Universelles Videomanagementsystem von Accellence Technologies GmbH

Externe Schnittstellen

Dieses Dokument ist geistiges Eigentum der Accellence Technologies GmbH.
Änderungen und Irrtümer vorbehalten.
Dieses Dokument darf nur mit der ausdrücklichen Zustimmung der Accellence Technologies GmbH verwendet, vervielfältigt oder weitergegeben werden.

Impressum

Herausgeber

Gesellschaft:	Accellence Technologies GmbH
Handelsregister:	HRB 110799 Hannover
Geschäftsführer:	Dipl.-Inf.(FH) Frank Christ, Dr.-Ing. Heinz Stephanblome
Redaktion:	Torsten Heinrich, Mike Plötz
Tel:	+49 (0)511 277 2400
Fax:	+49 (0)511 277 2499
E-Mail:	info@accellence.de
Internet:	www.accellence.de / www.vimacc.de
Anschrift:	Accellence Technologies GmbH Garbsener Landstraße 10, 30419 Hannover, Deutschland

© 2013-2020 Accellence Technologies GmbH Datum: 28.12.2020

Inhaltsverzeichnis

Abkürzungsverzeichnis

DCOM	Distributed Component Object Model
NTP	Network Time Protocol
OPC	OLE for Process Control
RFC	Request For Comments
RTP	Real-Time Transport Protocol
RTSP	RealTime Streaming Protocol
RTCP	Real-Time Control Protocol
SDP	Session Description Protocol (see RFC 4566)
TCP	Transmission Control Protocol
vimacc	Video Management System von Accellence Technologies GmbH

1 Einleitung

1.1 Zweck des Dokumentes

Das vorliegende Dokument ist ein Teildokument der System Dokumentation für das Produkt **vimacc** der Accellence Technologies GmbH.

vimacc ist eine universelle Videomanagementsoftware zur Übertragung, Anzeige, Auswertung und Archivierung von Video-, Audiodaten und zugehörigen Metadaten sowie zur Steuerung der Video- und Audiotechnik wie z.B. Kameras und Schaltkontakten eines digital vernetzten CCTV-Systems.

Dieses Dokument beschreibt die externen Schnittstellen eines **vimacc** Systems. Über diese Schnittstellen kann ein **vimacc** System in übergeordnete Managementsysteme integriert werden, um beispielsweise Videostreams abzurufen, Kameraaufschaltungen durchzuführen oder das gesamte **vimacc** Subsystem fernzusteuern.

1.2 Aufbau der Dokumentation

Die **vimacc** System Dokumentation besteht aus einer Reihe von Dokumenten, die jeweils einen Teilaspekt behandeln und in sich abgeschlossen sind.

Die folgende Aufstellung beschreibt kurz die zur Verfügung stehenden Dokumente, die in ihrer Gesamtheit die **vimacc** Videomanagementsoftware beschreiben:

- vimacc Systemdokumentation: Einführung
  Dieses Dokument skizziert zunächst die Problemstellung eines heutigen digitalen Videoüberwachungssystems und leitet daraus die Notwendigkeit einer universellen Videomanagementsoftware her. Anschließend gibt es einen Überblick über die allgemeinen Eigenschaften von **vimacc** und zeigt einige sich daraus ergebenen möglichen Einsatzgebiete.

- vimacc Systemdokumentation: Eigenschaften
  Dieses Dokument liefert eine umfassende technische Leistungsbeschreibung der **vimacc** Videomanagementsoftware.

- vimacc Systemdokumentation: Architektur
  Dieses Dokument gibt einen detaillierten Einblick in die Architektur von **vimacc** und stellt die zur Verfügung stehenden Software Komponenten und ihre Funktionen vor.

- **vimacc Systemdokumentation: Schnittstellen**
  Dieses Dokument.

- vimacc Systemdokumentation: Systemvoraussetzungen
  Dieses Dokument beschreibt die Minimalanforderungen an Hardware und Betriebssystem-Software, die erfüllt sein müssen, damit **vimacc** auf einer Hardware-Komponente installiert werden kann.

- vimacc Systemdokumentation: Systemplanung
  Dieses Dokument beschreibt die besonderen Randbedingungen, die bei der Planung eines modernen und vernetzten Videosystems zu berücksichtigen sind und kann somit einem Systemplaner als Hilfestellung dienen, ein **vimacc** System zu dimensionieren und zu planen.
  Darüber hinaus stellt es die zur Verfügung stehenden Software Editionen und deren Einsatzgebiete vor.

2 Schnittstellen

2.1 Allgemein

vimacc verfügt über eine Vielzahl von externen Schnittstellen, mit deren Hilfe ein **vimacc** System mit übergeordneten Managementsysteme oder untergeordneten Subsystemen Daten ausgetauscht werden kann, um die unterschiedliche Anwendungsfälle zu realisieren. Dazu gehören zum Beispiel:

- Abruf von Videostreams abzurufen zur Darstellung in einem übergeordneten Managementsystem
- Steuerung von **vimacc** um Kameraaufschaltungen (z.B. auf Monitorwänden) durchzuführen
- Abruf von Videostreams zur Übergabe an ein Video-Analyse-System (VCA-Video Content Analysis)
- Entgegennahme von Alarmsignalisierungen von Brandmelde- oder Einbruchmeldeanalagen oder Gegensprechanlagen
- Entgegennahme von Alarmsignalisierungen von eine Video-Analyse-Systemen

Abbildung 2.1 zeigt eine schematische Darstellung der möglichen externen Schnittstellen.

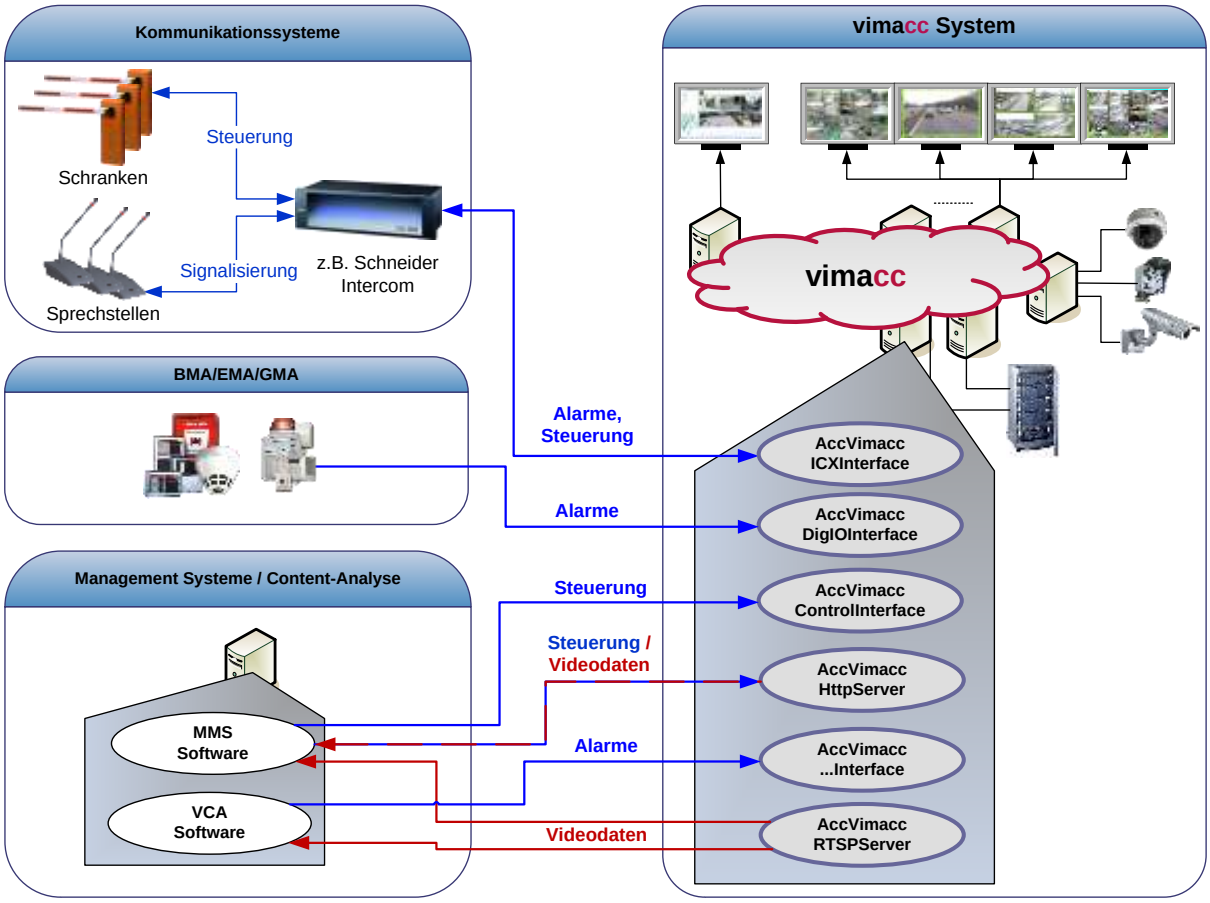

Abbildung 2.1: Externe Schnittstellen eines vimacc Systems

2.2 Abruf von Streaming-Daten via RTSP/RTP

2.2.1 Allgemein

Über das Real-Time Streaming Protocol (RTSP) können Audio- und Videostreams von einem **vimacc** System abgerufen und wiedergeben werden.
Hierzu muss innerhalb des **vimacc** Systems ein **vimacc** RTSP Streaming-Server betrieben werden (→ *vimacc Systemdokumentation: Architektur*).

Mittels RTSP können sowohl aufgezeichnete Streaming-Daten, als auch Live Streaming-Daten abgerufen werden.
Um einen Audio- oder Videostream vom *AccVimaccRTSPServer* abzurufen, muss der zugehörige Stream zunächst einmal eindeutig referenziert werden. Die Kennzeichnung eines solchen Streams erfolgt generell durch die Angabe einer eindeutigen RTSP-URL.
Die Abfrage dieser URLs für die zur Verfügung stehenden Live-Datenquellen oder die aufgezeichneten Audio- und Videostreams kann ebenfalls mittels RTSP durchgeführt werden. (Es ist aber auch genauso möglich, diese Parameter über eine **vimacc** HTTP Schnittstelle abzufragen und die RTSP-URL programmgesteuert zu erstellen.)

Die eigentliche Übermittlung der Streaming-Daten zum anfordernden Client erfolgt üblicherweise nicht über RTSP, sondern über das Real-Time Transport Protocol (RTP), wobei der *AccVimaccRTSPServer* die Übertragung über eine TCP/IP oder eine UDP/IP Verbindung unterstützt.
Die Steuerung dieser RTP Verbindung erfolgt parallel dazu über das Real-Time Control Protocol (RTCP), das ebenfalls über eine TCP/IP oder eine UDP/IP Verbindung etabliert werden kann.

Einen Überblick über die verwendeten Protokolle gibt die folgende Abbildung.

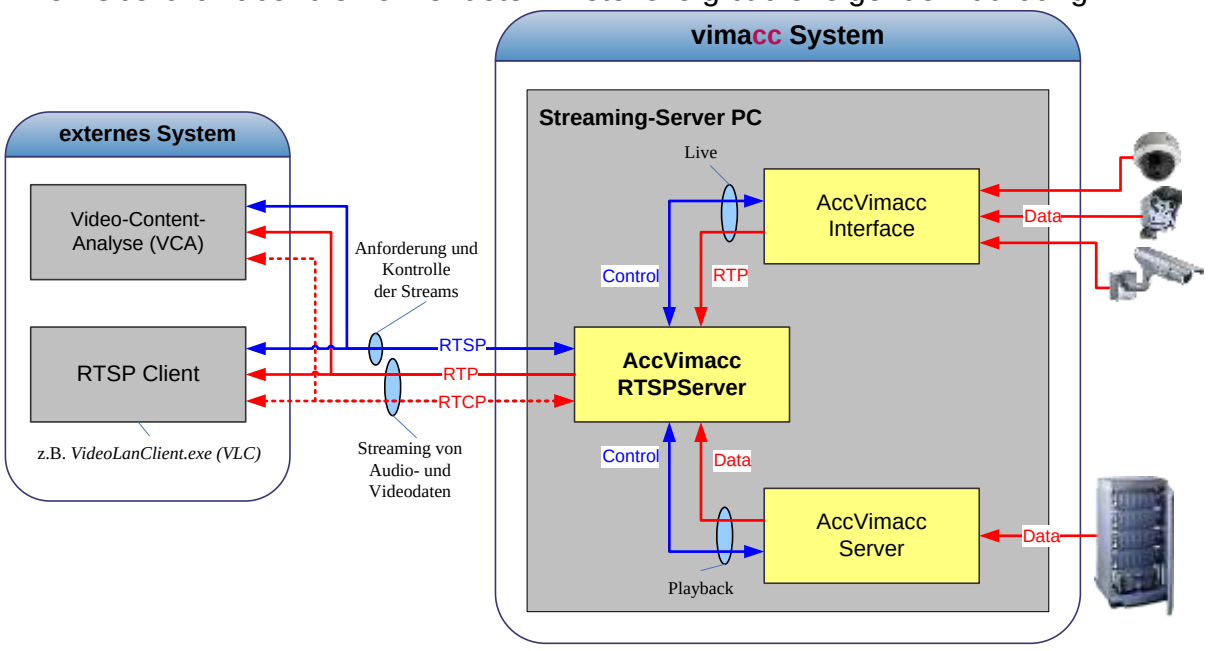

Abbildung 2.2: Abruf von Streaming-Daten über den *AccVimaccRTSPServer*

2.2.2 Kommunikation mittels RTSP

2.2.2.1 Verbindungsaufbau

Ein RTSP-Client baut zu dem RTSP Streaming-Server des **vimacc** Systems eine IP-Verbindung (TCP oder UDP) auf, um einen Stream abzurufen oder den zur Verfügung stehenden Inhalt des Streaming-Servers oder die zur Verfügung stehenden Streaming-Quellen abzurufen.

Bevor die Verbindung akzeptiert wird, fragt der Streaming-Server Benutzernamen und Passwort ab, um nur autorisierten Benutzer den Zugriff auf die Streaming-Daten zu gestatten. Diese Anmeldeinformationen sind über die **vimacc** Systemkonfiguration parametrierbar.

Es können mehrere RTSP-Clients gleichzeitig und unabhängig voneinander Verbindungen zu dem RTSP Streaming-Server aufbauen.

2.2.2.2 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP oder UDP

2.2.2.3 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP- oder UDP Socketverbindungen
- Darstellungsebene: ASCII-Klartext, Utf-8 kodiert

2.2.2.4 Anwendungsebene

Der **vimacc** RTSP Streaming-Server implementiert das RTSP Protokoll gemäß RFC 2326 `/RTSP/`, einschließlich der Verschachtelung der Binärdaten, so dass die Daten auch über TCP/IP übertragen werden können, wenn dies erforderlich sein sollte.

Die Kommunikation erfolgt üblicherweise durch das Versenden von RTSP-URLs vom Client zum Server.
Eine RTSP-URL (ohne Benutzername) hat die folgende Syntax:

rtsp://<video-server>:<RTP port>/<CameraID>/<live | playback>

Eine RTSP-URL (inklusive Benutzernamen und Passwort) hat die folgende Syntax:

rtsp://<username>:<password>@<video-server>:<RTP port>/
<CameraID>/<live | playback>

Das Schlüsselwort *live* oder *playback* am Ende der URL gibt an, ob ein Live Stream oder ein aufgezeichneter Stream abgerufen werden soll

Beispiel:

- Abruf der Live-Daten von Kamera *CAM-APM-C1* von dem RTSP Streaming-Server auf Rechner *cctv-server-1*, wobei der Stream auf den Port *5544* übertragen werden soll

  rtsp://cctv-server-1:5544/CAM-APM-C1/live

- Abruf der aufgezeichneten Daten von Kamera *CAM-APM-C1* von dem RTSP Streaming-Server *cctv-server-1*:

  rtsp://cctv-server-1:5544/CAM-APM-C1/playback

Hinweis:
Mit dem frei verfügbaren RTSP-Clients '*VLC media player*' (siehe `/VLC/`) kann die Basisfunktionalität des RTSP Streaming-Servers getestet werden. Nach Eingabe einer gültigen RTSP-URL und der vom Server abgefragten Benutzerdaten (Name und Passwort) kann der entsprechende Stream abgerufen und dargestellt werden.
Der '*VLC media player*' verfügt allerdings nur über eine sehr einfache Benutzerschnittstelle, so dass beispielsweise nicht die Liste der verfügbaren Streams abgefragt werden kann, oder auf der Zeitachse nicht zu einem absoluten Zeitpunkt navigiert werden kann.

2.2.2.5 Netzwerkressourcen

Der RTSP Streaming-Server ist per Default unter dem TCP-Port/UDP-Port 554 erreichbar. Dieser Port kann allerdings über die **vimacc** Systemkonfiguration geändert werden.

2.2.3 Streaming der Daten mittels RTP

2.2.3.1 Verbindungsaufbau

Während des Aufbaus der RTSP Verbindung handelt der RTSP Streaming-Server mit dem RTSP Client einen IP Port für die RTP Datenübertragung aus.
Generell gibt es hier zwei Möglichkeiten der Datenübertragung: Entweder wird eine UDP-Verbindung (unicast oder multicast) aufgebaut, dann wir der Port ausgetauscht und der Empfänger muss den UDP-Port öffnen, oder es wird eine Interleaving-Kennung ausgetauscht, die es ermöglicht die RTP-Daten direkt aus der RTSP-Verbindung zu extrahieren.

Der RTSP Streaming-Server kann zu mehreren RTSP-Clients gleichzeitig und unabhängig voneinander Verbindungen aufbauen.

2.2.3.2 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP (RTSP interleaved) oder UDP

2.2.3.3 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP- oder UDP Socketverbindungen
- Darstellungsebene: binär

2.2.3.4 Anwendungsebene

Der **vimacc** RTSP Streaming-Server beschreibt die Eigenschaften der Multimediadatenströme in Form von sogenannten SDP Dateien (Session Description Protocol), die gemäß RFC 4566 `/SDP/` gebildet werden, inklusive der Dekodierungsparameter gemäß RFC 3984 (RTP Payload Format for H.264 Video `/RTP H.264/`) und RFC 3016 (RTP Payload Format for MPEG-4 Audio/Visual Streams `/RTP MPEG-4/`).

2.2.3.5 Netzwerkressourcen

Der IP-Port (TCP oder UDP) zum Empfangen der RTP Streaming-Daten auf Seiten des Clients ist dynamisch und wird zwischen dem RTSP Streaming-Server und dem RTSP Client beim Verbindungsaufbau ausgehandelt.

2.2.4 Steuerung der Streams mittels RTCP

2.2.4.1 Verbindungsaufbau

Zur Steuerung des Streams und zur Aushandlung und Einhaltung von Quality of Service (QoS) Parametern baut ein RTSP-Client zu dem RTSP Streaming-Server eine zusätzliche IP-Verbindung (TCP oder UDP) auf.

Generell gibt es hier zwei Möglichkeiten der Datenübertragung: Entweder wird eine UDP-Verbindung (unicast oder multicast) aufgebaut mit "RTCP-Port = RTP-Port + 1", dann wird der Port ausgetauscht und der Empfänger muss den UDP-Port öffnen, oder es wird eine Interleaving-Kennung ausgetauscht, die es ermöglicht die RTCP Daten aus der RTSP-Verbindung zu extrahieren.

Es können mehrere RTSP-Clients gleichzeitig und unabhängig voneinander Verbindungen zu dem RTSP Streaming-Server aufbauen.

2.2.4.2 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP (RTSP interleaved) oder UDP

2.2.4.3 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP- oder UDP Socketverbindungen
- Darstellungsebene: ASCII-Klartext, Utf-8 kodiert

2.2.4.4 Anwendungsebene

Über die RTCP Verbindung werden Quality of Service (QoS) Parameter ausgehandelt und die eigentliche Steuerung des Streams (PLAY, STOP, PAUSE,…) durchgeführt.

2.2.4.5 Netzwerkressourcen

Der RTCP Port (TCP oder UDP, je nach Anforderung vom RTSP Client) ist üblicherweise der RTP-Port um eins erhöht:

```
RTCP-Port = RTP-Port + 1    (RTP-Port gerade Zahl!)
```

2.3 Steuerung über TCP

2.3.1 Allgemein

Das **vimacc** Control-Interface ermöglicht es, über ein einfaches textorientiertes Protokoll einen bidirektionalen Kommunikationskanal zu einem **vimacc**-System aufzubauen.
Über diese TCP-Verbindung kann das **vimacc**-System nahezu vollständig gesteuert werden. Weiterhin besteht die Option, Statusnachrichten und Systeminformationen zu erhalten. (→ *vimacc Systemdokumentation: Architektur*)

2.3.2 Verbindungsaufbau / Authentifizierung

Bei der Installation einer **vimacc-**Edition wird für das TCP Control-Interface der TCP-Port 4227 vorgesehen, das Interface ist per Default deaktiviert.
Es kann zwischen „keine Authentifizierung „ und „DIGEST Authentifizierung“ gewählt werden.

2.3.3 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP

2.3.4 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP Socketverbindung
- Darstellungsebene: ASCII-Klartext, UTF-8 kodiert

2.3.5 Anwendungsebene

Ein TCP-Client kann nach Aktivierung des Interfaces jederzeit eine TCP-Verbindung zu dem *TCP-Server* aufbauen. Nach erfolgreichem Verbindungsaufbau initiiert der Client eine Protokollsitzung, indem er sich authentifiziert.
Jedes Protokollkommando wird vom **vimacc** System stets über textbasierte Meldungen bestätigt.
Es existieren verschiedene Varianten des Protokolls mit unterschiedlichem Befehlsumfang (Basic, Alarm, All, Fallback).

2.3.6 Netzwerkressourcen

Das vimacc Control-Interface ist per Default unter TCP-Port 4227 erreichbar.
Die Konfiguration der Zugriffsparameter erfolgt im **vimacc** Configuration Center.

2.3.7 Telegrammbeschreibung

Die Kommandos der Telegramme sind in einem separaten Dokument beschrieben.
Siehe -> *vimacc_Systemdokumentation_Steuerschnittstelle_Control.pdf*

2.4 Steuerung über HTTP

2.4.1 Allgemein

Über einen http-Client (z.B. Webbrowser) besteht die Möglichkeit eine Verbindung zum *AccVimaccHTTPServer* aufzubauen (→ *vimacc Systemdokumentation: Architektur*). Über diese Schnittstelle kann das zugehörige **vimacc** System gesteuert werden und es besteht Zugriff auf die live- und playback-Videos. Die Video-Streams werden als JPEG-Einzelbild oder als MJPEG-Stream an einen Client übertragen.

2.4.2 Verbindungsaufbau / Authentifizierung

Bei der Installation einer **vimacc-**Edition wird der AccVimaccHttp-Server per Default auf Port 80 installiert.
Es kann zwischen verschiedenen Authentifizierungsmethoden gemäß RFC2617 (HTTP Basic and Digest Access Authentication `/http Authentication/)` gewählt werden:
- keine Authentifizierung (default)
- Basic Authentication
- Digest Authentication

2.4.3 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP

2.4.4 Darstellungs- und Sitzungsebene

- Sitzungsebene: HTTP
- Darstellungsebene: ASCII-Klartext, Utf-8 kodiert

2.4.5 Anwendungsebene

Der *AccVimaccHTTPServer* implementiert ein Subset des **vimacc** vimacc Controll-Interfaces (siehe Kapitel 2.3). Diese Schnittstelle ist für die Anwendungsebene ein textorientiertes Protokoll (ASCII-Klartext).

Ein http-Client (z.B. ein Webbrowser oder ein übergeordnetes Gefahren-managementsystem) kann jederzeit eine HTTP-Verbindung zu dem *AccVimaccHTTPServer* aufbauen. Nach erfolgreichem Verbindungsaufbau initiiert das übergeordnete Managementsystem eine Protokollsitzung, indem es sich beim *AccVimaccHTTPServer* authentifiziert.

Nach initiierter Protokollsitzung kann der http-Client/Webbrowser Steuerkommandos in Form von HTTP-GET-Requests mit Befehls-URL über die TCP-Verbindung

senden. Jedes Protokollkommando wird vom **vimacc** System stets über textbasierte Meldungen bestätigt.

2.4.6 Netzwerkressourcen

Der HTTP-Server des **vimacc** Systems ist per Default unter TCP-Port 80 erreichbar. Die Konfiguration des AccVimaccHttpServers erfolgt im **vimacc** Configuration Center.

2.4.7 Telegrammbeschreibung

Die Kommandos der Telegramme sind in einem separaten Dokument beschrieben.
Siehe -> *vimacc_Systemdokumentation_HTTP_Interface.pdf*

2.5 Steuerung über OPC

2.5.1 Allgemein

Der *AccVimaccOpcClient* implementiert eine OPC DA Schnittstelle, über die OPC Prozessdaten mit einen beliebigen OPC Server ausgetauscht werden können (→ *vimacc Systemdokumentation: Architektur*). *DA* steht dabei für *Data Access* und ist die OPC Spezifikation zur Übertragung von Echtzeitwerten über OPC.
Der *AccVimaccOpcClient* implementiert OPC DA Schnittstelle entsprechend der Schnittstellenspezifikation V1.0 oder V2.0 erfolgen (siehe `/OPC/`).

Für die Realisierung dieser Schnittstelle verwendet OPC die *DCOM* (Distributed Component Object Model) Technologie von Microsoft. DCOM kapselt dabei die Kommunikation zwischen den Komponenten, so dass es für die Anwendungen transparent ist, ob OPC die Daten nur innerhalb des lokalen Rechners versendet, oder über TCP/IP mit entfernten Rechnern kommuniziert.

2.5.2 Verbindungsaufbau

Um eine Verbindung zu einem konfigurierten OPC Server aufbauen zu können, muss der *AccVimaccOpcClient* wissen, wie er den Server adressieren kann.

Hierzu muss zunächst einmal dem *AccVimaccOpcClient* die IP Adresse des Rechners bekannt sein, auf dem der OPC Server betrieben wird.
Da der Verbindungsaufbau allerdings von DCOM durchgeführt wird, reicht an dieser Stelle die IP Adresse nicht aus. DCOM benötigt hierzu zusätzlich eine GUID (Global Unique Identifier), über die der OPC Server eindeutig referenziert werden kann. Diese GUID muss in der Registrierungsdatenbank des Windows Betriebssystems gespeichert werden und wird vom *AccVimaccOpcClient* über den Namen des OPC Servers (z.B. *OPC.SimaticNET* für den Simatic OPC Server der Firma Siemens) ermittelt. Als zweiten Konfigurationsparameter muss der *AccVimaccOpcClient* also den Namen des OPC Servers kennen.
(Das Anlegen der GUID in der Registrierungsdatenbank des Betriebssystems wird von dem **vimacc** Installationsprogramm bei der Installation des OPC Clients durchgeführt).

Die benötigten Konfigurationsparameter für den *AccVimaccOpcClient* sind in der **vimacc** Konfigurationsdatenbank hinterlegt.

Der Verbindungsaufbau zum OPC Server erfolgt mittels DCOM Methodenaufrufe, bei denen die beschriebenen Parameter übergeben werden.

2.5.3 Anwendungsebene

Nachdem sich der *AccVimaccOpcClient* über das OPC DCOM Interface dem konfigurierten OPC Server verbunden hat, erfolgt der Datenaustausch durch das periodische Lesen der vom OPC Server bereitgestellten sogenannten OPC

Prozessvariablen. Jede Prozessvariable wird als Paar aus einem Namen und dem zugehörigem Wert gespeichert.

Welche Prozessvariablen vom *AccVimaccOpcClient* gelesen werden, wird mittels einer Konfigurationsdatei festgelegt.
Ebenso wird innerhalb dieser Konfigurationsdatei definiert, welche Aktionen beim Setzen oder Zurücksetzen des Wertes einer Variablen vom *AccVimaccOpcClient* durchgeführt werden sollen.

2.5.4 Beschreibung der Prozessvariablen

Der Aufbau der Prozessvariablen ist im Prinzip beliebig und kann je nach Projekt variieren.
In der momentanen Implementierung (Stand Januar 2012) geht der *AccVimaccOpcClient* von der folgenden Syntax aus:

```
OPC Variablenname = <0|1>
```

Es wird also aktuell nur überprüft, ob die entsprechende Variable gesetzt (Wert=1) oder nicht gesetzt ist (Wert=0).

Ist eine Prozessvariable gesetzt, so wird die zu dieser Variablen konfigurierte Aktion ausgeführt. Die Zuordnung zwischen Prozessvariable und Aktion erfolgt über die Konfigurationsdatei des *AccVimaccOpcClient*.

Als Aktionen kennt der *AccVimaccOpcClient* momentan:

- Verbinde eine Kamera mit einem bestimmten Monitor
- Ändere die Aufzeichnungsbildrate zwischen den Werten *highrate* (z.B. 25 fps) und *lowrate* (z.B. 8 fps)

An einem Beispiel soll dies verdeutlicht werden:
Im OPC Server sei die Prozessvariable `Cctv.APM1.DG11_MON_H` definiert. In der Konfigurationsdatei vom *AccVimaccOpcClient* sei folgendes definiert:

```
[ProcessVars]
;OPC var => Camera|Monitor|Parameter (param=highrate|lowrate)
Cctv.APM1.DG11_MON_H=CAM-DG11|Videowall2/M05|param=highrate
```

Erkennt der *AccVimaccOpcClient*, dass der OPC Server den Wert der Prozessvariablen `Cctv.APM1.DG11_MON_H` auf 1 gesetzt hat, so wird er über die **vimacc** Config die Aufschaltung von Kamera `CAM-DG11` auf dem Monitor durchführen und die Aufzeichnungsrate auf die Bildrate erhöhen, die dem Parameter `highrate` zugewiesen ist (z.B. 25 fps).

2.6 Steuerung über ICX

2.6.1 Verbindungsaufbau

Der *AccVimaccICXClient* versucht sich mit einem konfigurierten ICX Server über eine TCP/IP Verbindung zu verbinden (→ *vimacc Systemdokumentation: Architektur*).

2.6.2 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP

2.6.3 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP
- Darstellungsebene: ASCII-Klartext, Utf-8 kodiert

2.6.4 Anwendungsebene

Die Komponente *AccVimaccICXClient* implementiert die Schnittstelle zwischen **vimacc** und einem Schneider Intercom System. Diese Schnittstelle wird realisiert durch das IP-Protokoll (ICX) der Firma Commend.

Über diese ICX Schnittstelle kann **vimacc** Ereignissen der Schneider Intercom Anlage Kameras und Szenarien zuordnen, so dass diese automatisch auf einen Monitor aufgeschaltet werden können und die Videos der entsprechenden Kameras in erhöhter Qualität gespeichert werden können.

Beim Eintreffen eines Rufes am Intercom System wird hierzu dem *AccVimaccICXClient* über die ICX Schnittstelle die rufende Sprechstelle mitgeteilt, woraufhin ein der entsprechenden Sprechstelle zugeordnetes Szenario aus der Konfiguration ausgelesen und über die **vimacc** *Config* aufgeschaltet werden kann und die entsprechenden Video-Bilder mit hoher Qualität aufgezeichnet werden können.
Nachdem das Ende des Gespräches signalisiert wurde, wird nach einer einstellbaren Zeit wieder auf die normale Aufzeichnungsqualität zurückgeschaltet.

2.6.5 Netzwerkressourcen

Der *AccVimaccICXClient* versucht, den konfigurierten ICX Server über einen konfigurierten TCP-Port zu erreichen.

2.6.6 Telegrammbeschreibung

Über die ICX Schnittstelle werden vom Schneider Intercom System folgende Daten an den *AccVimaccICXClient* übergeben (Task 8E 9E):

- Anrufende Sprechstelle (Ruf evtl. in Warteschleife)
- Annehmende Sprechstelle
- Statusinformation: Ruf wurde abgebaut (verzögert, da evtl. das Videobild nur umgeschaltet wird)
- Fehlermeldungen, Systemhinweise usw. werden von dem *AccVimaccICXClient*, soweit möglich, durch Meldungen an der ICX-Schnittstelle an die Schneider Intercom übergeben

2.7 Reporting über TCP/IP

2.7.1 Verbindungsaufbau

Eine Applikation kann über TCP/IP eine Verbindung zu einem *AccVimaccReportServer* aufbauen, um Report-Meldungen aus dem **vimacc** System zu erhalten (→ *vimacc Systemdokumentation: Architektur*).

2.7.2 Netzwerk- und Transportebene

- Netzwerkebene: IP
- Transportebene: TCP

2.7.3 Darstellungs- und Sitzungsebene

- Sitzungsebene: TCP
- Darstellungsebene: ASCII-Klartext, Utf-8 kodiert

2.7.4 Anwendungsebene

Die Komponente *AccVimaccReportServer* implementiert die Reporting-Schnittstelle eines **vimacc** Systems gemäß dem Protokoll RCP_VIMACC_REPORT.
Über diese RCP Schnittstelle (Remote Control Protocol) können Filterbedingungen definiert werden, die festlegen, welche Art von Meldungen **vimacc** an den verbundenen Event-Monitor senden soll.

Beim Eintreffen einer neuen Meldung *AccVimaccReportServer* wird diese zunächst in der Event-Datenbank des Server abgespeichert.
Darüber hinaus wird diese Meldung an alle verbundenen Event-Monitore gesendet, die die entsprechenden Filterbedingungen gesetzt haben.

Ferner können über das RCP_VIMACC_REPORT Protokoll Meldungen aus der Vergangenheit abgefragt werden. Diese werden dann, entsprechend der übergebenen Filterbedingungen aus der Event-Datenbank des Servers ausgelesen und an anfragenden Event-Monitor gesendet.

2.7.5 Netzwerkressourcen

Eventmonitore können sich über einen TCP-Port zu einem *AccVimaccReportServer* verbinden.
Der Default Port für das Protokoll RCP_VIMACC_REPORT ist 636. Dieser Port kann über die **vimacc** Systemkonfiguration geändert werden.

2.8 Zeitsynchronisation mittels NTP

Innerhalb eines **vimacc** Systems müssen alle Komponenten jederzeit über die gleiche Systemzeit verfügen, damit zeitliche Bezüge hergestellt werden können. Dies ist besonders wichtig im Zusammenhang mit der Speicherung und Wiedergabe der Streaming-Daten, denn nur wenn sichergestellt ist, dass in alle Komponenten die gleiche Systemzeit eingestellt ist, können die Streaming-Daten im richtigen zeitlichen Kontext wiedergegeben werden. Andernfalls wäre es nicht möglich, z.B. Aufzeichnungen einer Szene, die von mehreren Kameras gleichzeitig aufgenommen wurden, oder auch die zu einem Video gehörende, passende Tonspur im richtigen zeitlichen Zusammenhang wiederzugeben.
Die Zeitsynchronisation eines **vimacc** Systems und aller angeschlossenen Komponenten erfolgt immer über das Network Time Protocol (NTP).
Bei NTP handelt es sich um einen Standard zur Synchronisierung von Uhren in Computersystemen, bei dem die Zeitinformationen zwischen NTP-Servern und NTP-Clients über das Transportprotokoll UDP ausgetauscht werden.
NTP sieht grundsätzlich mehrere Hierarchiestufen von NTP-Servern vor. Bei jedem NTP-Server kann ein sogenanntes *Stratum* definiert werden, das die hierarchische Entfernung eines Zeitservers von der Zeitquelle beschreibt (siehe `/NTP/`).
Dadurch ist ein möglich ein System aufzubauen, bei dem mehrere Zeitserver nachgeschaltet betrieben werden können (siehe die folgende Abbildung).

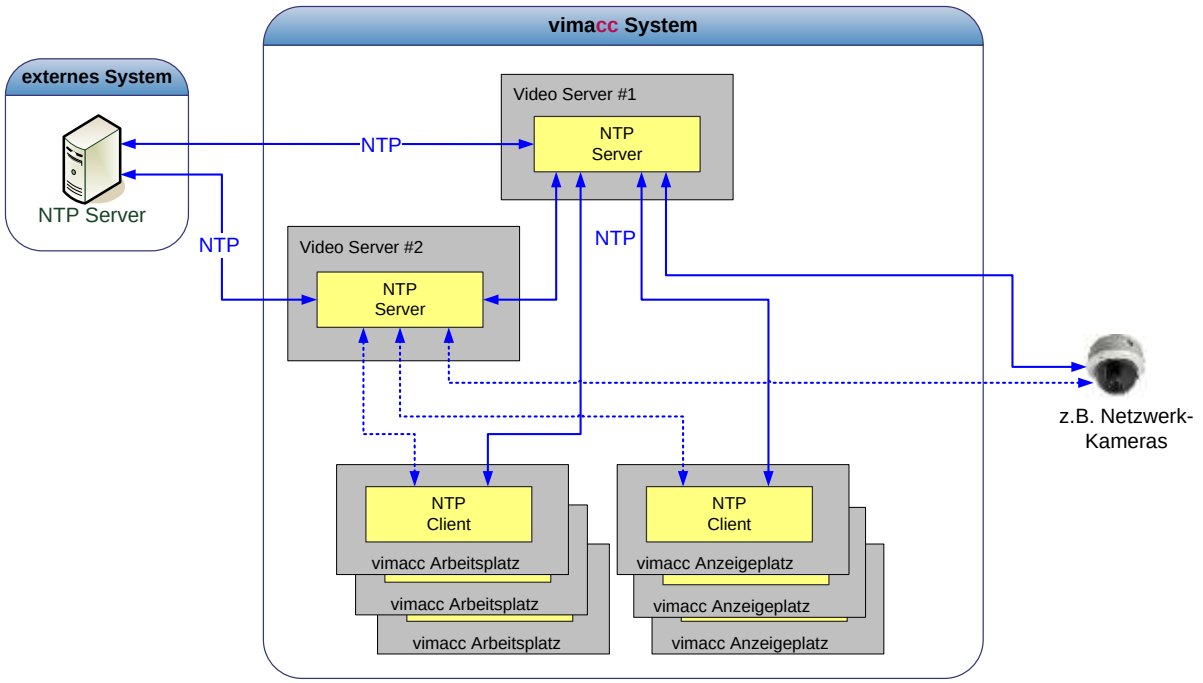

Abbildung 2.3: Zeitsynchronisation eines vimacc Systems

Es soll ein zentraler Zeitgeber (NTP-Server) für alle Teilsegmente benutzt werden. Dieser zentrale Server synchronisiert üblicherweise seine Zeit mittels DCF-Signal oder gegen einen Internet-Timeserver.

Innerhalb des **vimacc** Systems werden immer die zentralen, redundanten Server als NTP-Server betrieben. Diese NTP-Server werden mit einer geringe NTP Hierarchiestufe betrieben als der Zeitserver des übergeordneten Systems, d.h. die Server des **vimacc** Systems versuchen ihrerseits, sich mit dem Master-Zeitserver zeitlich zu synchronisieren.
Bei alle anderen Rechnern und angeschlossenen Geräten des **vimacc** Systems werden NTP-Clients betrieben, die so konfiguriert werden, dass die beiden **vimacc** Server deren Zeitserver darstellen, d.h. diese Komponenten haben wiederum ein geringe NTP Hierarchiestufe als die **vimacc** Server.

Das **vimacc** System kann auf diese Weise zunächst einmal zeitlich synchronisiert betrieben werden, auch wenn kein übergeordneter Zeitserver konfiguriert ist oder dieser nicht erreichbar ist. Ist dieser Master-Zeitserver allerdings über das Netzwerk erreichbar, so werden die NTP-Server des **vimacc** Systems ihre Systemzeit mit diesem Server synchronisieren, so dass schließlich das Gesamtsystem zeitlich vollständig synchronisiert werden kann.

Hinweis:
Die eingesetzten NTP Software-Komponente ist nicht Teil der **vimacc** Software, sondern es können frei verfügbare NTP Installationen verwendet werden. Beispielsweise kann die NTP Installation von *Meinberg* eingesetzt werden (siehe `/Meinberg/`).

3 Support / Hotline

Weitere Informationen unter www.vimacc.de

Haben Sie noch Fragen zu **vimacc®**?

Kontaktieren Sie unser Support-Team werktags von 9:00-17:00 MEZ/MESZ

- per Email an **support@accellence.de**
  oder
- telefonisch unter **+49 (0)511 277 2490**

4 Referenzierte Dokumente

/VLC/	VLC media player, http://www.videolan.org/
/RTSP/	Real Time Streaming Protocol (RTSP), RFC 2326, http://www.ietf.org/rfc/rfc2326.txt
/SDP/	Session Description Protocol (SDP), RFC 4566, http://www.ietf.org/rfc/rfc4566.txt
/RTP H.264/	RTP Payload Format for H.264 Video, RFC 3984 http://www.ietf.org/rfc/rfc3984.txt
/RTP MPEG-4/	RTP Payload Format for H.264 Video, RFC 3984 http://www.ietf.org/rfc/rfc3984.txt
/http Authentication/	HTTP Authentication, RFC 2617 http://www.ietf.org/rfc/rfc2617.txt
/OPC/	OPC Foundation http://www.opcfoundation.org/
/NTP/	Network Time Protocol and stratum levels http://en.wikipedia.org/wiki/Network_Time_Protocol
/Meinberg/	Meinberg NTP Service Installation http://www.meinberg-usa.com/ntp-software.php

Index