. The VAN Consortium.
[D06.5-1]	VAN Deliverable of work package 6 task 5, “Validation of Prototype Implementations - Preliminary Test Report”. The VAN Consortium.
[D07.1-1]	VAN Deliverable of work package 7 task 1, "Status and Analysis - Report on analysed public network technologies”. The VAN Consortium
[D07.2-1]	VAN Deliverable of work package 7 task 2, "Integration Concept - Architecture Specification”. The VAN Consortium
[D07.2-2]	VAN Deliverable of work package 7 task 2, "Integration Concept - VAN QoS over public networks specification; Network Management and common signaling specification”. The VAN Consortium
[D07.3-1]]	VAN Deliverable of work package 7 task 3, "Interfaces - Common Interface Specification Gateway Specification”. The VAN Consortium
[D09.2]	VAN Deliverable of work package 9 task 2, "IES Report”. The VAN Consortium
[IEC 61158]	International Standard IEC 61158-5 and IEC 61158-6 Type 10, Digital Data Communications for Measurement and Control, Fieldbus for use in Industrial Control Systems, Part 5-10: Application Layer service specification – type 10 elements Part 6-10: Application Layer protocol specification
[openvpn]	Homepage of OpenVPN: http://openvpn.net/

[PNTestTool] Test Tool to access the PROFINET CBA Server, download from www.profibus.com

[RFC 958] IETF (1985), RFC 958, Network Time Protocol (NTP), IETF, available at http://www.ietf.org

[RFC 1305] IETF (1992), RFC 1305, Network Time Protocol (Version 3) Specification, Implementation and Analysis, IETF, available at http://www.ietf.org

[VAN Groupware] VAN Groupware, available at www.van-eu.eu

[VB] Homepage of Virtual Box: www.virtualbox.org