DIPLOMARBEIT

Inventarisierung mit RFID

RFID Supported Inventory System

ausgeführt an der
Abteilung für Elektronik - Ausbildungsschwerpunkt Technische Informatik
der Höheren Technischen Bundeslehranstalt Hollabrunn

unter Betreuung von Dipl.-Ing. Franz Geischläger

durch

Sebastian Glaßner
und
Mathias Wallner-Haas

5BHELI
2005/06

Hollabrunn, 04.05.2006

HINWEISE

Die vorliegende Diplomarbeit wurde für die Abteilung Elektronik - Technische Informatik der HTBL Hollabrunn ausgeführt.

Die in dieser Diplomarbeit entwickelten Prototypen und Software-Produkte dürfen ganz oder auch in Teilen von Privatpersonen oder Firmen nur dann in Verkehr gebracht werden, wenn sie diese selbst geprüft und für den vorgesehenen Verwendungszweck für geeignet befunden haben.

Für alle Entwicklungen gilt die GNU General Software License [URL 1] der Free Software Foundation, Boston, USA in der aktuellen englischen Fassung.

Die Diplomarbeit erfüllt die “Standards für Ingenieur- und Technikerprojekte” entsprechend dem Rundschreiben Nr. 60 aus 1999 [URL 2] des BMBWK (GZ.17.600/101-II/2b/99).

Wir erklären, dass wir die Diplomarbeit eigenständig ausgeführt und alle verwendeten Quellen im Literaturverzeichnis angeführt habe.

Unterschriften:

Sebastian Glaßner Mathias Wallner-Haas

SCHLÜSSELBEGRIFFE

Inventarisierung

RFID

LabWindows CVI

PHP

MySQL Datenbank

SOAP Webservice

DANKSAGUNG

Der besondere Dank gilt Dipl.-Ing. Gerald Stoll für seine Bemühungen um eine Projekt Zusammenarbeit mit der Telekom Austria.
Weiters möchten wir uns bei den TA Mitarbeitern Ing. Rainer Graf, Markus Grasel, Dipl.-Ing. Harald Mairböck, Dipl.-Ing. Thomas Lehner und Roland Neustetter für ihre investierte Zeit bedanken, sowie bei Dipl.-Ing. Andreas Martin, Mitarbeiter des Infineon RFID Solutions Center in Graz, für eine sehr interessante Führung durch die Entwicklungsabteilung des Solutions Center.
Schließlich möchten wir uns noch bei unseren Kollegen Andreas Hagmann und Markus Popp für zahlreiche Hilfestellungen und Verbesserungsvorschläge bedanken.

GNU FREE DOCUMENTATION LICENSE

Herausgeber und Autoren dieses Dokuments sind

Sebastian Glaßner <sebastian.glassner@gmx.at>
Mathias Wallner-Haas <wallner-haas@a1.net>

http://rfid.5bheli.at

HÖHERE TECHNISCHE BUNDESLEHRANSTALT HOLLABRUNN

Abteilung: **Höhere Lehranstalt für Elektronik**

Ausbildungsschwerpunkt: **Technische Informatik**

DIPLOMARBEIT

Reife- und Diplomprüfung 2005/06

5BHELI

Thema: Inventarisierung mit RFID

Aufgabenstellung (Kurzfassung): Ergänzende Ausführungen siehe Beilage

Kandidaten / Kandidatinnen

Glassner Sebastian

Wallner-Haas Mathias

Betreuer / Betreuerin

DI Franz Geischläger

Externe Kooperationspartner

Firma / Institution ----

Betreuer / Kontaktperson ----

Schriftliche Kooperationsvereinbarung liegt vor ----

Budget 500,00 €

Bedeckung durch HTBL Hollabrunn

Geplante Verwertung der Ergebnisse

Laborinventarisierung

Erklärung

Die unterfertigten Kandidaten / Kandidatinnen haben gemäß § 34 (3) SchUG in Verbindung mit § 22 (1) Zi. 3 lit. b der Verordnung über die abschließenden Prüfungen in den berufsbildenden mittleren und höheren Schulen, BGBl. II Nr. 70 vom 24.02.2000 (Prüfungsordnung BMHS), die Ausarbeitung einer Diplomarbeit mit der umseitig angeführten Aufgabenstellung gewählt.
Die Kandidaten / Kandidatinnen nehmen zur Kenntnis, dass die Diplomarbeit in eigenständiger Weise und außerhalb des Unterrichtes zu bearbeiten und anzufertigen ist, wobei Ergebnisse des Unterrichtes mit einbezogen werden können.
Die Abgabe der vollständigen Diplomarbeit hat bis spätestens

05.05.2006 17:00:00

beim zuständigen Betreuer zu erfolgen.
Die Kandidaten / Kandidatinnen nehmen weiters zur Kenntnis, dass gemäß § 9 (6) der Prüfungsordnung BMHS nur der Schulleiter bis spätestens Ende des vorletzten Semesters den Abbruch einer Diplomarbeit anordnen kann, wenn diese aus nicht beim Prüfungskandidaten (bei den Prüfungskandidaten) gelegenen Gründen nicht fertiggestellt werden kann.

Kandidaten / Kandidatinnen - Unterschrift

Glassner Sebastian

Wallner-Haas Mathias

DI Franz Geischläger
Prüfer

DI Dr. Werner Reif
Abteilungsvorstand

HR Mag. DI Dr. Thomas Dietmaier
Direktor

Genehmigung:

St Pölten, am

DI Dr. Wilhelm König
Landesschulinspektor

HÖHERE TECHNISCHE BUNDESLEHRANSTALT HOLLABRUNN

Abteilung: **ELEKTRONIK**
Ausbildungsschwerpunkt: TECHNISCHE INFORMATIK

KANDIDATEN

Sebastian Glassner, Mathias Wallner-Haas

THEMA

Inventarisierung mit RFID

AUFGABENSTELLUNG

Es ist ein System zur Inventarisierung mit Hilfe von RFID Technologie zu entwerfen.

Zunächst sollen Untersuchungen bezüglich der Realisierungsmöglichkeiten der PC-Inventarisierung unter Verwendung von HF- und LF-Transpondern durchgeführt werden.

Die Software besteht aus Transportschicht und Verarbeitungslogik, deren Kommunikation über eine standardisierte XML-Schnittstelle erfolgt.

Die Transportschicht dient zur Aufbereitung und Weiterleitung der RFID-Reader Daten an die Schnittstelle.

In der Verarbeitungslogik erfolgt die Verwaltung der Daten unter Verwendung eines WAMP-Systems.

PROJEKTABLAUF

- Testen der Hardware: Reader sowie Transponder (Anbringung, Ausrichtung, etc.)
- Definition des XML-Interfaces
- Aufbereitung der Reader Daten (C)
- Einrichtung des Apache Webservers und der MySQL-Datenbank
- Übertragung der aufbereiteten Reader Daten mittels SOAP
- Verarbeitungslogik und dazugehörige Oberfläche entwerfen (PHP)
- Einbinden der Datenbank in die Überwachungslogik
- Zusammenfügen und Test des Gesamtsystems
- Vollständige Dokumentation der Hard- und Software. Dabei sind insbesondere auch eine Zusammenfassung der Arbeit in englischer Sprache, veröffentlichungsreife HTML-Seiten (Projekthompage) und eine Presseaussendung zu erstellen.

Kurzfassung

Unsere Diplomarbeit befasst sich mit dem Entwurf eines Systems zur Inventarisierung der Schulrechner an der HTL Hollabrunn mit Hilfe der Radio Frequency Identification (RFID) Technologie.
Auftraggeber ist die HTL Hollabrunn und der Betreuer des Projektes ist Dipl.-Ing. Franz Geischläger.

Durch das drahtlose Auslesen von passiven RFID Transpondern, die an den Gehäusen der Rechner angebracht werden, soll es ermöglicht werden, Informationen wie zum Beispiel deren Standort, Name, Konfiguration oder Reparaturstatus zu erhalten.
Die dafür benötigte Hardware wurde in Form des Multi-Function Evaluation Kit (Lesegerät + Demonstrations Software + Transponder) S4100 MFR von Texas Instruments, von der HTL Hollabrunn zur Verfügung gestellt.

Das Projekt besteht im Wesentlichen aus zwei Teilen, wobei der erste Teil, die Transportschicht als Client fungiert, während auf einem Server mittels Webinterface die Verwaltung einer Datenbank erfolgt. Zur Kommunikation zwischen den beiden Programmteilen wird das XML basierende und auf HTTP aufsetzende SOAP Protokoll verwendet.

Die Aufgaben der Transportschicht liegen hauptsächlich in der Aufbereitung der RFID Reader Daten, die bei Bedarf an den Webserver weitergeleitet werden können.
Die aus den Transpondern ausgelesenen Daten werden über die RS232 Schnittstelle nach Anfrage an die Transportschicht weitergeleitet.
Zum komfortablen Bearbeiten der Transponder Daten wird ein User Interface entwickelt, das dem Benutzer neben dem Echtzeitauslesen der Informationen auch das Beschreiben der RFID Tags ermöglicht.
Das Programm wird in der C++ Entwicklungsumgebung LabWindows CVI 8.0 realisiert. Um die Datenübertragung zum Server zu verwirklichen, wurde das SOAP-Protokoll mithilfe des Microsoft SOAP Toolkits 3.0 implementiert.

Das Webinterface soll die Verwaltung des Rechner Inventars über die Verknüpfung der eindeutigen Transponder ID mit Datensätzen aus der Datenbank zulassen.
Darüber hinaus sollen durch die Anwendung Wartungen der bzw. Servicetätigkeiten an den Rechnern festgehalten und nachvollziehbar gemacht werden. Um dies zu garantieren, ist eine Benutzerverwaltung notwendig. Allen Benutzern ist es möglich, die zu verwaltenden PCs mit ihren Wartungszuständen einzusehen.
Der Schulgebrauch des Systems wird durch die unterschiedliche Rechtevergabe an Schüler und Lehrer bewerkstelligt. Während dem übergeordneten Lehrer die gesamte Verwaltung der Wartungstätigkeiten und des Rechner Inventars zur Verfügung steht, ist der Schüler nur zum Hinzufügen von Servicevorgängen befähigt.
Der Administrator des Systems kann überdies die Benutzerverwaltung sowie die Datenbanksicherung regeln.
Die Webanwendung ist in PHP geschrieben und wird von einer MySQL Datenbank unterstützt, die mit dem Programm PHPmyAdmin erstellt wurde. Das hierfür verwendete Softwarepaket ist XAMPP für Windows.

Abstract

The main purpose of our engineering project is the conception of an inventory system for the computer inventory of the HTL Hollabrunn by using Radio Frequency Identification technology.
The ordering party of the project is the “HTL Hollabrunn”, and we are supervised by Dipl.-Ing. Franz Geischläger.

Information like the location, name, configuration or repair status of a computer should be wirelessly accessible by polling the passively powered RFID transponders which are mounted on the computer cases.
The required hardware, the Multi-function Evaluation Kit S4100 MFR produced by Texas Instruments, was provided by the HTL Hollabrunn.

Our project is split into two parts. The first part is the Transport Layer, which acts as a client whereas the second part, a server, provides the web-interface based administration of a database.

The Transport Layer is responsible for processing the data which is received from the RFID reader via COM port. The processed data is forwarded to the web-server on demand. In order to allow easy editing of the RFID tags a user interface is developed.
The program is fully realized with the ANSI C development environment CVI 8.0.

The web-interface allows administrating and searching inventoried computers which is made possible by linking the unique transponder identification number to data records of the database.
Moreover the application should log service activities that are accomplished on school computers. For this purpose and to allow school use, it is necessary to introduce different user rights. Every user is able to look up inventory and service activities. A teacher is authorized to edit inventory and service activities whereas a pupil is just entitled to add new services. The administrator controls the user accounts and he can backup the whole database.
The web-interface is written with PHP scripts and it supports a MySQL database. The required software is provided trough the software packet XAMPP.

The communication between the two software parts is established via the HTTP based SOAP protocol.

Inhaltsverzeichnis

ANHANG

1 Einleitung

1.1 Motivation

Die Verwendung RFID Technologie gewinnt durch die steigende Nachfrage nach den unterschiedlichen Einsatzgebieten, von Logistik Systemen bis hin zu kontaktlosen Kassensystemen, immer mehr an Bedeutung.
RFID vereint Elemente aus den verschiedensten Branchen. Dazu gehören zum Beispiel HF-Technik und EMV, Halbleitertechnik, Datenschutz und Kryptographie, Telekommunikation, Fertigungstechnik und viele verwandte Fachgebiete.
Darüber hinaus wird ein System zur Verwaltung der anfallenden Daten benötigt. Abhilfe schafft nur eine computerunterstützte Datenbank, die es ermöglicht, große Datenmengen in gewünschter Strukturierung zu speichern und administrierbar zu machen.

2 Lastenheft

Es ist ein System zu Inventarisierung der PCs an der HTL Hollabrunn mit Hilfe von RFID Technologie zu entwerfen.

Durch das Auslesen von Transpondern soll es ermöglicht werden, die in einer Datenbank abgelegten Rechnerinformationen mit den Schulrechnern zu verknüpfen.
Servicetätigkeiten an den Rechnern sollen über ein zentrales System nachvollziehbar festgehalten werden.

Das System soll auf zwei unabhängigen Anwendungen basieren. Der erste Teil soll zum Bearbeiten der Transponder und zum Ansteuern der Hardware dienen, während der zweite Teil die Verwaltung des Inventars über die Transponder ID realisiert. Die beiden Teile sollen über eine XML Schnittstelle miteinander kommunizieren.

3 Pflichtenheft

3.1 Zielbestimmung

3.1.1 Muss-Kriterien

Es ist ein System zu Inventarisierung der PCs an der HTL Hollabrunn mit Hilfe von RFID Technologie zu entwerfen.

Die Software soll im Wesentlichen aus zwei Teilen bestehen:

1) Transportschicht:
Ein Programm zur Aufbereitung der RFID-Reader-Daten zur Weiterleitung an die Logik über eine definierte XML Schnittstelle.

2) Webinterface:
Software zur Aufbereitung der Transportschicht-Daten (PHP) und Verwaltung mittels MySQL-Datenbank. Die Webanwendung soll auf einem Apache Webserver realisiert werden.
Darüber hinaus sollen durch die Anwendung Wartungen der bzw. Servicetätigkeiten an den Schulrechnern festgehalten und nachvollziehbar gemacht werden. Um dies zu garantieren ist eine Benutzerverwaltung notwendig. Das Webinterface erhält eine übersichtliche und einfach zu bedienende, grafische Oberfläche.

Zur Präsentation des Projektes im Internet ist eine Homepage zu erstellen.

3.1.2 Soll-Kriterien

Die Aufbereitung der RFID-Reader-Daten soll mit C oder C++ realisiert werden.

Zur Übertragung der aufbereiteten Daten zwischen der Transportschicht und der Logik soll das SOAP-Protokoll verwendet werden.

Die Projekthomepage soll standardkonform (valide) in XHTML programmiert werden.

3.1.3 Abgrenzungskriterien

Die Integration in das HTL – Inventarisierungssystem ist vorerst nicht vorgesehen, da das Projekt vorrangig Test- und Evaluierungszwecken dienen soll.
Datenübertragungen erfolgen unverschlüsselt.

3.2 Einsatz

3.2.1 Anwendungsbereiche

Das System wird zur Vereinfachung der Inventarisierung der PCs innerhalb der HTL Hollabrunn verwendet. Dabei wird die Entwicklung so ausgelegt, dass das System durch geringen Aufwand auf anderes Inventar der HTL (z.B.: Labor Messgeräte) umgelegt werden kann.

3.2.2 Zielgruppen

Das Projekt soll als Unterstützung für die in der HTL für PC-Inventarisierung verantwortlichen Personen dienen.
Dabei werden lediglich PC Grundkenntnisse, wie das Bedienen eines Browsers, vorausgesetzt.

3.2.3 Betriebsbedingungen

Zum Betrieb des Inventarisierungsmodells ist ein geschlossener, mit PCs ausgestatteter Raum erforderlich. Es soll sich nicht wesentlich von anderen Web Browser basierenden Anwendungen unterscheiden.
Betriebsdauer: täglich, 24h
Wartungsfreier Betrieb soll gewährleistet werden.
Die Sicherung der Datenbank muss vom Administrator durchgeführt werden.

3.3 Produkt Umgebung

3.3.1 Software

Client:

- Software zur Aufbereitung der Reader Daten (Transportschicht)
- Gängiger und aktueller Web Browser

Server:

- MySQL 4.1.12-nt Datenbank
- Apache Web Server mit PHP 5.0.4
- Webanwendung

3.3.2 Hardware

Client:

- Rechner mit Anbindung an das Netzwerk der HTBL Hollabrunn
- Rechner der die Anforderungen der o.g. Software unterstützt

Server:

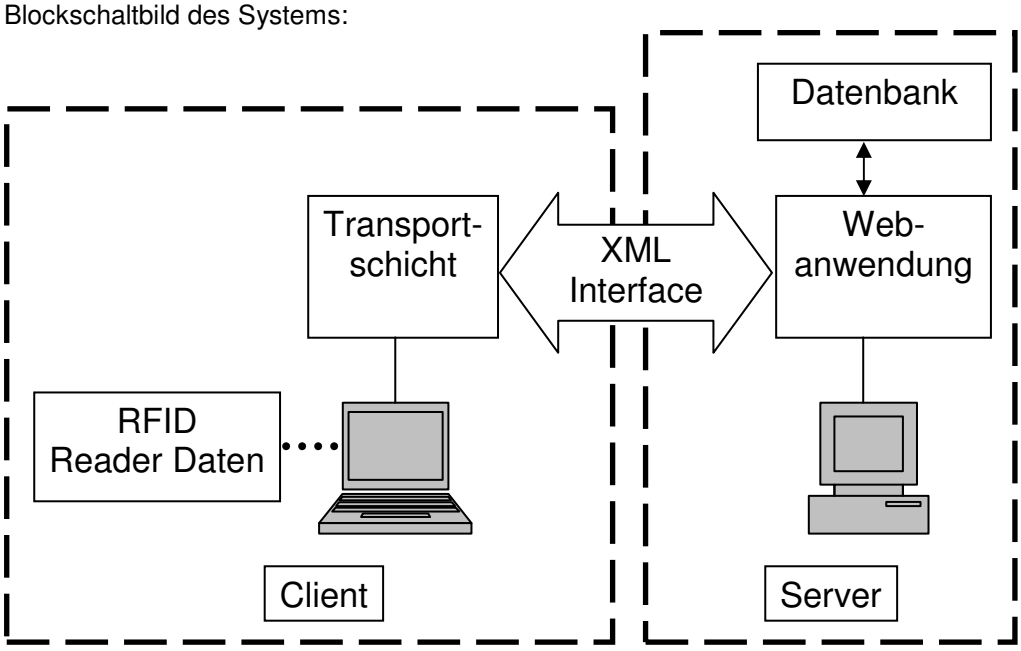
- Rechner mit Anbindung an das Netzwerk der HTBL Hollabrunn
- Rechner der die Anforderungen der o.g. Software unterstützt
- Ausreichend Rechen- und Festplattenkapazität

3.3.3 Orgware

Gewährleistung der permanenten Netzwerkanbindung durch den Administrator

3.4 Produktfunktionen

Blockschaltbild des Systems:



Die Eingangsschicht besteht aus den RFID-Lesegeräten. Die gelesenen Daten werden durch die Transportschicht über das XML Interface an das Webinterface geleitet.
Die Webanwendung verwaltet das System mit Hilfe einer Datenbank.

3.4.1 Eingangsschicht

Die Eingangschicht ist die Verbindung des RFID Lesegeräts mit dem Client. Durch sie erfolgt die Kommunikation der Hardware mit der Transportschicht.

3.4.2 Transportschicht

Das primäre Ziel der Transportschicht ist es die Daten der Eingangsschicht weiterzuverarbeiten. Die Verarbeitung der Daten und das Ansteuern des Lesegerätes kann dabei mit Hilfe einer Benutzeroberfläche gesteuert werden.
Die Transportschicht kann die Daten der Transponder aufbereiten und an den Datenbankserver weiterleiten. Neben diesen Funktionen sind auch die Möglichkeiten zur Aufzeichnung einer Logdatei und zur Ausführung eines Skriptes gegeben. Zur Informationsübergabe an den Server wird das SOAP-Protokoll direkt in die Anwendung integriert

3.4.3 XML Interface

Das XML Interface bietet die Möglichkeit zum Datenaustausch zwischen der Transportschicht und der Webanwendung an.
Als Schnittstelle soll ein möglichst vorteilhaftes Protokoll gewählt werden. Auf Grund von folgenden Vorteilen wird das SOAP Protokoll verwendet:
Standardisierung, Plattformunabhängigkeit, Offenheit, Robustheit und Skalierbarkeit.

3.4.4 Webanwendung

Programmstart:
Es besteht die Möglichkeit sich als Gast bzw. als Benutzer mit erweiterten Rechten auf der Web Oberfläche zu bewegen.

- **Gast:**
  Es ist kein Login notwendig. Für jeden Benutzer des Systems stehen folgende Funktionen zur Verfügung:
  - Anzeige aller eingetragenen PCs, mit allen möglichen Informationen die sich aus der Tabellenstruktur ergeben, in Form einer Tabelle. Zusätzlich wird in einer generierten Spalte die Anzahl der zu einem PC gehörenden Serviceeinträge angezeigt.
  - Bei einem Mausklick auf den gewünschten Rechnereintrag werden die dazugehörenden Serviceeinträge mit allen Inhalten aus der Datenbank angezeigt.
  - Die Ausgabe der Einträge kann angepasst werden. (alphabetisches Sortieren,…)
  - Suche nach bestimmten Kriterien (PC Name, Raum Name,…) und Anzeige der gefundenen Elemente.
  - Anzeige aller eingetragenen Services mit den Informationen aus der Tabellenstruktur in Form einer Tabelle.
  - Suche nach Rechnern unter Verwendung des RFID Lesegeräts.
  - Registrierungsmöglichkeit auf der Startseite

- **Registrierter User (deaktiviert):**
  Es ist ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines Gasts. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:
  - Einsehen und Bearbeiten der Accountdaten.
  - Er kann von einem Benutzer mit höherer Rechtekategorie aktiviert werden.
  - Es besteht die Möglichkeit sich abzumelden.

- **Registrierter User (aktiviert):**
  Es ist ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines deaktivierten registrierten Users. Zusätzlich steht ihm folgende Funktion zur Verfügung:
  - Bei der Anzeige der eingetragenen PCs kann zu jedem PC explizit ein Serviceeintrag hinzugefügt werden.

- **Master User:**
  Es ist ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines aktivierten registrierten Users. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:
  - Es können neue Rechner in die Datenbank aufgenommen werden.
  - Es können alle Einträge einzeln gelöscht werden. Wird ein PC Eintrag gelöscht werden alle dazugehörigen Serviceeinträge gelöscht.
  - Es können alle Einträge bearbeitet werden.

- **Administrator:**
  Es ist ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines Master Users. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:
  - Exportieren der Datenbank zur Sicherung
  - Wiederherstellen der Datenbank.
  - Benutzerverwaltung: Hinzufügen, Entfernen, Bearbeiten von Benutzern.
  - Aktivieren von registrierten Benutzern um ihnen das Erstellen von Services zu ermöglichen.

Benutzeroberfläche der Webanwendung:

- Struktur:
  Die Strukturierung der Oberfläche erfolgt entsprechend der Rechtekategorie des Benutzers.

- Bildschirmlayout
  Das Layout sowie das Design der Webanwendung wird überwiegend durch HTML bzw. CSS Komponenten bestimmt und ist über das gesamte System konsistent bzw. einheitlich.

3.4.5 Produktdaten

Es wird ein MySQL DBMS verwendet.
Folgende Datenstruktur soll mindestens in der Datenbank abgelegt werden:

Rechnerinformationen:
- Inventarnummer
- RFID
- Name
- Standort
- Erstellungsdatum
- Hardwarekonfiguration
- Kurzbeschreibung

Serviceinformationen
- Servicenummer
- Betroffener Rechner
- Datum
- Bearbeiter
- Beschreibung der Tätigkeit

Benutzerverwaltung
- Vorname
- Nachname
- Nick
- Passwort
- Rechte

3.4.6 Hardwarefunktionen

Das verwendete Texas Instruments S4100 Development Kit enthält 2 RFID-Reader, HF und LF Transponder verschiedener Bauformen, Anschlusskabel, Demo Software und ausführliche Dokumentation.
Die RFID Lesegeräte können Tags im Hochfrequenzbereich (13,56MHz) als auch Tags im Niederfrequenzbereich (134,2kHz) erkennen und beschreiben.
Erkannte RFID-Protokolle sind die der Texas Instruments RFID HF und LF Tags, sowie Tags nach ISO 15693 und ISO 14443 A/B.
Die Kommunikation zum PC erfolgt über die TTL I/O-Bausteine der Module.
Die Spannungsversorgung erfolgt über ein 230V~/5V= Netzteil.

3.5 Entwicklungsumgebung

3.5.1 Hardware

2 Notebooks:

- Targa Visionary XP-210 (Server)
- Acer Aspire 1710 (Client)

RFID Developement Kit:

- Texas Instruments S4100 Development Kit

3.5.2 Software

Verwendung	Software	Version
Betriebssystem	Windows XP	SP2 (NT 5.1)
Transportschicht	LabWindows CVI	8.0
Softwarepaket	XAMPP	1.4.14
Webanwendung	PHP	5.0.4
Datenbank	MySQL	4.1.12-nt
Datenbankadministration	phpMyAdmin	2.6.2
Webserver	Apache Webserver	2.0
HTML/PHP Editor	PHP Designer	2005
	PSPad editor	4.5.0
Blockschaltbilder, Flussdiagramme	MS Visio	2002
Texteditor, Tabellenkalkulation, Präsentationserstellung	MS Office	2003

4 Projektplanung

Juni 2005

Glaßner	Wallner-Haas
Projektauswahl	
Besuch des Infineon RFID Solutions Center in Graz bzw. der Telekom Austria Installationsstraße in Wien	
Besprechung der ungefähren Aufgabenstellung	
Beschäftigungsbereiche für Sommerferien erhalten	

Juli 2005

Glaßner	Wallner-Haas
Telekom Austria Besprechung	
Studieren der RFID Grundlagen, HTML, XML	

August 2005

Glaßner	Wallner-Haas
Studieren der RFID Grundlagen, HTML, XML	
Besuch der TA Logistik Zentrale	

September 2005

Glaßner	Wallner-Haas
Studieren der RFID Grundlagen, HTML, XML	
Endgültige Festlegung, Projekt Schulintern durchzuführen	
Neue Aufgabenstellung verfasst	
Kick Off Präsentation	
Entwicklungsumgebung für Software ausgewählt	
Projekthomepage begonnen	

Oktober 2005

Glaßner	Wallner-Haas
Pflichtenheft erstellt	Pflichtenheft erstellt
Beschäftigung mit PHP, MySQL, SOAP	RFID-Development Kit getestet
Gestaltung der Weboberfläche begonnen	RFID-Labels getestet
Datenbankmodellierung	Beschäftigung mit SOAP, C++

November 2005

Glaßner	Wallner-Haas
Entwicklung der Webanwendung	Beschäftigung mit CVI und MS SOAP Toolkit
Oberflächendesign	Ansteuerung des RFID-Readers durch CVI
Inventarverwaltung	RS232
	Fehlererkennung
	graf. Benutzeroberfläche
	Homepageaktualisierung

Dezember 2005

Glaßner	Wallner-Haas
Entwicklung der Webanwendung	Nutzung von RFID-Modulen des Readers
Inventarverwaltung	Marktforschung
Serviceverwaltung	PocketPCs mit RFID-CF-Cards
Benutzerverwaltung	RFID-Transponder

Jänner 2006

Glaßner	Wallner-Haas
Entwicklung der Webanwendung	Realisieren weiterer RFID-Reader-Funktionen
1.Projektpräsentation	1.Projektpräsentation
Kurzfassung erstellt	Kurzfassung erstellt

Februar 2006

Glaßner	Wallner-Haas
Weiterentwicklung der Webanwendung	Realisieren weiterer RFID-Reader-Funktionen und Beheben von Fehlern
Beschäftigung mit SOAP, WSDL	Kommunikation zwischen CVI Client und Apache Server über XML Interface
Kommunikation zwischen CVI Client und Apache Server über XML Interface	

März 2006

Glaßner	Wallner-Haas
Dokumentation	Dokumentation
2.Projektpräsentation	2.Projektpräsentation
Abgabe	Abgabe

April 2006

Glaßner	Wallner-Haas
Dokumentation	Dokumentation
Skripte ausbessern, kommentieren	Erstellen eines Installers
Schnellinstallationsanleitung	Quellcode optimieren und kommentieren
Projektpräsentation vor 4.Jg.	Projektpräsentation vor 4.Jg.

Mai 2006

Glaßner	Wallner-Haas
Dokumentation	Dokumentation
Fertigstellung und Abgabe der Diplomarbeit	Fertigstellung und Abgabe der Diplomarbeit

5 Realisierung

Zu Beginn der Arbeiten an unserem Projekt mussten wir den grundlegenden Systemaufbau festlegen. Wie bereits aus dem Pflichtenheft ersichtlich, besteht das System im Wesentlichen aus zwei Teilen: der Transportschicht und der datenbankbasierenden Webanwendung.
In den folgenden Unterpunkten wird die gewählte Realisierung näher erläutert.

5.1 Systemaufbau

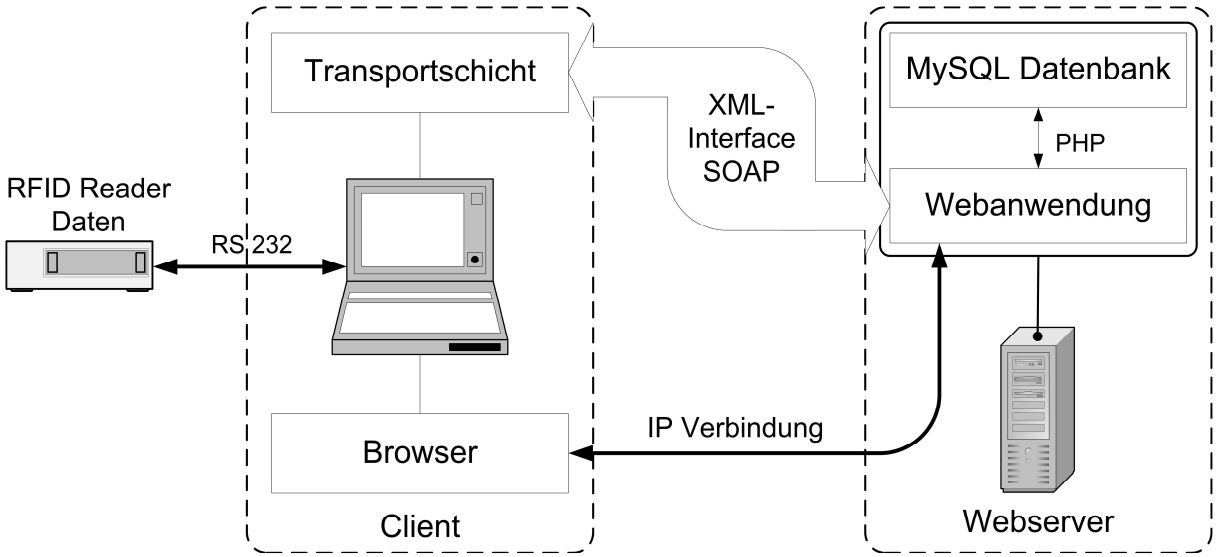
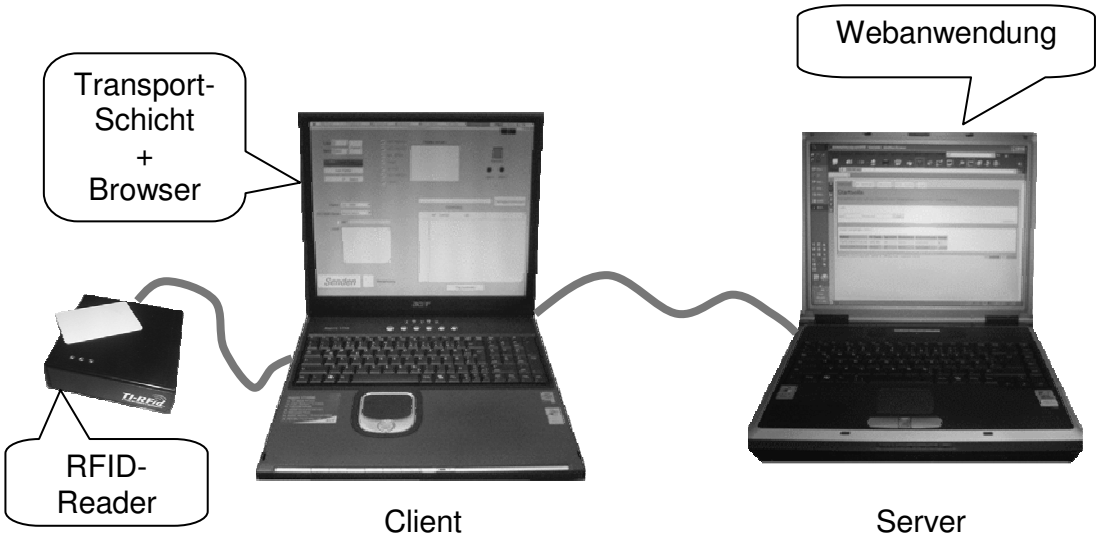


Abbildung 5.1 Blockschaltbild Systemaufbau

Wie aus dem Blockschaltbild ersichtlich ist die Transportschicht ein Programm das auf Seiten des Clients ausgeführt wird. Es kommuniziert über ein XML Interface mit der Webanwendung.
Gleichzeitig kann der Client mit Hilfe eines Browsers auf die am Server vorhanden Webanwendung und somit auch auf die Datenbank zugreifen.

5.2 Realisierter Aufbau

Zur Entwicklung des einzelnen Teile des Blockschaltbildes wurden zwei Notebooks und ein RFID Reader verwendet.



5.3 Hardware

Das Reader-Paket soll dem Projektteam die Möglichkeit bieten, eigene Applikationen möglichst ausgiebig testen zu können. Hierzu bietet sich das **Texas Instruments S4100 Development Kit** an.
Das Development Kit enthält 2 RFID-Reader, HF und LF Transponder verschiedener Bauformen, Ansschlusskabel, Demo Software und ausführliche Dokumentation.
Um die Kompatibilität zu anderen Anbietern zu gewährleisten, wird der Reader und die Anwendung auch mit Transpondern der Firma SmartTec evaluiert.

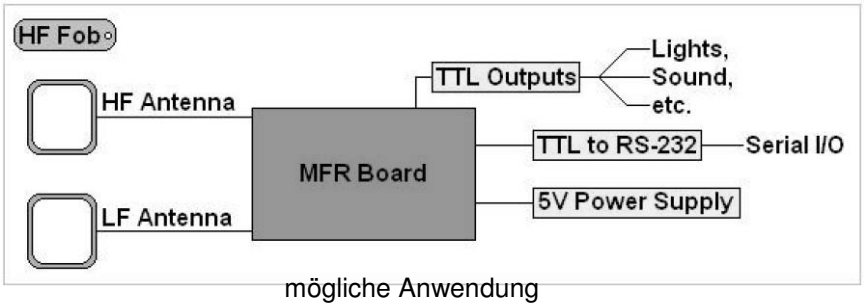
5.3.1 Texas Instruments S4100 Development Kit

Das Texas Instruments S4100 Development Kit wurde Anfang des Schujahres nach sorgfältiger Auswahl angekauft.
Zu den Vorteilen gegenüber Konkurrenzprodukten gehören:

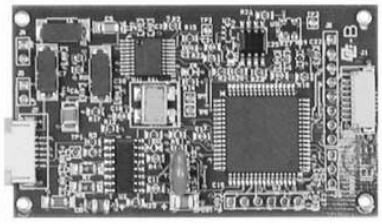
- Zwei Frequenzbereiche (HF u. LF)
- Eine Rohplatine und ein bereits fertig zusammengebauter Reader
- Hohe Vielfalt an Demo-Transpondern in verschiedenen Bauformen

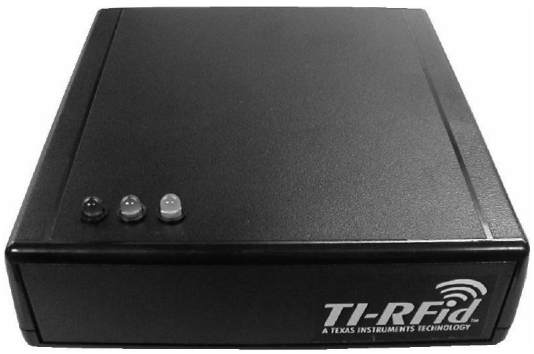
2 HF/LF RFID Module (eine Rohplatine und ein sofort funktionsfähiger Reader):
Die RFID Module können Tags im Hochfrequenzbereich (13,56MHz) als auch Tags im Niederfrequenzbereich (134,2kHz) erkennen und beschreiben. Erkannte RFID-Protokolle sind die der Texas Instruments RFID HF und LF Tags, sowie Tags nach ISO 15693 und ISO 14443 A/B.
Die Kommunikation zum PC erfolgt über die TTL I/O-Bausteine der Module.
Die Spannungsversorgung erfolgt über ein 230V~/5V= Netzteil.
Das Modul besitzt drei TTL Ausgänge (z.B.: zur Ansteuerung von Signalgebern).
Das Programm des RFID-Moduls kann per Firmware update ersetzt werden (man könnte sich die Firmware also auch selbst programmieren).

Der sofort funktionsfähige Reader ist in einem robusten Gehäuse verbaut.
Am TTL I/O ist ein UART Baustein angeschlossen, wodurch der Reader sofort mit beiliegendem RS232-Kabel am PC angeschlossen werden kann. Der Reader kann auch über RS485 Schnittstelle (Buchse vorhanden) angesteuert werden.
Die HF und LF Antennen sind angeschlossen.
An den TTL Ausgängen des Moduls sind zwei LEDs und ein Summer angeschlossen. Eine dritte LED signalisiert den betriebsbereiten Zustand (Spannungsversorung).



mögliche Anwendung


„rohes“ RFID-Modul



verbautes RFID-Modul

5.3.2 RFID-Transponder

Wie bereits erwähnt, stehen uns die **Texas Instruments Transponder** und die auf Anfrage kostenlos erhaltenen **SmartTEC** Transponder zur Verfügung.

Die Transponder von Texas Instruments bieten einen guten Überblick über die möglichen RFID-Einsatzgebiete. Jedoch sind nicht alle Transponder für den Einsatz in der Inventarisierung der PCs der HTL Hollabrunn geeignet. Nach einigen Untersuchungen bezüglich der Reichweite und der Störbarkeit der RFID-Übertragung, schieden sämtliche LF-Transponder aus. Diese sind stark Lageabhängig und die Reichweite ist wesentlich kürzer als bei den HF-Transpondern.
Ein weiterer Vorteil der HF-Transponder ist Pulkfähigkeit, was bedeutet das Antikollisionsverfahren unterstützt werden, welche erlauben mehrere Tags gleichzeitig einzulesen.

Tabelle 5.1 – Unterschiede der Transponder nach Frequenzbereich

Frequenz	**100-135 kHz**	**13,56 MHz**	**2,45 GHz**
Energie	Passiv	Passiv & Semiaktiv	Passiv & Aktiv
Speicher	bis 2kBit	bis 2kBit	bis 256kBit
Reichweite	bis 1,0m	bis ca. 1,7m	bis ca. 6m (p)
Metalleinfluss	hoch	hoch	niedrig
Flüssigkeiten	niedrig	niedrig	hoch
Pulkfähig	nicht realisiert	bis 100 St.	bis 500 St.
Kosten	niedrig	niedrig	hoch

Für unseren Anwendungszweck wurden also HF-Transponder gewählt. Die Labels sollen auf den Gehäusen der PCs aufgeklebt werden. Da sich Transponder des 13,56MHz-Frequenzbereichs noch stark von Metallteilen beeinflussen lassen, wählten wir die Kunststofffront des PC-Gehäuses als Transponder-Befestigungsstelle. Das RFID-Label ist im Idealfall selbstklebend und wird auf der Innenseite der Kunststoffverkleidung angebracht um es vor Beschädigungen zu schützen.

Es wurden entsprechende Anfragen an RFID-Transponderhersteller gesandt. Positive Antwort bekamen wir von der Firma SmartTEC, welche uns mit drei 13,56MHz-Transondern unterstützte. Die Transponder sind, wie von uns angefragt, selbstklebend, haben einen 112Byte großen Speicher und entsprechen der ISO15693 Norm. Der Transponder nutzt einen Philips Mikrocontroller.
SmartTEC machte uns ein Angbot für 600 solcher smart-LABELs zum Preis von 500€.

Nachdem wir unsere Wunschtransponder erhalten hatten, musste eine sinnvolle Belegung des internen Transponderspeichers gewählt werden.

Es wurde festgelegt, dass folgende Daten auf dem Transponder gespeichert werden müssen:
<u>Speicherbelegung des Transponders:</u>

16 Bytes: Erstellungsdatum/-uhrzeit
16 Bytes: Datum / Uhrzeit der letzen Änderung
16 Bytes: Inventarnummer
16 Bytes: Standort des Rechners
Restlicher Speicher (192 Bytes bei Texas Instruments und 48 Bytes bei SmartTEC-Transpondern):
beliebige Zusatzinformationen → üblicherweise: Ausstattung des PCs und durchgeführte Services

Alle Daten werden als ASCII-Strings abgespeichert, da wir das einfache Auslesen der Informationen mit den meisten RFID-Lesegerät-Programmen gewährleisten wollten. Somit belegt ein ASCII-Zeichen genau ein Byte.

Zusätzlich werden auf den Transpondern, die für die PC-Inventarisierung eingesetzt werden, die AFI (Application Family ID) auf 90_{HEX} (Item Management) und die DSFID (Data Storage Format ID) auf 91_{HEX} (willkürlich für die PC-Inventarisierung gewählt) gesetzt. Anhand dieser Daten erkennt die Transportschicht, ob es sich um einen HTL-PC-Inventarisierungstransponder handelt.

5.4 Transportschicht

Die Transportschicht wurde in der ANSI C Entwicklungsumgebung National Instruments Labwindows CVI 8.0 (als Demoversion auf der Projekt-CD) programmiert. Zur einfachen Nutzung von SOAP im Zielprogramm wurde das Microsoft SOAP Toolkit 3.0 installiert. Die Funktionen des Toolkits werden im CVI als Gerätetreiber angezeigt.

5.5 Webanwendung + Datenbank

Das Inventarisierungssystem basiert auf einem Server-Client Modell. Das heißt ein Client kann mit Hilfe eines Webbrowsers auf den auf den Server zugreifen.
Die Verbindung zwischen dem Server und dem Client erfolgt mit Hilfe einer IP-Verbindung, die in einem lokalen Netzwerk realisiert ist. Um mit der Webanwendung arbeiten zu können, müssen auf der Clientseite nur Standardprogramme installiert sein. Das heißt es muss ein Betriebssystem und ein Webbrowser verfügbar sein. In Abbildung 5.2 ist der prinzipielle Aufbau einer Server-Client Verbindung zu erkennen.

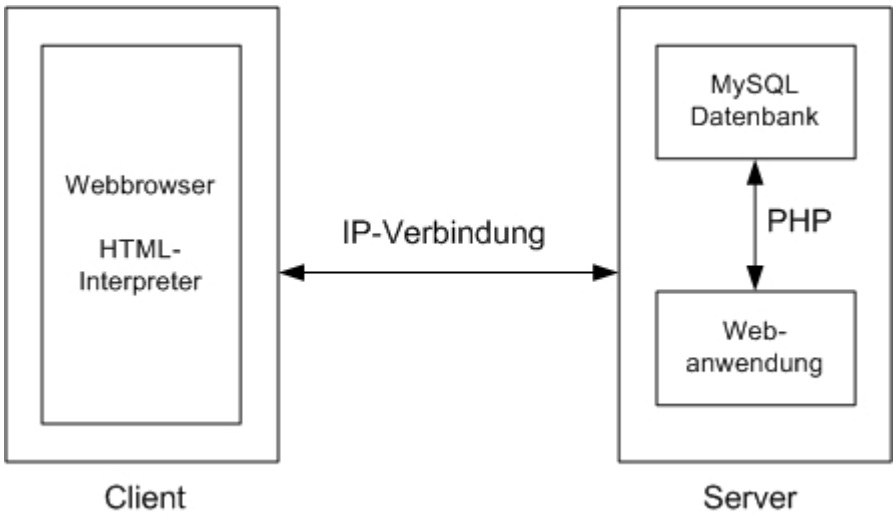


Abbildung 5.2 Server-Client Verbindung

In den folgenden Abschnitten wird die Auswahl der für die Webanwendung verwendeten Software erläutert.

5.5.1 Webserver

Als Webserver kam von Anfang an nur **Apache HTTP Server** in Frage.
Ein großer Vorteil des Apache ist, dass er wie alle Produkte der Apache Software Foundation, kostenlos als Open Source unter der Apache-Lizenz verfügbar ist. Der Apache bietet die Möglichkeit, mittels serverseitiger Skriptsprachen Webseiten dynamisch zu erstellen. Unterstützt werden dabei unter anderem PHP und Perl. Darüber hinaus ist Apache für Unix sowie auch für Windows-NT Plattformen verfügbar.

5.5.2 Datenbank

Die Wahl des Datenbank Management Systems fiel auf **MySQL**.
MySQL ist eine in der verwendeten Version unter der GPL stehende Datenbank. Sie war von Anfang an für große Datenmengen, hohe Verfügbarkeit, extreme Stabilität und sehr gute Performance ausgelegt. MySQL ist ein vielseitigen DBMS, das sich hervorragend an das jeweilige Anforderungsprofil anpassen kann. MySQL wird sehr häufig zusammen mit dem Webserver Apache und PHP eingesetzt.

5.5.3 Webanwendung

Um die Web-Oberfläche Benutzerfreundlich zu gestalten setzt sich diese aus einer Kombination von verschiedenen Programmen zusammen. Das Grundgerüst ist mit **HTML** programmiert. Um dynamische Abfragen und Anzeigen zu gestalten wurde **PHP** verwendet.
Durch den Einsatz der serverseitigen Skriptsprache PHP wird die Webanwendung zur eigentlichen Schnittstelle zwischen der Datenbank und dem restlichen System.
In der Version 5 bringt PHP die sogenannte SOAP Extension mit sich, mit der Webservices realisiert werden können.

5.5.4 Betriebssystem

Das verwendete Betriebssystem zur Entwicklung und zum Betrieb des Webservers ist Microsoft Windows XP. Ein Grund für die Auswahl von Windows XP war das Vorhandensein des Softwarepaketes XAMPP für Windows (siehe folgendes Kapitel) sowie die Erfahrung im Umgang mit dem weitverbreiteten Betriebssystem.

5.5.5 XAMPP für Windows

Ein großer Vorteil der in den Kapiteln 5.5.1 - 5.5.3 aufgeführten Softwareprodukte liegt in dem Softwarepaket XAMPP.

XAMPP setzt sich hauptsächlich aus folgenden Teilen zusammen:

- X: Windows oder Linux
- A: Apache HTTP Server
- M: MySQL
- P: PHP
- P Perl

Folglich bildet das Paket die ideale Entwicklungsumgebung für das zu entwickelnde System. XAMPP ermöglicht eine einfache und rasche Installation. Die Philosophie hinter XAMPP ist Anfängern und Profis einen einfachen Einstieg in die Welt des Apache zu ermöglichen. XAMPP ist so vorkonfiguriert, dass möglichst alle Features von Apache und Co aktiviert sind. So, wie es z. B. für einen Entwickler am angenehmsten ist. XAMPP ist ebenso wie seine Komponenten kostenlos und kann unter http://www.apachefriends.org/de/xampp.html heruntergeladen werden.

6 Theoretische Grundlagen

In diesem Kapitel soll grundlegendes Wissen für die Technologien, die in diesem Projekt angewandt werden, übermittelt werden.

6.1 RFID (Radio Frequency Identification)

RFID ist die Abkürzung für Radio Frequency Identification, was so viel heißt, wie die Identifikation über Funk. Man kann somit Daten auf einem Transponder berührungslos und ohne Sichtkontakt lesen und speichern. Dieser Transponder kann an Objekten angebracht werden, welche dann anhand der darauf gespeicherten Daten automatisch und schnell identifiziert werden können. In unserem Projekt können wir die Schul-PCs mit dieser Technik also drahtlos identifizieren.
In den folgenden Abschnitten wird auf diese Technik genauer eingegangen.

6.1.1 Grundsystem

Das RFID-Grundsystem besteht im Wesentlichen aus dem Lesegerät dem und der zugehörigen Middleware (z.B.: PC mit Reader-Ansteuerungssoftware), die das System im gewünschten Bereich (z.B.: PC Inventarisierung) integriert.

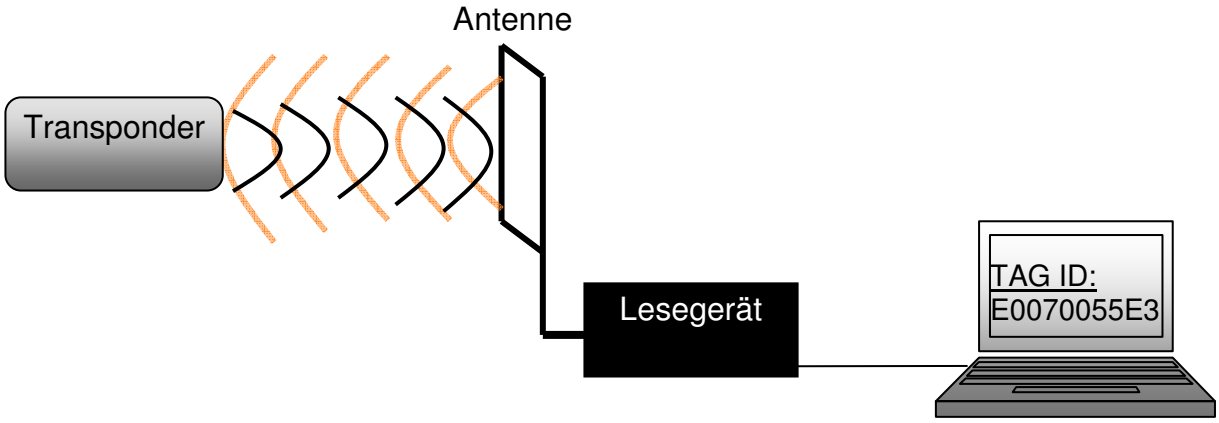


Abbildung 6.1 RFID-Grundsystem

6.1.2 Begriffsbeschreibung

Lesegerät (auch RFID-Reader genannt)
Das Lesegerät erhält einen Befehlsstring vom PC und führt den entsprechenden Befehl aus. Der Befehl kann die Einstellungen des Readers ändern (z.B.: Modulationstiefe, Wortbreite, Baudrate usw.) und die Peripherie ansteuern (z.B.: LEDs, Transponder, usw.).
Jedes Lesegerät ist im Grunde auch fähig zu schreiben, trotzdem wird es RFID-Reader genannt.
Die Ansteuerung der Transponder erfolgt über die Antenne, die an das Lesegerät angeschlossen ist. Über diese Antenne wird ein elektromagnetisches Feld erzeugt, welches die passiven Transponder mit Energie versorgt, und die Anweisungen und Daten übertragen.

Transponder (auch RFID-Etikett, -Chip, -Tag, -Label oder Funketikett genannt)
Transmitter + Responder
Die Transponder sind die Datenträger die kabellos über die Antennen ausgelesen werden können.

6.1.3 Typischer Abfragevorgang

1.) Die Readeransteuerungs-Software sendet über die RS232-Schnittstelle des Rechners einen Befehl an das Lesegerät
2.) Das Lesegerät führt den Befehl nach erfolgreicher Fehlerprüfung aus
3.) Betrifft der Befehl einen RFID-Transponder, so wird die Antenne des Lesegeräts, je nach Einstellung, angesteuert um die Befehle an den Transponder weiterzugeben
4.) Der Prozessor des Transponders führt die erhaltenen Befehle aus und gibt, je nach Befehl, die entsprechende Antwort an das Lesegerät zurück
5.) Das Lesegerät schickt die Antwort der Transponders an den PC weiter
6.) Die Readeransteuerungs-Software kontrolliert den Antwortstring auf Fehler und gibt das Ergebnis aus

6.1.4 Frequenzbereiche

Tabelle 6.1 RFID-Frequenzbereiche

Parameter	Niedrig-frequenz	Hochfrequenz	Ultrahoch-frequenz	Mikrowelle
Frequenz	125 – 134 kHz	13,56 MHz	868 bzw. 915 MHz	2,45 bzw. 5,8 GHz
Leseabstand	bis 1,2 m	bis 1,2 m	bis 4 m	bis zu 15 m (in Einzelfällen bis zu 1 km)
Lesegeschwin-digkeit	langsam	je nach ISO-Standard	schnell	sehr schnell (aktive Transponder)
Feuchtigkeit	kein Einfluss	kein Einfluss	negativer Einfluss	negativer Einfluss
Metall	negativer Einfluss	negativer Einfluss	kein Einfluss	kein Einfluss
Weltweit akzeptierte Frequenz	ja	ja	teilweise (EU/USA)	teilweise (nicht EU)
Heutige ISO-Standards	11784/85 und 14223	14443, 15693 und 18000	14443, 15693 und 18000	18000
Typische Transponder-Bautypen	Glasröhrchen-Transponder, Transponder im Plastikgehäuse, Chipkarten Smart Label, Chipkarten	Smart Label, Industrie-Ttransponder	Smart Label, Industrie-Ttransponder	Großformatige Transponder
Beispielhafte Anwendungen	Zutritts- und Routen-kontrolle, Wegfahr-sperren, Wäsche-reinigung, Gasablesung	Wäschereinigung, Asset Management, Ticketing, Tracking & Tracing, Pulk-Erfassung	Paletten-erfassung, Container-Tracking	Straßenmaut, Container-Tracking

6.1.5 Einsatzgebiete

Das Spektrum der Anwendungsgebiete ist weit gefächert. Besonders in den folgenden Bereichen gewinnt die RFID-Technologie zunehmend an Bedeutung:

- Supply Chain Management (Logistik)
- Zutrittskontrolle (z.B.: Skilift, Therme)
- Electronic Article Surveillance (momentan 1bit-Transponder, also nur „da“ oder „nicht da“)
- Tieridentifikation
- drahtlose Schnittstelle f. Sensoren
- **Inventarisierung (z.B.: PCs an der HTL)**

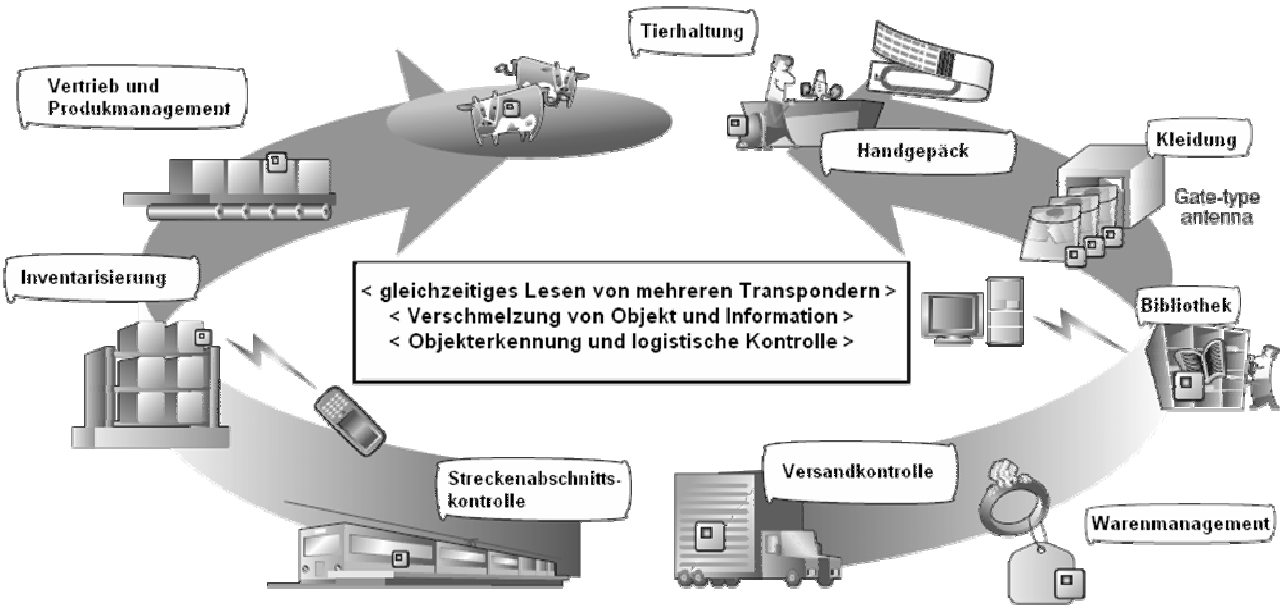


Abbildung 6.2 Einsatzgebiete

6.1.6 Transponderbauformen

Entsprechend der Einsatzgebiete, gibt es verschiedene Transponderbauformen. Im Texas Instruments Evaluation Kit sind einige Bauformen enthalten:

Abbildung 6.3 Transponderbauformen

Tabelle 6.2 – Transponderbauformen

Nr.	Typ	Frequ.	Anwendungsbereich
1	Transponder-Inlay	HF/LF	Warenidentifikation
2	Transponder im Kreditkartenformat	HF/LF	Zugangskontrolle – Firma, Auto, …
3	Schlüsselanhänger	LF	Zugangskontrolle – Firma, Auto, …
4	Transponder im Kunststoffgehäuse	LF	Warenidentifikation – Kleidung
5	selbstklebendes HF-Label	HF/LF	Warenidentifikation – PCs, Medikamente
6	HF-Identifikationslabels	HF/LF	Wäscherei
7	LF-Glastransponder	LF	Tieridentifikation (Transponder unter die Haut)

6.1.7 Energieversorgung

Ein deutliches Unterscheidungs-Merkmal stellt die Art der Energieversorgung der RFID-Transponder dar. Man unterscheidet zwischen aktiven und passiven Transpondern.

Passive Versorgung:

Kleine batterielose RFID-Transponder besitzen keine eigene Energieversorgung und müssen ihre Versorgungsspannung durch Induktion aus den Funksignalen der Basisstationen gewinnen. Dies reduziert zwar die Kosten und das Gewicht der Chips, gleichzeitig verringert es aber auch die Reichweite.

Diese Art von RFID-Transpondern ist im Stückpreis sehr günstig, was in folgenden Sektoren ausschlaggebend ist:
Produktauthentifizierung bzw. -auszeichnung, Zahlungssysteme und Dokumentenverfolgung da die Kosten pro Einheit hier ausschlaggebend sind.
Alle Transponder, die in diesem Projekt verwendet wurden (Transponder des TI-Eval-Kits eingeschlossen), werden passiv über das Lesegerät mit Energie versorgt.

Aktive Versorgung:

RFID-Transponder mit eigener Energieversorgung erzielen eine erheblich höhere Reichweite und besitzen einen größeren Funktionsumfang, verursachen aber auch erheblich höhere Kosten pro Einheit.

Deswegen werden sie dort eingesetzt, wo die zu identifizierenden oder zu verfolgenden Objekte eine lange Lebensdauer haben:
z.B. bei wieder verwendbaren Behältern in der Containerlogistik oder bei Lastkraftwagen im Zusammenhang mit der Mauterfassung (Go-box für Österreichs Straßen).

Grundlegend kann man die Transponder also folgendermaßen trennen:

Tabelle 6.3 Transponder-Unterschiede: aktiv - passiv

Eigenschaft	aktiv	passiv
Bauform	groß	klein
Gewicht	hoch	gering
Senderreichweite	weit	kurz
Speicherplatz	viel	wenig
Anschaffungspreis	hoch	gering
Wartungsaufwand	hoch	gering

6.1.8 Transponderaufbau

Wir wählten selbstklebende HF-Labels für unseren Anwendungszweck (siehe Realisierung).

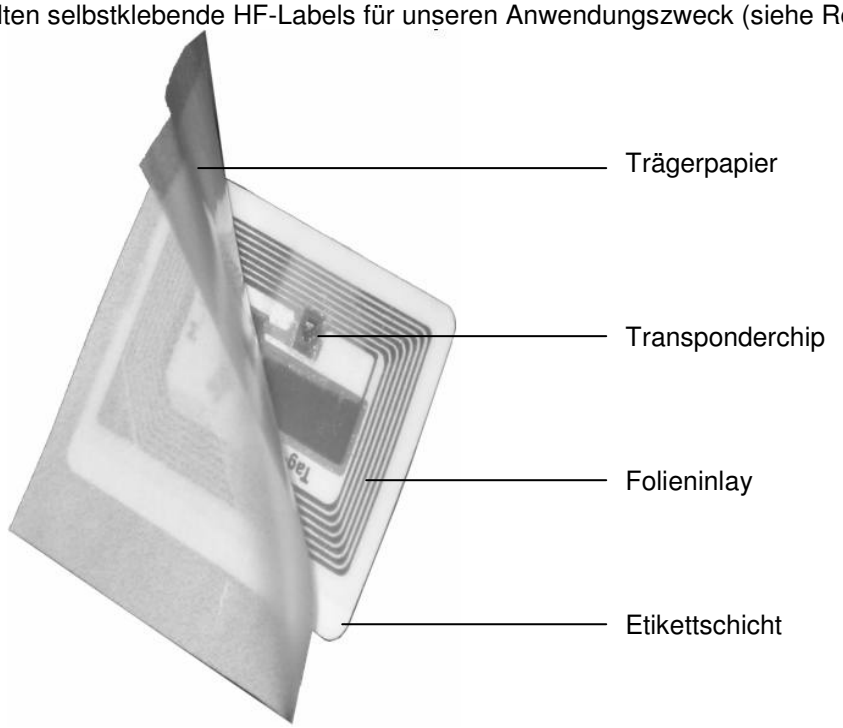


Abbildung 6.4 Transponderaufbau

Trägerpapier:	Das Trägerpapier (Liner) dient zum Schutz der Klebeschicht und des Inlays und zur Befestigung des Transponders solange dieser nicht anderswo angeklebt wird.
Transponderchip:	Der Mikrocontroller und der Speicher des Transponders sind in diesem Chip integriert. Der Chip (ca. 4mm² groß) ist direkt an der Antenne (dem Inlay) angeschlossen, über welche er mit Energie versorgt wird, Befehle entgegennimmt und Daten zurücksendet.
Folieninlay:	Das Inlay ist eine Kupferschicht, die als Antenne dient. Über sie, wird Energie aus dem elektromagnetischem Feld des Lesegerätes gewonnen und Daten übertragen.
Etikettschicht:	Die Etikettschicht ist eine Kunststoff bzw. Papierschicht, auf welcher sich das Folieninlay, der Transponderchip und der Klebstoff befinden.

6.1.9 Standardspeicher

Auf ISO 15693-Transpondern gibt es verschiedene Speicherbereiche:

UID (Unique ID)
8 Byte
Die Unique ID wird vom Hersteller angegeben und ist die wichtigste Kennummer im RFID-Bereich. Die Nummer ist einzigartig und wird bei Tranpondersuchen zurückgegeben. Über diese Nummer kann der Transponder explizit angesprochen werden, womit Antikollisionsverfahren möglich sind. Die UID kann also nicht geändert werden.

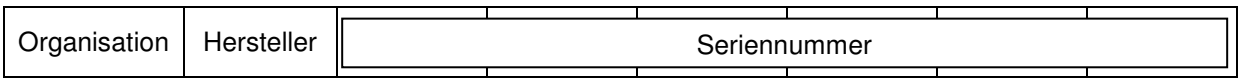


Abbildung 6.5 Aufbau der ISO15693 UID

- Organisation: Gibt an welche Organisation die Transponder genehmigt hat:
  - $E0_{HEX}$: ISO/IEC7816-6/AM1 (*am häufigsten verwendet*)
  - $E2_{HEX}$: EPCglobal

- Hersteller: Gibt den Hersteller des RFID-Mikrochips an:

Tabelle 6.4 RFID-Mikrochiphersteller mit ISO15693 UID-Nummer

Organisation Hersteller	$E0_{HEX}$ + Kennnummer	Organisation Hersteller	$E0_{HEX}$ + Kennnummer
Motorola	01_{HEX}	Mitsubishi	$0D_{HEX}$
ST Microelectronics	02_{HEX}	Samsung	$0E_{HEX}$
Hitachi	03_{HEX}	Hyundai	$0F_{HEX}$
Philips	04_{HEX}	LG	10_{HEX}
Infineon	05_{HEX}	Emosyn	11_{HEX}
Cylinc	06_{HEX}	Inside	12_{HEX}
Texas Instruments	07_{HEX}	Orga	13_{HEX}
Fujitsu	08_{HEX}	Sharp	14_{HEX}
Matsushita	09_{HEX}	Atmel	15_{HEX}
NEC	$0A_{HEX}$	EM-Marin	16_{HEX}
Oki	$0B_{HEX}$	KSW Microtec	17_{HEX}
Toshiba	$0C_{HEX}$	XICOR	19_{HEX}

Uns wurden uns Tags von Texas Instruments (mit Texas Instruments Chip) und von SmartTEC (mit Philips Chip) zu Verfügung gestellt.

- Seriennummer: 6 Bytes, die vom Hersteller gewählt werden

AFI (Application Family ID)
1 Byte
Die Application Family ID gibt an, für welchen Zweck der Transponder genutzt wird. Sie kann also frei vom Benutzer gewählt und verändert werden.

Tabelle 6.5 AFI-Bedeutung

AFI Most Significant Nibble	AFI Least Significant Nibble	Meaning LRI64 Devices respond from	Examples / Note
0	0	All families and sub-families	No applicative preselection
x	0	All sub-families of family X	Wide applicative preselection
x	y	Only the Yth sub-family of family X	
0	y	Proprietary sub-family Y only	
1	0, y	Transport	Mass transit, Bus, Airline,...
2	0, y	Financial	IEP, Banking, Retail,...
3	0, y	Identification	Access Control,...
4	0, y	Telecommunication	Public Telephony, GSM,...
5	0, y	Medical	
6	0, y	Multimedia	Internet services....
7	0, y	Gaming	
8	0, y	Data Storage	Portable Files...
9	0, y	Item Management	
A	0, y	Express Parcels	
B	0, y	Postal Services	
C	0, y	Airline Bags	
D	0, y	RFU	
E	0, y	RFU	
F	0, y	RFU	

Für die HTL PC-Inventarisierung haben wir 90_{HEX}, also Item Management, gewählt.

DSFID (Data Storage Format ID)
1 Byte
Die DSFID gibt an, in welchem Format die Daten auf dem Transpoder gespeichert sind. Dieses Byte kann ebenfalls vom Anwender bestimmt und verändert werden.
Wir wählten willkürlich 91_{HEX} für PC-Inventarisierungsdaten.

6.1.10 weiterer Speicherbereich, Veränderbarkeit

typisch 128 bis 256 Bytes

Zusätzlich zu den bereits genannt Daten, kann der Transponder einen weiteren Speicherbereich haben. Die Größe dieses Speichers bestimmt der Hersteller. Die realisierte Speichergröße kann vom Transponder abgefragt werden. Unsere Transponder haben 112 (SmartTEC) bzw. 256 (TI) Bytes zusätzlichen Speicher integriert.
Der Speicher wird blockweise angesprochen. Wenn man also etwas auf diesen Speicher schreiben oder von ihm lesen will, so kann man mit einem Befehl immer nur die Blöcke ansprechen. Ein Block entspricht entweder 4 (typisch) oder 256 Bytes (sehr selten). Will man z.B. das 7te Byte schreibschützen so muss man dies am gesamten 2. Block (also die Bytes 4 bis 7) machen.

Veränderbarkeit des Speichers

Read-Only-Transponder

Diese Transponder erlauben nur das Auslesen des Speichers. Häufig wird nur eine Seriennummer gespeichert. Der Speicher fällt daher sehr klein aus, genügt aber um z.B.: Waren zu unterscheiden.

Vorteile: sehr günstig in der Produktion, vor unbefugter Manipulation geschützt
Nachteile: auch nicht von Befugten änderbar

Einsatzgebiete: Warenmanagement, Inventarisierung, Tierhaltung

Write-Once-Read-Many Transponder (WORM)

WORM-Transponder kann man nur einmal beschreiben und beliebig oft auslesen.

Vorteile: Der Anwender kann den Speicher einmal beschreiben und ab da an ist der Speicher vor jedem anderen Schreibzugriff geschützt.
Nachteil: bei Umstellungen, nicht mehr änderbar

Einsatzgebiete: z.B.: Warenmanagement - Artikelbeschreibung und Preis kann somit vom Handel gespeichert und nicht vom Käufer manipuliert werden.
wird mit dem gleichen Prinzip auch bei Zutrittskontrollen verwendet (Skipass)

Read-Write-Transponder
Das Tag erlaubt mehrmaliges Lesen und Beschreiben des Speichers.

Vorteile: Speicherinhalt beliebig veränderbar, kann aber auch schreibgeschützt werden (→WORM)
Nachteile: teurer in der Herstellung, auch Unbefugte könnten auf ungeschützte Tags schreiben

Einsatzgebiete: bargeldlose Bezahlung, Zutrittskontrolle (z.B.: Therme), Inventarisierung mit Speicherung von sich ändernden Zusatzangaben (Reparaturen)

6.1.11 Transponderzustände

Ein Transponder kann mit bestimmten Befehlen in verschiedene Modi versetzt werden:
1.) Silent: Der Transponder reagiert auf keine allgemeine Anfrage mehr. Er muss mit seiner UID angesprochen werden.
2.) Selected: Der Transponder reagiert nur auf Befehle, die das selected-Flag gesetzt haben.
3.) Normal: Der Transponder reagiert auf alle Befehle, bei denen das selected-Flag nicht gesetzt ist.

6.2 XML (eXtensible Markup Language)

Die Extensible Markup Language (engl. für „erweiterbare Auszeichnungs-Sprache“), ist ein Standard zur Erstellung maschinen- und menschenlesbarer Dokumente in Form einer Baumstruktur, der vom World Wide Web Consortium (W3C) definiert wird. XML definiert dabei die Regeln für den Aufbau solcher Dokumente. Für einen konkreten Anwendungsfall ("XML-Anwendung") müssen die Details der jeweiligen Dokumente spezifiziert werden. Dies betrifft insbesondere die Festlegung der Strukturelemente und ihre Anordnung innerhalb des Dokumentenbaums. XML ist damit ein Standard zur Definition von beliebigen in ihrer Grundstruktur jedoch stark verwandten Auszeichnungssprachen. Eine Sprache zur Definition anderer Sprachen nennt man Metasprache. XML ist eine vereinfachte aber erweiterbare (eXtensible) Teilmenge von SGML (Standard Generalized Markup Language).

Die prominenteste der Sprachen, die mit Hilfe von XML definiert werden, ist derzeit zweifellos die neue Fassung von HTML, XHTML genannt. Aber auch andere bekannte Sprachen, wie etwa WML (Beschreibungssprache für das WAP-Protokoll, das Internet-Inhalte aufs Handy-Display bringt), kommen ins Gespräch. Welcher Erfolg all diesen Sprachen beschieden ist, entscheidet der Markt. Daneben bietet XML jedoch auch die Möglichkeit an, völlig neue Sprachen zu definieren, die für eigene Datenstrukturen optimal angepasst sind, und die aber trotzdem nicht proprietär sind, sondern einem standardisierten Regelwerk folgen.

Die Leistung von XML besteht darin, dass man mit den Konzepten und Regeln, die es bereitstellt, eigene Auszeichnungssprachen definieren kann, die ähnlich funktionieren wie HTML. All diese Sprachen bestehen immer wieder aus Elementen, markiert durch Tags, deren Verschachtelungsregeln, und aus Attributen mit erlaubten Wertzuweisungen. Daneben gibt es Regeln, wie man solche Sprachen - in XML auch Namensräume (Namespace) genannt - in andere Sprachen importieren und somit eine Sprache innerhalb einer anderen Sprache benutzen kann.

Ein großer Vorteil von XML ist, dass die Dateien leicht lesbar (für Software und Mensch) sind und deshalb kaum Codier- und Decodieraufwand für die Erstellung von XML-Dateien notwendig ist.

6.2.1 XML-Beispiel

Beschreibt einen Scriptbefehl:

```
<projekt sprache="ANSIC" name="RFID-invent" typ="exec" stand="18-03-
2006">
  <modul name="script" stand="22-03-2006" pfad="C:\RFID\script.rfs">
   <funktion name="auslesen" stand="22-03-2006">
    <beschreibung>
     Auslesen aller Tag-Informationen
    </beschreibung>
   </funktion>
  </modul>
</projekt>
```

Beschreibt die Funktionen und Variablen eines Programms (z.B. bei Webservices):

```
<message name="TagDataRequest">
 <part name="id" type="xsd:string"/>
 <part name="ip" type="xsd:string"/>
 <part name="name" type="xsd:string"/>
 <part name="location" type="xsd:string"/>
 <part name="group" type="xsd:string"/>
</message>

<message name="TagDataResponse">
 <part name="ack" type="xsd:string"/>
</message>

<portType name="SoapServerPortType">
 <operation name="sendTagData">
  <input message="tns:TagDataRequest"/>
  <output message="tns:TagDataResponse"/>
 </operation>
</portType>
```

Beschreibt die Informationen eines RFID-Transponders:

```
<SOAPSDK4:RFIDget xmlns:SOAPSDK4="urn:RFIDproject">
 <RFID>e0 07 00 00 06 d5 50 88</RFID>
 <type>ISO 16593 HF</type>
 <content>hello world!</content>
 <about>PC Reparatur</about>
 <INr>GK06-20</INr>
 <birth>16.08.2005</birth>
 <lastchange>09.09.2005</lastchange>
</SOAPSDK4:RFIDget>
```

Die Beschreibung über XML ist natürlich nicht auf den Bereich der Technik begrenzt. Eigentlich alles, was irgendwelche benennbaren und beschreibbaren Strukturen aufweist kann mit XML beschrieben werden. Mit welcher Software man diese Daten visualisieren, abspielen oder anderweitig verarbeiten kann, ist damit noch nicht festgelegt. Es geht zunächst nur mal darum, Daten sinnvoll zu strukturieren und vollständig zu beschreiben.
Welche Elemente und Attribute erlaubt sind und wie diese Angeordnet sein dürfen wird in der DTD (Document Type Definition) beschrieben. Diese werden in einer anderen Syntax definiert. Wir mussten uns aber nicht damit beschäftigen, da bereits DTDs für SOAP (siehe nächstes Kapitel) vorhanden sind.

Die allen Beispiele enthalten ganz unterschiedliche Elemente, Attribute, Wertzuweisungen und typische Verschachtelungen. Gemeinsam ist ihnen jedoch, dass sie offensichtlich aus bestimmten erlaubten Elementen, Attributen, Wertzuweisungen und Regeln zur Verschachtelung bestehen.
Diese Beispiele sind nicht komplett frei erfunden. Wir haben XML für SOAP und somit für die Beschreibung eines Webservices und Übertragung der Daten benötigt.

6.3 SOAP

SOAP (ursprünglich für *Simple Object Access Protocol*) ist ein Protokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und Remote Procedure Calls durchgeführt werden können. SOAP stützt sich auf die Dienste anderer Standards, XML zur Repräsentation der Daten und Internet-Protokolle der Transport- und Anwendungsschicht (vgl. TCP/IP-Referenzmodell) zur Übertragung der Nachrichten. Die gängigste Kombination ist SOAP über HTTP und TCP. Ursprünglich war SOAP die Abkürzung für Simple Object Access Protocol (*Einfaches Objekt-Zugriffs-Protokoll*), seit Version 1.2 ist SOAP jedoch offiziell keine Abkürzung mehr, da es erstens keineswegs einfach (Simple) ist und da es zweitens nicht (nur) dem Zugriff auf Objekte (Object Access) dient.

6.3.1 Struktur einer SOAP-Nachricht

Eine SOAP-Nachricht ist nach dem Head-Body-Pattern modelliert. Ein SOAP-Envelope ist ein Container, der ein (optionales) Header-Element und ein Body-Element enthält. Weiterhin wird hier der verwendete Namensraum festgelegt.
Im Head-Bereich der Nachricht werden die Metainformationen der Nachricht untergebracht. Diese können Informationen über das Routing der Nachricht, über eine eventuelle Verschlüsselung und / oder über die Zugehörigkeit zu einer Transaktion umfassen.
Im Body der Nachricht sind, wie auch bei HTML, die Nutzdaten untergebracht. Diese Daten müssen vom Empfänger der Nachricht interpretiert werden, mögliche Zwischenstationen können diese auch ignorieren. Die Daten können dabei unter anderem für entfernte Methodenaufrufe, (Fehler-)Meldungen oder reine Daten (z. B. Abbildung einer Klassenstruktur) stehen.
Anschließend können mögliche Anhänge folgen, diese werden abhängig von dem Transportprotokoll an die Nachricht angehängt. Binärdateien (Sound, Video, Grafik etc.) können durch Nutzung von MIME-Mechanismen angebunden werden.

6.3.2 SOAP über HTTP

Ein per HTTP übertragenes SOAP-Paket hat folgende Struktur:
(die Rahmen sind nicht Bestandteil, sondern verdeutlichen lediglich die Struktur):

```
+-------------------------------------------------HTTP-Header--+
| POST /realtimedata/transponders HTTP/1.1                     |
| Host: htlsoapserver                                          |
| Content-Type: text/xml; charset=utf-8                        |
| ...                                                          |
+--------------------------------------------------------------+

  <?xml version="1.0" encoding="UTF-8"?>
+-------------------------------------------------SOAP-Envelope--+
| <SOAP-ENV:Envelope                                           |
|  xmlns:SOAP-ENV="http://..."                                 |
|  ...                                                         |
| +-----------------------------------SOAP-Header (optional)--+ |
| | <SOAP-ENV:Header>                                        | |
| |   <t:transaction                                         | |
| |    xmlns:t="..." ...                                     | |
| | </SOAP-ENV:Header>                                       | |
| +----------------------------------------------------------+ |
| +-----------------------------------------------SOAP-Body--+ |
| | <SOAP-ENV:Body>                                          | |
| |   <m:getRFID xmlns:m="rfid-uri">                         | |
| |     <room>net2</room>                                    | |
| |   </m:getRFID>                                           | |
| | </SOAP-ENV:Body>                                         | |
| +----------------------------------------------------------+ |
| </SOAP-ENV:Envelope>                                         |
+--------------------------------------------------------------+
```

Im beim HTTP-Protokoll üblichen HTTP-Header würde bei HTML-Dateien als Content-Type "text/html" definiert. Für SOAP muss "text/xml" definiert sein, da XML-Dateien übertragen werden. Der HTTP-Header endet mit einer leeren Zeile (CRLF).
Anschließend folgt der XML-Teil, beginnend mit "<?xml ...".
Der XML-Teil besteht hauptsächlich aus dem so genannten "Envelope"-XML-Element. Dieses wiederum enthält die beiden XML-Elemente "Header" und "Body", wobei das "Header"-Element auch entfallen kann.
Das "Body"-Element muss enthalten sein. Hierin wird der eigentliche Inhalt plaziert, also die Daten, eine Meldung, eventuell eine Fehlermeldung oder ein RPC-Funktionsaufruf.

An diesem einfachen Beispiel wird bereits deutlich, dass SOAP die Menge der zu übertragenden Daten erheblich aufbläht. Dies ist nicht nur ein Problem der beanspruchten Netzwerkbandbreite, auch das Generieren und Auswerten (Parsen) der Nachrichten erfordert sehr viel Rechenzeit, verglichen mit anderen, einfacheren RPC-Protokollen.
Die Vorteile der Strukturierung und einfacheren Lesbarkeit erleichtern die saubere Implementierung in den Anwendungen. Ein weiterer Vorteil ist, dass das die Daten aufgrund des XML-Formats und der Nutzung von TCP/IP als Transportschicht nur von sehr wenigen Firewalls geblockt wird.
Wie die SOAP-Nachricht schlussendlich aussehen darf/soll wird im WSDL-Dokument bestimmt.

6.3.3 WSDL (Web Services Description Language)

Die WSDL definiert eine plattform-, programmiersprachen- und protokollunabhängige XML-Spezifikation zur Beschreibung von Netzwerkdiensten (Web Services) zum Austausch von Nachrichten.

WSDL ist eine Metasprache, mit deren Hilfe die Funktionalität eines Web Service beschrieben werden kann. Es werden im Wesentlichen die Operationen definiert, die von außen zugänglich sind, sowie die Parameter und Rückgabewerte dieser Operationen. Im einzelnen beinhaltet ein WSDL-Dokument funktionelle Angaben zu:

- der Schnittstelle
- Zugangsprotokoll und Details zum Deployment
- Alle notwendigen Informationen zum Zugriff auf den Service, in maschinenlesbarem Format

Beschreibungselemente

Services werden durch sechs XML-Hauptelemente definiert:

- Datentypen (*types*):
  Definition der Datentypen, die zum Austausch der *messages* benutzt werden.
- Mitteilungen (*message*):
  Abstrakte Definitionen der übertragenen Daten, bestehen aus mehreren logischen Teilen, von denen jedes mit einer Definition innerhalb eines Datentypsystems verknüpft ist.
- Port-Typen (*portType*):
  Eine Menge von abstrakten Arbeitsschritten (vier Typen von ausgetauschten Mitteilungen):
  - One-way: Der Service bekommt eine Input-Message vom Client.
  - Request-response: Der Service bekommt einen Request (Input-Message) vom Client und sendet eine Antwort (Output-Message).
  - Solicit-response: Der Service sendet eine Message und erwartet eine Antwort vom Client.
  - Notification: Der Server sendet eine Output-Message.
- Bindung (*binding*)
  Bestimmt das konkrete Protokoll und Datenformat für die Arbeitsschritte und Mitteilungen, die durch einen bestimmten Port-Typ gegeben sind.
- Ports (*port*)
  Spezifiziert eine Adresse für eine Bindung, also eine Kommunikationsschnittstelle, üblicherweise eine URI
- Services (*service*)
  Fassen eine Menge von verwandten Ports zusammen

WSDL wird häufig in Kombination mit SOAP und dem XML-Schema verwendet, um Web Services im Internet anzubieten. Ein Client, der einen Webservice aufruft, kann WSDL lesen, um zu bestimmen, welche Funktionen auf dem Server verfügbar sind. Alle verwendeten speziellen Datentypen sind in der WSDL-Datei in XML-Form eingebunden. Der Client kann nun SOAP verwenden, um eine in WSDL gelistete Funktion letztlich aufzurufen.
Genau das wurde von uns realisiert. Unser WSDL-Dokument und die Realisierung der Übertragung der Daten über SOAP kann im Kapitel 8.10 „Webinterface“ nachgelesen werden.

6.4 HTML

6.4.1 Allgemeines

Die Hypertext Markup Language (HTML) beschreibt Dokumente als Hypertext, typischerweise beschreibt sie Webseiten. HTML beschreibt Informationen im Sinn einer Auszeichnungssprache und wurde vom World Wide Web Consortium bis Version 4.01 weiterentwickelt, die Weiterentwicklung geschieht allerdings seither als XHTML.
Mehr Informationen über HTML findet man unter http://de.selfhtml.org/html/index.htm.

6.4.2 Grundgerüst

Eine gewöhnliche HTML-Datei besteht grundsätzlich aus folgenden Teilen:

- Dokumenttyp-Deklaration (Angabe zur verwendeten HTML-Version)
- Header (Kopfdaten. z.B. Angaben zu Titel u.ä.)
- Body (Körper - anzuzeigender Inhalt, also Text mit Überschriften, Verweisen, Grafikreferenzen usw.)

Im folgenden Teil wird das Grundgerüst einer XHTML Datei gezeigt.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Text des Titels</title>
</head>
<body>
</body>
</html>
```

6.4.3 Tabellen

Tabellen sind ein wichtiger Bestandteil von HTML. Damit die Funktion verdeutlicht werden kann folgt hier ein Beispiel:

HTML Code

```
<table>
      <tr>
            <th>Titel 1</th>
            <th>Titel 2</th>
      </tr>
      <tr>
            <td>
            Spalte 1, Zeile 1
            </td>
            <td>
            Spalte 2, Zeile 1
            </td>
      </tr>
      <tr>
            <td colspan=2>
            Spalte 1 bzw. 2, Zeile 2
            </td>
      </tr>
</table>
```

Interpretierter HTML Code

Titel 1	Titel 2
Spalte 1, Zeile 1	Spalte 2, Zeile 1
Spalte 1 und 2, Zeile2	

Mit dem Parameter <table> wird eine Tabelle geöffnet, mit dem Ausdruck </table> wird diese wieder geschlossen, zwischen diesen Tags können jetzt Spalten und Zeilen bzw. daraus folgende Zellen realisiert werden. Der Tabellenkopf wird mit <th> (table head) und </th> festgelegt.
Durch <tr> bzw. </tr> werden Zeilen dargestellt, welche jetzt in eine beliebige Anzahl an Zellen unterteilt werden. Dies erfolgt mit dem Ausdruck <td> bzw. </td>. In HTML ist es allgemein so, dass alle Tags die geöffnet werden mit dem gleichen Ausdruck und einem enthaltenen „/“ wieder geschlossen werden.

6.4.4 Links

Verweise (Links) sind ein entscheidender Bestandteil jedes Hypertext-Projekts. Mit ihrer Hilfe lässt sich ein Projekt strukturieren. Erst durch Verweise wird aus der losen Dateisammlung ein zusammenhängendes Web-Projekt. Sie können Projekt interne als auch Projekt externe Seiten verlinken. Syntax Beispiel:

```
<a href=Startseite.php>Home</a>
```

Mit <a> wird ein Anker Element gekennzeichnet. Als Wert an das href-Attribut wird das gewünschte Verweisziel zugewiesen.
Einem Link kann gefolgt werden, wenn der Benutzer diesen mit der Maus anklickt. Der Text „Home“ zwischen den Tags wird angezeigt. Die hier angenommene Datei Startseite.php wird aufgerufen.

Es gibt auch Situationen, in denen man einem PHP-Skript Werte übergeben möchte, ohne ein Formular zu verwenden. Dies ist mit folgender Methode möglich:

```
<a href="datei.php?var1=Wert1&var2=Wert2">Linktext</a>
```

Allgemeiner formuliert: An das Verweisziel (in unserem Fall datei.php) wird mit einem „?“ beginnend der Variablenname und mit einem Gleichheitszeichen der Wert angehängt; weitere Werte mit einem „&“ statt „?“ . Es dürfen dabei keine Leerzeichen entstehen.

6.4.5 Grafiken

Um Grafiken in Ihre HTML-Dateien einzubinden, werden Grafikdateien an gewünschten Stellen im HTML-Quelltext referenziert. Diese Aufgabe wird mit dem Tag <img> gelöst:

```
<img src=./bild.jpg alt=“Hier sollte ein Bild sein!“ />
```

Durch diesen Ausdruck wird die Datei bild.jpg eingebunden. Wird diese nicht gefunden wird alternativ der Text „Hier sollte ein Bild sein!“ ausgegeben. Durch den Punkt („.“) in der Pfadangabe wird die Datei im Verzeichnis des HTML Dokumentes erwartet. Damit auf Grafiken in überliegenden Ordnern zugegriffen werden kann, muss dieser Punkt durch „..“ ersetzt werden. Grafikdateien können auch absolut referenziert werden.

6.4.6 Formulare

Ein Formular dient dazu, vom Benutzer eingegebene Daten an den Server zu übermittelt. HTML allein kann mit Formularen nicht besonders viel anfangen, jedoch in Verbindung mit PHP sind diese eine mächtige und sehr wichtige Erweiterung.
Formulare können sehr unterschiedliche Aufgaben haben. So werden sie zum Beispiel eingesetzt:

- um bestimmte, gleichartig strukturierte Auskünfte von Anwendern einzuholen,
- um Anwendern das Suchen in Datenbeständen zu ermöglichen,
- um Anwendern die Möglichkeit zu geben, selbst Daten für einen Datenbestand beizusteuern,
- um dem Anwender die Möglichkeit individueller Interaktion zu bieten, etwa um aus einer Produktpalette etwas Bestimmtes zu bestellen

Definition eines Formular Bereiches:

HTML Code

```
<form name=Formular action=test.php?action=send method=post>
      Formularfeld: <br />
      <input type="text" name="name" size="30" maxlength="30">
      <input type="submit" value="Formular absenden">
</form>
```

Interpretierter HTML Code

Formularfeld:

Formular absenden

Durch den Button (`type="submit"`) wird das Formular abgeschickt. Zunächst wird die Datei test.php aufgerufen, weiters wird die URL durch eine GET-Variable `action` erweitert. Die Übertragungsart ist POST, somit sind die vom Benutzer eingegebenen Daten nicht sichtbar.

6.4.7 Allgemeine Block Elemente

Mehrere Elemente wie Text, Grafiken, Tabellen usw., können in einem gemeinsamen Bereich eingeschlossen werden. Diese allgemeinen Block Elemente bewirken nichts weiter als eine neue Fließtextzeile zu beginnen. Ansonsten haben sie keine Eigenschaften.
Allgemeine Block Elemente sind dazu gedacht, um mit Hilfe von CSS formatiert zu werden. Syntax:

```
<div class="rot">Dieser Text ist rot und fett gedruckt</div>
```

Mit der entsprechenden Style Sheet Klasse „rot" die als class-Attribut angegeben wird erfolgt die Formatierung des Textes zwischen den `<div></div>` Tags.
Die allgemeinen Block Elemente werden zum Aufbau des Web-Projekt Seiten verwendet. Denn sie sind im Gegensatz zu Tabellen Layouts dafür vorgesehen.

6.5 CSS

6.5.1 Allgemeines

Stylesheets sind eine unmittelbare Ergänzung zu HTML. Es handelt sich dabei um eine Sprache zur Definition von Formateigenschaften einzelner HTML-Elemente bzw. Klassen.
Ein weiteres wichtiges Leistungsmerkmal von CSS ist die Möglichkeit, zentrale Formate zu definieren. Im Kopf einer HTML-Datei können zentrale Definitionen zum Aussehen eines Elements notiert werden. Alle Elemente der entsprechenden HTML-Datei erhalten dann die Formateigenschaften, die einmal zentral definiert sind. Stylesheet-Definitionen können auch in einer separaten Datei vorgenommen werden.
Genau wie HTML wird auch CSS vom W3-Konsortium normiert. Es handelt sich also um einen firmenunabhängigen, offen dokumentierten und frei verwendbaren Standard.

Eine umfassende Beschreibung der Cascading Style Sheets findet man unter:
http://de.selfhtml.org/css/index.htm

6.5.2 CSS Format Definition

In einem Stylesheet wird einer vordefinierten HTML Klasse eine bestimmte Formatierung zugewiesen:

```
p{
  font-family: Arial;
  font-size: 12px;
}
```

p steht für die Klasse und in den geschwungenen Klammern stehen die CSS-Eigenschaften mit den dazugehörigen Werten, die der Klasse zugewiesen werden. Es gibt sehr viele vordefinierte Klassen in CSS, hier einige Beispiele:

`body:` In dieser Klasse stehen alle Definitionen die das den Körper des HTML Dokuments betreffen (Hintergrundfarbe, Textfarbe, Schriftgröße, …)

`a:` Diese Klasse dient der Formatierung von Hyperlinks. In diesem Fall gibt es noch weitere Pseudoklassen:

`a:link`	noch nicht verwendete Links
`a:visited`	Besuchte Links
`a:hover`	Verweise bei „MouseOver"
`a:active`	aktive Links

In CSS gibt es außerdem die Möglichkeit, eigene Klassen zu erstellen. Dies geschieht, in dem man vor der Klassenbezeichnung einen Punkt einfügt. Mit der folgenden Klasse können z.B die Inhalte eines allgemeinen Block Elements rot, fett gedruckt und mit der Schriftgröße 14 dargestellt werden.
Das Beispiel aus Kapitel 6.4.7 benutzt diese Klasse:

```
.rot{
  font-size: 14px;
  color: red;
  font-weight: bold;
}
```

6.5.3 Einbindung einer CSS Datei

So bindet man ein Style Sheet in eine HTML Datei ein:

```
<link rel="stylesheet" type="text/css" href="./style/stylesheet.css">
```

6.6 PHP

6.6.1 Allgemeines

PHP ist die Abkürzung für „PHP: Hypertext Preprocessor", eine weitverbreitete Open Source Skriptsprache speziell für Webentwicklungen. PHP läßt sich in HTML einbinden. Die Syntax erinnert an C, Java und Perl und ist einfach zu erlernen. Das Hauptziel dieser Sprache ist es, Webentwicklern die Möglichkeit zu geben, schnell dynamisch generierte Webseiten zu erzeugen. Aber PHP kann für weitaus mehr eingesetzt werden.
Was PHP von clientseitigen Sprachen wie Javaskript unterscheidet, ist dass der Code auf dem Server ausgeführt wird. Wird ein PHP Skript auf einem Server ausgeführt empfängt der Besucher nur das Ergebnis. Das heißt er hat keine Möglichkeit herauszufinden wie der zugrunde liegende Code aussieht. Weiter Informationen sind unter http://www.php.net/docs.php einsehbar.

6.6.2 Einbettung in HTML

Der in HTML eingebettete PHP-Code steht zwischen speziellen Anfangs- und Schlusstags, mit denen man in den PHP-Modus und zurück wechseln kann. Meistens werden die Tags <?php ..... ?> und <? ..... ?> verwendet

Die Einbindungen sind in folgendem Beispiel ersichtlich:

```
<html>
	<head>
		<title>Beispiel</title>
	</head>
	<body>
	<?php
		echo "Hallo, ich bin ein PHP-Skript!";
	?>
	<?
		echo "Hallo, ich bin auch ein PHP-Skript!";
	?>
	</body>
</html>
```

Durch den Befehl „echo“ erfolgt die Ausgabe.

6.6.3 Variablen

Alle Variablen in PHP beginnen immer mit einem Dollarzeichen „$“. Als Bezeichnungen sind Buchstaben und Unterstriche möglich. Die Länge der Bezeichnungen ist unbegrenzt. Weiters wird zwischen Groß- und Kleinschreibung unterschieden. In PHP müssen Variablen nicht deklariert werden, denn sie werden automatisch bei ihrer ersten Verwendung deklariert.

Beispiele für variable Variablen:

```
$action = "search";
$mein_name = "Sebastian Glassner";
$wert = 21;
```

Globale Variablen
Globale Variablen sind in allen Sichtbarkeitsbereichen eines Skripts verfügbar. Sie sind gekennzeichnet durch die Zeichenfolge „$_“.

`$_SERVER[Attribut]`
: Variablen, die vom Webserver gesetzt werden oder anderweitig direkt mit der ausführenden Umgebung des aktuellen Skripts zusammenhängen.
Attribute:
'SERVER_NAME'
Der Name des Server Hosts unter dem das aktuelle Skript ausgeführt wird.
'DOCUMENT_ROOT'
Das Wurzelverzeichnis des Webservers.
'HTTP_USER_AGENT'
String, der den Typ und Namen des Browsers angibt, der auf die Seite zugreift.
'REMOTE_ADDR'
Die IP-Adresse des Rechners, der die aktuelle Seite angefordert hat.

`$_GET`
: Variablen, die dem Skript über HTTP GET geliefert werden.

`$_POST`
: Variablen, die dem Skript über HTTP POST geliefert werden.

`$_SESSION`
: Variablen, die aktuell in der Session eines Skripts registriert sind.

6.6.4 Datentypen

PHP unterstützt folgende Datentypen:

- skalare Typen:
  - Boolean
  - Integer
  - Fließkomma-Zahl (float)
  - String / Zeichenkette
- zusammengesetzte Typen:
  - Array
  - Object
- spezielle Typen:
  - Resource
  - NULL

6.6.5 Kontrollstrukturen

Wie in C gibt es in PHP Kontrollstrukturen, die beim Programmieren nicht wegzudenken wären. Dazu zählen:

Bedingungen:

```
if (Bedingung) {Anweisung;}
  else {Anweisung;}

switch (Variable) {
  case Bedingung1:              //Variable = Bedingung1
       Anweisung;
       break;
  case Bedingung2:              //Variable = Bedingung2
       Anweisung;
       break;
}
```

Schleifen:

```
while (Bedingung) {Anweisung;}          //Kopfgesteuert

do {Anweisung;} while (Bedingung);      //Fußgesteuert

for (i=Startwert; i=Endwert; i=Schrittweite) {Anweisungen;}      //Zählschleife
```

Kommentare:

Kommentare werden mit „//“ (einzeilig) bzw. mit „ /* */ “ gekennzeichnet.

6.6.6 Funktionen

Funktionen dienen im Allgemeinen dem Zusammenfassen mehrerer Befehle zu einem Aufruf. Dadurch werden Programme einfacher lesbar, weil klar ist, wozu ein Befehlsblock dient.
Die Syntax um eigene Funktionen zu programmieren lautet wie folgt:

```
function foo($arg_1, $arg_2, ..., $arg_n) {
  echo "Example function.\n";
  return $retval;
}
```

Die Funktion bekommt die Argumente `$arg_1` bis `$arg_n` übergeben und gibt den Wert der Variablen `$retval` zurück.

6.6.7 Allgemeine Funktionen

Um in PHP die Programmierung zu erleichtern bzw. zu beschleunigen sind im PHP-Standard bestimmte Funktionen vordefiniert.

Auf den folgenden Seiten werden die benutzten PHP Funktionen, alphabetisch geordnet, beschrieben.

array()

Mit der Funktion array() erzeugt man aus gegebenen Werten ein Array.
Zusätzlich gibt es die Möglichkeit, innerhalb eines Arrays wiederum Arrays zu definieren und somit mehrdimensionale Arrays zu erzeugen.
Syntax: `array array(Wert1,Wert2,..,WertN)`

array_merge()

fügt die Elemente von zwei oder mehr Arrays zusammen, indem die Werte des einen an das Ende des anderen angehängt werden. Das daraus resultierende Array wird zurückgegeben.
Syntax: `array array_merge(array array1, array array2 [, array …])`

close()

Schließt ein zuvor mit `dir(pfad)` geöffnetes Verzeichnis. `close()` ist eine Methode der Klasse `dir`.

```
class dir {
      void close ( void )
}
```

Syntax: `$directory->close();`

date()

Mit date() kann man eine Zeitangabe formatieren oder auswerten. Die Zeitangabe wird im Parameter `timestamp` übergeben. Wird dieser Parameter leer gelassen, nimmt die Funktion die aktuelle Zeit.
Der Parameter format ist ein String, der festlegt, welche Informationen über die Zeitangabe benötigt werden..
Syntax: `string date(string format [, int timestamp])`

dir()

Mit `dir()` kann ein Verzeichnis geöffnet werden. `dir()` gehört zur gleichnamigen Klasse `dir`.

```
class dir {
      dir ( string directory )
}
```

Syntax: `$directory->dir(`*Pfad*`);`

eregi()

Mit eregi() kann man in einer Zeichenkette (Zeichenkette) nach Übereinstimmungen suchen, die durch den regulären Ausdruck in Suchmuster angegeben wurden.
Syntax: `int eregi(string Suchmuster, string Zeichenkette [, array regs])`

exit ()

Mit exit() kann man die Ausführung eines Skripts beenden. Dabei ist darauf zu achten, dass das Skript nicht mehr fortgesetzt werden kann.
Syntax: `void exit(void)`

explode()

Zerteilt einen String anhand eines definierten Trennzeichens. Separator ist dabei das Trennzeichen.
Syntax: `array explode (string separator, string string [, int limit])`

fclose ()

Mit fclose() kann man eine offene Datei (fp) schließen. Bei erfolgreichem Schließen der Datei wird true, sonst false zurückgeliefert.
Syntax: `int fclose(int fp)`

file_exists()

Mit file_exists() kann man überprüfen, ob eine Datei (filename) auf dem Server existiert. Im Erfolgsfall gibt diese Funktion true, sonst false zurück.
Syntax: `int file_exists(string filename)`

fopen()

Öffnet eine Datei vom lokalen Dateisystem. Der Parameter mode legt fest, auf welche Weise und für welche Zugriffsarten die Datei geöffnet wird. Folgende Werte gibt es:

a - Öffnet die angegebene Datei nur zum Schreiben und positioniert den Dateizeiger auf das Ende der Datei. Sollte die angegebene Datei nicht existieren, so wird versucht sie anzulegen.
a+ - Öffnet die angegebene Datei zum Lesen und Schreiben und positioniert den Dateizeiger auf das Ende der Datei. Sollte die angegebene Datei nicht existieren, so wird versucht sie anzulegen.
r - Öffnet die angegebene Datei zum Lesen und positioniert den Dateizeiger auf den Anfang der Datei.
r+ - Öffnet die angegebene Datei zum Lesen und Schreiben und positioniert den Dateizeiger auf den Anfang der Datei.
w - Öffnet die angegebene Datei zum Schreiben und positioniert den Dateizeiger auf den Anfang der Datei. Die Länge der Datei wird auf 0 Byte gesetzt. Sollte die angegebene Datei nicht existieren, so wird versucht sie anzulegen.
w+ - Öffnet die angegebene Datei zum Lesen und Schreiben und positioniert den Dateizeiger auf den Anfang der Datei. Sollte die angegebene Datei nicht existieren, so wird versucht sie anzulegen.

Syntax: `int fopen(string filename, string mode [, int use_include_path])`

fwrite ()

Mit fwrite() kann man Binärdaten (string) in eine Datei (fp) schreiben lassen. Optional kann die Länge angegeben werden.
Syntax: `int fwrite(int fp, string string [, int length])`

header()

header() wird zum Senden von HTTP Anfangsinformationen (Headern) benutzt. Damit können zum Beispiel Weiterleitungen auf andere Webseiten realisiert werden.
Syntax: `int header (string string [, bool replace [, int http_reponse_code]])`

include_once()

Der include_once Befehl hat zur Folge, dass die angegebene Datei eingelesen und ausgewertet wird. Der Code wird nur einmal eingebunden. Damit soll verhindert werden, dass zwei identische Funktionen inkludiert werden und somit ein Fehler produziert wird.
Syntax: `include_once("file_name")`

ini_set()

Mit ini_set können Konfigurationsparameter eines SOAP Servers geändert werden.
Syntax:
```
string ini_set ( string varname, string newvalue )
```

read()

Dient zum Auslesen eines mit `dir()` geöffneten Verzeichnisses. `read()` gehört zur Klasse `dir`.
```
class dir {
      string read ( void )
}
```
Syntax:
```
$content = $directory->read()
```

session_destroy()

Mit `session_destroy()` beendet man eine aktuelle Session und löscht alle Daten, die innerhalb der Session genutzt wurden. Zusätzlich wird auch die Session-ID gelöscht.
Syntax:
```
bool session_destroy(void)
```

session_register()

Mit `session_register()` registriert man eine oder mehrere Variablen (name usw.) innerhalb einer Session.
Syntax:
```
bool session_register(mixed name [, mixed ...])
```

session_start()

Mit `session_start()` erstellt man eine neue Session oder führt eine schon bestehende Session fort, deren ID über eine GET-Variable oder ein Cookie übermittelt wurde.
Syntax:
```
bool session_start(void)
```

session_unregister ()

Mit `session_unregister()` hebt man die Registrierung einer Variablen (name) wieder auf.
Syntax:
```
bool session_unregister(string name)
```

shell_exec()

Führt einen Befehl in der Kommandozeile aus.
Syntax:
```
string shell_exec ( string cmd )
```

sizeof()

Die Funktion sizeof() zählt die Anzahl der Elemente eines Arrays und gibt das Ergebnis zurück.
Syntax:
```
int sizeof(array array)
```

strlen()

Mit `strlen()` kann man sich die Länge einer Zeichenkette (str) zurückgeben lassen.
Syntax:
```
int strlen(string str)
```

strtolower()

Mit `strtolower()` kann man den Inhalt einer Zeichenkette (str) in Kleinbuchstaben umwandeln.
Syntax:
```
string strtolower(string str)
```

strtr()

Mit `strtr()` kann man innerhalb eines Strings (`str`) bestimmte Zeichen austauschen lassen. Dabei werden die Zeichen `from` durch die Zeichen `to` ersetzt. Sollten `from` und `to` von unterschiedlicher Länge sein, so werden die überzähligen Zeichen einfach ignoriert.
Syntax: `string strtr(string str, string from, string to)`

time()

Mit time() kann man sich den aktuellen UNIX-Zeitstempel zurückgeben lassen. Dieser Zeitstempel enthält die Anzahl der Sekunden seit Beginn der Unix-Epoche (01.01.1970 um 00:00:00 Uhr).
Syntax: `int time(void)`

trim()

Mit trim() kann man innerhalb eines Strings (str) überflüssige Zeichen am Anfang und Ende entfernen lassen.Überflüssige Zeichen sind: \n, \r, \t, \v, \0, Leerzeichen
Syntax: `string trim(string str)`

6.6.8 Spezielle Funktionen

SOAP Funktionen

PHP bietet ab der Version 5.0 implementierte Funktionen an, die das SOAP Protokoll unterstützen. Diese Funktionen sind allerdings bei der Grundinstallation von XAMPP deaktiviert. Sie müssen durch entsprechende Konfiguration der php.ini Datei aktiviert werden. (siehe Kapitel 8.1.5)

SoapClient()

Erstellt einen neuen SOAP Client. Als Parameter wird der Pfad zur WSDL Datei angegeben.
Syntax: `$client = new SoapClient("file.wsdl");`

SoapServer()

Erstellt einen neuen SOAP Server. Als Parameter wird der Pfad zur WSDL Datei angegeben.
Syntax: `$server = new SoapServer("file.wsdl");`

handle()

Dient zum Abarbeiten einer SOAP Anfrage. handle() gehört zur Klasse SoapServer():

```
class                              SoapServer                                  {
void          handle          (          [string          soap_request]          )
}
```

Syntax: `SoapServer -> handle()`

addFunction()

Fügt eine Funktion zum SOAP Server hinzu. Das heißt der SOAP Client kann bei entsprechender Anfrage auf die Funktion zugreifen. Die hinzuzufügende Funktion muss die in einer WSDL Datei spezifizierten Argumente in der richtigen Reihenfolge enthalten.
addFunction() gehört zur Klasse SoapServer:

```
class                              SoapServer                                  {
void          addFunction          (          mixed          functions          )
}
```

Syntax: `SoapServer -> addFunction ('function_name')`

MySQL Funktionen

Die MySQL Funktionen ermöglichen die Kommunikation eines PHP Skriptes mit einer MySQL Datenbank. Im folgenden Abschnitt sind die wichtigsten Funktionen aufgelistet.

mysql_connect()

Diese Funktion öffnet eine Verbindung zu einem Datenbank Server. Sie ist unbedingt nötig um mit einer Datenbank arbeiten zu können. Als Argumente werden der Hostname, der Benutzername des Datenbankadministrators und das entsprechende Kennwort übergeben. Mit `or die` kann eine Fehlermeldung bei nicht erreichen des Datenbank Servers generiert werden.

Syntax:
```
int mysql_connect(
[string hostname[:port][:/path/to/socket]
[, string Benutzername [, string Kennwort]]]
)
or die (string Fehlermeldung)
```

mysql_select_db()

Mit `mysql_select_db()` wählt man eine Datenbank aus. Als Parameter wird der Datenbankname gefordert.

Syntax:
```
int mysql_select_db(string Datenbankname [, int Verbindungs-Kennung])
```

mysql_query()

Mit `mysql_query()` sendet man eine SQL-Anfrage an einen Datenbankserver. Die Funktion `mysql_query()` liefert im Erfolgsfall true, sonst false zurück.

Syntax:
```
int mysql_query(string Anfrage [, int Verbindungs-Kennung])
```

mysql_num_rows()

Mit `mysql_num_rows()` kann man sich anhand einer Ergebnis-Kennung die Anzahl der Datensätze eines Ergebnisses zurückgeben lassen.

Syntax:
```
int mysql_num_rows(int Ergebnis-Kennung)
```

mysql_fetch_array()

Mit `mysql_fetch_array()` kann man sich anhand einer Ergebnis-Kennung Datensätze in einem assoziativen Array übergeben lassen. Dabei werden die Feldnamen innerhalb der Tabelle als Schlüssel des Arrays genutzt. Im Erfolgsfall liefert diese Funktion den aktuellen Datensatz, sonst wird false zurückgegeben.

mysql_close()

Mit `mysql_close()` schließt man anhand der Verbindungskennung (Verbindungs- Kennung) eine Verbindung zu einer MySQL-Datenbank.

Syntax:
```
int mysql_close([int Verbindungs-Kennung])
```

Einfaches Beispiel:

```
<?     $db  = mysql_connect ("localhost","User","")
       or die ("Keine Verbindung möglich!");          //Verbindung herstellen
       mysql_select_db (Datenbank1);                  //Datenbank auswählen
       $sql = "SELECT * FROM tabellenname";           //SQL Code
       $res = mysql_query($sql);                      //Anfrage ausführen
       $anz = mysql_num_rows($res);                   //Ergebnisanzahl ermitteln
       echo $anz;                                     //Anzahl ausgeben
       echo "<br />";                                 //Leerzeile ausgeben
       while ($row = mysql_fetch_array($res)){        //Alle Ergebnisse abgefragt?
         echo $row['Spalte1];                         //1.Spalte ausgeben
         echo "<br />";                               //Leerzeile ausgeben
         echo $row['Spalte2];                         //2.Spalte ausgeben
         echo "<br />";                               //Leerzeile ausgeben
       }
       mysql_close($db);                              //Verbindung beenden
?>
```

6.7 MySQL

Auf den folgenden Seiten stehen grundlegende Informationen zu Datenbanksystemen bzw. zur Abfragesprache SQL.

6.7.1 Komponenten eines Datenbanksystems

Eine Datenbank (DB, engl. Data Base) ist eine systematische Sammlung von Daten.
Zur Nutzung und Verwaltung der in der DB gespeicherten Daten benötigt der Anwender ein Datenbank-Verwaltungssystem (DBMS, engl. Data Base Management System).
Die Kombination aus DB und DBMS ist das Datenbanksystem (DBS, engl.: Data Base System), das jedoch häufig fälschlicherweise als Datenbank bezeichnet wird.

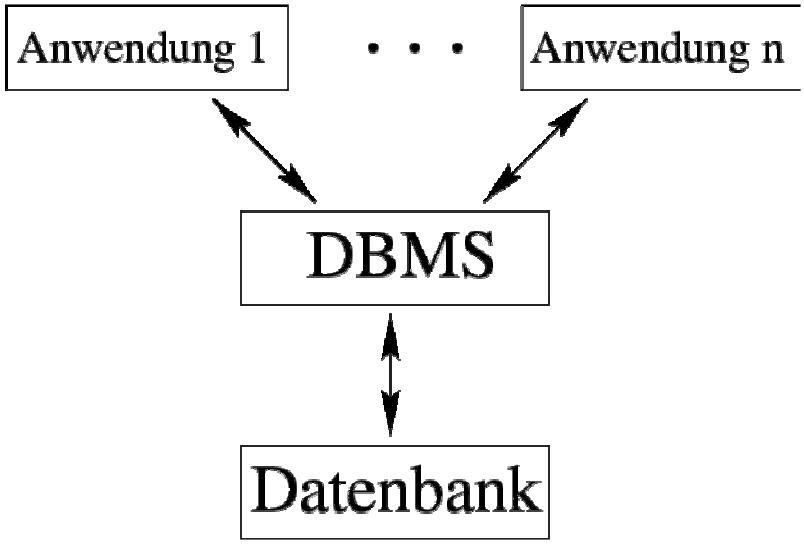


Abbildung 6.6 Struktur eines DBS

6.7.2 Ebenen eines Datenbanksystems

Ein Datenbanksystem besteht aus vier Ebenen:

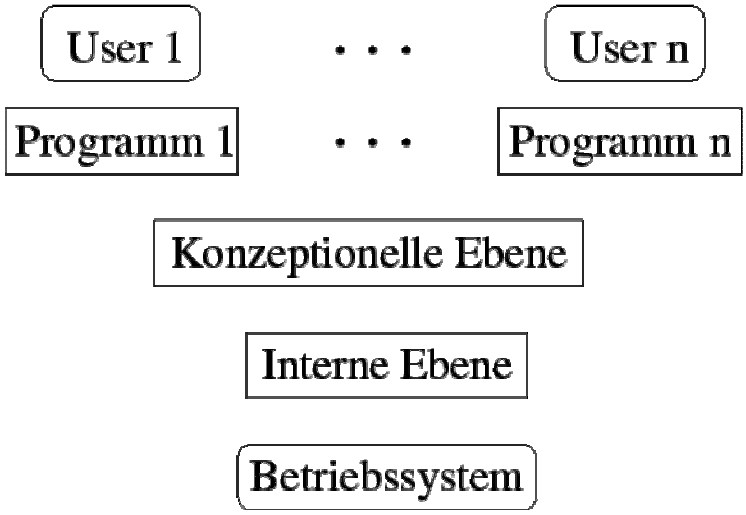


Abbildung 6.7 Ebenen eines DBS

1. **Betriebssystem**

   Dies ist die unterste Ebene, auf der jede Computeranwendung basiert.

2. **Interne Ebene**

   Auf der internen Ebene erfolgt die physische Speicherung der Daten. Die Speicherlogik, die dabei verwendet wird, hängt vom DBMS ab und kann dem Entwickler ziemlich egal sein, da er lediglich über die konzeptionelle Ebene auf die DB zugreift. Den Anwender braucht weder die interne noch die konzeptionelle Ebene zu kümmern, da er erst über die oberste, nämlich die externe Ebene, auf die DB zugreift.

3. **Konzeptionelle Ebene**

   Auf der dritten, der konzeptionellen Ebene, wird das Datenmodell beschrieben. Durch ein Datenmodell wird eine reale Anwendung umgelegt. Im Datenmodell sind die Strukturen der Daten und ihre Beziehung zueinander festgelegt. Nach der Art, wie die Beziehungen in dem Datenmodell geregelt werden, unterscheidet man zwischen hierarchischen, vernetzten und relationalen Datenmodellen. Wir verwenden das **relationale Datenmodell**, da es das modernste ist und sich durchgesetzt hat.

   Tabellenstruktur
   Beim relationalen Datenmodell werden die Daten in Tabellen angeordnet. Jede Tabelle hat einen eindeutigen Relationsnamen. Alle Zeilen der Tabelle (ohne die Spaltenüberschriftszeile) werden als Relation, jede einzelne Zeile davon als Tupel bzw. Datensatz, die Spaltenüberschriften als Attributnamen oder Attribute und alle Attributnamen zusammen werden als Relationsschema bezeichnet.

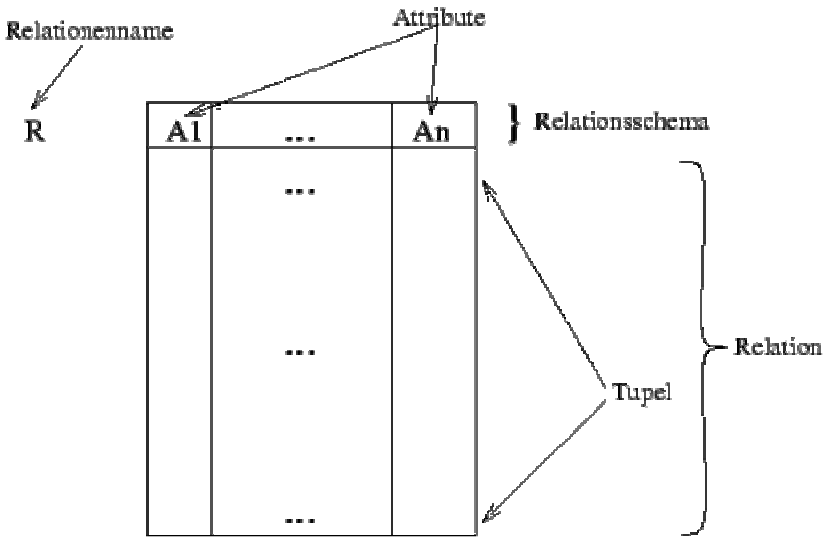


Abbildung 6.8 Tabellenstruktur

   Schlüssel
   Damit man jede Zeile gezielt ansprechen kann, wird ein Schlüsselattribut eingeführt. Der Schlüssel muss immer eindeutig sein und wird auch als Primärschlüssel bezeichnet. .
   Zum Einrichten der DB mit ihren Tabellen bedient man sich der Data Definition Language (DDL).

4. **Externe Ebene**

   Auf der obersten Ebene befindet sich der Anwender, der auf das DBS mit einer Daten-Abfragesprache (DQL, engl.: Data Query Language), einer Daten-Manipulationssprache (DML, engl.: Data Manipulation Language) oder einer eigenen Anwendung zugreift.

6.7.3 SQL

SQL steht für Structured Query Language, zu deutsch strukturierte Abfrage-Sprache.
Die SQL ist die eigentliche Schnittstelle zur Datenbank.
Ein DBS kann auf drei Arten gesteuert werden: Entweder dialogorientiert, im Batch-Betrieb oder durch andere Programme (PHP Skripte). Die Syntax bleibt dabei jedoch die selbe.
SQL lässt sich in drei Klassen einteilen:

1. Data Definition Language (DDL)
   - DDL dient z.B zum Anlegen, Löschen, Bearbeiten von Datenbanken und Tabellen
   - Befehle: CREATE, DROP, ALTER
2. Data Control Language (DCL)
   - DCL dient z.B. zur Rechtevergabe
   - Befehle: GRANT, REVOKE
3. Data Manipulation Language (DML)
   - DML dient z.B. zur Manipulation von Datensätzen (Tupel) oder der Abfrage von Daten
   - Befehle: SELECT, INSERT, DELETE, UPDATE

Syntaxbeschreibung

DDL, DCL:
Die Aufgaben der DDL bzw. DCL wurden bei unserem Projekt von dem Programm phpMyAdmin übernommen, dass unter Kapitel 8.1.6 näher beschrieben ist.

DML:

Tabelle füllen:

```
INSERT INTO tabelle [(spalte_1, spalte_2, … , spalte_n)]
       VALUES (wert_1, wert_2, … , wert_n);
```

Mit INSERT wird der Tabelle tabelle ein Datensatz d.h. eine Zeile hinzugefügt. Die Angabe der Spaltenbezeichnungen ist optional, und nur benötigt wenn nicht alle Spalten beschrieben werden sollen. Die einzutragenden Werte müssen in der entweder in der Reihenfolge der angegebenen Spalten oder wenn die Spaltenbezeichnungen nicht angegeben sind in der Reihenfolge der in der Tabelle definierten Attribute angeordnet werden.

Tabelle leeren:

```
DELETE FROM tabelle [WHERE where_definition];
```

DELETE löscht Zeilen aus tabelle, die mit der in where_definition angegebenen Bedingung übereinstimmen, und gibt die Anzahl der gelöschten Datensätze zurück.
Wird keine where_definition angegeben wird der Inhalt der gesamten Tabelle gelöscht.

Tabelle aktualisieren:

```
UPDATE tabelle
    SET spalten_name1=ausdruck1, [spalten_name2=ausdruck2, ...]
    [WHERE where_definition]
```

UPDATE aktualisiert Spalten in bestehenden Tabellenzeilen mit neuen Werten. Die SET-Klausel gibt an, welche Spalten geändert werden sollen und welche Werte ihnen zugewiesen werden. Die WHERE-Klausel legt - falls angegeben - fest, welche Zeilen aktualisiert werden sollen. Ansonsten werden alle Zeile aktualisiert.

Tabelle auslesen:

```
SELECT select_ausdruck,...
    [FROM tabellenreferenz
        [WHERE where_definition]
        [GROUP BY { spalten_name } [ASC|DESC],...]
        [HAVING where_definition]
        [ORDER BY { spalten_name } [ASC|DESC],...]
        [LIMIT zeilen]
```

SELECT wird benutzt, um ausgewählte Zeilen aus einer oder mehreren Tabellen abzurufen. select_ausdruck gibt die Spalten an, die Sie abrufen wollen.
GROUP BY dient zum Gruppieren von Elementen. Mit HAVING können gruppierte Elemente zusätzlich aussortiert werden.
Mit LIMIT kann die Auswahl auf eine angegebene Anzahl von Datensätzen reduziert werden.
Einem SELECT-Ausdruck kann mit AS ein Alias zugewiesen werden. Der Alias wird als Spaltenname verwendet und kann bei ORDER BY- oder HAVING-Klauseln benutzt werden.
Um in absteigender Reihenfolge zu sortieren, wird dem Namen der Spalte in der ORDER BY-Klausel das DESC-Schlüsselwort zugewiesen.
SELECT Abfragen können auch verschachtelt werden.

MySQL stellt arithmetische, sogenannte BUILD IN Funktionen und Mustersuch-Funktionen für SELECT Befehle zur Verfügung:

- **Arithmetische Funktionen**: +, - , / , *

- **BUILD IN Funktionen:**
  können nur als select_ausdruck verwendet werden.
  - `SUM(spaltenname)`
    Summiert ausgewählte Spaltenelemente
  - `MIN(spaltenname)`
    liefert den kleinsten Spaltenwert
  - `MAX(spaltenname)`
    liefert den größten Spaltenwert
  - `COUNT(spaltenname)`
    liefert die Anzahl der selektierten Spaltenelemente

  Beispiel: `SELECT MAX(spalte1) as maximalwert FROM tabelle;`

- **Mustersuch-Funktionen:**
  - **LIKE Operator** in der WHERE-Klausel
    Vergleicht ein Spaltenelement mit einer Zeichenkette. Das Zeichen % steht dabei für ein beliebige Anzahl beliebiger Zeichen.

    Beispiel: `SELECT * FROM tabelle1 WHERE spalte1 LIKE %A%`
    Diese Anfrage gibt alle Elemente der Tabelle tabelle1 aus in denen in spalte1 eine Zeichenfolge mit dem Buchstaben „A“ vorkommt.

  - **REGEXP Operator** in WHERE-Klausel
    MySQL erlaubt auch die Verwendung von regulären Ausdrücken. Der hierbei benötigte Operator heißt REGEXP.
    Jokerzeichen: ^ stimmt mit dem Anfang einer Zeichenkette überein
    $ stimmt mit dem Ende einer Zeichenkette überein
    . stimmt mit einem beliebigen Zeichen überein
    x* simmt mit jeder Folge von 0 od. mehr x Zeichen überein

    Beispiel: `SELECT * FROM tabelle1 WHERE spalte1 REGEXP ^A$;`
    Gibt alle Tupel der tabelle1 aus die in spalte1 die exakte Zeichenfolge A aufweisen.

    Beispiel: `SELECT * FROM tabelle1 WHERE spalte1 REGEXP .*;`
    Gibt alle Tupel der tabelle1 aus die in spalte1 eine beliebige Zeichenfolge aufweisen.

7 Transportschicht

In diesem Kapitel sollen die Zwecke, die Funktionsweise und die Bedienung der Transportschicht beschrieben werden.

7.1 Grundsystem

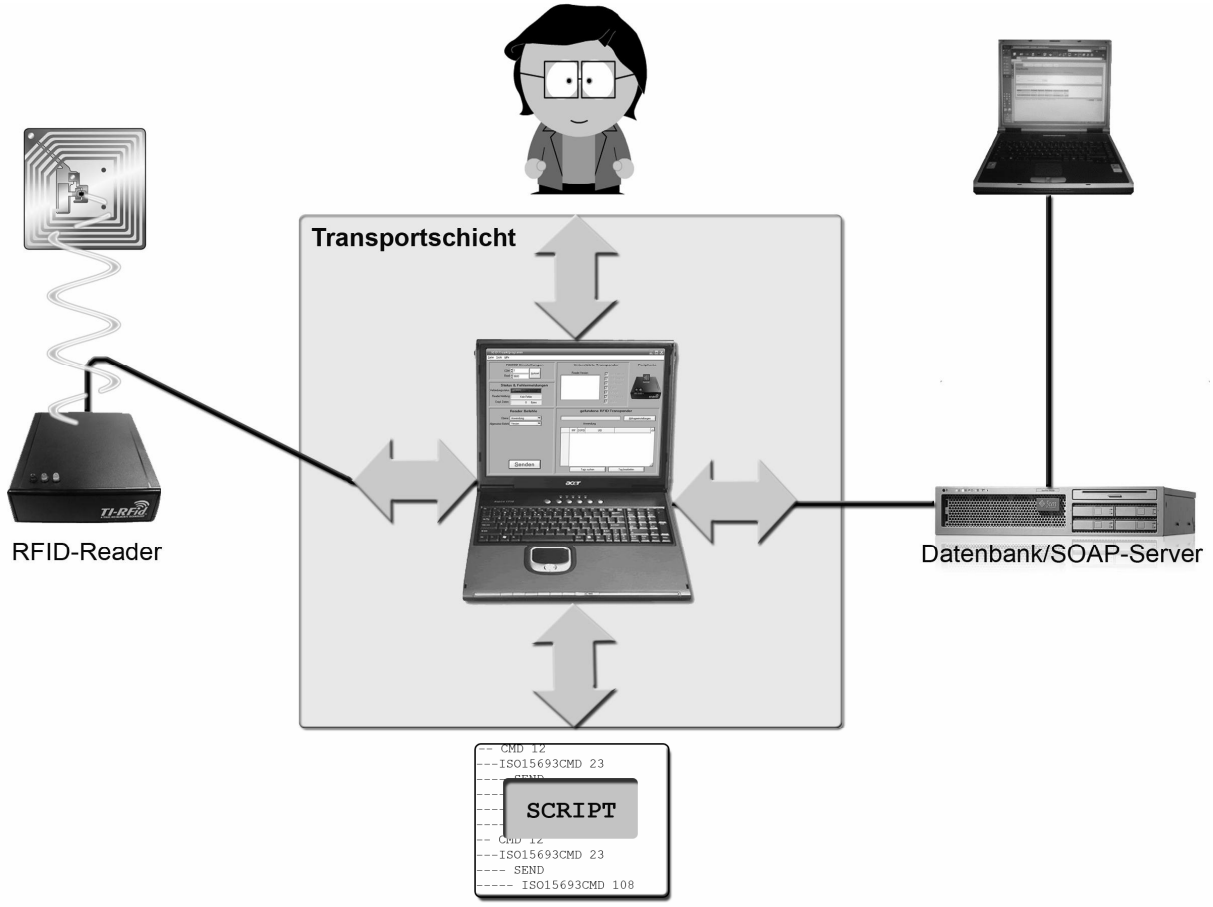


Abbildung 7.1 Transportschicht

Die Transportschicht dient zu folgenden Zwecken:

- Senden und Empfangen von RFID-Reader-Daten
- Senden und Empfangen von SOAP-Daten
- User Interface für Interaktion mit Personen
- Ausführen eines Skripts zur Automatisierung der Transportschicht

Um ein Programm mit diesen Fähigkeiten effektiv realisieren zu können, wählten wir LabWindows CVI 8.0 als Entwicklungsumgebung. Sämtliche Quellcodes der Transportschicht sind also ANSI C - Codes. Das Programm selbst wurde RFIDinvent getauft und kann mit dem Installer auf der Projekt-CD (\Transportschicht\Installer\RFID-invent 1_0 (Installer).msi) installiert werden.

Das CVI-Projekt befindet sich auf der Projekt-CD unter \Transportschicht\Entwicklung\Source.
Eine Demo-Version von CVI kann mit \Transportschicht\Entwicklung\CVI 8.0.zip installiert werden. Diese ist 30 Tage lauffähig, jedoch haben die damit kompilierten Programme ab dem 8. Tag nach der Installation nur eine Laufzeit von 10 Minuten.
Die Neuninstallation von CVI nach den 30 Testtagen gelang nur, wenn vor der Neuninstallation CVI 7 installiert wurde.

7.2 Senden und Empfangen von RFID-Daten

Das Ziel dieses Kapitels ist das Anwendungs-Layer-Protokoll und dessen Funktionen, welche vom Multi-Function Reader (MFR) unterstützt werden, zu erläutern.

Der Mikrocontroller des Readers kann zwar neu programmiert werden, das war jedoch nicht notwendig, da das TI-Programm bereits alle benötigten Funktionen beherrscht.
Das Anwendungs-Layer-Modul and verschiedene Bibliotheken bilden die Basisanwendung.
Der Anwendungs-Layer dient als Schnittstelle zu den Bibliotheken und bietet deren Funktionen an.
Der Reader unterstützt folgende Funktionen:
1) 14443 Layer 2,3,4 beide Typen A & B
2) 15693 Layer 2 and 3
3) Tag-it$_{TM}$
4) TI-RFID LF (DST, RO, RW)
5) Serieller E/A mit TTL-Pegel

7.2.1 Paketstruktur

Um nun den Reader ansteuern zu können, muss man wissen wie ein Paket zur Befehlsübertragung aufgebaut ist.

Der RFID-Reader ist als Slave-Gerät vorgesehen. D.h. es wird von einem anderen Gerät (z.B.: PC) über eine serielle Schnittstelle gesteuert. Um diese Beziehung zu unterstützen, verwendet der Reader eine standard Protokollstruktur.
Dieses Kapitel beschreibt, wie der Master (PC) auf die einprogrammierten Funktionen des Slaves (MFR) zugreifen kann indem Anfrage- und Antwort-Beispiele erläutert werden.

Jedes Paket hat die gleichen Overhead-Bytes um die Befehle übertragen zu können und um deren Gültigkeit und Vollständigkeit überprüfen zu können. Diese Regelung und die variable Länge des Befehlsstrings ermöglichen den Paketen, die variierenden Bedingungen der Befehle jeder Bibliothek fehlerfrei umsetzen zu können.

7.2.2 Kommunikationsstring

Das folgende Bild beschreibt den Aufbau eines Anfrage bzw. Antwort-Strings. Es wird also der nötige Overhead für die serielle Übertragung der Daten beschrieben:

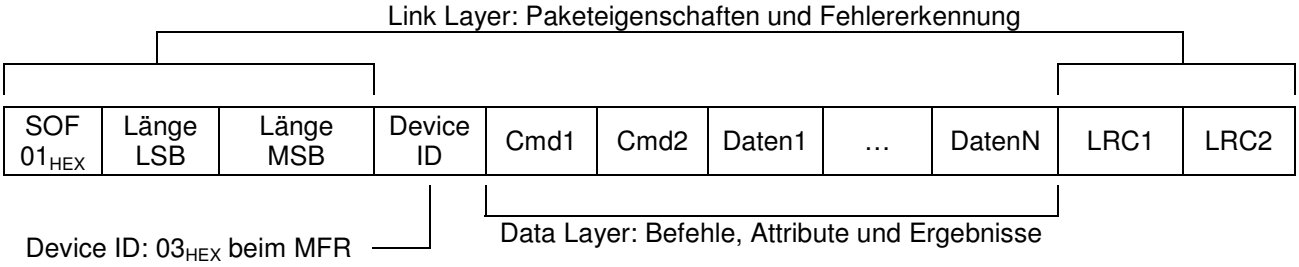


Abbildung 7.2 Aufbau des MFR Befehls- bzw. Antwortstrings

Device ID:

Die Device ID ist bei unserem Reader immer 03_{HEX}

Data Layer:

Der Data Layer besteht aus dem Befehl (Cmd1&2) und seinen Attributen (Daten1,2,…N):

Befehl: Der Befehl besteht aus zwei Bytes: Cmd1 und Cmd2.
Cmd1 beschreibt den Layer, also die Bibliothek der Funktion.

Bei unserem Reader gibt es folgende Layer ($Byte_{HEX}$):

01: Application (grundlegende Funktionen; z.B.: Baudrate, Transpondersuche,…)
02: ISO 14443-A (HF Transponder nach ISO14443A)
03: ISO 14443-B (HF Transponder nach ISO14443B)
04: ISO 15693 (HF Transponder nach ISO15693 – für die PC-Inventarisierung)
05: TagIt (HF Transponder von Texas Instruments)
06: LFDST (LF Data Storage Transponder)
07: ISO 14443-4 (HF Transponder nach ISO144434)
08: Apollo (HF Transponder)

Cmd2 beschreibt den Befehl des Layers. Der Befehl kann nur in diesem Layer vorhanden sein, aber auch in anderen Layer, wobei jeder Layer dann den Befehl anders interpretieren kann.

z.B.: gibt es den Befehl „Set Baudrate“ nur im Application Layer;
gibt es den Befehl „Find Token“ in allen Layern aber es kommen verschiedene Antworten (z.B.: bei Application: 1 UID irgendeines Typs, bei ISO15693: bis zu 16 ISO15693-UIDs)

Die Befehle können in den Texas Instruments Handbüchern nachgelesen werden (dem MFR-Kit beigelegte CD unter \Documentation\; jeder Layer hat ein eigenes Handbuch)

Attribute: Die Attribute sind je nach Befehl sehr verschieden.
Häufige Attribute sind z.B.: UniqueID# (Einzigartige ID des Labels, 8 Bytes)
Selected (Nur ausgewählte Transponder, 1 Byte)
Schreibmethode (polled oder burst mode, 1 Byte)

Bei einer Reader-Antwort entsprechen die Attribute den angefragten Daten. Das erste Datenbyte (Daten1) ist das Status Byte. Es gibt an ob der Vorgang erfolgreich abgeschlossen wurde (00_{HEX}) oder nicht ($>00_{HEX}$).

Für das Byte sind folgende Werte und Bedeutungen bekannt:

Tabelle 7.1 Fehlercodes

Kein Fehler	0	HF ASIC Not RBLOCK	45
Kein Transponder	1	14443B SDESELECT Error	46
Ungültiges RF Format	5	14443 Daten falsch	47
Ungültige Geräte ID	14	Many CID not support Transponders	48
Ungültige Aktion	16	Kollision BPSK und/oder CID	49
Falscher Download Zustand	17	Kollision BPSK Decode	50
Schreiben fehlgeschlagen	18	14443B abgebrochen	51
Ungültige Adresse	19	Transponderpuffer voll	52
Ungültige Baudrate	20	14443A Uplink Parity	53
Ungültige Prüfziffern	21	14443A ATS	54
Kein Timer verfügbar	22	14443A PPS	55
Ungültige Entity ID	23	14443A Cascade Level	56
Daten abgeschnitten	24	14443A SAK CRC	57
kein Datenlesen	25	BPSK Kein Fehler	64
Ungültiges Start Byte	26	BPSK bad frame wait	65
Ungültige CRC	27	BPSK bad value baud	66
Befehl ungleich Antwort	28	BPSK annulliert	67
14443A Daten falsch	32	BPSK Zwischenträger	68
14443B Daten falsch	33	BPSK TR0 Timeout	69
14443B kein Transponder	34	BPSK RCV Überlauf	70
HF ASIC Empfangs-Timeout	35	BPSK kein SOF	71
HF ASIC abgebrochen	36	BPSK kein EOF	72
HF ASIC ATQB Error	37	BPSK TR1 Timeout	73
HF ASIC Prot-Typ	38	BPSK CRC Fehler	74
HF ASIC ungültige CID	39	BPSK frame error	75
HF ASIC ungültige NAD	40	Modul abgebrochen	76
HF ASIC CID Low Power	41	Parameterfehler	77
14443B HLTB Fehler	42	Kollision erkannt	87
14443B ungültiger Block-Typ	43	Apollo Lebenskreis	96
HF ASIC Not IBLOCK	44	Apollo Daten falsch	97

Link Layer:

SOF (Start Of Frame) Der Befehls- bzw. Antwortstring/-frame beginnt immer mit 01_{HEX} als erstes Byte

Länge (LSB, MSB): Geben die Anzahl der zu übertragenden Bytes an = Framegröße
z.B.: *1xSOF + 2xLänge + 1xDeviceID + 2xCmd + 0xDaten + 2xLRC*
das Frame ist 8 Bytes groß (das ist die Mindesgröße eines Frames)
→ Länge: MSB=00_{HEX} LSB=08_{HEX}

LRC1 + LRC2: Longitudinal Redundancy Check (Längssummenprüfung)

Die Längssummenprüfung wird durch XOR-Verknüpfung der Bytes einer Bytefolge berechnet, wobei das Ergebnis der vorangegangenen Verknüpfung mit dem jeweils folgenden Byte verknüpft wird. Da die Methode nicht sehr fehlersicher aber sehr schnell ist, kommt sie normalerweise nur bei kleinen Datenblöcken (ca. 32 Byte) zum Einsatz. Viele häufige Fehlerformen werden nicht erkannt. Vertauschungen innerhalb eines Datenblockes können ebenso wenig bemerkt werden, wie sich gegenseitig aufhebende Mehrfachfehler.
Durchschnittlich ist eine Anfrage an den Reader ca. 16 Byte lang. Eine Antwort hat jedoch manchmal (eigentlich nur beim Auslesen des Speichers) die Länge von 255 Byte, sonst aber um die 12 Byte. Da bei einer so kurzen seriellen Kabelverbindung (50cm zw. Reader und PC) aber praktisch nie Bitfehler auftreten, ist die Längssummenprüfung mehr als ausreichend.

Vorgehensweise zur Längssummenprüfung:
1.) Byte 1 und Byte 2 werden über XOR miteinander verknüpft.
2.) Das Ergebnis der Verknüpfung wird mit dem jeweils folgenden Byte wiederum mittels XOR verknüpft.
3.) Der zweite Schritt wird solange wiederholt, bis alle Bytes (natürlich außer den LRC-Bytes selbst) verknüpft worden sind.

Beispiel:
Abfrage der Reader-Modul/Bibliotheks-Versionsnummern:
01_{HEX} + 08_{HEX} + 00_{HEX} +03_{HEX} + 01_{HEX} + 40_{HEX} + LRC1 + LRC2 (Invertierte LRC1)

Tabelle 7.2 Berechnung des LRC Bytes

Operation	$Byte_{HEX}$	$Byte_{BIN}$							
XOR	01	0	0	0	0	0	0	0	1
	08	0	0	0	0	1	0	0	0
XOR	= 09	0	0	0	0	1	0	0	1
	00	0	0	0	0	0	0	0	0
XOR	= 09	0	0	0	0	1	0	0	1
	03	0	0	0	0	0	0	1	1
XOR	= 0A	0	0	0	0	1	0	1	0
	01	0	0	0	0	0	0	0	1
XOR	= 0B	0	0	0	0	1	0	1	1
	40	0	1	0	0	0	0	0	0
LRC1 =	**4B**	0	1	0	0	1	0	1	1

LRC1 = $4B_{HEX}$
LRC2 = ⌐LRC1 = ⌐01001011 = 10110100 = $B4_{HEX}$ = LRC2

Die gültige Anfrage an den Reader würde also lauten:
01_{HEX} + 08_{HEX} + 00_{HEX} +03_{HEX} + 01_{HEX} + 40_{HEX} + $4B_{HEX}$ + $B4_{HEX}$

Wird zur Kontrolle die LRC jetzt über alle Datenbytes errechnet ist das Ergebnis immer 00. Somit ist die Kenntnis der ursprünglichen LRC für die Kontrolle nicht nötig, da ein von 0 abweichendes Ergebnis immer auf einen Fehler hinweist.

7.2.3 CVI-Code für die Verarbeitung des Kommunikationsstring

Nun, da wir wissen, wie ein gültiges Paket aussehen muss, können wir uns mit der Realisierung der Reader-Verbindung beschäftigen:

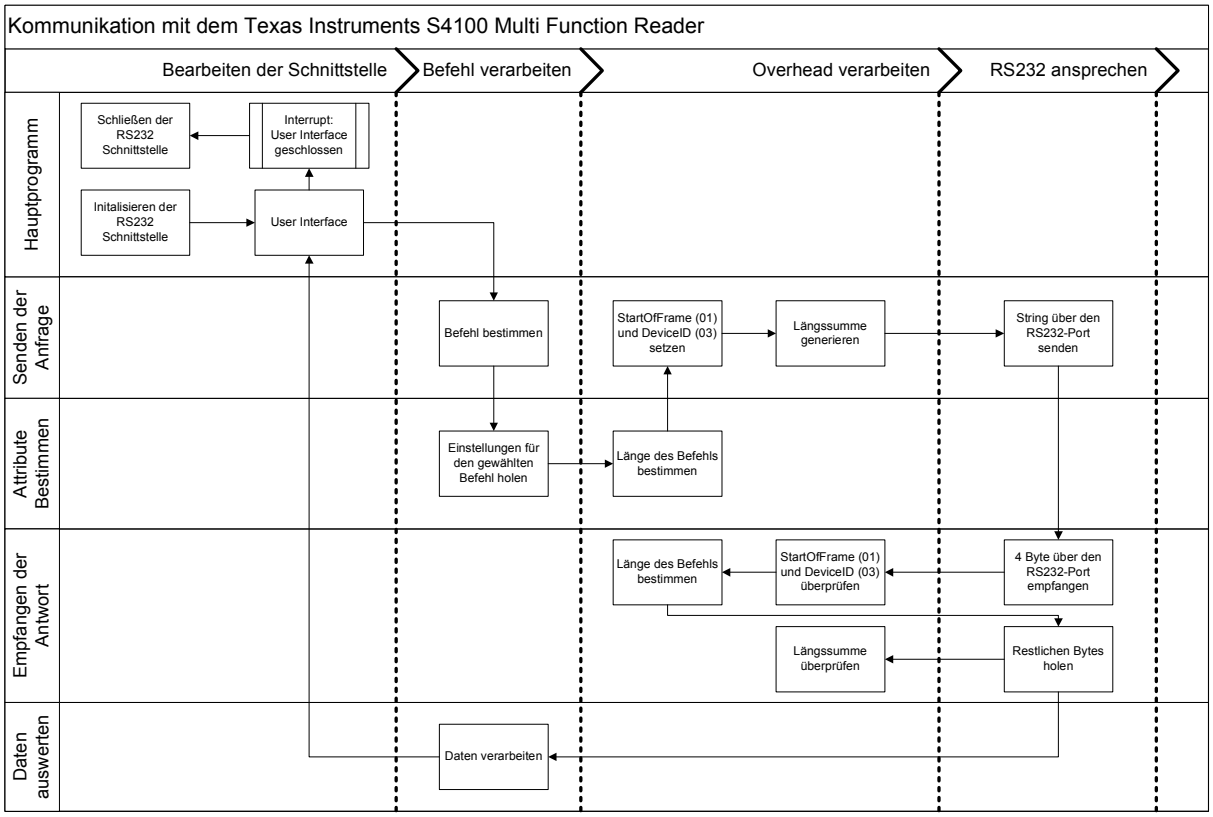


Abbildung 7.3 Ablaufdiagramm: Kommunikation mit dem Reader

RS232-Verbindung

Um den RFID-Reader ansprechen zu können, müssen wir eine Datenverbindung aufbauen. Der RFID-Reader kann über die RS232 und die RS485 angesprochen werden. Wir nutzen die, bei PCs verbreitete, RS232 (COM) Schnittstelle.

Initialisierung der COM-Schnittstelle (im Hauptprogramm):

```
OpenComConfig (comport, "", baudrate, 0, 8, 1, 512, 512);
// Com Schnittstelle öffnen und konfigurieren
// (Com-Port + Baudrate einstellbar, keine Parity, 8 Datenbits, 1 Stopbit)
SetComTime (comport, 0.5);
// Com-Timeout (Reader braucht Zeit zum auswerten)
```

Schließen der COM-Schnittstelle (im Hauptprogramm, wenn das UserInterface geschlossen wird):

```
CloseCom (comport);
```

Generieren bzw. Auswerten der Kommunikationsstrings

Die folgenden Programme dienen zur Generation des gültigen Sendestrings (sendRFID()) und zur Überprüfung des darauf folgenden Empfangsstrings (readRFID()). Diese Programme werden bei jeder Kommunikation mit dem Reader aufgerufen und bilden somit den Kern der MFR-Kommunikation.

Befehl an Reader Senden:

```
unsigned char *req;  // Anfrage an den Reader

int sendRFID(void)
{     int error=0;
       LRC=0;       //Longitudinal Redundancy Check
       laenge=8;    //Länge der Anfrage ist die Mindestanzahl der Sendebytes
       GetCtrlVal (panelHandle, TOP_LAYER, &req[4]); //Layer bestimmen
       switch(req[4])
       {case 1: GetCtrlVal (panelHandle, TOP_CMD,          &req[5]); break;
          //Allgemeiner Befehl
        case  2:  GetCtrlVal  (panelHandle,  TOP_ISO14443ACMD,  &req[5]);  break;
         //ISO 14443 A Befehl
        case  3:  GetCtrlVal  (panelHandle,  TOP_ISO14443BCMD,  &req[5]);  break;
         //ISO 14443 B Befehl
        case  4:  GetCtrlVal  (panelHandle,  TOP_ISO15693CMD,    &req[5]);  break;
         //ISO 15693   Befehl
        case 5: GetCtrlVal (panelHandle, TOP_TAGITCMD,     &req[5]); break;
         //ISO Tag-it  Befehl
        case 6: GetCtrlVal (panelHandle, TOP_LFDSTCMD,     &req[5]); break;
         //ISO LF DST  Befehl
        case  7:  GetCtrlVal  (panelHandle,  TOP_ISO144434CMD,  &req[5]);  break;
         //ISO 14443-4 Befehl
        case 8: GetCtrlVal (panelHandle, TOP_APOLLOCMD,    &req[5]); break;
         //Apollo Befehl
       }
       putvalues();        //Zusatzattribute bestimmen (später erklärt)
       req[0]=1;           //StartOfFrame
       req[1]=laenge%256;  //LSB der Anzahl der Sendebytes
       req[2]=laenge/256;  //MSB der Anzahl der Sendebytes
       req[3]=3;           //DeviceID (immer 3)
       req[laenge-2]=0;    //LRC zurücksetzen
       for(i=0;i<laenge-2;i++)              //LRC Bytes berechnen
        req[laenge-2]=req[laenge-2]^req[i]; //~LRC Byte
       req[laenge-1]=255-req[laenge-2];     //LRC+~LRC muss FFhex sein
       error=ComWrt (comport, req, laenge); //Auf COM schreiben
       readRFID();                          //Readerantwort holen
       return error;
}
```

Antwort vom Reader empfangen:

```
unsigned char *ans;  // Antwort vom Reader

int readRFID(void)
{ int error=0;
       laenge=0; //Länge der Antwort zurücksetzen
       LRC=0;    //LRC zurücksetzen
       error=ComRd (comport, ans, 4);          //Headerbytes lesen
       if ((ans[0]==1)&&(ans[3]==3))           //Startbyte SOF + Device ID
        { LRC=((ans[0]^ans[1])^ans[2])^ans[3]; //LRC der 1. 4 Bytes (^=XOR)
          laenge=ans[1]+265*ans[2];             //Länge der Antwort einlesen
          ComRd (comport, ans, laenge-4);       //weitere Daten holen
          for(i=0;i<laenge-4;i++)               //LRC der restl. Bytes berechnen
            LRC=LRC^ans[i];                     //LRC Byte
          if((LRC==ans[laenge-5])&&(ans[laenge-5]+ans[laenge-6]==255))
                                                //richtige LRC+~LRC muss FFhex sein
            SetCtrlVal (panelHandle, TOP_FEEDBACK, "Checksum OK");
                                               //positiven Checksumstatus ausgeben
          else SetCtrlVal (panelHandle, TOP_FEEDBACK, "Wrong Checksum");
                                                //negativen Checksumstatus ausgeben
          erroraction(); //was tun wenn ein Fehler zurückgegeben wird
          getvalues();   //Auswerten des Datenstrings (später erklärt)
        }
       return error;
}
```

7.2.4 Verarbeiten der Readerdaten

In `sendRFID()` wird das Unterprogramm für das Holen der Attribute aufgerufen (`putvalues()`) und in `readRFID()` wird das Unterprogramm für das Auswerten der empfangenen Daten aufgerufen (`getvalues()`).

Die Unterprogramme sind relativ umfangreich und werden daher nicht zur Gänze erläutert. Da sie aber sehr häufig genutzt werden und die meisten RFID-Funktionen dort verarbeitet werden, wird die Funktionsweise erläutert.
Somit soll die Erweiterung bzw. Änderung der Funktionsverarbeitung durch einen Dritten ermöglicht werden. In Form einer Anleitung wird erläutert wie man eine Funktion hinzufügen kann.

Man nehme an, die Funktion „Write AFI“ aus der ISO 15693 Bibliothek soll realisiert werden:

1.) Auf der Projekt-CD unter TI\Documentation den „ISO 15693 Library Reference Guide“ suchen

2.) Im Dokument das Kapitel „Write AFI“ suchen → „Write AFI Request ($6C_{16}$)“ auf Seite 38

3.) Hier findet man nun eine kurze Beschreibung sowie die Request- und Response Packets

4.) Im User Interface (RFID.uir) wird nun der neue Befehl eingetragen:
Dazu wird der ISO15693CMD Ring erweitert → $6C_{HEX} = 108_{DEZ}$

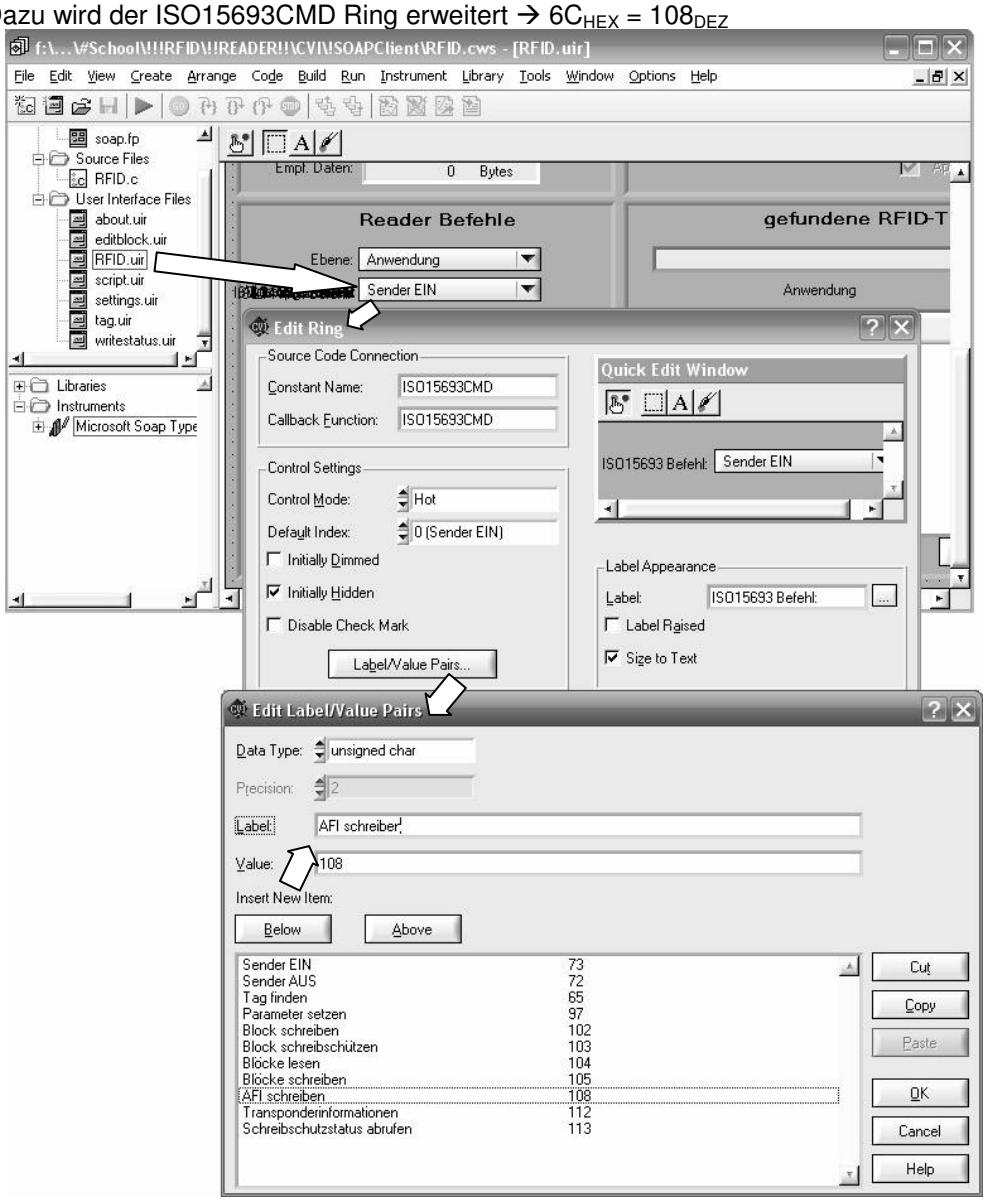


Abbildung 7.4 neuen Befehl einprogrammieren

Würde der Befehl keine Attribute haben und nichts (außer dem Status-Byte) zurückliefern. So wären wir nun fertig. Der Befehl könnte ausgewählt und vom Reader verarbeitet werden.

5.) Laut Reference Guide hat der Befehl noch 3 (+ 1 optionales) Attribut. Diese müssen im C-Code (`putvalues()`) für diesen Befehl definiert werden:

```
int putvalues(void)
{ ...
  ...
  switch (req[5])   //Befehlsbyte auswerten
  {case 65:         //-- RFID TAG REQUEST
          {...
           laenge++;//Zusatzbyte
           break;
          }
   case 105:        //-- WRITE MULTIPLE BLOCKS
          {...
           laenge+=...;//Zusatzbytes
           break;
          }
   case 108:       //-- WRITE AFI
        {GetCtrlVal (settingsHandle, SETTINGS_SELECT, &h); //selected Häkchen
         req[6]=h;                                         //Selected Befehl?
         checkburst(); //schaut in Settings, ob Tag im burst beschrieben wird
         GetCtrlVal (panelHandle, TOP_POLLED, &h);//holt Burst-Status
         req[7]=h;     //setzt burst im Request
         GetCtrlVal(tagHandle, TAG_AFI, &req[8]); //AFI angeben (hier aus Feld
         laenge+=3;    //Die Mindestlänge 8 (aus sendRFID() wird um 3 erhöht)
         //Optional könnte man hier noch die 8 Bytes für die UID angeben
         //und die Länge dementsprechend um 8 erhöhen
         //z.B.: strcpy(&req[9],"\x07\xe4\x00\x55\x03\x15\x07\xe0"); laenge+=8;
         //Achtung: Die UID wird von hinten nach vorne angegeben
         break; //nicht vergessen, sonst gibt es unerwünschte Effekte
        }
   case 112:        //-- GET SYSTEM INFORMATION
       {...
        break;
       }
      }
      ...
      ...
}
```

Nun kann der Befehl vom Reader verstanden werden. Das Programm gibt die Daten des Statusfeld automatisch aus (siehe Kapitel: User Interface). Die Auswertung des ISO15693 Response Flags müsste im Unterprogramm `putvalues()` ähnlich `getvalues()` eingetragen werden. Ist dort kein entsprechender Eintrag vorhanden, so gibt das Programm die zusätzlichen Bytes über die Standardschnittstelle (meistens die Konsole) aus.

7.3 User Interface

In diesem Kapitel soll der Umgang mit dem User Interface, also der Benutzeroberfläche der Transportschicht, erläutert werden. Es soll also den Nutzern des Programms dabei behilflich sein, die gewünschten Ergebnisse zu bekommen. Das Kapitel und dessen Abschnitte werden deshalb stark einem Benutzerhandbuch ähneln.

7.3.1 Hauptoberfläche

Die Hauptoberfläche wird sofort nach dem Start des Programms geladen und angezeigt. Sie beinhaltet die meisten Funktionen sieht somit auf den ersten Blick recht „überladen“ aus. Durch die Einteilung in Funktionsblöcke, kann man schnell den Überblick gewinnen.

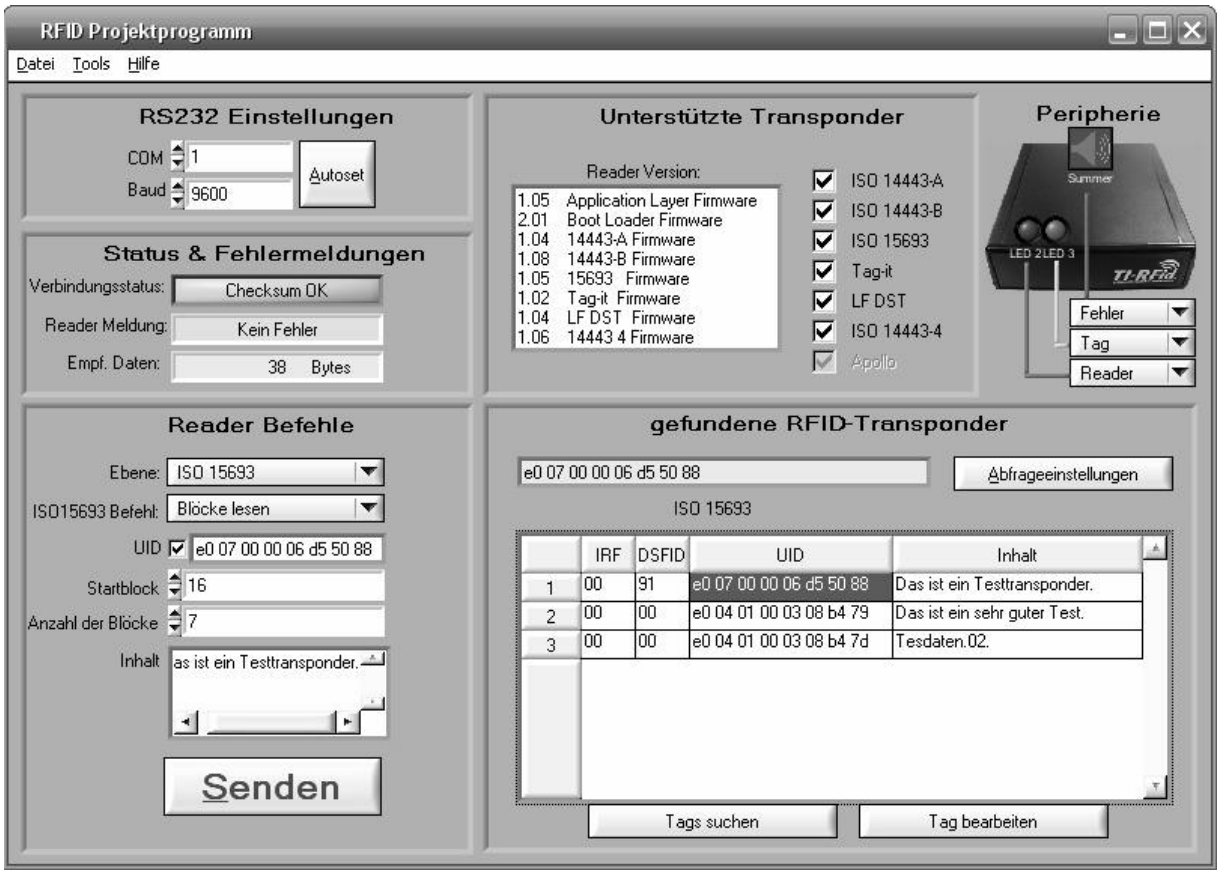


Abbildung 7.5 Hauptoberfläche

In den folgenden Abschnitten werden die Funktionsblöcke erklärt.

7.3.2 Menüleiste

Die Menüleiste hat drei Teilbereiche:

- Datei
- Tools
- Hilfe

Datei

Hier werden alle Operationen, die etwas mit Dateien zu tun haben, aufgerufen. Über das Menü kann z.B.: ein Script gestartet werden (→Datei→Script→starten) oder die Aufzeichnung einer Logdatei gestartet werden (→Datei→Log→aufzeichnen). In beiden Fällen wird sich ein Fenster zur Auswahl des Files öffnen.
Nach der Auswahl läuft das Skript oder die Aufzeichnung und in der Menüleiste können diese auch wieder beendet werden.

Tools:
Hier werden Aufrufe für kommende kleine Tools platziert z.B.: Log→Script-Umwandler,

Hilfe:
Hier kann der Aufruf einer Hilfedatei platziert werden

7.3.3 RS232 Einstellungen

Das Programm kommuniziert über einen COM-Port des PCs mit dem Reader. Die RS232 des Readers hat die folgenden Grundeinstellungen:
8 Datenbits, ein Stoppbit, keine Parität, 9600Baud, keine Flussteuerung

Die Baudrate des Readers kann aber umgestellt werden (siehe Kapitel: Reader Befehle). Deshalb muss das Programm mehrere Baudraten unterstützen. Die COM-Nummer der RS232 Schnittstelle des PCs natürlich auch veränderbar sein, denn die gewünschte RS232-Schnittstelle wird nicht immer der COM1 sein.

Für den Fall, dass man nicht weiß an welcher Schnittstelle der Reader angeschlossen ist und auf welche Baudrate dieser eingestellt ist, wurde eine Autoset-Funktion integriert, die alle noch nicht geöffneten COM-Ports mit den möglichen Baudraten durchprobiert bis eine gültige Antwort vom Reader kommt. COM-Ports, die nicht geöffnet werden können, werden übersprungen. Die Suche wird nach dem tausendsten COM-Port abgebrochen.
Auf manchen PCs konnten wir feststellen, dass manche Modems auf den Reader-Befehl reagieren und eine Antwort senden, sodass die Suche abgebrochen wird. Dass es sich dabei nicht um den Reader handelt, erkennt man an unsinnigen Fehlermeldungen, die angezeigt werden. Um dieses Problem zu umgehen, sollte die Readersuche mit einem höheren COM-Port begonnen werden.

7.3.4 Status und Fehlermeldungen

Dieser Bereich gibt Auskunft über die Ergebnisse der Readerverbindung.

Im Verbindungsstatus wird angezeigt ob eine die Verbindung hergestellt werden konnte. Falls nicht, kommt die Fehlermeldung, dass das der Reader keine Antwort gab oder dass die Checksumme der Daten nicht stimmt.

Die Reader Meldung gibt das bereits erwähnte Statusbyte bereits mit Übersetzung (siehe Tabelle: Fehlercodes) aus.

In beiden Feldern, werden positive Ergebnisse durch grünen und Fehler durch roten Hintergrund angezeigt.

Weiters wird noch angezeigt, wieviele Bytes zuletzt vom Reader empfangen wurden.

7.3.5 Unterstützte Transponder

Entsprechend der bereits erwähnten Bibliotheken, die der Reader nutzt, können mehrere Transpondertypen angesprochen werden. Im „Reader-Version“-Feld, kann man die Versionen der Bibliotheken sehen und mit den Häkchen rechts davon, kann man Funktionen der Bibliotheken deaktivieren.

7.3.6 Peripherie

Das Reader-Board besitzt drei Ausgänge, wobei zwei davon an die LEDs (gelb & grün) angeschlossen sind und der dritte einen Summer ansteuert.

Durch die Auswahllisten kann man angeben, wann die Ausgänge gesetzt werden sollen.
z.B.: SOAP - bei jeder Kommunikation mit dem Server
Tag - wenn ein Transponder gefunden wurde
Fehler - wenn ein Fehler auftritt

Die Peripheriezustände können auch durch einfaches Klicken auf die Symbole (die LED- und Lautsprecher-Icons) verändert werden.

7.3.7 Reader Befehle

Dieser Bereich dient zur direkten Ansteuerung des Readers. In diesem Bereich können die Befehle der Bibliotheken angegeben und an den Reader gesandt werden (siehe Kapitel: Verarbeiten der Readerdaten). Im Normalfall wird dieser Bereich nicht vom Nutzer benutzt, denn die meisten benötigten Funktionen sind in Form von eigenen Buttons an passenden Stellen der Benutzeroberfläche realisiert. Dennoch bildet dieser Bereich den bereits erwähnten Kern der MFR-Kommunikation.
Würde man also einen Button betätigen der einen Befehl an den Reader schickt (z.B.: „Tags suchen“) so wird im „Reader Befehle“-Feld die notwendige Bibliothek (z.B.: Anwendung) und der entsprechende Befehl (Tag finden) eingestellt und anschließend das Unterprogramm (`sendRFID()`) aufgerufen. Man könnte den Befehl also auch händisch einstellen und anschließend auf den „Senden“-Button, welcher einfach nur `sendRFID()` aufruft, drücken.

Je nach eingestelltem Befehl, werden (sofern entsprechend realisiert) die zugehörigen Attribute eingeblendet.

Im Kapitel: „Verarbeiten der Readerdaten“ wurde bereits erklärt, wie man eine noch nicht realisierte Reader-Funktion im Programm integrieren kann, nun soll gezeigt werden, wie man die entsprechenden Attribute für die Funktion anzeigen lassen kann.

1.) wenn notwendig (also nicht bereits vorhanden), Attributsfeld erzeugen.
Die Felder für die Eingabe der Attribute werden sollten als ‚hidden’ initialisiert werden.
2.) Das Einblenden der Attribute wird vom Unterprogramm `cmdrefresh()` gesteuert. Man muss also einen Eintrag für das Einblenden der gewünschten Attribute für den Befehl machen.
z.B.: Einblenden der UID, des Polled(Burst)-Häkchens und des (neu erzeugten) AFI-Felds für den Befehl „Write AFI Request“ (108_{DEZ}; $6C_{HEX}$)

```
int cmdrefresh (void)
{ ...
 if((req[5]>=102&&req[5]<=105)||req[5]=108||req[5]==112||req[5]==113)
 //write(multiple)blocks & get(multiple)Block(s) & system info & security
 {SetCtrlAttribute (panelHandle, TOP_enterUID,   ATTR_VISIBLE, 1);
  SetCtrlAttribute (panelHandle, TOP_enteredUID, ATTR_VISIBLE, 1);
 }
 else
 {SetCtrlAttribute (panelHandle, TOP_enterUID,   ATTR_VISIBLE, 0);
  SetCtrlAttribute (panelHandle, TOP_enteredUID, ATTR_VISIBLE, 0);
 }
 ...
 if((req[5]>=102&&req[5]<=103)||req[5]==105||req[5]=108)//Schreibvorgänge
      SetCtrlAttribute (panelHandle, TOP_POLLED,   ATTR_VISIBLE, 1);
 else SetCtrlAttribute (panelHandle, TOP_POLLED,   ATTR_VISIBLE, 0);
 ...
 if(req[5]=108) SetCtrlAttribute (panelHandle, TOP_AFI, ATTR_VISIBLE, 1);
 else           SetCtrlAttribute (panelHandle, TOP_AFI, ATTR_VISIBLE, 0);
 ...
 return 1;
}
```

Nun werden die Attribute bei Auswahl des Befehls im User Interface angezeigt.

Bereits realisierte Befehle

Nun seien noch einige bereits realisierte Befehle beschrieben:

Anwendungsebene [01_{HEX}]

Version:
Version Request (40_{16})

Die Funktion gibt alle Bibliotheks-/Modul-Versionen zurück.

Die Transportschicht schreibt die Readerversionen in das „Reader-Version“-Textfeld.

Tag finden:
Find Token Request (41_{16})

Die Anzahl der Schleifen, die der Reader nach Transpondern suchen soll, wird übergeben.
Je mehr Schleifen angegeben sind, umso länger dauert der Lesevorgang. Das bedeutet, dass der RS232-Timeout gegebenenfalls erhöht werden muss (pro Schleife ~0.5 Sekunden).
Wird in einer Schleife ein Transponder gefunden, so werden alle übrigen Schleifen nicht mehr ausgeführt, sondern sofort der gefundene Transponder zurückgeschickt.

Die Funktion selbst sucht nach ISO14443A, ISO14443B, ISO15693, Tag-It und DST-LF Transpondern.
Je nach Prioritätseinstellung wird aber nur einer der Transpondertypen (wenn vorhanden) angezeigt.

- Wird kein Transponder gefunden, so wird in der Readerantwort das Statusbyte 01_{HEX} (kein Transponder) zurückgegeben.
- Wird ein Transponder gefunden, so werden dessen UID und der Transpondertyp zurückgesandt.
- Findet der Reader mehr als einen Transponder eines Transpondertyps (z.B.: 2 x ISO15693) so wird ebenfalls nur eine UID angezeigt, das Statusbyte in der Readerantwort hat aber nicht mehr den Wert 00_{HEX} (kein Fehler) sondern 57_{HEX} (Kollision entdeckt).

Tag Priorität
Set Token Priority (42_{16})

Wie bereits erwähnt, unterstützt der MFR mehrere verschiedene Transpondertypen. Es gibt Funktionen, für die es wichtig ist, zu wissen, welcher Transponder „Vorrang“ hat. Der oben genannte Befehl „Tag finden“ greift auf diese Prioritätsliste zu.

Die Einträge der Liste können durch einfaches Drag and Drop erweitert oder verringert werden. Die Prioritätsliste ist auf maximal fünf Einträge begrenzt. Befindet sich eine Bibliothek nicht innerhalb der der Prioritätsliste, so werden die entsprechenden Transponder von den Funktionen ignoriert.

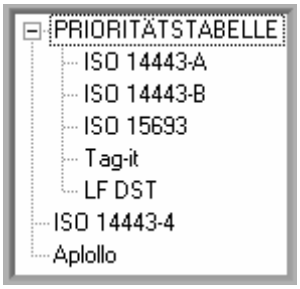


Abbildung 7.6 Prioritätstabelle

Ausgang setzen
Set Driver Request (43_{16})

Das Reader-Modul besitzt drei Ausgänge. Beim verbauten MFR ist das Modul an zwei Leuchtdioden (1xgrün,1xgelb) und einem Summer angeschlossen.
Deshalb können die Ausgänge über die entsprechenden Icons im bereits beschriebenen Peripheriebereich ein- bzw. ausgeschalten werden.

Der Befehls selbst sendet einen Befehl (ON, OFF oder Toggle) an die angegebenen Treiber. Will man also z.B.: den Befehl „Ausgang1 und Ausgang3 einschalten, Ausgang2 ausschalten“ realisieren, so muss man zwei Befehle (einen zum Einschalten und einen zum Ausschalten) mit der entsprechenden Angabe der Ausgänge senden.

Baudrate setzen
Set USART 0 Baud Rate (46_{16})

Der MFR kann die Baudrate der seriellen Verbindung manipulieren.
Der Reader unterstützt:
9600 baud (voreingestellt)
19200 baud
38400 baud
57600 baud
115200 baud

Wird eine andere Baudrate eingestellt, so gibt der Reader die Fehlermeldung „Ungültige Baudrate" zurück.

ISO15693-Ebene [04_{HEX}]

Sender ein / aus
Transmitter ON Request (48_{16})
Transmitter OFF Request (49_{16})

Mit diesem Befehl kann man Funktionalitäten der Module/Bibliotheken aktivieren bzw. deaktivieren. Beim ISO15693-Modul ist z.B.: das Antikollisionsverfahren deaktiviert. Es können also nicht mehrere Transponder eingelesen werden.
Diese Funktion ist auch im Bereich „Unterstützte Transponder" zugänglich.

Tag finden
Find Token Request (41_{16})

Die Funktion „Tag Finden" ist bereits aus der Anwendungsebene bekannt (gleicher Name, gleicher Befehl nur auf einem anderen Layer). Der Befehl wird mit dem gleichen Attribut (Anzahl der Schleifen) aufgerufen. Es gibt jedoch folgende Unterschiede zwischen dem Befehl, ausgeführt in der Anwendungsebene und diesem Befehl, ausgeführt in der ISO15693-Ebene:

Find Token in der Anwendungsebene	Find Token in der ISO15693-Ebene
mehrere Transpondertypen	nur ISO15693-Transponder
maximal ein Transponder anzeigbar	bis zu 16 Transponder gleichzeitig

Block schreiben
Write Block Request (66_{16})

Dieser Befehl schreibt einen Block (meistens 4 Byte) auf den beschreibbaren Speicher des Transponders (natürlich nur, wenn dieser Block nicht schreibgeschützt ist). Es müssen die Blocknummer, also welcher Block beschrieben werden soll, sowie die Anzahl der Bytes, die ein Block hat (kann mit „Transponderinformationen" ausgelesen werden) und natürlich die zu schreibenden Daten angegeben werden.

Das Transportschicht-Programm verwendet diesen Befehl bei jedem Schreibvorgang auf den erweiterten Speicher.

Block schreibschützen
Lock Block Request (67_{16})

Dieser Befehl sperrt den gewählten Block des Transponderspeichers. Das bedeutet, dass der Speicher danach nicht mehr beschreibbar ist.
Achtung! Dieser Befehl kann nicht rückgängig gemacht werden.
Auch hier muss natürlich die Blocknummer und die Anzahl der Bytes pro Block angegeben werden.

Blöcke lesen
Read Multiple Blocks Request (68_{16})

Mit diesem Befehl kann man mehrere Blöcke aus dem erweiterten Speicher lesen. Man gibt dazu den Startblock und die Anzahl der zu lesenden Blöcke an.
Achtung! Wenn die Readerantwort 255 Bytes übersteigen würde, bekommt man nur die Fehlermeldung „Daten abgeschnitten“ zurück. Die Transportschicht teilt den Lesevorgang des ganzen Speichers deshalb in mehrere kleinere Lesevorgänge auf.

Blöcke schreiben
Write Multiple Blocks Request (69_{16})

Mit diesem Befehl kann man mehrere Blöcke in den Speicher des Transponders schreiben. Leider hat dieser Befehl bei keinem unserer Transponder funktioniert, weshalb wir die Blöcke zum Schreiben immer einzeln übertragen müssen. Wir stellten fest, dass andere Programme deshalb dieselbe Vorgehensweise nutzten. Dies konnten wir anhand von RS323-Mitschnittprogrammen (z.B.: „Serial Monitor) feststellen.

AFI / DSFID schreiben
Write AFI Request ($6C_{16}$)
Write DSFID Request ($6E_{16}$)

Mit diesen Befehlen kann man die AFI (Application Family ID) oder die DSFID (Data Storage Format ID) auf ein Tag schreiben (Bleistift-Symbol im Tagdetail-Panel).

AFI / DSFID schreiben
Lock AFI Request ($6D_{16}$)
Lock DSFID Request ($6F_{16}$)

Mit diesen Befehlen kann man die AFI (Application Family ID) oder die DSFID (Data Storage Format ID) auf ein Tag schreibenschützen (Schloß-Symbol im Tagdetail-Panel).

Transponderinformationen
Get System Information (70_{16})

Mit diesen Befehlen kann man diverse Informationen über einen Transponder abfragen. Folgende Informationen kann man mit diesem Befehl erhalten:

Info Flags: (1Byte) Gibt an, welche Informationen der Transponder geben kann (DSFID, AFI, Speichergröße, Herstellerinfo)

UID: (8Bytes) Die einigartige Seriennummer des Transponders

DSFID: (1 Byte) Data Storage Format ID: Gibt an, in welchem Format, die Daten gespeichert wurden

AFI: (1 Byte) Application Family ID: Gibt an, für welchen Anwendungszweck der Transponder verwendet wird.

Speichergröße: (2 Bytes) Gibt die Größe des erweiterten Speicherbereichs an. Das erste Byte gibt die Anzahl der Blöcke und das zweite Byte die Anzahl der Bytes pro Block an.

Herstellerinfo: (1 Byte) wird vom Hersteller angegeben und kann nicht verändert werden

Schreibschutzstatus aufrufen
Get Multiple Block Security Status (71_{16})

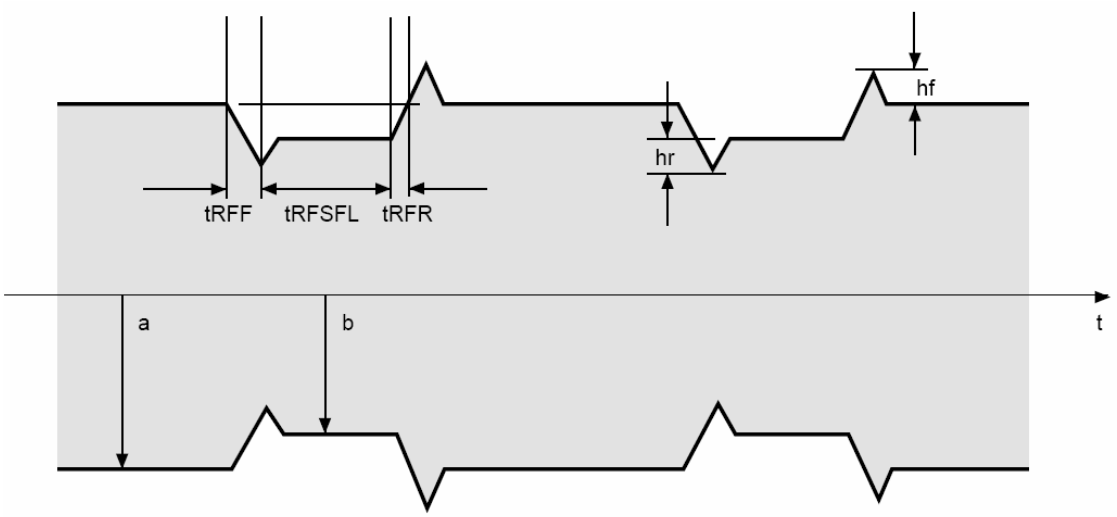
Mit diesem Befehl kann man den Schreibschutzstatus (<locked> oder <not locked>) mehrerer Blöcke auslesen. Pro angefragten Block (Startblock + Anzahl der Blöcke) wird bei der Antwort ein Byte übertragen. Wenn das Byte 00_{HEX} ist, dann ist der Block nicht schreibgeschützt. Wenn das Byte 01_{HEX} ist, dann ist der Block schreibgeschützt.

Parameter setzen
Set Parameters Request (61_{16})

Mit diesem Befehl kann man Parameter zur Übertragung der Daten zwischen Reader und Transponder setzen:

- Baud: High / Low: Gibt die Baudrate an
- UplnkMod: AM / FM: Modulationsverfahren
- DnlnkDepth: 10% / 100%: Modulationstiefe
- DnlnkCode: 1of4 / 1of256: Kodierung
- NonVolatile: ja /nein: Einstellungen über Reset hinaus behalten

Standardeinstellung: High Baud Rate, FM uplink, 10% downlink modulation, 1of4 coding



Parameter	Min	Max
hr	–	0.1 x (a-b)
hf	–	0.1 x (a-b)

Abbildung 7.7 10% Modulationstiefe

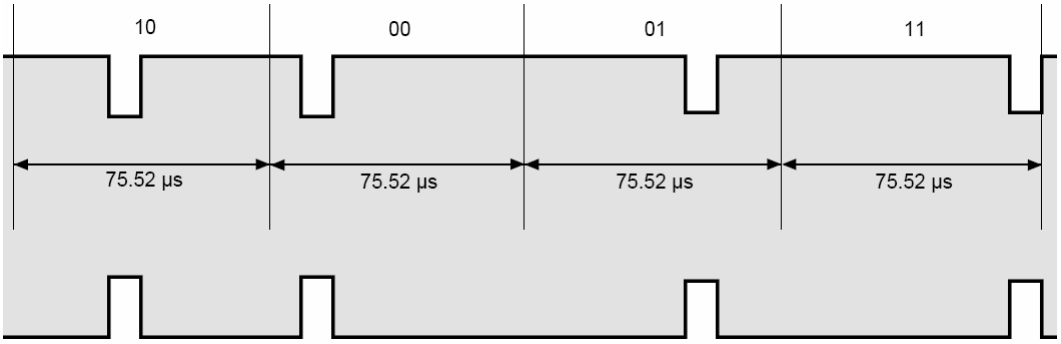


Abbildung 7.8 1-out-of-4-encoding

7.3.8 Gefundene RFID-Transponder

Dieser Bereich der Hauptoberfläche wird am häufigsten genutzt. Hier kann mit einem Knopfdruck (<Tag suchen> nach einem Transponder gesucht werden. Daraufhin wird der Befehl „Tag finden“ ausgeführt. Sobald die Transponder in der Liste stehen, kann man einen davon mit einem Klick markieren und mit dem Button <Tag bearbeiten> kommt man in die Detailansicht des Transponders.

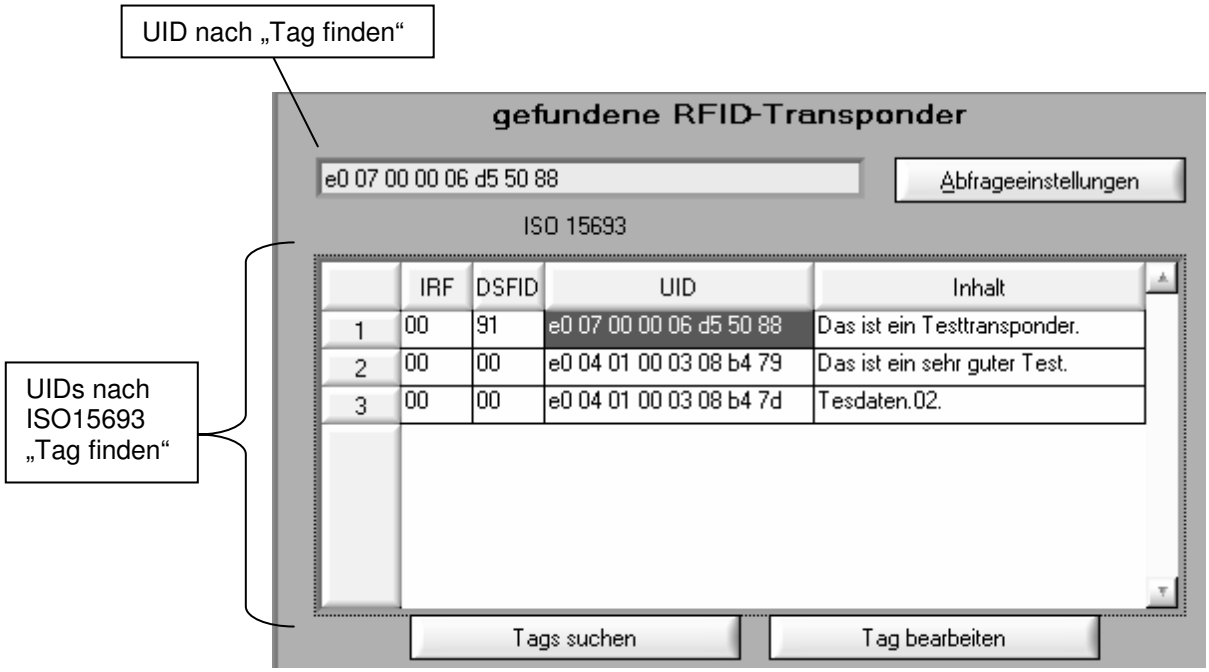


Abbildung 7.9 gefundene RFID-Transponder

IRF: Inventory Response Flag
DSFID: Data Storage Format ID
UID: Unique ID
Inhalt: Vorschau des Speicherinhalts

Durch einen Klick auf <Abfrageeinstellungen> öffnet sich ein Panel mit diversen Einstellmöglichkeiten.

7.3.9 Einstellungspanel

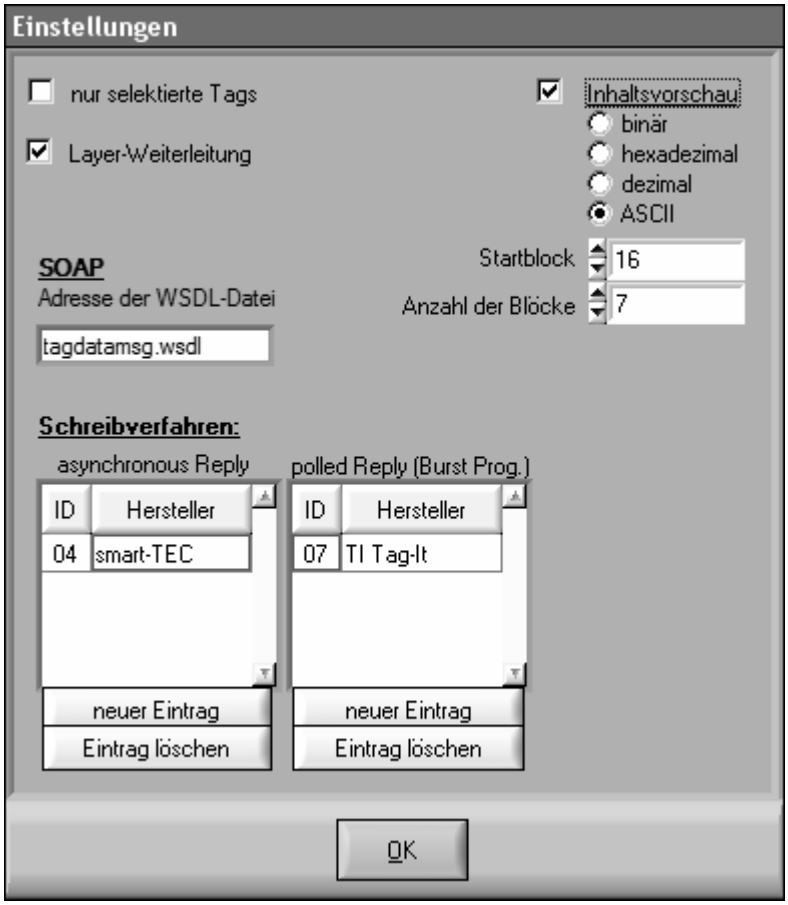


Abbildung 7.10 Einstellungspanel

Einstellungsmöglichkeiten:

nur selektierte Tags
Transponder können mit dem Befehl „Select Request“ (noch nicht in der Transportschicht implementiert) markiert werden. Bei manchen Befehlen, kann man bestimmen, dass nur so markierte Transponder angesprochen werden. Das Programm erkennt an diesem Häkchen, ob es auf markierte oder auf alle Transponder zugreifen soll.

Layer Weiterleitung
Mit der Layer-Weiterleitung kann man Befehle fix miteinander verknüpfen. So könnte man definieren, dass auf ein „Tag finden“ der Anwendungsebene, ein „Tag finden“ der speziellen Ebene folgt. So hat man den Vorteil, dass man alle Transponderarten findet (Anwendungsebene) und alle Transponder der gefundenen Transponderart (spezielle Ebene, z.B.: ISO15693).

Die Weiterleitung wird im Unterprogramm `nextcmd()` aufgerufen. In diesem Programm wird auf den zuletzt ausgeführten Befehl geprüft (also das Cmd1 der Antwort) und anhand dessen entschieden, welcher Befehl als nächster ausgeführt werden soll.

SOAP
Hier kann der Pfad zur WDSL-Datei angegeben werden.
Pfad der WSDL-Datei kann ein lokaler Pfad - z.B.: „C:\WSDL-Dateien\test.wsdl“ (absolut) oder „test.wsdl“ (relativ; in diesem Fall im Programmverzeichnis) - aber auch ein Pfad auf einem Server z.B.: http://172.31.11.21/SOAP/test.wsdl sein.

Falls die WSDL-Datei des Servers nicht geladen werden kann, empfiehlt es sich die lokale Kopie zu verwenden. Diese wird bei der Installation in den Programmordner kopiert → Pfad:“tagdatamsg.wsdl“.

Damit man sich zum Server verbinden kann, müssen die IP-Adressen in der WSDL-Datei geändert werden.

Inhaltsvorschau
Mit diesem Häkchen kann man die Inhaltsvorschau der „gefundene Transponder"-Tabelle einschalten. Die Funktion ist standardmäßig abgeschaltet um unnötige Kommunikationsvorgänge zwischen dem MFR und dem PC zu vermeiden.
Die Inhaltsvorschau kann in den ersichtlichen Formaten angezeigt werden. Zusätzlich kann man den Startblock und die Anzahl der Blöcke, die gelesen werden sollen, einstellen.

Schreibverfahren:
Der Speicher von Transpondern kann auf zwei verschiedene Arten beschrieben werden:

- polled
- asynchron

Es kann nicht abgefragt werden, auf welche Weise ein Transponder beschrieben werden muss, weshalb es eine gute Lösung ist, Tabellen anzugeben, welche Transponder auf welche Art angesprochen werden sollen.
Wir hatten nur zwei verschiedene Hersteller für die Transponder: smart-TEC (Philips) und Texas Instruments. Daher sind momentan nur diese beiden Einträge in den entsprechenden Listen vorhanden.
In den Listen muss die ID angegeben werden, die Spalte „Hersteller" dient nur zur besseren Übersicht. Die ID ist das zweite Byte der UID und somit das Herstellerkennzeichen (siehe Theoretische Grundlagen).

7.3.10 Tagdetails

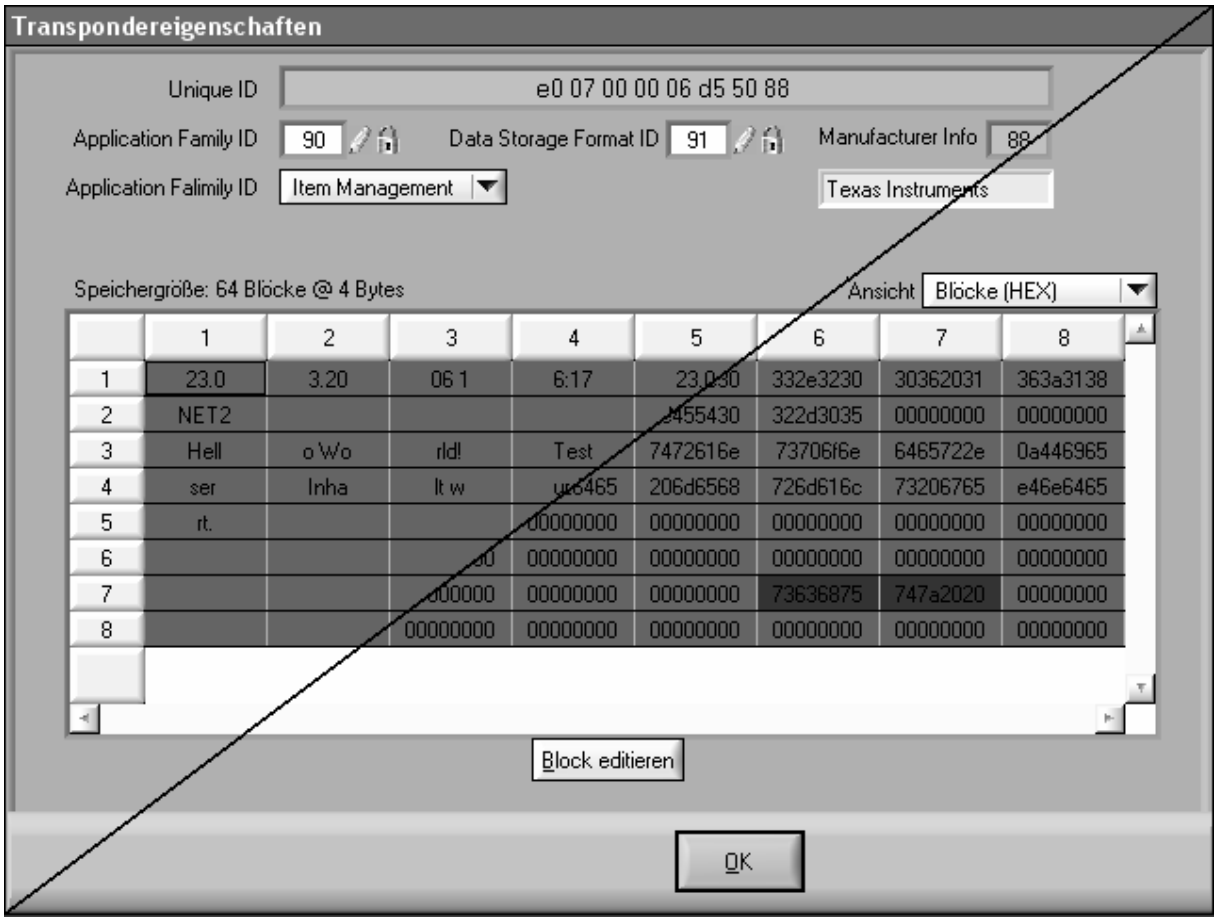


Abbildung 7.11 Tagdetail – Blockansicht (Links als ASCII-Zeichen, Rechts als HEX-Werte)

Das Tagdetail-Panel (oder auch Transpondereigenschaften-Panel) offenbart die ganze Information, die man von einem Transponder abfragen kann.

Es zeigt folgende Informationen:

- **Unique ID** (mit Interpretation des Herstellerbytes z.B.: 07_{HEX} = Texas Instruments)
- **Application Family ID** + Editier- und Schreibschutzfunktionen (als Hex-Zahl und entsprechender String z.B.: 90_{HEX} = Item Management)
- **Data Storage Format ID** + Editier- und Schreibschutzfunktionen
- **Manufacturer Info**
- **Speichergröße** (Blockanzahl x Bytes pro Block)
- **Speicherinhalt** (mit verschiedenen Ansichten)
- **Schreibgeschützte Bytes** (in der Block-Ansicht)

7.3.11 Speicheransichten

Das Programm stellt drei verschiedene Ansichtsarten für den erweiterten Speicher zu Verfügung. Die Ansichten werden über das Auswahlmenü <Ansicht> umgeschaltet.

Blockansicht:

In dieser Ansicht kann man genau sehen, welche Bytes in welchem Block stehen. Man kann in der im Auswahlmenü <Ansicht> zwischen der Blockansicht mit Hexadezimalwerten und ASCII-Zeichen wählen.

Weiters wird auch der Schreibschutzstatus angezeigt: Beschreibbare Blöcke werden mit grünem Hintergrund dargestellt, gesperrte (schreibgeschützte) Blöcke mit rotem Hintergrund.

In der Blockansicht hat man auch die Möglichkeit einen einzelnen Block auszuwählen und zu editieren indem man den gewünschten Block mit einem Klick markiert und dann den <Block editieren> - Button betätigt.

Es öffnet sich nun ein kleines Fenster, mit dem man den Inhalt des Blocks neu beschreiben kann. Mit dem <Format> Auswahlmenü kann man wählen ob man ASCII-Zeichen oder Hexadezimalwerte angibt. Mit dem <schreibgeschützt> Auswahlfeld kann man den Block schreibschützen. Sobald das erfolgreich durchgeführt wurde, wird der Block im Tagdetail-Panel mit einem roten Hintergrund versehen und wenn man wieder das „Block editieren"-Panel aufruft, ist das schreibgeschützt Häkchen bereits gewählt und da man in dem Panel ja nichts mehr mit den Funktionen machen kann, sind die Buttons und Eingabefelder gedimmt.

Abbildung 7.12 Block editierbar

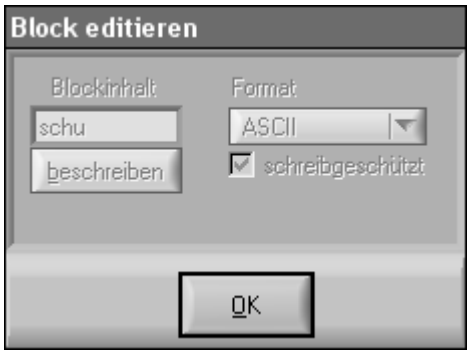


Abbildung 7.13 Block schreibgeschützt

Textansicht:

Diese Ansicht wurde programmiert um fremde Transponder, die die Daten als ASCII-Zeichen speichern, auszulesen. Bei dieser Ansicht sucht das Programm, wo Strings anfangen und aufhören und gibt diese dann in einer Liste aus. In jeder Zeile wird angegeben, bei welchem Block der String anfängt.

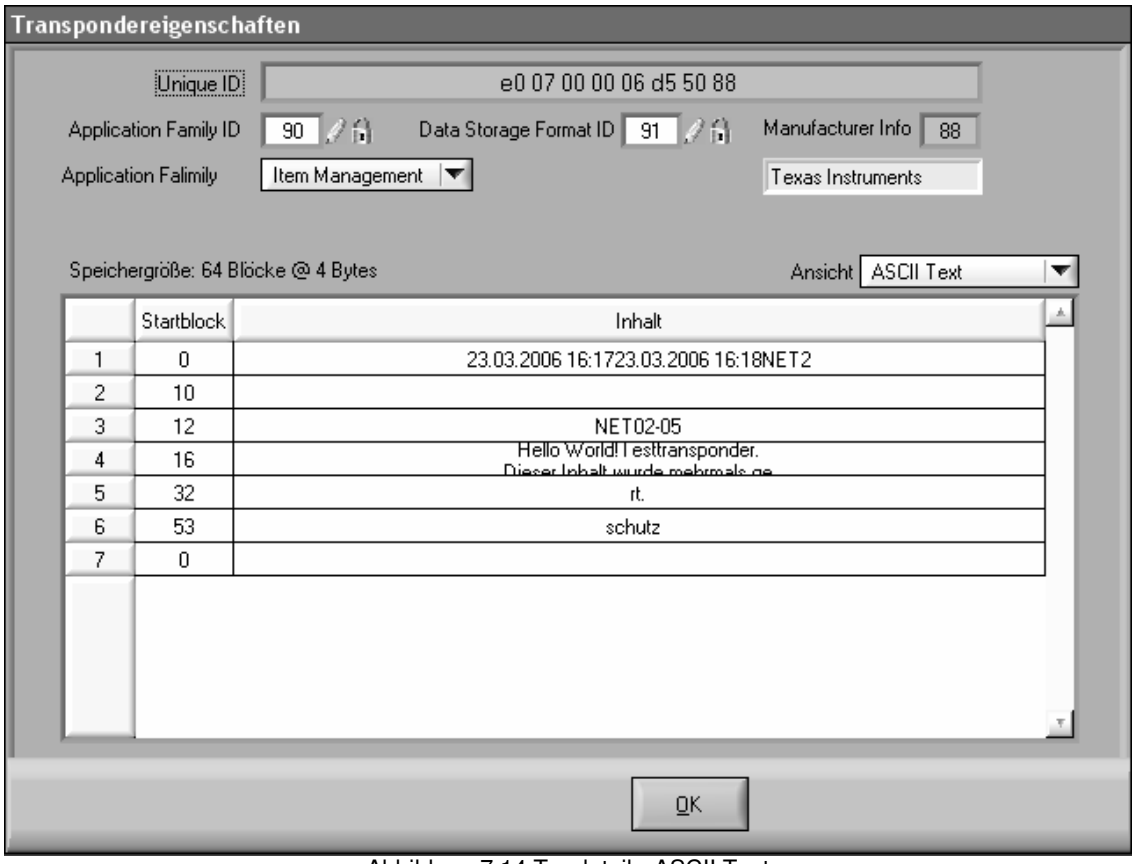


Abbildung 7.14 Tagdetail - ASCII Text

PC-Inventarisierung-Ansicht:

Diese Ansicht wurde entwickelt um als Eingabemaske für HTL-PC-Inventarisierungsdaten zu dienen.

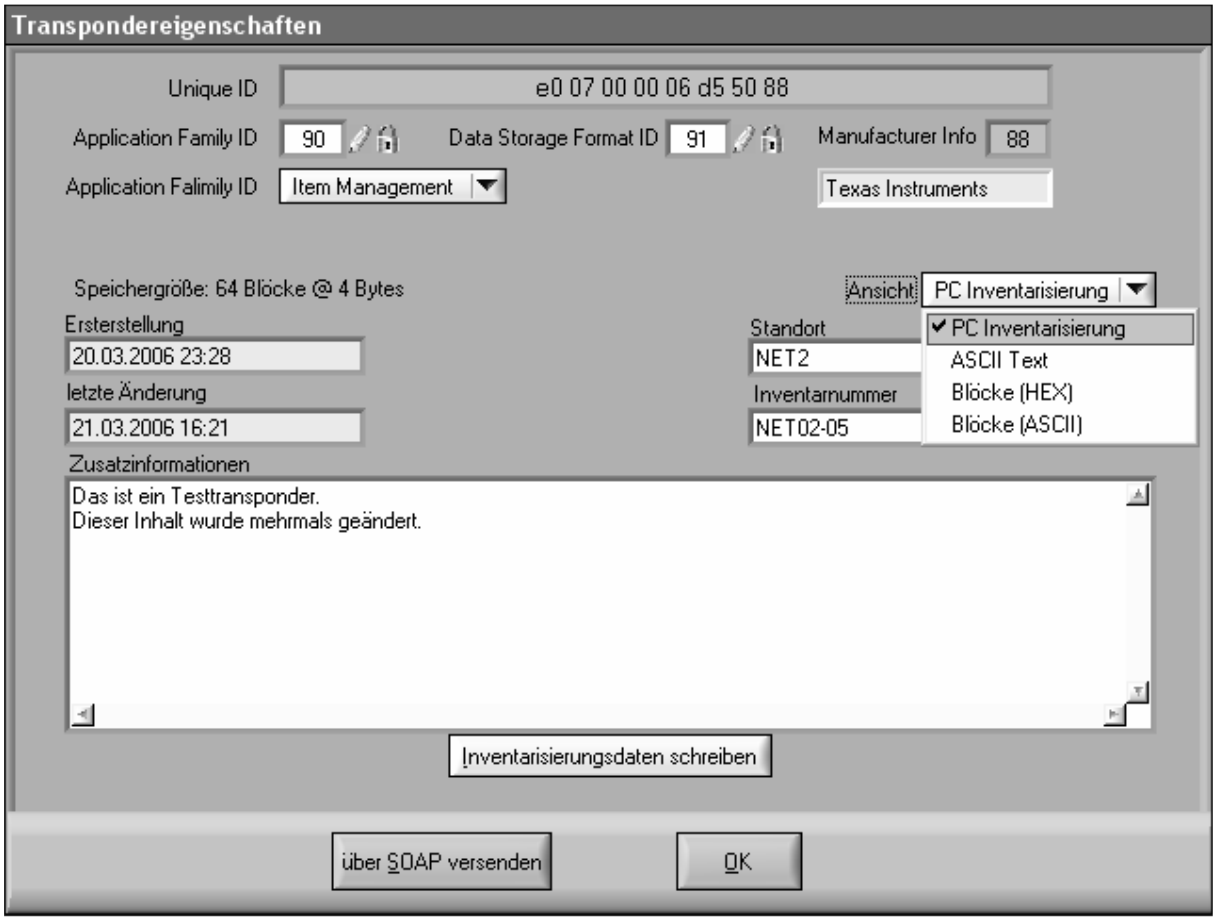


Abbildung 7.15 Tagdetail - PC-Inventarisierung-Ansicht

Das Programm erkennt HTL-PC-Inventarisierungstransponder an der AFI (90_{HEX}) und an der DSFID (91_{HEX}). Stimmt einer der beiden Werte nicht, so werden alle Felder als leer angezeigt. Man kann nun Standort und Inventarnummer eintragen, eventuell auch Zusatzdaten, und das Programm kann die Daten auf das Tag schreiben. Dabei schreibt es die Ersterstellungszeit und die Zeit der letzten Änderung ebenfalls auf den Transponder. Die AFI und die DSFID werden auf die richtigen Werte gesetzt und beim nächsten Auslesen sind die Felder ausgefüllt. Schreibt man nun nochmals auf den Transponder, so wird nur die Zeit und das Datum der letzten Änderung geändert, die Zeit und das Datum der Erstellung bleiben erhalten.

Durch den Button <Inventarisierungsdaten schreiben> werden die Daten auf den Transponder geschrieben. Es öffnet sich ein kleines Statuspanel, durch welches man erkennen kann, wie viele Blöcke insgesamt geschrieben werden müssen und wie weit der Schreibvorgang des Teilbereichs fortgeschritten ist.
Nach beendetem Schreibvorgang wird eine Zusammenfassung angezeigt. Man kann daraus herauslesen welche Blöcke erfolgreich und welche ohne Erfolg beschrieben wurden.

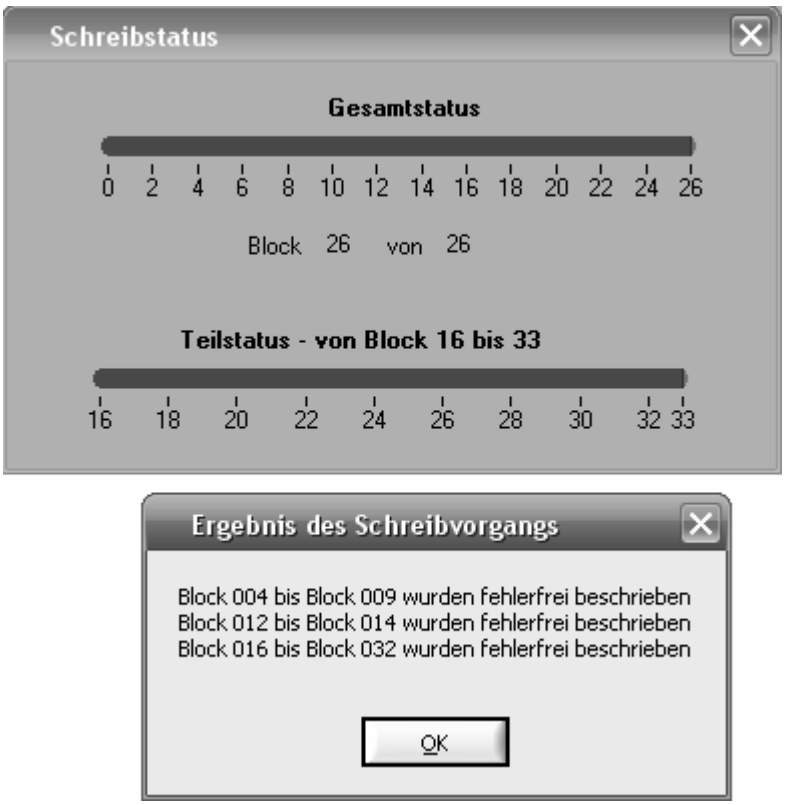


Abbildung 7.16 Erfolgreicher Schreibvorgang

In der PC-Inventarisierung-Ansicht ist auch der Button für die manuelle Übertragung der Daten an den Datenbankserver vorhanden: <über SOAP senden>.

Daraufhin öffnet sich ein Fenster mit den SOAP-Verbindungsnachrichten. Der Balken über dem Fenster verdeutlicht noch, ob die Übertragung geglückt (grün) oder fehlgeschlagen ist (rot).

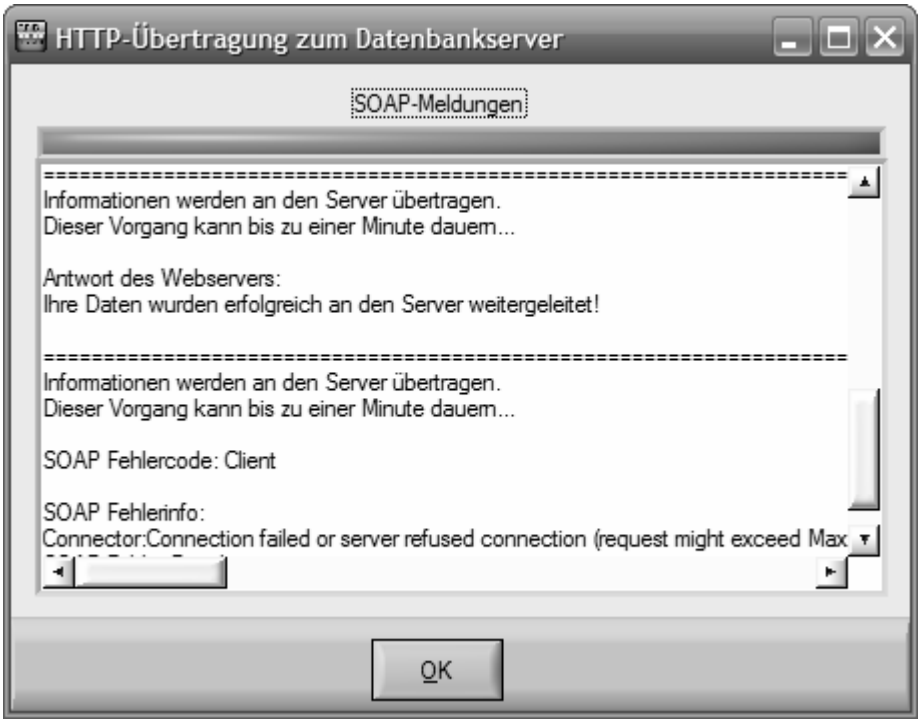


Abbildung 7.17 SOAP-Status

7.4 SOAP-Verbindung

Die SOAP-Verbindung wird vom Unterprogramm `soaprequest()` aufgebaut:

```
int soaprequest()
{ ...
  ...
  LPOLESTR functionName = L"sendTagData"; //Name der Webservice-Funktion
  ...
  //Paramtertypen (5 Strings)
  unsigned int     paramTypes[] = { CAVT_CSTRING, CAVT_CSTRING, CAVT_CSTRING,
                                    CAVT_CSTRING, CAVT_CSTRING};
  ...
  ...
  GetCtrlVal(tagHandle, TAG_UID, id);           // Unique-ID holen
  GetCtrlVal(tagHandle, TAG_INVENTNR, name);    // Inventarnummer des Rechners
  GetCtrlVal(tagHandle, TAG_PLACE, location);   // Standort des Rechners
  GetNumTableRows (panelHandle, TOP_TAGS, &h); // Anzahl der Tags
  sprintf(group, "%d", h);                      // als String

  //Erzeugen der SOAP-Client Objekts
  __caErrChk (MSSOAPLib30_NewISoapClient (NULL, 1, LOCALE_NEUTRAL, 0, &hClient));

  //Initialisieren des SOAP-Client-Objekts mit der Funktion; Angabe der WSDL-Datei
  __caErrChk              (MSSOAPLib30_ISoapClientMSSoapInit              (hClient,
NULL,"C:\\tagdatamsg.wsdl", "", "", ""));
  ...
  ...
  //Webservicefunktion aufrufen
  __caErrChk (CA_MethodInvokeEx (hClient, &errorInfo, &IID_IDispatch, functionId,
              CAVT_CSTRING, &ack, sizeof (paramTypes) / sizeof (paramTypes[0]),
              paramTypes, id, ip, name, location, group));

  MessagePopup ("Antwort des Webservers", ack);
}
```

Man kann erkennen, dass die Unique ID, die Inventarnummer, der Standort und die Anzahl der gelesenen Transponder (für zukünftige Anwendungen) an den Server übertragen werden.

Will man einen Parameter mehr über SOAP übertragen, so muss man nur

1. einen Paramter mehr bei den `paramTypes[]` hinzufügen
2. Die WSDL-Datei dementsprechend umändern
3. die Variable oder den Wert des Parameters beim Aufruf der Funktion (`CA_MethodInvokeEx`) übergeben

7.5 Skript

Damit der Benutzer nicht immer selbst die Buttons betätigen muss, besitzt die Transportschicht die Fähigkeit Skripte ausführen zu können. Zu diesem Zweck wurde eine sehr einfache Skriptsprache entwickelt.

Eigenschaften der Skriptsprache:

- Die Datei hat die Erweiterung .rfs (im Installer enthaltene Skriptdatei: script.rfs und start.rfs)
- Jede Zeile beginnt mit einem "-". (z.B.: - SOAP)
- Die Anzahl der "-" gibt die Abhängigkeit an → Ein Befehl mit 2x"-" wird nur ausgeführt wenn der vorhergehende Befehl mit einem "-" keine Fehlermeldung verursachte. (z.B.: wird --- *SOAP* nicht ausgeführt wenn das vorhergehende -- *SEND* nicht erfolgreich ausgeführt werden konnte.
- Das Skript wird nach der letzten Zeile wieder von vorne gestartet

Die Befehle, die abgearbeitet werden können, sind im Unterprogramm `runcmd()` realisiert. Will man also den Skript-Befehlssatz erweitern so braucht man nur in diesem Programm den entsprechenden Eintrag machen:

```
int runcmd (char *cmd)
{ char command[30];       //der Befehl
  char attribut[30];      //und das Attribut
  int noerror=0;          //Fehler?, Wichtig für Abhängigkeiten
  StringUpperCase (cmd); //damit nicht an Groß- /Kleinschreibung gebunden
  sscanf(cmd, "%s %s", command, attribut);  //Aufteilen in Befehl und Attribut
  ...
  if(strcmp(command,"ISO15693CMD")==0)          //Einstellen eines ISO15693 Befehls
   {sscanf(attribut, "%d", &h);                 //Aus Attribut -> Zahlenwert machen
    SetCtrlVal(panelHandle,TOP_ISO15693CMD, h); //den Befehl einstellen
    noerror=1;                                  //kein Fehler
   }

  if(strcmp(command,"SEND")==0)                 //schicken des Befehls an den MFR
   {sendRFID();                                 //Aufruf der Sendefunktion
    GetCtrlAttribute(panelHandle, TOP_ERROR, ATTR_TEXT_BGCOLOR, &noerror);
    if (noerror==0xFF5555) noerror=0;           //Auf Fehler prüfen
    else noerror=1;
   }

  if(strcmp(command,"SOAP")==0)                 //schicken der Daten über SOAP
     {SetCtrlVal(tagHandle, TAG_view, 1);       //Einstellen der richtigen Ansicht
       fillincontent();                         //Inhalt der Felder füllen
       soaprequest();                           //über SOAP senden
       noerror=1;                               //Fehlerfrei
     }
       ...
  return noerror;

}
```

Eine Skriptzeile wird also in Befehl und Attribut aufgeteilt und anschließend weiterverarbeitet. Es muss bei der Erstellung der Skriptdatei auf die Leerzeichen geachtet werden.
Eine Skriptzeile ist also folgendermaßen aufgebaut:

1. ein oder mehrere „-“
2. Leerzeichen
3. Befehl
4. Leerzeichen
5. Attribut
6. optional: Leerzeichen + Zusatzattribut

Einfaches Beispiel für eine Skriptdatei: Lesen von Transponderdaten und Ausgabe über SOAP

```
- layer 4
-- iso15693CMD 65
--- SEND
---- UID LIST
----- ISO15693cmd 112
------ SEND
------- SOAP
```

Hier ist jeder Befehl von seinem vorhergehenden abhängig. D.h. wenn ein Befehl einen Fehler zurückliefert startet das Skript wieder von vorne.

Um das Skript zu starten, muss in der Menüleiste → Datei → Script → starten gewählt werden. Um das Skript zu stoppen: Datei → Script → beenden.

7.5.1 Startskript

Beim Start des Programms wird ebenfalls eine Skriptdatei ausgeführt. Dieses Skript wird nur einmal durchlaufen und dient zur Konfiguration des Programms. Die Skriptdatei wird ebenfalls von `runcmd()` verarbeitet. Es können also alle Skriptbefehle im Startskript aufgerufen werden.

Die Startskriptdatei hat den Dateinamen start.rfs und befindet sich im Programmverzeichnis.

7.5.2 Realisierte Script-Befehle

LAYER	Stellt die Ebene des Befehls ein (siehe Punkt 7.2.2)
ISO15693CMD	Stellt den gewünschten Befehl ein
UID	Kopiert die UID aus der Auflistung der gefundenen Tags
SEND	Schickt den Befehl an den Reader
SOAP	Schickt die Tagdaten über die XML-Schnittstelle
LOOPS	Stellt die Schleifenanzahl für den „Tag suchen“-Befehl ein
WSDL	Stellt den Pfad für die WSDL-Datei ein
LED2	Stellt ein, wann die LED2 angesteuert werden soll (z.B.: soap, tag, error)
LED3	Stellt ein, wann die LED3 angesteuert werden soll (z.B.: soap, tag, error)
SUMMER	Stellt ein, wann der Summer angesteuert werden soll (z.B.: soap, tag, error)
ASYNCH	Fügt einen Eintrag zur Liste der asynchron anzusprechenden Tags hinzu
POLLED	Fügt einen Eintrag zur Liste der synchron anzusprechenden Tags hinzu

Beispiele für die Anwendung der Befehle sind in den Skriptdateien script.rfs und start.rfs enthalten.

7.6 Logdatei

Das Programm kann auf Wunsch alle Schreib- und Lesevorgänge zwischen Reader und PC aufzeichnen. In der Log-Datei kann man Datum und Uhrzeit des Vorgangs einsehen und welche Daten übertragen worden sind. Im Logfile werden auch SOAP-Aktivitäten aufgezeichnet.

Ein Logfile könnte also folgendermaßen Aussehen:

```
17.03.2006 09:06:35 | Sendestring:     01 14 00 03 04 68 00 00 3d 02 88 50 d5 06 00 00 07
e0 a9 56
17.03.2006 09:06:36 | Empfangsstring: 01 16 00 03 04 68 00 00 00 00 00 00 00 00 00 00
00 00 00 78 87
17.03.2006 09:06:36 | Sendestring:     01 13 00 03 04 71 00 00 3f 88 50 d5 06 00 00 07 e0
b7 48
17.03.2006 09:06:36 | Empfangsstring: 01 4a 00 03 04 71 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3d c2
17.03.2006 09:06:37 | SOAP-Anfrage: ID:e0 07 00 00 06 d5 50 88   Inventarnummer:NET02-01
Standort:NET2          Transponderzahl1
17.03.2006 09:06:39 | Sendestring:     01 09 00 03 04 41 01 4f b0
17.03.2006 09:06:39 | Empfangsstring: 01 09 00 03 04 41 01 4f b0
```

Um die Aufzeichnung zu starten, muss in der Menüleiste → Datei → Aufzeichnung → starten gewählt werden. Um das Skript zu stoppen: Datei → Aufzeichnung → beenden.

7.7 HTL-Programm vs. Demoprogramme

Was unterscheidet die Transportschicht von der mitgelieferten RFID-Demonstrations-Software:

Tabelle 7.3 Vergleich zu anderen Programmen

Funktion	Texas Instruments Demoprogramm	DataBrokers PIRF Lite 2.0	HTL Hollabrunn RFIDinvent
Erkennen mehrerer Transpondertypen	ja	ja	ja
Ansteuerung der Ausgänge / selbst konfigurierbar / manuell ansteuerbar (Button)	ja / nein / nein	ja / nein / nein	ja / ja / ja
Einstellen der COM-Schnittstelle / mit Autoset	ja / nein	ja / nein	ja / ja
Lesen und Beschreiben von LF Transpondern	nein	ja	nein
Lesen und Beschreiben von HF Transpondern	nein	ja	ja
Beschreiben im asynchronous reply mode (nicht-TI-Tags)	nein	nein	ja
Fortschrittsbalken beim Schreibvorgang	nein	nein	ja
Detailinformationen über einen Transponder	nein	nein	ja
Aufzeichen einer Logdatei	nein	ja	ja
Ausführen von Scripts	nein	nein	ja
SOAP - Kommunikation	nein	nein	ja
DSFID & AFI sehen, beschreiben und sperren	nein	nein	ja
bestimmte Blöcke schreiben	nein	nein	ja
Blöcke sperren	nein	nein	ja
Konfigurationsmöglichkeiten	keine	wenig	mehr
Anzeige von gesperrten Blöcken	nein	nein	ja

8 Webanwendung

8.1 Lokale Entwicklungsumgebung (XAMPP)

8.1.1 Allgemeines

Wie bereits in Kapitel 5 erwähnt wurde als lokale Entwicklungsumgebung für die Webanwendung die Software Distribution XAMPP verwendet.
Die genutzte Version, XAMPP 1.4.14 für Windows, setzt sich aus mehreren Komponenten zusammen. Folgende wurden verwendet:

- PHP 5.0.4
- Apache 2.0.54 (Win32)
- phpMyAdmin 2.6.2-pl1
- MySQL 4.1.12-nt

8.1.2 Installation

Die Installation gestaltet sich als sehr einfach. Nach dem Download der XAMPP Version muss die .exe Datei einfach ausgeführt werden.

Abbildung 8.1 Auszuführende Datei

Bei der Installation legt XAMPP folgende Verzeichnisstruktur an:

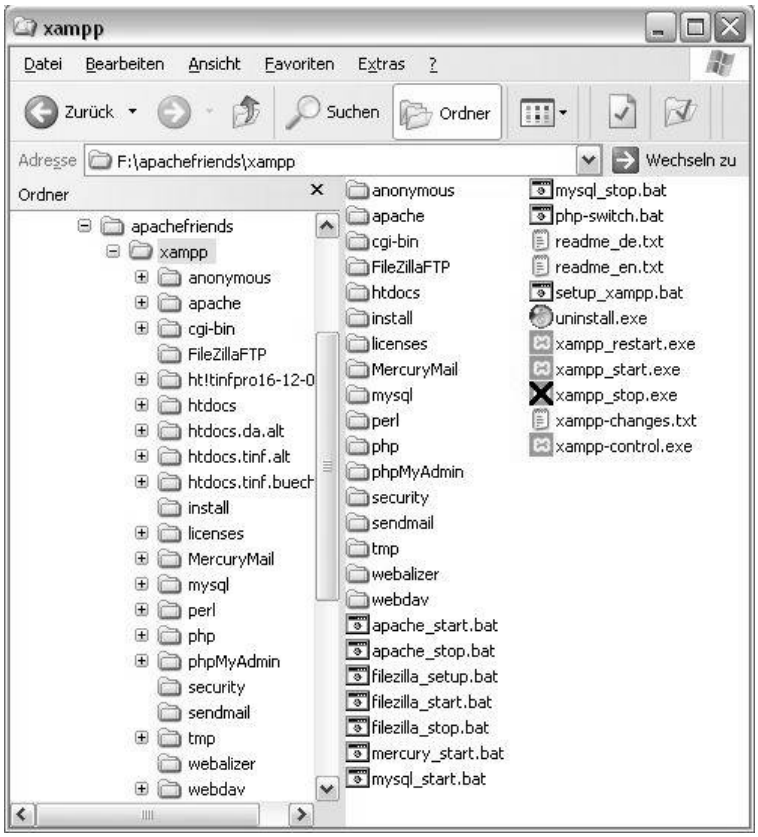


Abbildung 8.2 XAMPP Verzeichnisstruktur

Standardmäßig wird XAMPP in das Verzeichnis C:\apachefriends\xampp installiert. Dieser Pfad kann natürlich geändert werden.

Wichtige Verzeichnisse:

htdocs\	Dieser Ordner ist das Root Verzeichnis des Webservers. In ihm befindet sich das entwickelte HTML bzw. PHP Projekt.
apache\bin\	Hier befindet sich die Datei php.ini, die zum Konfigurieren des PHP API dient.

Deinstallation

Ebenso einfach wie die Installation erfolgt auch die Deinstallation von XAMPP (Ausführen von uninstall.exe im Installationsverzeichnis). Bei der Deinstallation sollte jedoch darauf geachtet werden, bereits erstellte Webanwendungen bzw. das htdocs Verzeichnis vorher zu sichern.

8.1.3 Praktisches Arbeiten mit XAMPP

Die im Installationsverzeichnis vorhandenen Anwendungen xampp_start.exe und xampp_stopp.exe dienen zum Starten bzw. Beenden von XAMPP.

Aufgetretene Probleme:

Es hat den Anschein, dass es einige Anwendungen gibt, die sich nicht mit XAMPP vertragen. Aus eigener Erfahrung verhinderte zum Beispiel das für VOIP Telefonie verwendete Tool Skype das Starten des Webservers.

8.1.4 Testen der Installation

Nachdem man XAMPP auf seinem System gestartet hat, sollte man, wenn man nun mit einem Browser auf die URL `http://localhost/xampp` zugreift, die Startseite von XAMPP sehen. Der Screenshot in Abbildung 8.3 zeigt die zu erwartende Ausgabe.

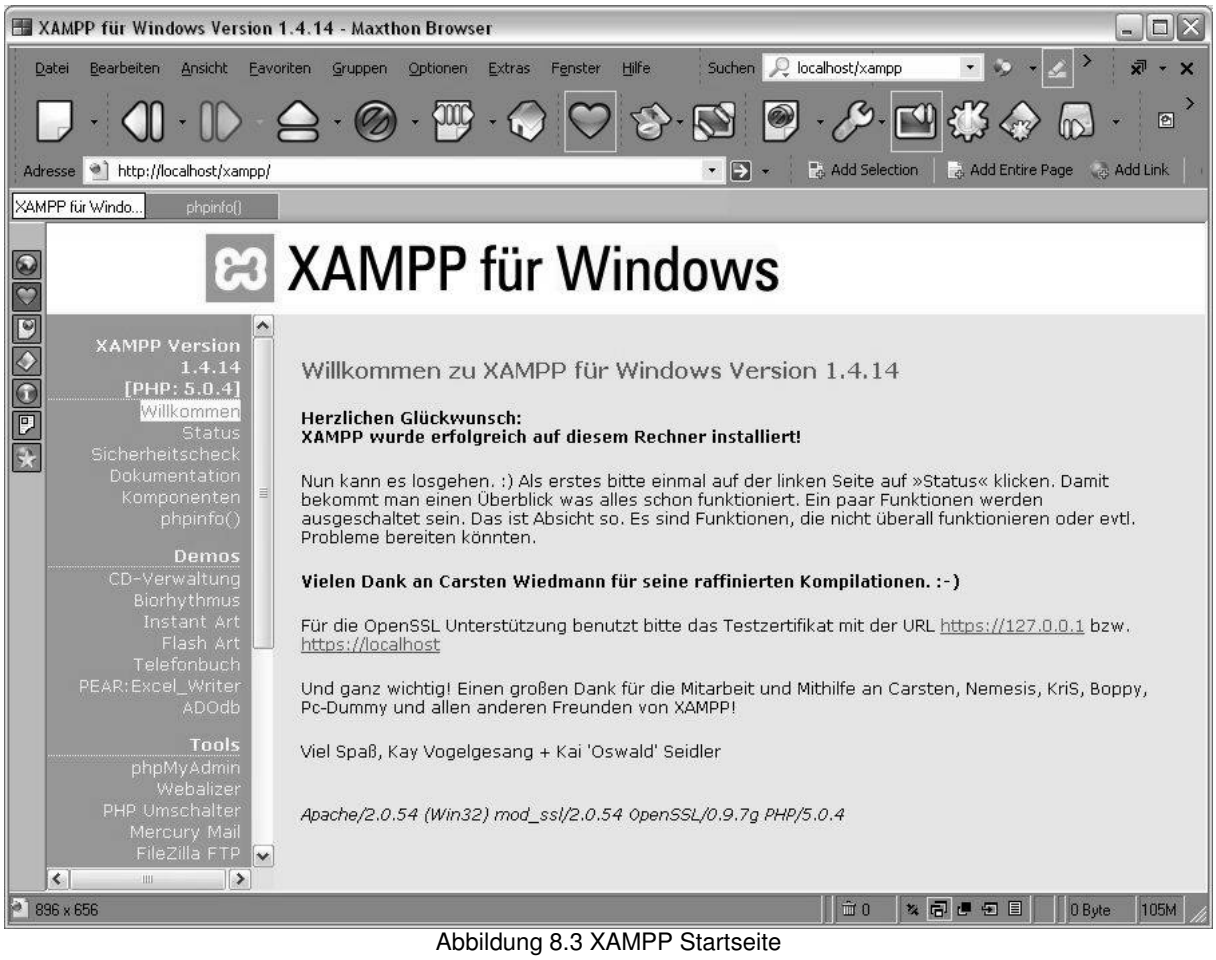


Abbildung 8.3 XAMPP Startseite

Zum testen der PHP API kann das folgende in ein HTML Gerüst eingebettete PHP Skript verwendet werden:

```
<html>
	<head>
		<title>PHP Skript</title>
	</head>
	<body>
		<h1>Es ist jetzt der <?php echo date("j. F Y H:i"); ?> Uhr</h1>
	</body>
</html>
```

Die PHP Funktion `date()` zeigt das aktuelle Datum und die Uhrzeit an.
Die Datei kann anschließend unter test.php im htdocs Verzeichnis abgespeichert werden.
Der Aufruf der PHP Seite im Browser mit `http://localhost/test.php` sollte folgende Ausgabe erzeugen:

Abbildung 8.4 PHP API testen

Erscheint diese Ausgabe so ist die Installation von XAMPP erfolgreich verlaufen.

8.1.5 Aktivieren der PHP SOAP-Extension

Zur Kommunikation mit dem CVI SOAP Client Programm benötigten wir eine Möglichkeit einen SOAP Server zu erstellen.
PHP enthält ab der Version 5.0!! die sogenannte SOAP-Extension. Allerdings ist diese Extension standardmäßig deaktiviert. Um sie zu aktivieren muss das Semikolon vor dem folgenden Eintrag in der php.ini Datei entfernt werden.
Achtung: Bei der Installation von XAMPP werden mindestens drei Konfigurationsdateien mit dem Namen php.ini erzeugt. Die Änderung muss in der php.ini Datei in dem Verzeichnis <Installationspfad>\xampp\apache\bin vorgenommen werden.

```
597 ;extension=php_radius.dll
598 ;extension=php_rar.dll
599 ;extension=php_shmop.dll
600 ;extension=php_snmp.dll
601 extension=php_soap.dll
602 ;extension=php_sockets.dll
603 ;extension=php_stats.dll
604 ;extension=php_sybase_ct.dll
605 ;extension=php_threads.dll
```

Abbildung 8.5 SOAP-Extension aktivieren

Um die Änderung zu übernehmen muss der Server neu gestartet werden (am besten mit restart.exe).

8.1.6 phpMyAdmin

phpMyAdmin ist ein in der XAMPP Distribution enthaltenes Tool, das es ermöglicht, mit wenigen Mausklicks Datenbanken zu administrieren. Im Prinzip macht PHPMyAdmin nichts anderes, als Eingaben in MySQL-Befehle umzuwandeln. Durch das Tool kann viel Zeit und Mühe eingespart werden, da die Datenbankadministration mit den selben Möglichkeiten ohne Kommandozeile möglich ist.
Das Programm selbst ist eine Zusammensetzung von PHP-Skripten bzw. Dateien und ist unter `http://localhost/phpmyadmin` erreichbar.

Datenbank anlegen:
Das anlegen einer Datenbank ist direkt über die Startseite von phpMyAdmin möglich.
Durch den Button „Anlengen“ (Markierung 2) wird die Datenbank mit dem gewünschten Namen aus dem Formularfeld (Markierung 1) angelegt.

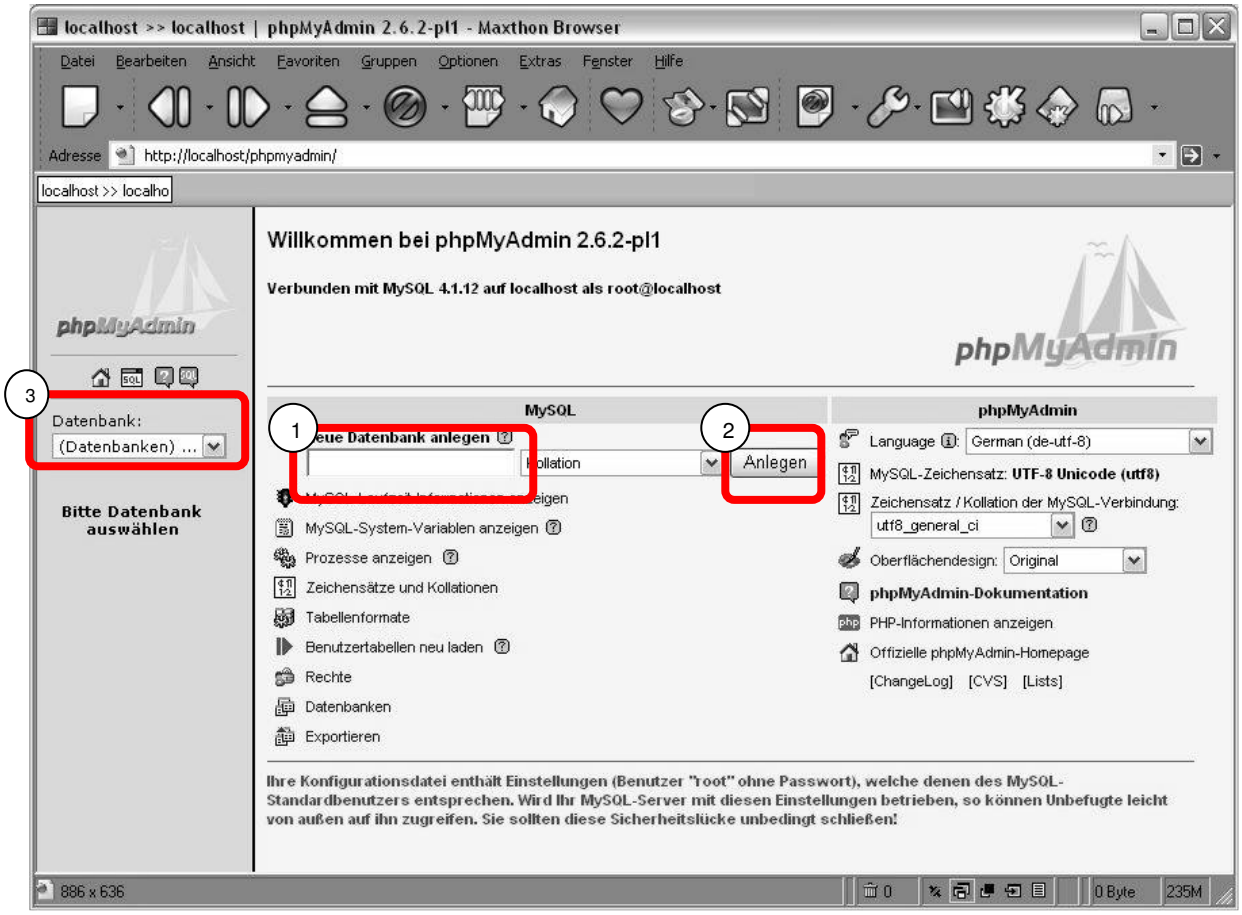


Abbildung 8.6 phpMyAdmin – Startseite

Im folgenden Beispiel wurde eine Datenbank mit dem Namen test_tb angelegt. Im Drop-Down Menü (Abbildung 8.6, Markierung 3) kann diese anschließend ausgewählt werden um eine Tabelle hinzuzufügen.

Abbildung 8.7 phpMyAdmin – Tabelle anlegen

In die Eingabefelder (Abbildung 8.7, Markierung 1) muss der Name und die Spaltenanzahl der Tabelle angegeben werden. In diesem Fall eine Tabelle mit 3 Spalten und dem Namen test_table.

Nach dem Bestätigen der Eingaben mit OK erscheint folgende Eingabemaske:

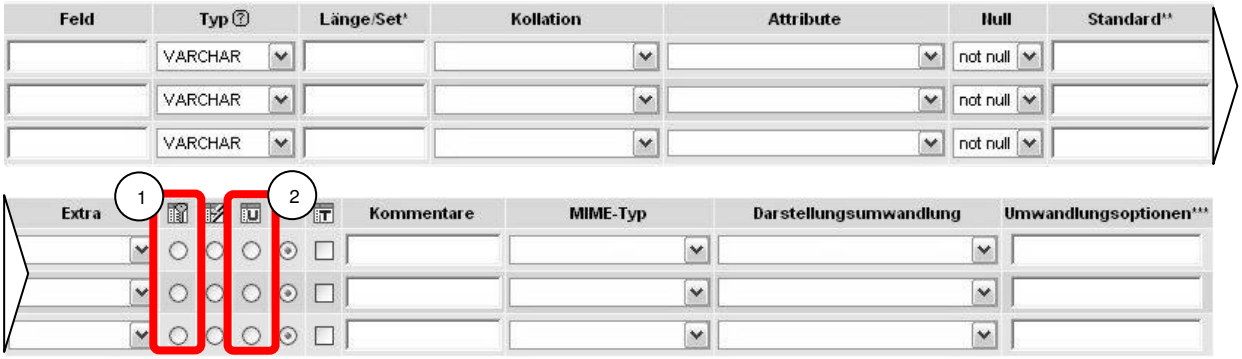


Abbildung 8.8 phpMyAdmin – Tabellenstruktur

Erklärungen zu Abbildung 8.8:

Feld:

Name der Spalte, keine Umlaute, Leerzeichen.

Typ:

Auswahl des Datentyps der Spalte
Hier die wichtigsten Datentypen:

- INT
  Deklariert eine Ganzzahl
- VARCHAR
  Deklariert Zeichenfolgen mit variabler Zeichenanzahl (max. 256 Zeichen pro Spaltenelement).
- TEXT
  Deklariert eine Text-Spalte mit einer maximalen Länge von 65535 Zeichen pro Spaltenelement.
- TIMESTAMP
  Deklariert einen Zeitstempel. Eine TIMESTAMP-Spalte ist nützlich, um Datum und Zeit einer INSERT- oder UPDATE-Operation zu speichern, weil sie automatisch auf das Datum und die Zeit der jüngsten Operation gesetzt wird, wenn man nicht selbst einen Wert zuweist.

.

Länge:

Wird bei unserer Anwendung nur für VARCHAR Werte gebraucht um die maximale Zeichenanzahl anzugeben.

Kollation:

Zeichensatz

Null:

Hier kann festgelegt werden ob einer Spalte bei jedem Eintrag ein Wert zugewiesen werden muss (NOT NULL) oder nicht (NULL). NULL bedeutet das ein Feld leer ist.

Standard:

Festlegung eines Standardwertes für eine Spalte. Dieser Wert wird bei jedem neuen Datensatz in der entsprechenden Spalte gespeichert, sofern kein anderer Wert in der INSERT Anweisung angegeben wird.

Extra:

INT Werten kann hier die Eigenschaft auto_increment zugewiesen werden. Das heißt, dass bei jedem neuen Tabelleneintrag der entsprechende Spaltenwert um 1 gegenüber dem letzen Tabelleneintrag erhöht wird.

Primary Key (Abbildung 8.8, Markierung 1):
Die Spalte wird als Primärschlüssel deklariert. Das heißt alle Spalteneinträge müssen eindeutig und vorhanden (NOT NULL) sein. Es kann nur einen Primärschlüssel pro Tabelle geben.

Unique (Abbildung 8.8, Markierung 2):
Unique kennzeichnet einen Spalteneintrag als eindeutig.

Die hier nicht näher beschriebenen Felder aus Abbildung 8.8 werden für unsere Zwecke nicht zwingend benötigt. Sie können einfach leer bleiben.

Mit einem Klick auf 

wird die Tabellenstruktur übernommen.

Das Befüllen der Tabellen mit Werten übernimmt in weiterer Folge ein PHP Skript. Grundlagen zum Ansprechen einer Datenbank mit PHP siehe Kapitel 6.3.8

8.2 Datenbankmodellierung

8.2.1 Allgemeines

Die Entwicklung einer Datenbank vollzieht sich in mehreren Schritten. Zunächst ist festzustellen, welche Informationen die Anwender vom Datenbanksystem erwarten.
Aufgrund der Festlegungen im Pflichtenheft kann man sich überlegen, welche Tabellen benötigt werden. Ferner muss festgelegt werden, welche Datentypen den einzelnen Tabellenspalten zugewiesen werden.
Die Hauptaufgabe der Datenbankmodellierung ist das Verhindern von redundanten Daten sowie das Sicherstellen der eindeutigen Identifizierbarkeit von Datensätzen.

8.2.2 ER-Modell

Für die Webanwendung wurde eine Datenbank verwendet die aus 5 Tabellen besteht.
Die Bezeichnung der entwickelten Datenbank lautet `pcinventar`.

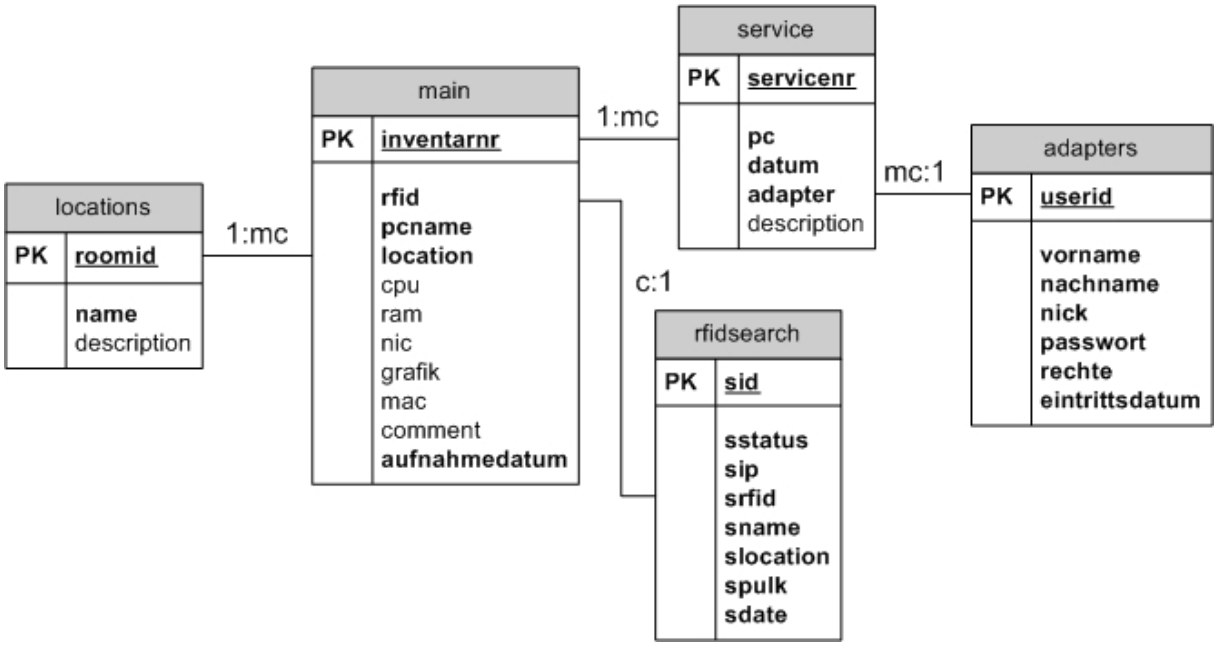


Abbildung 8.9 ER-Modell

8.2.3 Tabellenstruktur

Anmerkungen: *kursiv* → Primary Key, **fett** → Unique;

Tabellenstruktur für Tabelle **adapters**:

Feld	Typ	Null	Standard
userid	int(11)	Nein	
vorname	varchar(30)	Nein	
nachname	varchar(30)	Nein	
nick	varchar(20)	Nein	
passwort	varchar(20)	Nein	
rechte	int(11)	Nein	0
eintrittsdatum	timestamp	Ja	CURRENT_TIMESTAMP

Diese Tabelle dient als Benutzertabelle. In ihr werden die Registrierungsdaten der Systemuser gespeichert.

Tabellenstruktur für Tabelle **locations**:

Feld	Typ	Null	Standard
roomid	int(11)	Nein	
name	varchar(30)	Nein	
description	text	Ja	NULL

In dieser Tabelle werden die Computer-Räume gespeichert.

Tabellenstruktur für Tabelle **main**:

Feld	Typ	Null	Standard
inventarnr	int(11)	Nein	
rfid	varchar(100)	Nein	
pcname	varchar(30)	Nein	
location	int(11)	Nein	0
cpu	varchar(30)	Ja	NULL
ram	varchar(30)	Ja	NULL
nic	varchar(30)	Ja	NULL
grafik	varchar(30)	Ja	NULL
mac	varchar(30)	Ja	NULL
comment	text	Ja	NULL
aufnahmedatum	timestamp	Ja	CURRENT_TIMESTAMP

In der Tabelle „main“ werden die Rechner mit ihrer Hardwarekonfiguration abgelegt. Sie dient der Verwaltung des Rechner Inventars.

Tabellenstruktur für Tabelle **rfidsearch**:

Feld	Typ	Null	Standard
sid	int(11)	Nein	
sstatus	varchar(20)	Nein	
sip	varchar(20)	Nein	
srfid	varchar(100)	Nein	
sname	varchar(30)	Ja	NULL
slocation	varchar(30)	Ja	NULL
spulk	int(11)	Nein	0
sdate	timestamp	Ja	CURRENT_TIMESTAMP

Diese Tabelle wird als Zwischenspeicher bei der Übertragung der Tag-Daten über das XML-Interface genutzt.

Tabellenstruktur für Tabelle **service**:

Feld	Typ	Null	Standard
servicenr	int(11)	Nein	
pc	int(11)	Nein	0
datum	timestamp	Ja	CURRENT_TIMESTAMP
adapter	int(11)	Nein	0
description	text	Ja	NULL

In dieser Tabelle werden die an den Rechnern getätigten Wartungen festgehalten.

SQL Syntax:

```
-- Datenbank: 'pcinventar'

-- --------------------------------------------------------

--
-- Tabellenstruktur für Tabelle 'adapters'
--

CREATE TABLE 'adapters' (
  'userid'          int(11)      NOT NULL auto_increment,
  'vorname'         varchar(30)  NOT NULL default '',
  'nachname'        varchar(30)  NOT NULL default '',
  'nick'            varchar(20)  NOT NULL default '',
  'passwort'        varchar(20)  NOT NULL default '',
  'rechte'          int(11)      NOT NULL default '0',
  'eintrittsdatum' timestamp     NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  ('userid'),
  UNIQUE KEY 'nick' ('nick')
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_general_ci
;

-- --------------------------------------------------------
```

```
--
-- Tabellenstruktur für Tabelle ‘locations’
--

CREATE TABLE ‘locations’ (
  ‘roomid’          int(11)      NOT NULL auto_increment,
  ‘name’            varchar(30)  NOT NULL default ‘‘,
  ‘description’     text,
  PRIMARY KEY  (‘roomid’)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_general_ci
;

-- --------------------------------------------------------

--
-- Tabellenstruktur für Tabelle ‘main’
--

CREATE TABLE ‘main’ (
  ‘inventarnr’      int(11)      NOT NULL auto_increment,
  ‘rfid’            varchar(100) NOT NULL default ‘‘,
  ‘pcname’          varchar(30)  NOT NULL default ‘‘,
  ‘location’        int(11)      NOT NULL default ‘0’,
  ‘cpu’             varchar(30)  default NULL,
  ‘ram’             varchar(30)  default NULL,
  ‘nic’             varchar(30)  default NULL,
  ‘grafik’          varchar(30)  default NULL,
  ‘mac’             varchar(30)  default NULL,
  ‘comment’         text,
  ‘aufnahmedatum’   timestamp    NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  (‘inventarnr’),
  UNIQUE KEY ‘rfid’ (‘rfid’)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_general_ci
;

-- --------------------------------------------------------
--
-- Tabellenstruktur für Tabelle ‘rfidsearch’
--

CREATE TABLE ‘rfidsearch’ (
  ‘sid’             int(11)      NOT NULL     auto_increment,
  ‘sstatus’         varchar(20)  NOT NULL     default ‘‘,
  ‘sip’             varchar(20)  NOT NULL     default ‘‘,
  ‘srfid’           varchar(100) NOT NULL     default ‘‘,
  ‘sname’           varchar(30)  default NULL,
  ‘slocation’       varchar(30)  default NULL,
  ‘spulk’           int(11)      NOT NULL     default ‘0’,
  ‘sdate’           timestamp    NULL         default CURRENT_TIMESTAMP,
  PRIMARY KEY  (‘sid’)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_general_ci
;

-- --------------------------------------------------------

--
-- Tabellenstruktur für Tabelle ‘service’
--

CREATE TABLE ‘service’ (
  ‘servicenr’       int(11)      NOT NULL auto_increment,
  ‘pc’              int(11)      NOT NULL default ‘0’,
  ‘datum’           timestamp    NOT NULL default CURRENT_TIMESTAMP,
  ‘adapter’         int(11)      NOT NULL default ‘0’,
  ‘description’     text,
  PRIMARY KEY  (‘servicenr’)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_general_ci
;
```

8.3 Oberfläche

8.3.1 Allgemeines

Das aussehen der Oberfläche wird grundlegend durch HTML bzw. CSS Komponenten bestimmt. Sie ist über das ganze System konsistent und einheitlich.
Zusätzlich wird die Oberfläche der Rechtekategorie eines Benutzers angepasst. Die Realisierung der dynamischen Oberfläche bzw. des dynamischen Menüs ist in Kapitel 8.5 ersichtlich.

8.3.2 Screenshot der Oberfläche

Abbildung 8.10 zeigt einen Screenshot der realisierten Oberfläche mit Administrator Rechten.
Markierung 1 kennzeichnet das dynamische Menü, das sich der Rechtekategorie des Benutzers anpasst. Das Menü dient der Navigation und ist immer, d.h auf allen Unterseiten des Projektes vorhanden.
Markierung 2 hebt den Aktionsbereich hervor.

Abbildung 8.10 Oberfläche

8.3.3 Seitenaufbau - Dialogstruktur

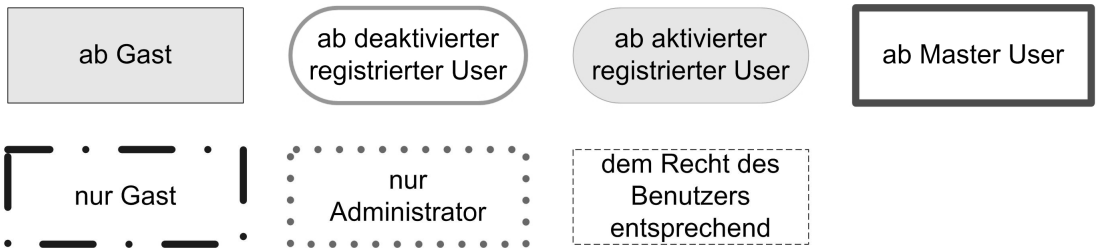
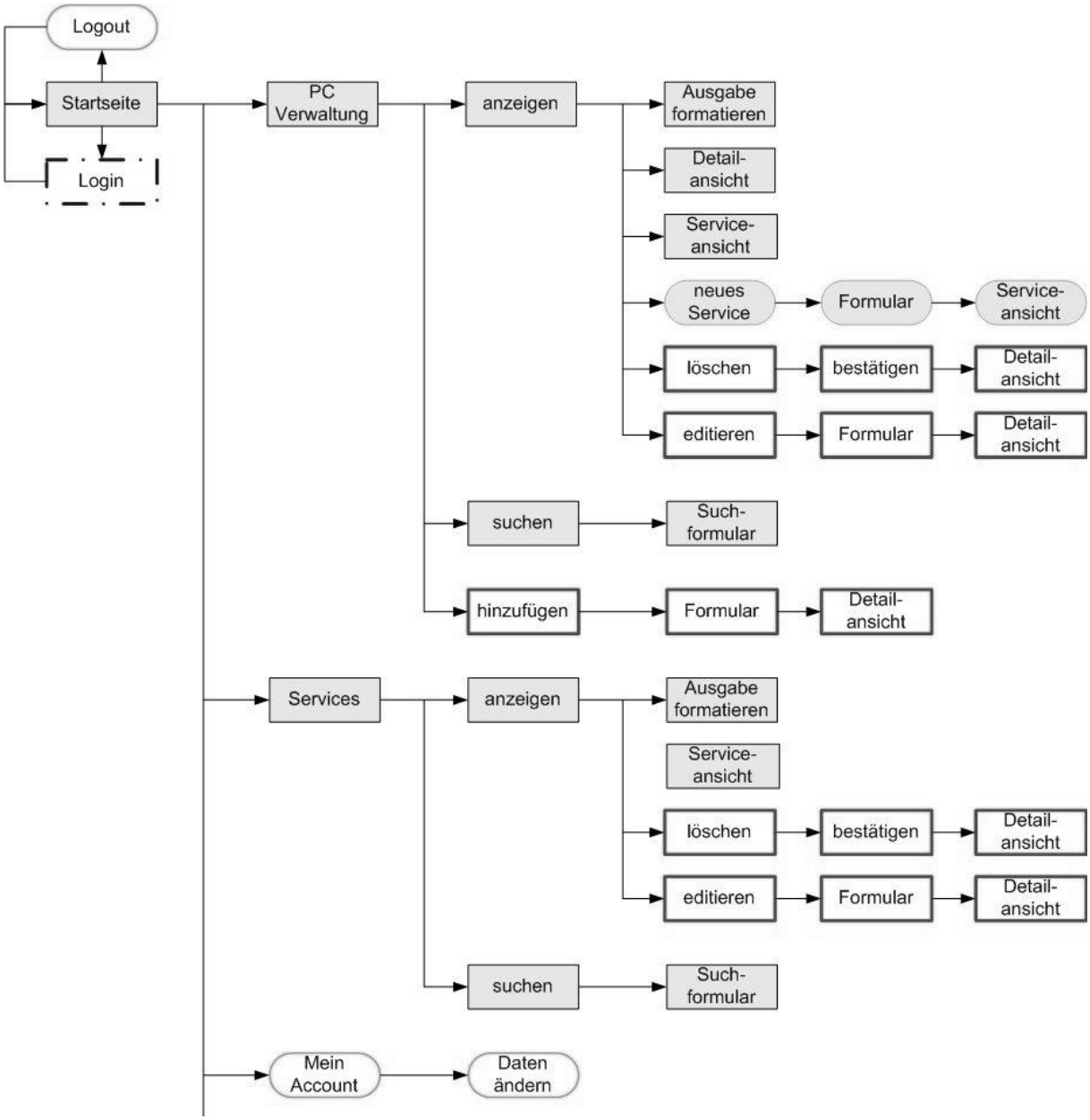


Abbildung 8.11 Seitenaufbau – Legende



Fortsetzung auf folgender Seite →

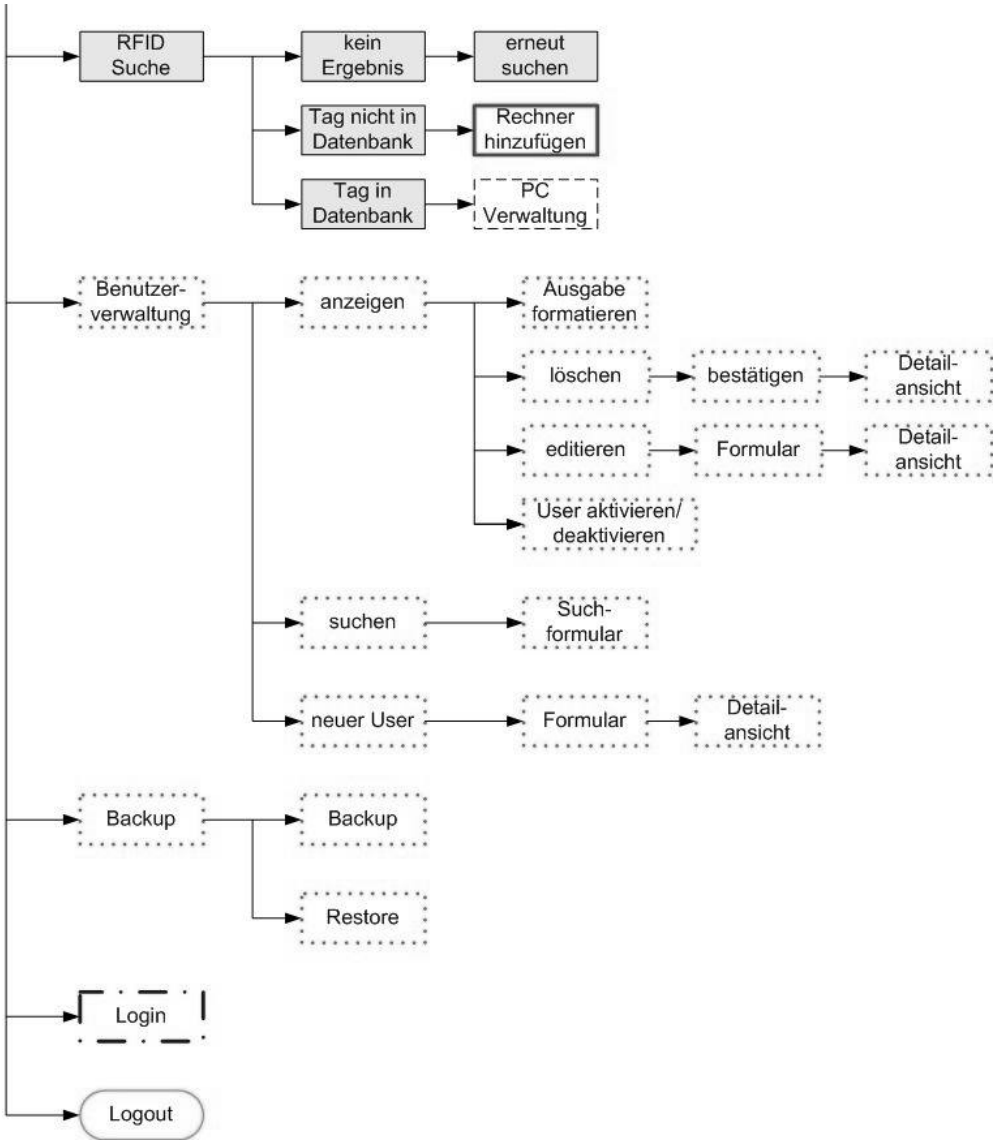


Abbildung 8.12 Seitenaufbau - Dialogstruktur

Abbildung 8.12 zeigt die Struktur der Oberfläche. Aus der Legende (Abbildung 8.11) sind die unterschiedlichen Rechtekategorien ersichtlich. Abbildung 8.12 gibt Aufschluss über den Umfang des dynamischen Menüs.

Beispiel:

Dem Administrator stehen wie auch aus Abbildung 8.10 erkennbar die Menüpunkte Startseite, PC Verwaltung, Services, Mein Account, RFID Suche, Backup, und Logout zur Verfügung, während der Gast nur Zugriff auf Startseite, PC Inventar, Services, RFID Suche und Login hat.

In den entsprechenden Unterteilungen der Hauptmenüpunkte, sind weitere Unterscheidungen anhand der Rechte-Kategorie des Benutzers in Bezug auf den Umfang der nutzbaren Funktionen getroffen. So kann etwa ein Master User im Menüpunkt PC Inventar, Einträge hinzufügen, wohingegen ein Gast nur Lese-Rechte besitzt.

8.3.4 Ordner-, Verzeichnisstruktur

Zur Verwirklichung der Webanwendung wurde die Ordnerstruktur aus Abbildung 8.13 im Root-Verzeichnis des Webservers (htdocs) angelegt.

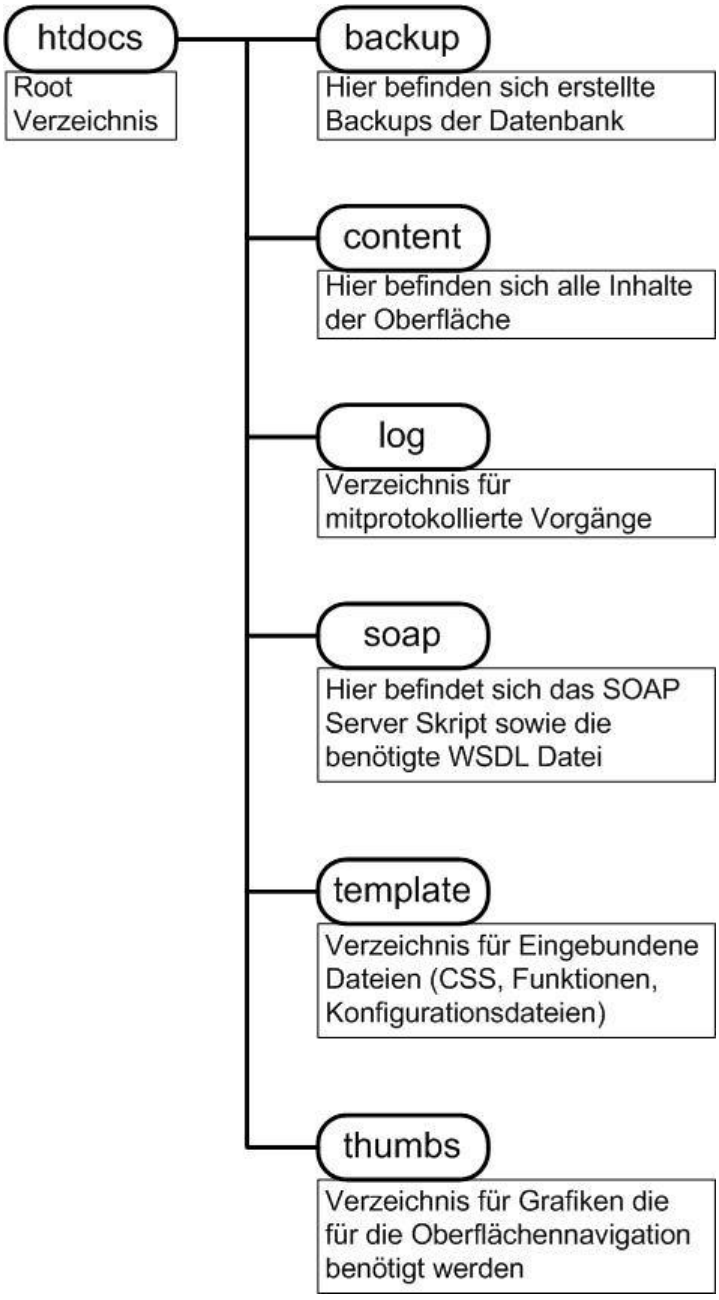


Abbildung 8.13 Verzeichnisstruktur

Zusätzlich befindet sich das Verzeichnis XAMPP im Root-Ordner. Es enthält die Startseite von XAMPP. Das Verzeichnis und die darin gespeicherte Oberfläche von XAMPP wird nicht zwingend benötigt, kann jedoch hilfreich sein, da sie Verweise zu diversen Manuals, bzw. Informationen über den aktuellen Sicherheitsstatus der Entwicklungsumgebung zur Verfügung stellt.

8.4 Grundsystem

8.4.1 Allgemein

Das ganze System wurde so geschrieben, dass der erzeugte HTML – Code alle Regeln des XHTML 1.0 Standards erfüllt. Dies beinhaltet auch die Formatierung mit Cascading Style Sheets (CSS).

Auf den folgenden Seiten werden die Skripte des Grundsystems näher Beschrieben. Zur Erklärung werden jedoch nur Quellcode-Ausschnitte verwendet. Die Quellcodes der einzelnen Skripte befinden sich auf der Projekt CD.

8.4.2 index.php

Das wichtigste Element des Grundsystems ist die Index Seite (index.php) im Root-Verzeichnis. Je nach angeforderter Seite werden ein Template und der eigentliche Inhalt geladen. Das heißt bei jedem Seitenaufruf wird zuerst die index.php aufgerufen.

Beschreibung der Index Datei

In der index.php werden zunächst die beiden Skripte config.php und functions.php mit include_once() eingebunden. Die beiden Skripte stellen der Webanwendung Funktionen zur Verfügung.
Für nähere Informationen siehe Kapitel 8.4.3 und 8.4.4.

Nach dem Start einer Session mit erfolgt der Verbindungsaufbau zum Datenbank Server und schließlich die Auswahl der Datenbank pcinventar.

Im nächsten Schritt wird versucht das Skript mit dem Namen $content.php einzubinden. Beim ersten Aufruf der Index Seite ist die $content Variable noch nicht gesetzt. Folglich existiert auch keine Datei $content.php. Wegen der file_exits() Funktion wird deshalb das main.php Skript eingebunden, welches der Startseite entspricht.

Weiters werden noch die Skripte menue.php (Kapitel 8.4.5) und html_head.php (Kapitel 8.4.6) eingebunden.

Anschließend wird die in menue.php generierte variable $menu_var ausgegeben. Sie repräsentiert das dynamische Menü.
Der darauf folgende Abschnitt dient der Anzeige des Seiten-Titels der ebenfalls in menue.php erzeugt wird. Wenn man in das System eingeloggt ist wird außerdem der Nickname in der ersten Zeile unterhalb des Menüs rechts neben dem Seiten-Titel angezeigt.

Die Gestaltung des Aktionsbereiches wird durch die Ausgabe der Variable $content_var vorgenommen. Sie ist eine der wichtigsten Variablen des ganzen Systems und sie entspricht dem gewählten Seiteninhalt. Sie wird von allen Skripten im Verzeichnis content\ gesetzt.

Abschließend erfolgt noch die Ausgabe des HTML Footer's, der das HTML Grundgerüst abschließt, und die Beendigung der Datenbankverbindung (mysql_close()) sowie der Session (session_destroy()).

8.4.3 config.php

In dieser Datei werden die wichtigsten Pfade ausgehend vom Root-Verzeichnis als Funktionen definiert. Dies hat den Sinn, das bei einer eventuellen Umbenennung eines Unterverzeichnisses nur die config.php aktualisiert werden muss, sofern die angegebenen Funktionsnamen auf den weiteren Seiten verwendet werden. Die Datei befindet sich im Verzeichnis template

```
function domain () {                //gibt die Servernamen zurück
       $str=$_SERVER["SERVER_NAME"];
       return "http://$str/";
}

function content_dir () {           //gibt den rel. Pfad des content Verz. zurück
       return "content/";
}

function template_dir () {          //gibt den rel. Pfad des template Verz. zurück
       return "template/";
}
```

8.4.4 functions.php

In functions.php im Verzeichnis template sind selbst definierte Funktionen gespeichert. Die jeweiligen Funktionen sollen nicht nur Schreibarbeit ersparen, sie sollen auch für die notwendige Übersichtlichkeit sorgen.
Hier eine kurze Erklärung der erstellten Funktionen.

sql_error( )

- wird bei Zugriffsfehlern auf Tabellen verwendet
- gibt den SQL Code und anschließend den MySQL Error Text aus

```
function sql_error ($sql, $db) {
  return "Fehler!<br />$sql<br />" . mysql_error ($db);
}
```

rechtekat()

- wird zum Eruieren der Rechtekategoriebezeichnung aus der RechteID verwendet
- übergeben wird die RechteID
- die RechteID ist eine Nummer die Rechtekategorie verkörpert
- Beispiel:
  RechteID von Gast = 0: Der Wert 0 wird der Funktion übergeben und im Array an der Stelle 0 steht die Bezeichnung Gast → „Gast“ wird zurückgeliefert

```
function rechtekat ($r) {
  $str = array ("Gast", "Registrierter User", "Master User" , "Administrator",
                "Registrierter User (aktiviert)");
  return $str[$r];
}
```

Rechte()

- Diese Funktion hat die selbe Aufgabe wie rechtekat(), mit dem Unterschied, dass keine für die Ausgabe formatierten Werte, sondern für den Anwender nicht sichtbare Kurzbezeichnungen zurückgegeben werden.
- Verwendet wird sie in UserRight() und damit praktisch in fast allen Skripten des Projektes.

```
function Rechte ($r) {
  $str = array ("guest", "reg", "master" , "admin", "reg_aktiv");
  return $str[$r];
}
```

UserRight()

- ermittelt die RechteID eines Users aus der adapters Tabelle anhand der SessionID
- Sofern kein User eingeloggt ist wird der String „guest“ zurückgeliefert, was einem Gast User entspricht.

```
function UserRight ($db) {
  if ($_SESSION ['userid']) {
    $sql="SELECT rechte FROM adapters WHERE userid=" . $_SESSION['userid'];
    $res=mysql_query ($sql);
    if ($res) {
      $row=mysql_fetch_array ($res);
      return Rechte($row['rechte']);
    }
    else {
      $content_var = sql_error ($sql, $db);
    }
  }
  else{
    return Rechte (0);
  }
}
```

xxxx_image()

- Diese Funktion bindet ein Bild mit dem Namen xxxx ein.
- Sie wird für alle Navigationssymbole verwendet.

```
function xxxx_image () {
    return "<img class=\"thumb\" src=\"". domain() . "thumbs/xxxx.gif\"
              title=\"Ändern\" alt=\"ändern\"/>";
}
```

datfor()

- formatiert einen MySQL Standard-Datumsstring wie z.B. 2006-03-09 in die Form 09.03.2006 um
- verwendet für die formatierte Ausgabe eines Datums

```
function datfor ($dat){
    $str=explode(' ', $dat);
    $datestr=explode('-', $str[0]);
    $timestr=$str[1];
    $datefor=$datestr[2].".".$datestr[1].".".$datestr[0]." $timestr";
    return $datefor;
}
```

8.4.5 menue.php

In menue.php wird das Navigationsmenü in die Variable $menue_var geschrieben, die später in index.php ausgegeben wird.
Dafür werden 2 Dimensionale (2D) Arrays verwendet. Dem Gast, registrierten User, Master User und Administrator werden je ein 2D-Array zugewiesen. Dieses ist z.B. beim Gast wie folgt aufgebaut:

```
$guest=Array (
              Array ("Startseite", "main"),
              Array ("PC Inventar", "g_inventar"),
              Array ("Services", "g_service"),
              Array ("RFID Suche", "rfid")
);
```

Ein 2D Array kann wie eine Tabelle mit zwei Spalten angesehen werden. Also:
$guest:

#	SPALTE1	SPALTE2
0	Startseite	main
1	PC Inventar	g_inventar
2	Services	g_services
3	RFID Suche	rfid

SPALTE1 ist der Titel des Menüpunktes, SPALTE2 der Dateiname des entsprechenden Skriptes und # ist der Index.

Abhängig vom Login Status, der mit der Funktion UserRight() ermittelt werden kann, wird das entsprechende 2D Array in das Array $menue kopiert anschließend noch der Menüpunkt Login bzw. Logout mit array_merge() angehängt.

Mit diesem Code Abschnitt werden in weiterer Folge Links aus dem 2D Array $menue generiert:

```
$menue_var="<div class=\"menue\">";
for ($i=0; $i<sizeof($menue); $i++) {
       $item  = $menue[$i];              //Zeile des 2D Arrays in 1D Array schreiben
       $class = "menue_item";            //Klasse für inaktiven Menüpunkt
       if ( $content == $item[1] ) {     //Aktuellen Menüpunkt hervorheben
       $class = "menue_active";          //Klasse für aktiven Menüpunkt
       $title = $item[0];                //Seitentitel entspricht item[0]
    }
       $menue_var .= "<div class=\"$class\">
       <a class=\"menue\" href=\"?content=" . $item[1] . "\">" . $item[0] ."</a>
       </div><div class=\"menue_spacer\"></div>";
}

$menue_var .= "\n<div class=\"menue_spacer2\">
<a style=\"width: 100%; margin: 0px;\"></a></div>";
$menue_var .= "\n</div>";
```

Die Tags dienen der Formatierung des Menüs.

Wird die Variable $menue_var zum Beispiel mit der Menüstruktur des „Gastes“ beschrieben, entsteht bei Ausgabe der Variable mit „echo“ in der index.php dieses Menü:

Abbildung 8.14 Menü Gast

Bei einem Klick auf „Startseite“ wird als „content“ „main“ übergeben und in index.php wird darauf das Skript main.php eingebunden.

8.4.6 html_head.php

Der HTML Kopf Bereich dient einerseits der Generierung des Seitentitels mit

```
<title>Inventarisierung mit RFID - <?php echo $title;?></title>
```

wobei $title dem aktiven Menüpunkt entspricht (siehe 8.4.5 menue.php), andererseits wird im Kopfbereich auch die CSS Datei eingebunden.

Da die zur Zeit am meisten verbreiteten Browser (Mozilla Firefox, Internetexplorer) bestimmte HTML oder CSS Elemente unterschiedlich interpretieren, wurden zwei Style Sheets angelegt. In Abhängigkeit vom verwendeten Browsertyp der mittels $_SERVER["HTTP_USER_AGENT"] ermittelt wird, wird der entsprechende Style Sheet eingebunden.

8.5 Benutzerverwaltung

8.5.1 Allgemein

Um zu garantieren, dass Wartungen bzw. Servicetätigkeiten an den Rechnern nachvollziehbar festgehalten werden können ist eine Benutzerverwaltung notwendig.

Dazu stellt das System ein Login-System bereit, das auf PHP-Sessions basiert.
Die Rechte des Users werden von jedem Inhaltselement, das in die index.php geladen wird selbstständig überprüft. Wenn der User nicht genügend Rechte besitzt wird er auf sie Startseite verwiesen. Diese Seite erfordert z.B. Administrator – Rechte:

```
if (UserRight ($db)!= "admin") {
  header("Location: " . domain () . "?content=main");
  exit ();
}
```

Standardmäßig, dass heißt wenn kein User angemeldet ist wird der User „Gast“ angenommen der die wenigsten Rechte besitzt. Der Gast hat in weiterer Folge die Möglichkeit sich zu registrieren (siehe Kapitel 8.5.4).
Das Login-System ist durch die beiden Skripte login.php und logout.php realisiert.

8.5.2 login.php

Die Möglichkeit sich anzumelden steht natürlich nur einem Gast User zur Verfügung.

Beim Aufruf des Inhaltselement „login“ über das Navigationsmenü, wird ein Formular über die Inhaltsvariable $content_var ausgegeben, das zum Eintragen des Benutzernamen und des Passwortes auffordert. Die Groß- bzw. Kleinschreibung des Namens ist egal.

Abbildung 8.15 Login

Beim Absenden des Formulars mit dem Button „Login“ wird die Variable $login gesetzt und das login.php Skript erneut aufgerufen.
Da $login nun einen Wert besitzt wird der eindeutige Benutzername aus dem Feld „Nick“ mit der Spalte „nick“ aus der Tabelle „adapters“ verglichen.Ist der eingegebene Name nicht vorhanden erscheint die Fehlermeldung Nickname wurde nicht gefunden! über dem Formular.
Sofern der Nickname existiert, wird das Passwort überprüft. Stimmt es nicht mit dem Datenbankeintrag überein wird Passwort falsch! ebenfalls über dem Formular ausgegeben.

Wenn die Login Informationen korrekt sind wird die UserID aus der Tabelle „adapters“ in der Session-Variable userid, die mit `$_SESSION['userid']` abgefragt werden kann, registriert (session_register()).
Zum Abschluss des erfolgreichen Anmeldevorgangs erfolgt eine Weiterleitung zur Startseite.

Die Login-Möglichkeit steht dem Gast User auch auf der Startseite main.php zur Verfügung.

8.5.3 logout.php

logout.php steht nur für Angemeldete Benutzer zur Verfügung.
In logout.php wird die Registrierung der Session-Variable `$_SESSION['userid']` `session_unregister()` aufgehoben wenn der Button „Logout“ betätigt wird. Das heißt die aktuelle Sitzung wird beendet. Es erfolgt außerdem eine Weiterleitung zur Startseite. Der Benutzer des Systems ist von nun an wieder mit „Gast“ Rechten unterwegs.

Abbildung 8.16 Logout

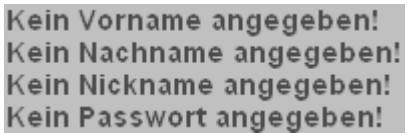
Die Logout-Möglichkeit steht dem angemeldeten User auch auf der Startseite main.php zur Verfügung.

8.5.4 registierung.php

Wie aus Abbildung 8.15 Markierung 1 ersichtlich ist, steht einem Benutzer der sich als Gast auf der Oberfläche bewegt auch die Registrierung zur Verfügung.
Mit Hilfe der Registrierung werden die im Formular angegebenen Account-Daten in die Tabelle „adapters“ eingetragen sofern diese gültig sind.

Abbildung 8.17 Registrierungsformular

Abbildung 8.17 zeigt das Registrierungsformular. Alle Felder werden für einen gültigen Eintrag benötigt. Fehlen Angaben erscheint eine entsprechende Fehlermeldung:



Ist der gewählte Nickname schon vorhanden, wird z.B.

angezeigt.

Nach erfolgreicher Registrierung wird eine Zusammenfassung der Account-Daten ausgegeben.
Der angelegte User kann sich nun mit allen Rechten der Kategorie „registrierter User“ auf der Oberfläche bewegen.
Alle User Gruppen die höhere Rechte besitzen können nur von einem Administrator in die „adapters“ Tabelle eingetragen werden.

8.6 Gast User

8.6.1 Allgemein

Um sich als Gast auf der Oberfläche zu bewegen ist kein Login notwendig. Für jeden Benutzer des Systems stehen folgende Funktionen zur Verfügung:

- Anzeige aller eingetragenen PCs, mit allen möglichen Informationen die sich aus der Tabellenstruktur ergeben, in Form einer Tabelle. Zusätzlich wird in einer generierten Spalte die Anzahl der zu einem PC gehörenden Serviceeinträge angezeigt.
- Bei einem Mausklick auf den gewünschten Rechnereintrag werden die dazugehörenden Serviceeinträge mit allen Inhalten aus der Datenbank angezeigt.
- Die Ausgabe der Einträge kann angepasst werden. (alphabetisches Sortieren,…)
- Suche nach bestimmten Kriterien (PC Name, Raum Name,…) und Anzeige der gefundenen Elemente.
- Anzeige aller eingetragenen Services mit den Informationen aus der Tabellenstruktur in Form einer Tabelle.
- Suche nach Rechnern unter Verwendung des RFID Lesegeräts.
- Registrierungsmöglichkeit auf der Startseite

8.6.2 Startseite (main.php)

main.php ist die Startseite der Webanwendung. Beim aufruf der URL des Servers wird sie angezeigt. In Abbildung 8.18, Markierung 1 ist der in Kapitel 8.5.2 beschriebene Login Bereich zu erkennen.
Darunter (Markierung 2) befindet sich eine Tabelle die die beiden zuletzt hinzugefügten Service Dokumentationen anzeigt.
Die Aktionsbuttons in Markierung 3 verweisen zur Detailansicht der jeweiligen Rechner (Kapitel 8.6.3)

Abbildung 8.18 Startseite

8.6.3 PC Inventar (g_inventar.php)

In PC Inventar erfolgt eine Auflistung aller vorhandenen Rechner in Form einer Tabelle.

Über die Buttons in Abbildung 8.19, Markierung 1 kann die Ausgabe der inventarisierten Rechner angepasst werden:

- Mit „anzeigen“ werden alle eingetragenen Rechner angezeigt.
- Mit „suchen“ kann nach allen Rechnermerkmalen, die auch aus der Kopfzeile der Tabelle ersichtlich sind, gesucht werden. Dazu wird ein Formular benutzt.

Beim Klick auf eines der beiden Elemente in Markierung 1 wird die index.php erneut aufgerufen. Wird z.B. „suchen“ ausgewählt werden der Index Datei die Parameter
?content=g_inventar&action=suche übergeben. Mit einer switch() Kontrollstruktur wird in Abhängigkeit der Variable $action die entsprechende Funktion ausgeführt. In diesem Beispiel die Suchfunktion.

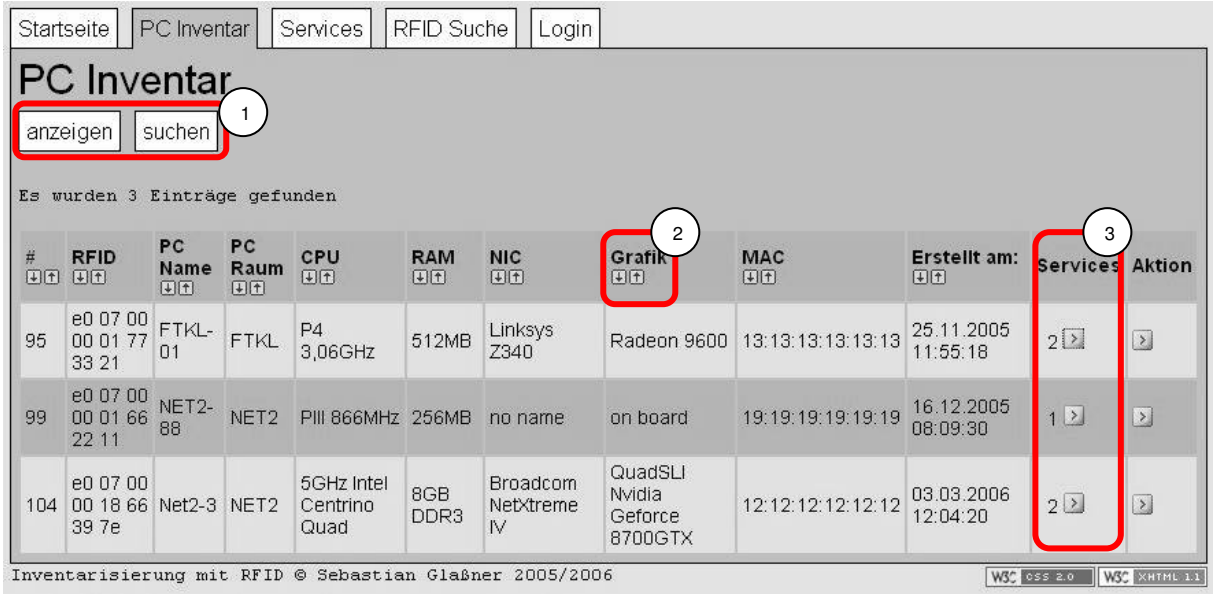


Abbildung 8.19 PC Inventar

Markierung 2 zeigt Aktionsbuttons mit denen die Rechner in der Tabelle auf- bzw. absteigend nach der ausgewählten Spalte sortiert werden können.

Markierung 3: In dieser Spalte wird die Anzahl der vorhandenen Servicetätigkeiten des Rechners angezeigt. Der Aktionsbutton verweist auf die Serviceansicht (Abbildung 8.20).

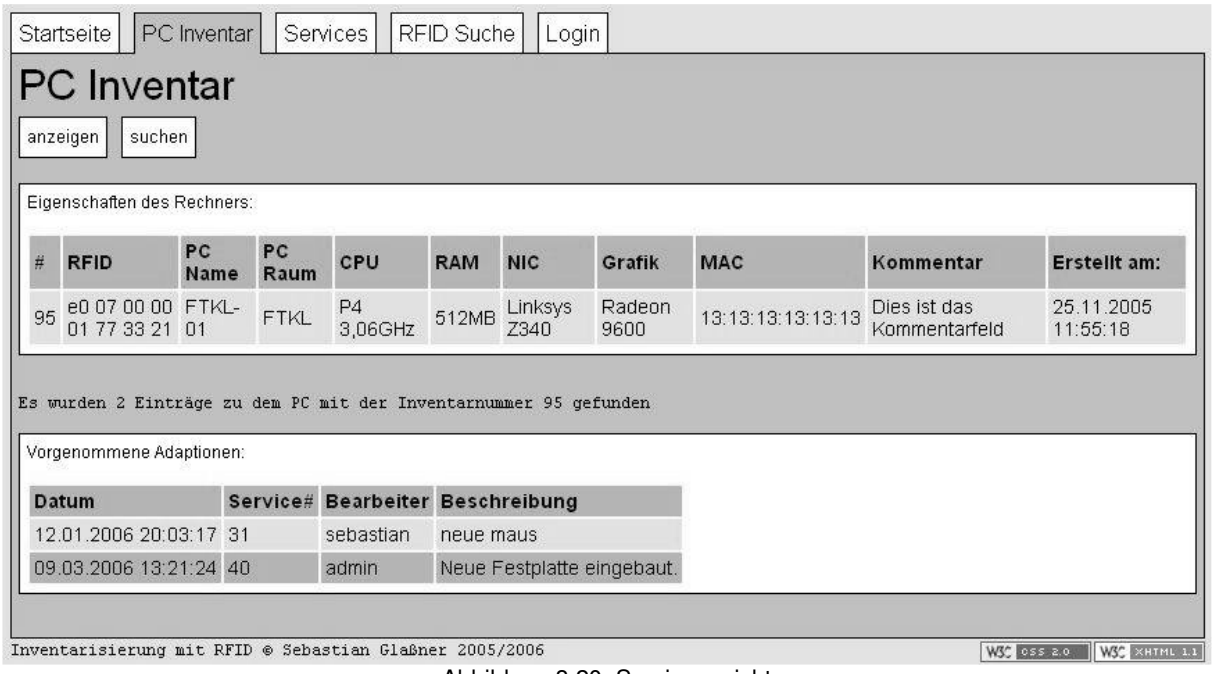


Abbildung 8.20 Serviceansicht

In dieser Ansicht sind zunächst die Eigenschaften des Rechners angeführt. Darunter werden die an dem PC vorgenommenen Wartungen nach dem Bearbeitungsdatum geordnet gezeigt.

8.6.4 Services (g_service.php)

Der Menüpunkt „Services“ ist ähnlich dem Skript g_inventar.php aufgebaut. Mit „anzeigen“ wird die in Abbildung 8.21 dargestellte Ausgabe erzeugt, während mit Suchen das Suchformular (Abbildung 8.22) anzeigt. Die Aktionsbuttons verweisen wieder auf die Serviceansicht.

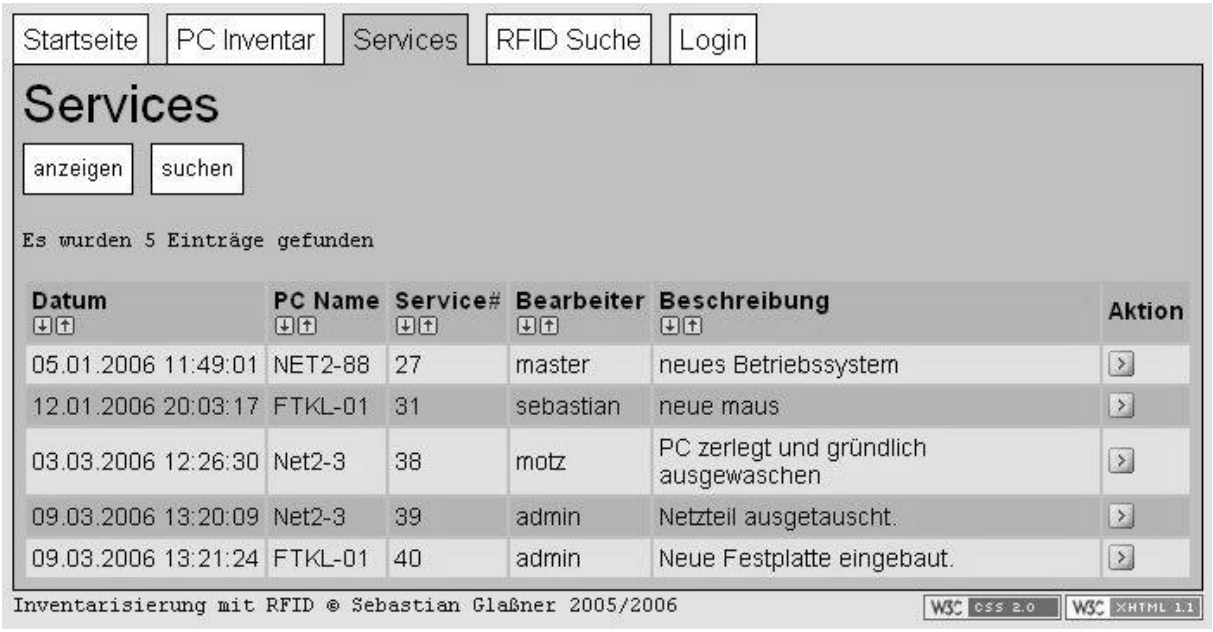


Abbildung 8.21 Services

Abbildung 8.22 Service - Suchformular

Beim Absenden des Formulares mit „GO“ wird der Eintrag „admin“ der Ausgabefunktion in g_service.php übergeben, die mit Hilfe von regulären Ausdrücken in der Datenbank nach passenden Einträgen sucht.

Als Ergebnis würde diese Suche alle Wartungseinträge ausgeben deren Bearbeiter der User mit dem Nick „admin“ ist.

8.6.5 RFID Suche (rfid.php)

Eine von der Transportschicht erzeugte SOAP Anfrage sendet die aus einem Tag ausgelesenen Daten an das SOAP Server Skript. Dieses Skript trägt die erhaltenen Informationen in die Tabelle „rfidsearch“ ein (Die genauen Aufgaben und Funktionen des SOAP Server Skriptes sind im Kapitel 8.10 näher erläutert). Zu den übermittelten Daten zählt auch die IP Adresse des Clients, die somit auch in der Tabelle gespeichert wird.

Ruft der Benutzer am Client über einen Browser den Menüpunkt „RFID Suche“ auf, bzw. betätigt er die Schaltfläche „erneut Suchen“ (Abbildung 8.23, Markierung 1), werden die Daten des soeben eingelesenen Transponders angezeigt.

Um zu ermöglichen, dass die RFID Suche von mehreren Clients aus möglich ist wird zunächst die IP Adresse des Clients mit $_SERVER(„REMOTE_ADDR“) im rfid.php Skript ermittelt.
Anschließend wird eine Datenbankabfrage gestartet, die den zuletzt hinzugefügten und noch nicht abgefragten Eintrag aus der Tabelle „rfidsearch“ sucht, bei dem die in rfid.php ermittelte IP Adresse mit der bei der SOAP Übertragung im Server-Skript ermittelten IP übereinstimmt.

Die Abfrage der Suchergebnisse mittels Webanwendung kann daher nur vom Rechner ausgeführt werden, an dem der Transponder eingelesen wurde.

Verläuft die Datenabfrage aus der Tabelle „rfidsearch“ erfolgreich wird in der Tabelle „main“ nach Einträgen gesucht die die Unique ID Nummer des eingelesenen Transponders in der Spalte „rfid“ aufweisen. Sofern der Transponder und somit der Rechner in der Datenbank gefunden wird erscheint folgende Ausgabe:

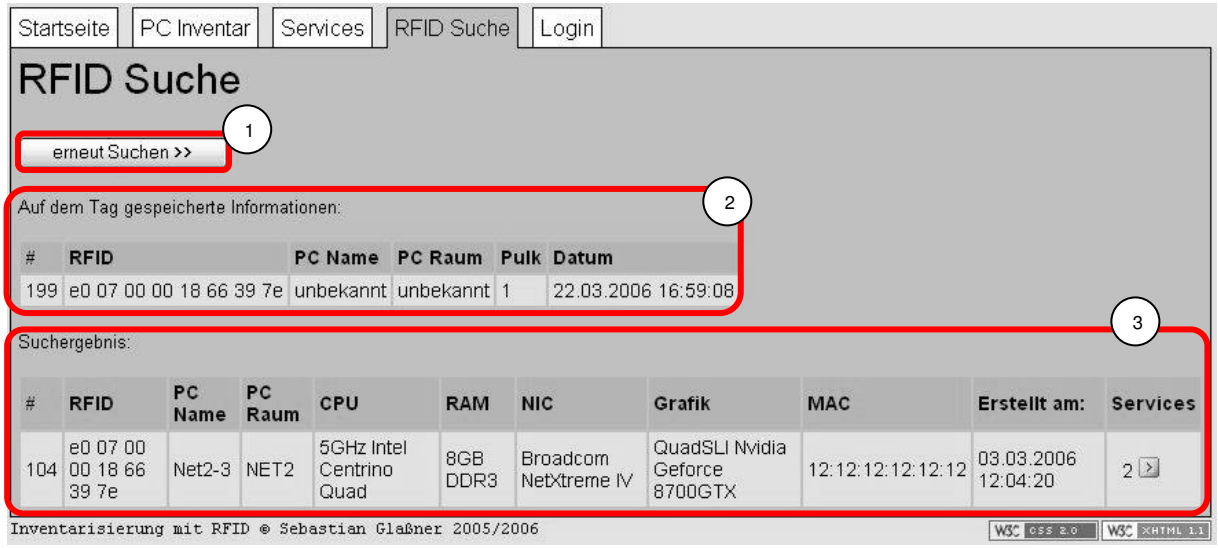


Abbildung 8.23 Erfolgreiche RFID Suche

In Markierung 2 werden die auf dem Transponder gespeicherten Daten angezeigt.
Markierung 3 zeigt den mit der Unique ID des Transponders verknüpften Datenbankeintrag an.

Befindet sich der Rechner nicht in der Datenbank erscheint statt des Inhalts von Markierung 3 diese Fehlermeldung:

Nachdem die Abfrage des eingelesenen Transponders aus der Tabelle „rfidsearch“ mit Erfolg durchgeführt wurde, wird der Spalteneintrag „sstatus“ auf „inaktiv“ gesetzt.
Das heißt wird die Anweisung „erneut Suchen“ abermals ausgeführt, erscheint eine Ausgabe wie sie in Abbildung 8.24 zu sehen ist.

Wird der Menüpunkt „RFID Suche“ aufgerufen ohne dass zuvor ein Transponder eingelesen wurde, wird dieser Seiteninhalt angezeigt.

Abbildung 8.24 Erfolglose RFID Suche

8.7 Registrierter User

8.7.1 Allgemein

Um sich als registrierter User auf der Weboberfläche zu bewegen ist zunächst eine Registrierung und anschließend ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines Gasts. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:

- Einsehen und Bearbeiten der eigenen Accountdaten.
- Es besteht die Möglichkeit sich abzumelden.
- Er kann von einem Benutzer mit höherer Rechtekategorie aktiviert werden.

Durch die Aktivierung eines registrierten Users durch einen Administrator (siehe Administrator-Benutzerverwaltung) erhält dieser die Möglichkeit zu jedem PC explizit ein Serviceeintrag hinzuzufügen.

8.7.2 Startseite (main.php)

Die Startseite des registrierten Users entspricht der des Gast Users (es wird auch das gleiche Skript verwendet), mit dem Unterschied, dass über die Bedingung

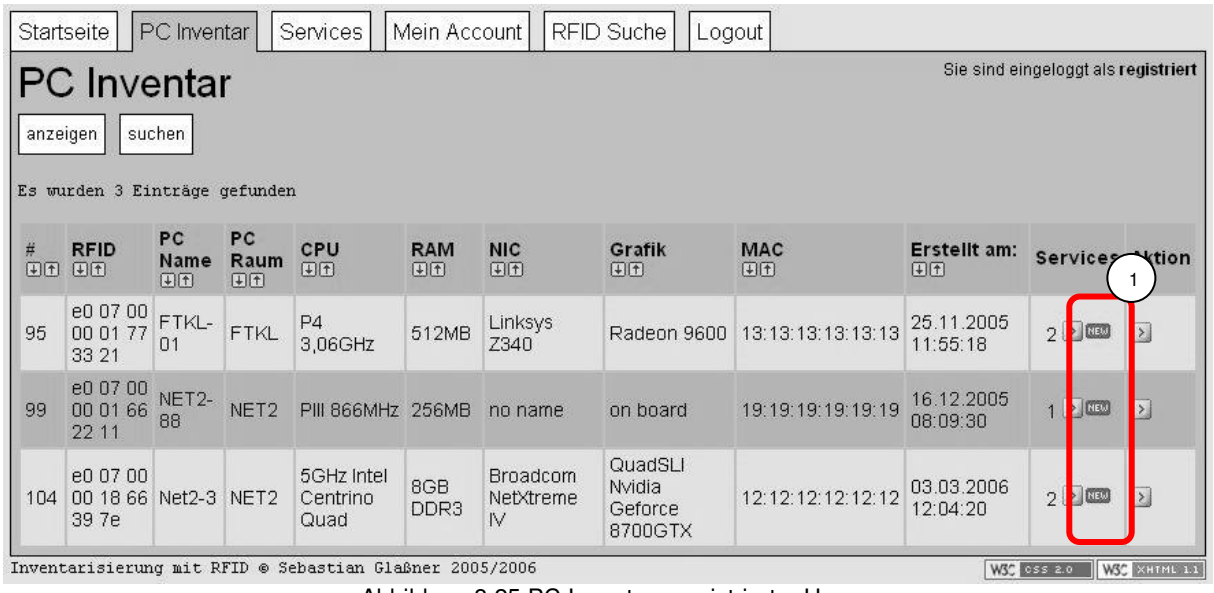
```
if (!$_SESSION['userid']) {
	//Login Formular
}
else{
	//Logout Formular
}
```

anstelle des Login Formulars das Logout Formular ausgegeben wird.

8.7.3 PC Inventar (r_inventar.php)

Vom Aufbau her entspricht dieses Skript ebenfalls dem des Gastes (Kapitel 8.6.3). Für einen deaktivierten registrierten User weist es auch keine zusätzlichen Funktionen auf.

Ist der User hingegen aktiviert, steht im die Möglichkeit Servicevorgänge an Schulrechnern zu dokumentieren zur Verfügung.
Dazu sollte z.B zuerst der gewartete Rechner gesucht werden um ihm anschließend über den Button „new“ (Abbildung 8.25, Markierung 1) einen neuen Serviceeintrag zuzuweisen.



#	RFID	PC Name	PC Raum	CPU	RAM	NIC	Grafik	MAC	Erstellt am:	Services	Aktion
95	e0 07 00 00 01 77 33 21	FTKL-01	FTKL	P4 3,06GHz	512MB	Linksys Z340	Radeon 9600	13:13:13:13:13:13	25.11.2005 11:55:18	2	
99	e0 07 00 00 01 66 22 11	NET2-88	NET2	PIII 866MHz	256MB	no name	on board	19:19:19:19:19:19	16.12.2005 08:09:30	1	
104	e0 07 00 00 18 66 39 7e	Net2-3	NET2	5GHz Intel Centrino Quad	8GB DDR3	Broadcom NetXtreme IV	QuadSLI Nvidia Geforce 8700GTX	12:12:12:12:12:12	03.03.2006 12:04:20	2	

Abbildung 8.25 PC Inventar - registrierter User

Bei einem Klick auf den Button „new“ öffnet sich ein Formular in welchem der Wartungsvorgang beschrieben werden kann.

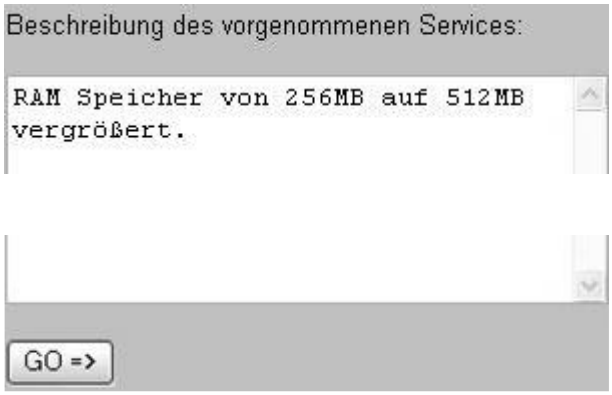


Abbildung 8.26 neues Service

Nach dem Absenden des Formulars wird auf die Serviceansicht gewechselt (siehe Kapitel 8.6.3).

8.7.4 Mein Account (account.php)

Wird über den Menüpunkt „Mein Account“ das Skript account.php eingebunden, werden die Daten des aktuell eingeloggten Benutzers sichtbar.
Über die Schaltfläche „Daten ändern“ kann der User seine persönlichen Angaben ändern.

Abbildung 8.27 Mein Account

8.7.5 RFID Suche (rfid.php)

Siehe Kapitel 8.6.5

8.8 Master User

8.8.1 Allgemein

Um sich als Master User auf der Oberfläche zu bewegen ist ein Login notwendig. Dieser Benutzer besitzt alle Rechte eines aktivierten registrierten Users. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:

- Es können neue Rechner in die Datenbank aufgenommen werden.
- Es können alle Einträge einzeln gelöscht werden. Wird ein PC Eintrag gelöscht werden alle dazugehörigen Serviceeinträge gelöscht.
- Es können alle Einträge bearbeitet werden.

8.8.2 Startseite (main.php)

Siehe Kapitel 8.7.2 Startseite des registrierten Users.

8.8.3 PC Verwaltung (m_verwaltung.php)

Das Skript m_verwaltung.php dient der Administration der Schulrechner. Der Aufbau entspricht im Wesentlichen dem Skript r_inventar.php. Jedoch stehen dem Master User einige zusätzliche Funktionen zur Verfügung.

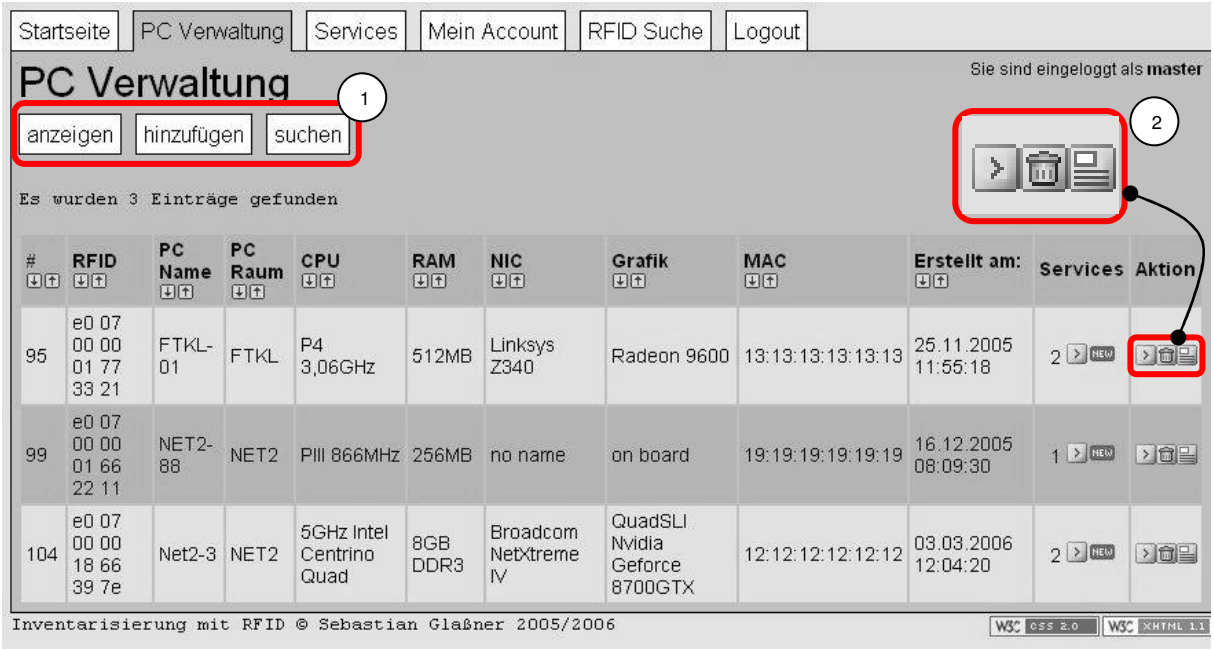


Abbildung 8.28 PC Verwaltung – Master User

In Abbildung 8.28, Markierung 1 ist die Schaltfläche „hinzufügen“ zu erkennen. Diese Funktion ermöglicht es neue Rechner in die Datenbank aufzunehmen.

Abbildung 8.29 zeigt das Formular mit den möglichen Eingaben. Die mit * markierten Felder sind für einen neuen Eintrag gefordert.
Das Drop Down Menü des Feldes „PC Raum“ listet alle in der Tabelle „locations“ vorhandenen Räume auf. Über den Link „Räume verwalten…“ (Abbildung 8.29, Markierung 1) wird das Skript raumverwaltung.php aufgerufen.

In der Raumverwaltung können neue PC Räume hinzugefügt werden bzw. vorhandene PC Räume entfernt werden. Es können jedoch nur jene Räume aus der Datenbank Tabelle „locations“ gelöscht werden, denen keine Rechner zugewiesen sind. Um sie trotzdem löschen zu können müssen alle Rechner aus der Tabelle „main“ entfernt werden, die sich auf den gewünschten Computer Raum beziehen.

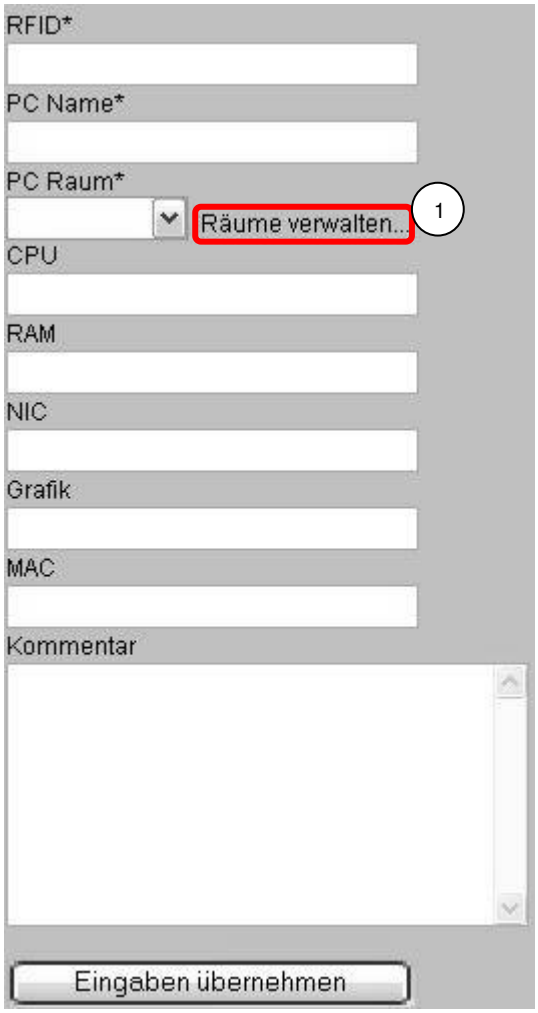


Abbildung 8.29 Rechner hinzufügen

Abbildung 8.30 Raumverwaltung

In Abbildung 8.28, Markierung 2 sind die Aktionsbuttons vergrößert dargestellt.

Mit wird die Detailansicht des Rechners geöffnet. Sie zeigt zusätzlich zu den Angaben in der Tabelle das Kommentarfeld an.

Inventar Nummer	95
RFID	e0 07 00 00 01 77 33 21
PC Name	FTKL-01
PC Raum	FTKL
CPU	P4 3,06GHz
RAM	512MB
NIC	Linksys Z340
Grafik	Radeon 9600
MAC	13:13:13:13:13:13
Kommentar	Dies ist das Kommentarfeld
Aufnahmedatum	25.11.2005 11:55:18

Abbildung 8.31 Detailansicht

Mit kann ein Rechner editiert werden. Hierzu wird das in Abbildung 8.29 abgebildete Formular mit den vorhandenen Spalteneinträgen des Rechners angezeigt.

Mit dem Button kann der ausgewählte Rechner aus der Tabelle „main“ gelöscht werden. Um unabsichtliches Löschen zu verhindern, erscheint bei Klick auf den „löschen“ Button diese Bestätigungsmaske, welche die Detailansicht des zu entfernenden Rechners enthält:

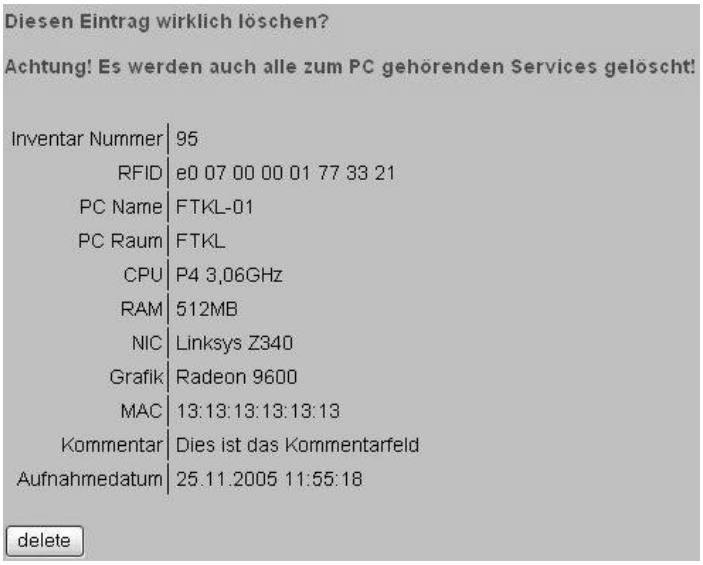


Abbildung 8.32 Löschbestätigung

Der Master User kann auch wie ein aktivierter registrierter User neue Serviceeinträge erstellen (Abbildung 8.28). In der Spalte Services ist hierzu der Button NEW vorhanden.
Die Schaltfläche dient zum Umschalten auf die Serviceansicht (Abbildung 8.20).

8.8.4 Services (m_service.php)

Die grundsätzliche Aufbau dieses Skriptes entspricht dem von g_service.php (Kapitel 8.6.4).
Der Master User hat zusätzlich wie im Menüpunkt „PC Verwaltung“ die Möglichkeit Services zu dokumentieren, diese zu bearbeiten und löschen.

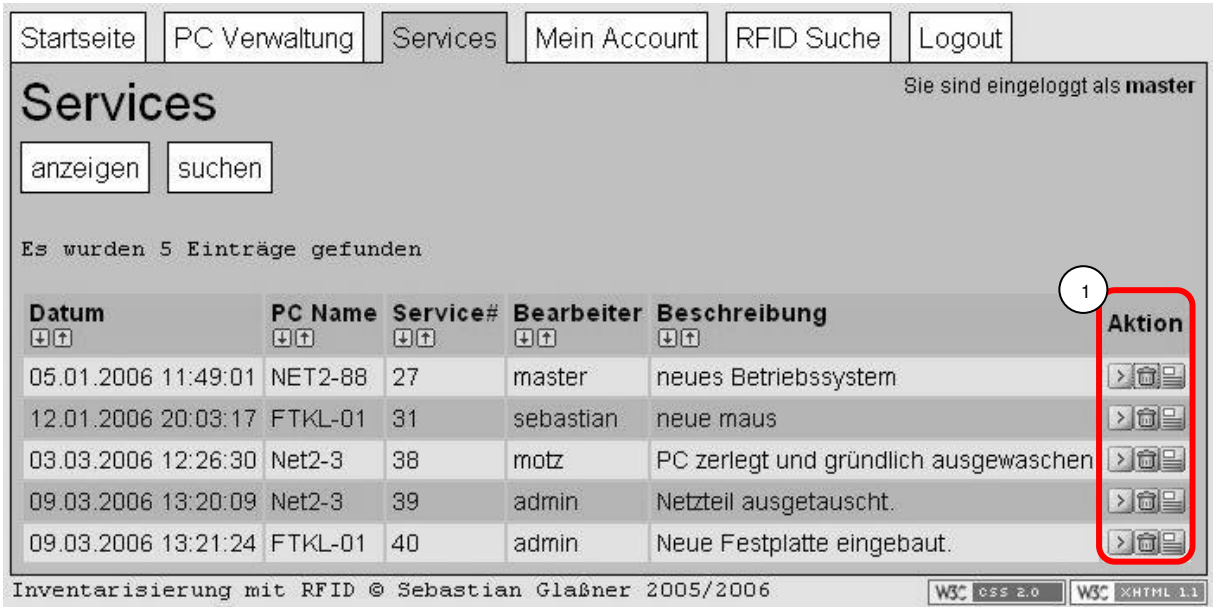


Abbildung 8.33 Services – Master User

Abbildung 8.33, Markierung 1:

→ Serviceansicht (siehe Abbildung 8.20)

→ Eintrag löschen (Abbildung 8.34)
Bevor ein der Löschvorgang durchgeführt wird, muss er in der Detailansicht bestätigt werden.

→ Eintrag editieren (Abbildung 8.35)
Die hervorgehobenen Felder können nicht editiert werden. In der Beschreibung des Servicevorgangs können Änderungen vorgenommen werden.

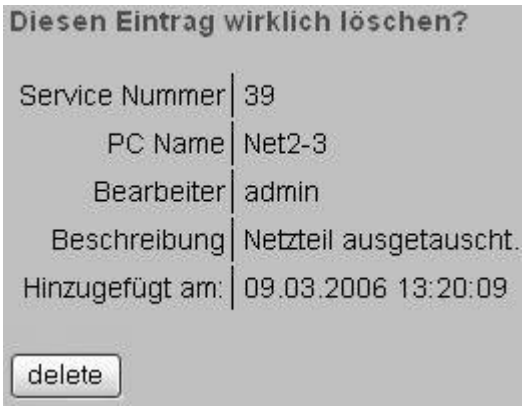


Abbildung 8.34 Service löschen

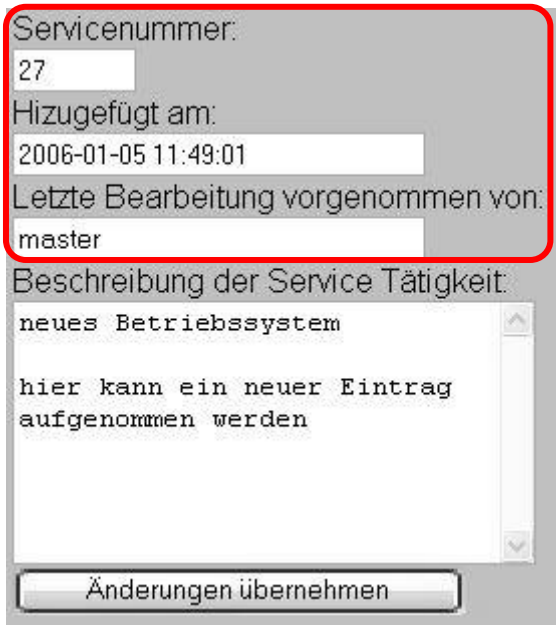


Abbildung 8.35 Service ändern

8.8.5 Mein Account (account.php)

Siehe 8.7.4 Mein Account – registrierter User

8.8.6 RFID Suche (rfid.php)

Es wird das selbe Skript wie beim registrierten User verwendet (siehe Kapitel 8.7.5).
Es stehen auch die gleichen Möglichkeiten zur Verfügung.

Darüber hinaus wird mit Hilfe der selbst definierten Funktion UserRight() ein Skript-Abschnitt für den Master User ergänzt, mit dem folgendes verwirklicht werden kann:

> Wenn eine unbekannte RFID Transponder ID eingelesen wird, das heißt wenn noch kein Datenbankeintrag existiert, der mit dieser ID verknüpft ist, so besteht ab dem Master User die Möglichkeit einen neuen Rechner mit der eingelesenen Nummer in die Datenbank aufzunehmen.
> Dazu wird ein Link eingefügt, der beim Betätigen die ID an das Formular zum Hinzufügen eines Rechners übergibt (Abbildung 8.36, Markierung 1).

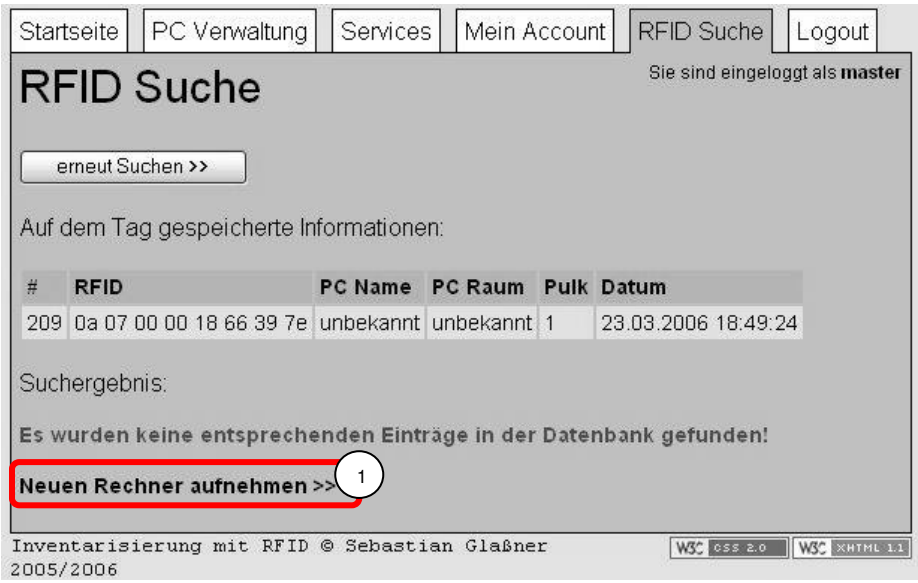


Abbildung 8.36 RFID Suche – Master User

8.9 Administrator

8.9.1 Allgemein

Um sich als Administrator auf der Oberfläche zu bewegen ist wiederum ein Login notwendig. Der Administrator besitzt alle Rechte eines Master Users. Zusätzlich stehen ihm folgende Funktionen zur Verfügung:

- Exportieren der Datenbank zur Sicherung
- Wiederherstellen der Datenbank
- Benutzerverwaltung: Hinzufügen, Entfernen, Bearbeiten von Benutzern
- Aktivieren von registrierten Benutzern um ihnen das Erstellen von Services zu ermöglichen

8.9.2 Startseite (main.php)

Siehe Kapitel 8.7.2 Registrierter User

8.9.3 PC Verwaltung (m_verwaltung.php)

Siehe Kapitel 8.8.3 Master User

8.9.4 Services (m_service.php)

Siehe Kapitel 8.8.4 Master User

8.9.5 Mein Account (account.php)

Siehe Kapitel 8.7.4 Registrierter User

8.9.6 RFID Suche (rfid.php)

Siehe Kapitel 8.8.6 Master User

8.9.7 Benutzerverwaltung (admin_benutzer.php)

Über die Benutzerverwaltung hat ein Administrator die Möglichkeit, einzelne Benutzer in ihren Rechten einzuschränken bzw. diese zu erweitern.

Abbildung 8.37 Benutzerverwaltung

Mit den Funktionsschaltflächen (Abbildung 8.37, Markierung 1) kann der Administrator eigenständig neue Benutzer anlegen bzw. explizit nach bestimmten Benutzern suchen.

Die Aktionsbuttons (Abbildung 8.37, Markierung 1) bzw. dienen zum Löschen und Editieren des entsprechenden Benutzers.
Durch die Schaltflächen und können zum Aktivieren/Deaktivieren eines registrierten Users herangezogen werden. Durch die Aktivierung wird einem registrierten User die Möglichkeit gegeben, Serviceeinträge zu erstellen.

Wird ein mit die Edit-Funktion aufgerufen erscheint dieses Formular:

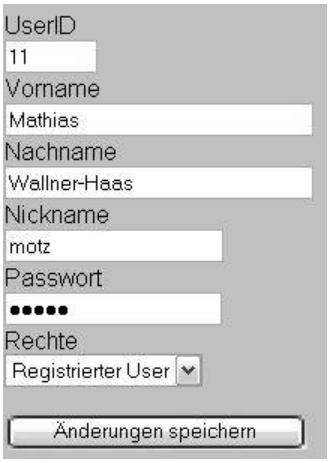


Abbildung 8.38 Benutzer editieren

Der Administrator kann hier alle Accountdaten eines beliebigen Benutzers editieren.
Dies beinhaltet auch das Passwort und die Rechtekategorie.

Master User bzw. andere Administratoren können nur von einem User mit Administrator-Rechten erstellt werden.

Das Feld UserID kann nicht bearbeitet werden.

8.9.8 Backup (admin_backup.php)

Die Backup Funktion ist notwendig um Sicherungen der Datenbank zu erstellen. Backup Dateien werden im Verzeichnis htdocs/backup gespeichert.

Über die Schaltfläche „Backup erstellen“ (Abbildung 8.39, Markierung 1) wird die Sicherung der Datenbank vorgenommen. Dazu wird vom Skript ein Kommandozeilenbefehl auf dem Server ausgeführt.
Der mit `shell_exec()` ausgeführte Befehl wird vom MySQL DBS zur Verfügung gestellt.

```
<Installationsverzeichnis>\mysql\bin\mysqldump --user Sebastian -psebastian!?
--database pcinventar > <Installationsverzeichnis>\htdocs\backup\<dateiname>
```

Als Dateiname wird mit der Funktion time() und der Endung .bak festgelegt. time() liefert einen UNIX Timestamp zurück, welcher einen eindeutigen Dateinamen garantiert.

Das Skript bietet darüber hinaus die Option, Backup Dateien wiederherzustellen. In Markierung 2 werden alle Inhalte des Verzeichnisses htdocs\backup aufgelistet. Durch Anklicken des gewünschten, verlinkten Backup Files wird die Datenbank Wiederherstellung durch das Kommando

```
<Installationsverzeichnis>\mysql\bin\mysqldump --user Sebastian -psebastian!?
--database pcinventar < <Installationsverzeichnis>\htdocs\backup\<dateiname>
```

durchgeführt. Das zuletzt erstellte Backup weist als Dateiname den höchsten UNIX Timestamp Wert auf.

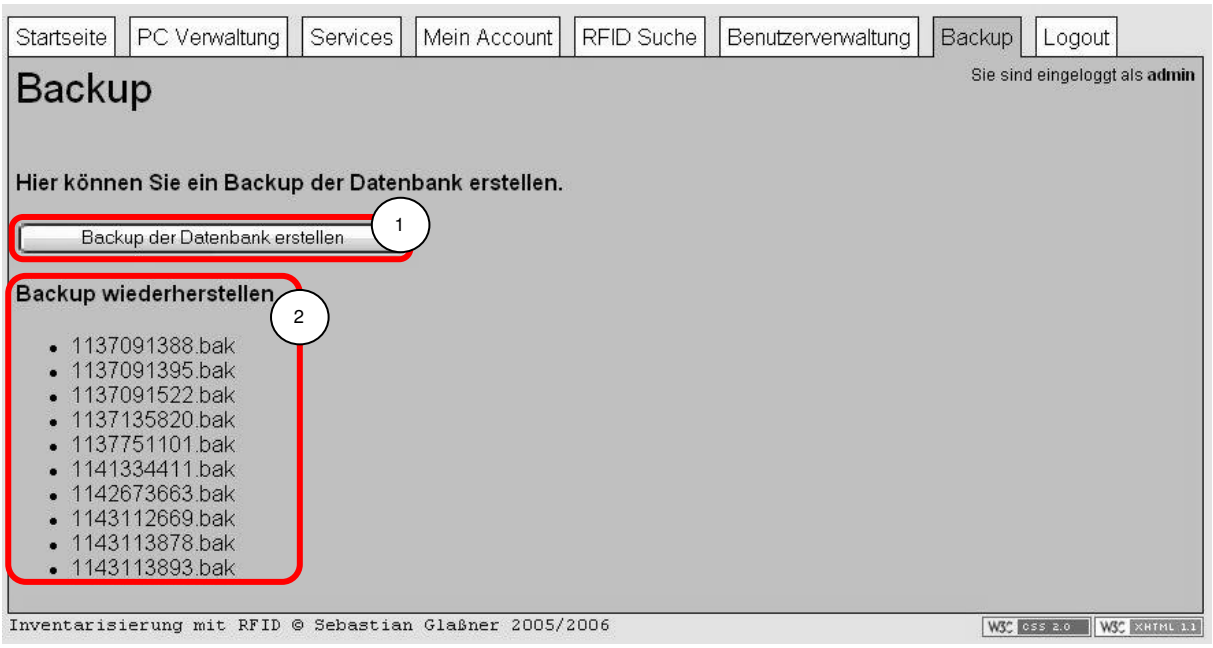


Abbildung 8.39 Backup – Administrator

8.10 Webservice

8.10.1 Allgemein

Ein Webservice ist eine Softwarekomponente, die unter Verwendung von bestehenden Technologien wie HTTP (für den Transport der Daten) und XML (für die Formatierung der Daten) „über das Web" genutzt werden kann. In diesem Kapitel wollen wir die Programmierung von und mit Webdiensten in PHP auf Grundlage der XML-Standards (SOAP) und Web Service Description Language (WSDL) betrachten. SOAP dient der Formatierung der Daten, WSDL der Beschreibung der durch den Webdienst angebotenen Funktion(en).
Dank der verbesserten Unterstützung für XML-Technologien bietet PHP 5 eine ebenso effiziente wie komfortable Schnittstelle für die Arbeit mit Webdiensten an.
Am Server befindet sich ein Skript (SoapServer.php im Verzeichnis htdocs\soap), welches das eigentliche Webservice anbietet.

Die Möglichkeiten eines Webservices sollen anhand des folgenden Beispiels näher erläutert werden.

8.10.2 Beispiel

Server
Zunächst muss ein Serverskript erstellt werden, welches den Webdienst zur Verfügung stellt

```
<?php
function  addiere($sum1, $sum2) {
    $res = $sum1 + $sum2;
    $datei = fopen("F:\\datei.txt", "w");
    fwrite ($datei, "$sum1+$sum2=");
    fwrite ($datei, $res);
    return $res;
}
$server = new SoapServer('http://localhost/mywsdl.wsdl');
$server->addFunction('addiere');            //Funktion zum Server hinzufügen
$server->handle();                          //Hier wird die Abfrage abgearbeitet
?>
```

In diesem einfachen Beispiel wurde ein Soap Server erstellt, die zwei Werte addiert, und das Ergebnis über SOAP an den Client zurück schickt. Das Ergebnis wird zusätzlich in einer Datei am Server gespeichert.

Die für eine SOAP Übertragung benötigte WSDL Datei ist wie folgt aufgebaut:

Kopfbereich
Im Kopfbereich erfolgen allgemeine Angaben zur WSDL Datei. Hier befindet sich auch der einleitende `<definitions>` Tag, der den Beginn der Definition kennzeichnet.

```
<?xml version ='1.0' encoding ='UTF-8' ?>
<definitions name='TestServer'                                  //Name der Definition
  xmlns:tns=' http://172.31.11.21/mywsdl.wsdl '                 //Pfad zur WSDL Datei
  targetNamespace="http://172.31.11.21/mywsdl.wsdl"
  xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/'            //Standards für WSDL
  xmlns:xsd='http://www.w3.org/2001/XMLSchema'
  xmlns:soapenc='http://schemas.xmlsoap.org/soap/encoding/'
  xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
  xmlns='http://schemas.xmlsoap.org/wsdl/'>
```

Message
Zwischen den Message Tags werden die einzelnen Nachrichten definiert. Mit der Eigenschaft "name" wird der Name für diese Nachricht festgelegt.
Die Zeile mit dem "part" bestimmt die Variablen, die übergeben werden sowie ihren Datentyp.

```
<message name='addiereAnfrage'>                                 //Name der Funktion
  <part name='sum1' type='xsd:float'/>
  <part name='sum2' type='xsd:float'/>
</message>
<message name='addiereAntwort'>                                 //Name der Funktion
  <part name='Result' type='xsd:float'/>
</message>
```

Port Type
Hier wird festgelegt, welche Funktion welche „Message" bekommt. Also in unserem Beispiel bekommt die Operation „addiere" eine Input Message und eine Output Message zugeschrieben. Als Operation wird die eigentlich Funktion bezeichnet.
Konkret heißt das, dass hier festgelegt wird, dass die Message „addiereAnfrage" der Funktionsaufruf an den Server ist (Eingangsnachricht) und die Message „addiereAntwort" den Rückgabewert darstellt (Ausgangsnachricht)..

```
<portType name='TestServerPortType'>
  <operation name='addiere'>
    <input message='tns:addiereAnfrage'/>
    <output message='tns:addiereAntwort'/>
  </operation>
</portType>
```

Binding
Hier wird die Methode festgelegt, mit der der Client mit dem Server kommuniziert und welche Verschlüsselung genommen wird. Die vorgenommenen Definitionen sind Standards.

```
<binding name='TestServerBinding' type='tns:TestServerPortType'>
  <soap:binding style='rpc'
    transport='http://schemas.xmlsoap.org/soap/http'/>
  <operation name='addiere'>
    <soap:operation soapAction='urn:xmethodsTestServer#addiere'/>
    <input>
      <soap:body use='encoded' namespace='urn:xmethodsTestServer'
        encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
    </input>
    <output>
      <soap:body use='encoded' namespace='urn:xmethodsTestServer'
        encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
    </output>
  </operation>
</binding>
```

Service
Im Service-Tag wird festgelegt mit welcher Datei der Client kommunizieren muss.

```
<service name='TestServerService'>
  <port name='TestServerPort' binding='tns:TestServerBinding'>
    <soap:address location='http://172.31.11.21/wsdlserver.php'/>
  </port>
</service>
</definitions>
```

Wird nun ein Client erstellt der die Funktion „addiere“ aufruft und zwei Variablen übergibt kann folgende SOAP Nachricht aufgezeichnet werden.
Die Aufzeichnung der Nachricht erfolgte mit dem Netzwerk Sniffer Ethereal.

SOAP Anfrage:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SOAP-ENV:Envelope
xmlns:SOAPSDK1="http://www.w3.org/2001/XMLSchema"
xmlns:SOAPSDK2="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAPSDK3="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <SOAPSDK4:addiere xmlns:SOAPSDK4="urn:xmethodsTestServer">
            <sum1>371</sum1>
            <sum2>9</sum2>
      </SOAPSDK4:addiere>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Antwort:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="urn:xmethodsTestServer"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
        <SOAP-ENV:Body>
            <ns1:addiereResponse>
                <Result xsi:type="xsd:float">
                    380
                </Result>
            </ns1:addiereResponse>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Aus diesem Beispiel ist zu erkennen das der Server auf die Anfrage die beiden Zahlen 371 und 9 zu addieren, das Ergebnis 380 zurück liefert.

8.10.3 SOAP Server

Die verwendeten Funktionen sind in Kapitel 6.6.8 Theoretische Grundlagen erklärt.
Das Skript wird bei der Anfrage eines Clients ausgeführt. Die Struktur der Anfrage muss der WSDL Definition entsprechen.
Bei einem korrekten Kommunikationsaufbau werden die übermittelten Transponder Daten in der Tabelle „rfidsearch“ abgelegt.
Es werden folgende Daten über SOAP übertragen:

- Unique ID
- Inventarnummer des Rechners
- Standort des Rechners
- Anzahl der gelesenen Transponder (für zukünftige Mehrfachabfrage-Anwendungen)

Als Antwort erhält der Client den String „Ihre Daten wurden erfolgreich an den Server weitergeleitet!“ zurück.

Die Weiterverarbeitung der Daten erfolgt anschließend im Skript rfid.php (Kapitel 8.6.5)

8.10.4 Verwendete WSDL Datei

siehe Projekt CD

9 Zusammenfassung und Ausblick

9.1 Zusammenfassung

Das Projekt ist im geforderten Ausmaß voll funktionsfähig. Das heißt die zugrunde liegende Aufgabenstellung wurde erfüllt.

Das System ermöglicht folgende Anwendungen:

- Das RFID-Lesegerät kann angesteuert werden.
- RFID Transponder können ausgelesen und beschrieben werden.
- Durch Anbringen eines Tags an einem Rechner kann dieser eindeutig identifiziert werden. Gibt es in der Datenbank Datensätze, die mit der eindeutigen Identifikationsnummer dieser Tags verknüpft sind, so können Informationen wie der Name, Standort oder der Reparaturstatus des Rechners, an dem der Transponder angebracht ist, eingesehen werden.
- Wartungs- und Servicetätigkeiten an den Rechnern können nachvollziehbar festgehalten werden.
- Die Verwaltung von Rechnern und Reparaturvorgängen kann über eine einfach zu bedienende grafische Weboberfläche vorgenommen werden.
- Zur Kommunikation zwischen den beiden realisierten Software-Teilen wird das XML-basierende SOAP-Protokoll verwendet.

9.2 Ausblick

Das Projekt könnte durch entsprechendes editieren der Skripte und Quellcodes auf andere Inventarisierungssysteme wie zum Beispiel Labormessgeräteinventarisierung, Bücherinventarisierung in Bibliotheken,… umgelegt werden.
Darüber hinaus könnte auch ein Zugangskontrollsystem entwickelt werden, das dieses Projekt als Vorlage benutzt.

In Zukunft wird die RFID Technologie immer mehr an Bedeutung gewinnen. Im Zusammenspiel mit einem Datenbanksystem schafft dieses Projekt eine technisch hoch interessante und aktuelle Ausgangsbasis für ähnliche Anwendungen.

A Abbildungsverzeichnis

5 Realisierung

6 Theoretische Grundlagen

7 Transportschicht

8 Webanwendung

B Literaturverzeichnis

B.1 Bücher, Zeitschriften, Diplomarbeiten

Bücher:

Titel	Autor	Verlag
RFID Handbuch 3.Auflage 2002	Klaus Finkenzeller	Hanser

Zeitschriften:

Name	Ausgabe	Titel
CHIP Professional	05/2005	Web-Design professionell

Diplomarbeiten:

Titel	Autor/en	Ausführung
Niederösterreichische Volkskultur Mitgliederverwaltungsdatenbank	Zehetner, Aschenbrenner	HTL Hollabrunn 2004/2005
Radlbrunn Homepage	Aichinger, Hörmann	HTL Hollabrunn 2004/2005

B.2 Internet

URL	Beschreibung
http://www.de.selfhtml.org	HTML, CSS Dokumentation
http://ffm.junetz.de/members/reeg/	MySQL, PHP Manual
http://www.selfphp.info/	PHP Funktionsreferenz, Befehlsverzeichnis
http://www.php.net/docs.php	PHP Manual
http://www.mysql.com/documentation/.	Offizielle MySQL Dokumentation
http://de.wikipedia.com	Online Lexikon
http://www.tutorials.de/forum/php-tutorials/166733-php5-einstieg-soap.html	Webservice Erstellung, SOAP Grundlagen
http://www.professionelle-softwareentwicklung-mit-php5.de/erste_auflage/programming-php.soap.html	Webdienst Erstellung
http://www.apachefriends.org/de/index.html	XAMPP Download, Docs
http://www.codeguru.com/	C-Codes
http://sine.ni.com	National Instruments Homepage
http://www.ti.com/tiris/	RFID-Bereich von Texas Instruments
http://home.wtal.de/ranzurmall/visualc/	Visual C++ Grundkenntnisse
http://cplus.kompf.de	Grundkenntnisse für objektorientiertes Programmieren
http://www.pruefziffernberechnung.de/L/LRC.shtml	LRC Berechnung
http://www.st.com/stonline/books/pdf/docs/9716.pdf	Transponder Datenblatt

B.3 CDs

Name	Inhalt	Herausgeber
TagTracker RFID Test Suite 2.1	Demoprogramme	Databrokers
Texas Instruments Evaluation Kit CD (*cd*\documentation\)	Datenblätter zur MFR-Hardware; Datenblätter zur API	Texas Instruments

C Glossar

Apache	Der am meisten verbreitete Web Server, hat heute ca. zwei Drittel Marktanteil. Apache wird von professionellen Site-Betreibern bevorzugt, weil sich das Produkt als stabil und effizient erwiesen hat und neue Versionen selten Sicherheitsprobleme haben.
AFI	Die Application Family ID gibt an, für welchen Verwendungszweck der Transponder gedacht ist. (siehe Theoretische Grundlagen → auf Tag gespeicherte Daten)
Button	Frei übersetzt würde man von einem „Absendeknopf“ sprechen. Damit können z.B. Formulare abgesendet werden.
Client	Als Client (englisch für „Klient, Mandant“) wird ein Computerprogramm bezeichnet, das die Ressource einer zentralen Station (eines Servers) nutzt. Der Client muss dazu Kontakt zum Server aufbauen und kann dann dessen Angebot nutzen. Die Kommunikation erfolgt im allgemeinen über ein Rechnernetz. Das heißt, der Server befindet sich auf einem anderen Rechner als der Client. Das Verfahren wird als Client-Server-Prinzip bezeichnet
Datensatz	Ein Datensatz ist ein „Eintrag“ in der Datenbanktabelle.
DISPID	Dispatch ID, dient zum Ansprechen eines Webservices. Diese ID wird bei SOAP über den Namen des Webservice gefunden.
DSFID	Die Data Storage Format ID gibt an, in welchem Format die Daten auf dem Transponder gespeicher sind (siehe Theoretische Grundlagen → auf Tag gespeicherte Daten)
HTML-Tag	Tag ist die Bezeichnung für einen HTML Befehl. Ein Tag steht immer in Spitzen Klammern.
Link	Ein Link ist ein Verweis auf eine bestimmte Datei.
Logdatei	Eine Logdatei (engl. log file) beinhaltet das automatisch erstellte Protokoll aller oder bestimmter Aktionen von einem oder mehreren Nutzern an einem Rechner, ohne dass diese davon etwas mitbekommen oder ihre Arbeit beeinflusst wird.
LRC	Longitudinal Redundancy Check: Verfahren zur Fehlerprüfung bei blockweiser Übertragung, bei dem aus einem Datenblock ein Paritätsbyte berechnet wird, das als Redundanzinformation zusätzlich zum Block übertragen wird.

MFR	Multi Function Reader: RFID Lesegerät, welches mehrere Frequenzbereiche unterstützt
PHP	PHP ist eine Erweiterung für Internet-Server (ähnlich ASP für Unix und Windows), die es ermöglicht, schnell und mit verhältnismäßig wenig Aufwand dynamische Websites für Multimedia- oder E-Commerce-Anwendungen im Internet zu erstellen.
RFID	Radio Frequency Identification (engl. für Funk-Erkennung) ist eine Methode, um Daten berührungslos und ohne Sichtkontakt lesen und speichern zu können.
RFID-Transponder **RFID-Tag** **RFID-Smart Label** **RFID-Token**	Teil des RFID-Systems. Eine Sende/Empfangs-Einheit, die nach einer Anfrage, die gewünschten Daten (z.B.: Identifikationsnummer) zurückliefert. Ein Transponder ist ein - meist drahtloses - Kommunikations-, Anzeige- oder Kontrollgerät, das eingehende Signale aufnimmt und automatisch darauf antwortet.Der Begriff Transponder ist zusammengesetzt aus den Begriffen Transmitter und Responder. Ein RFID Tag besteht aus einen Mikrochip und einer Antenne. Auf dem Chip ist ein Code gespeichert, der bestimmte Informationen enthält. Als Antwort auf ein auslösendes Radiosignal eines Lesegerätes können diese Transponder selbst Signale senden oder empfangen. Man unterscheidet dabei aktive und passive RFID Tags.
RPC	Remote Procedure Call, ein Protokoll für verteilte Anwendungen
Server	Ein Server ist ein Programm, welches auf die Kontaktaufnahme eines Client-Programmes wartet und nach Kontaktaufnahme mit diesem Nachrichten austauscht.
Skript	Skriptsprachen (üblich auch Scriptsprachen) sind Programmiersprachen, die vor allem für kleine, überschaubare Programmieraufgaben gedacht sind. Skripte sind Programme die von einem Interpreter (ein anderes Programm) ausgeführt werden.
SOAP	SOAP ist ein Protokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und Remote Procedure Calls durchgeführt werden können. SOAP stützt sich auf die Dienste anderer Standards, XML zur Repräsentation der Daten und Internet-Protokolle der Transport- und Anwendungsschicht (vgl. TCP/IP-Referenzmodell) zur Übertragung der Nachrichten. Die gängigste Kombination ist SOAP über HTTP und TCP
UID	Unique ID: Einzigartige Identifikationsnummer des Transponders
User	User ist der englische Ausdruck für Benutzer.

WSDL	Die Web Services Description Language definiert einen plattform-, programmiersprachen- und protokollunabhängigen XML-Standard zur Beschreibung von Netzwerkdiensten (Webservices) zum Austausch von Nachrichten.
XML	Die Extensible Markup Language, abgekürzt XML, ist ein Standard zur Erstellung maschinen- und menschenlesbarer Dokumente in Form einer Baumstruktur. XML definiert dabei die Regeln für den Aufbau solcher Dokumente. Für einen konkreten Anwendungsfall ("XML-Anwendung") müssen die Details der jeweiligen Dokumente spezifiziert werden. Dies betrifft insbesondere die Festlegung der Strukturelemente und ihre Anordnung innerhalb des Dokumentenbaums.

D Installationsanleitung

D.1 Webanwendung

D.1.1 Konventionen

<Inst.Vz>

Dieser Ausdruck wird im folgenden Dokument für den Pfad des Installationsverzeichnisses der Softwaredistribution verwendet.
Im Standardfall lautet der Pfad *C:\apachefriends\xampp* . Ansonsten muss der Pfad dem bei der Installation gewählten angepasst werden.

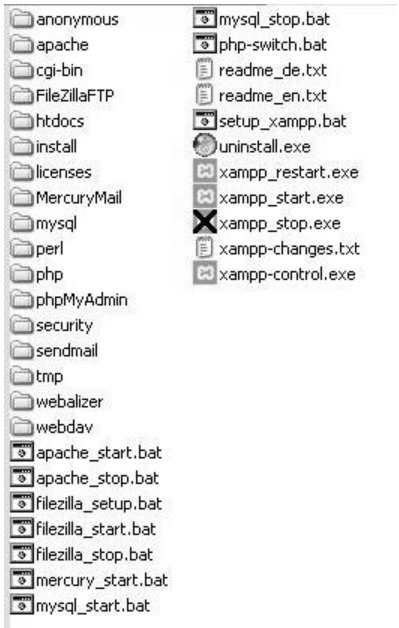
D.1.2 XAMPP installieren

Der Start der Installation erfolgt durch Ausführen der Datei *xampp-win32-1.4.14-installer.exe*.
Die Datei befindet sich auf der Projekt CD im Verzeichnis
\Webanwendung\Software_Entwicklungsumgebung

Installationsablauf:

1. Sprache auswählen… Deutsch > OK
2. Instalationsassistent… Weiter >
3. Lizenzvertrag… Annehmen >
4. Installationsverzeichnis wählen (Installation benötigt ca. 160MB) > Installieren
5. Fertigstellen >

Bei der Installation legt XAMPP folgende Struktur innerhalb des Installationsverzeichnisses an:



XAMPP Verzeichnisstruktur

Wichtige Verzeichnisse:

<Inst.Vz>htdocs\ Dieser Ordner ist das Root Verzeichnis des Webservers. In ihm befindet sich das entwickelte HTML bzw. PHP Projekt.

<Inst.Vz>apache\bin\ Hier befindet sich die Datei php.ini, die zum Konfigurieren des PHP API dient.

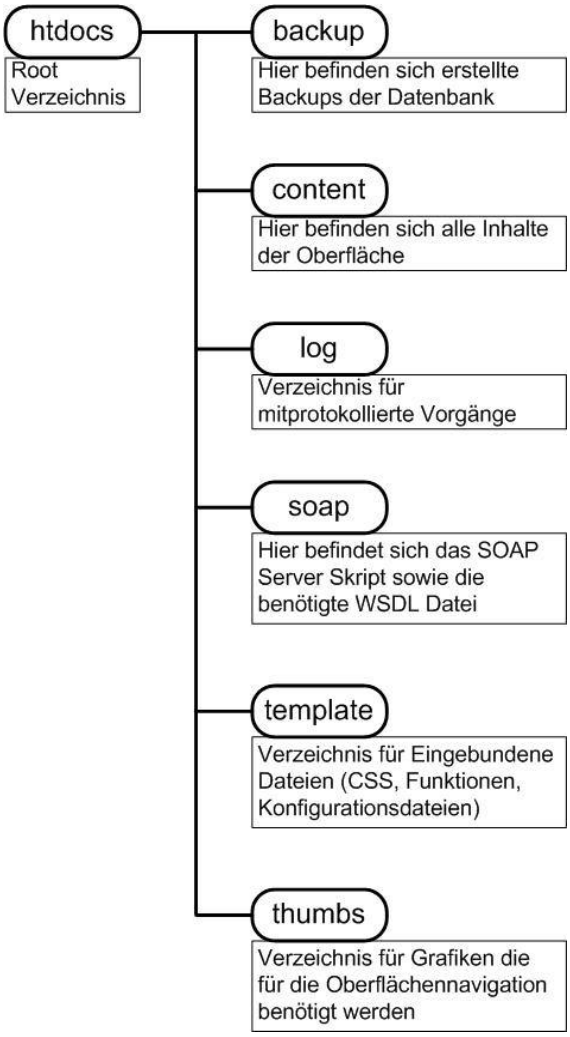
D.1.3 Praktisches Arbeiten mit XAMPP

Die in *<Inst.Vz>* vorhandenen Anwendungen *xampp_start.exe* und *xampp_stopp.exe* dienen zum Starten bzw. Beenden von XAMPP.

D.1.4 Webprojekt anlegen

Das Webprojekt muss in den Ordner *<Inst.Vz>htdocs* kopiert werden. Die folgende Abbildung zeigt die Verzeichnisse der Webanwendung.
Das Webprojekt befindet sich auf der Projekt CD im Verzeichnis *Webanwendung\htdocs\ .*

XAMPP legt bei der Installation ein Beispielprojekt im Verzeichnis *<Inst.Vz>htdocs* an. Dieses muss vor dem gelöscht werden.
Anschließend kann das Webprojekt (Inhalt des htdocs Ordner auf der Projekt CD) in das Verzeichnis *<Inst.Vz>htdocs* am Rechner kopiert werden.



Verzeichnisstruktur des Webprojektes

D.1.5 Datenbank

Nun muss die Datenbank angelegt werden.
Im ersten Schritt muss hierzu der Webserver gestartet werden. Dazu muss die Anwendung *xampp_start.exe* im Verzeichnis *<Inst.Vz>* ausgeführt werden.

Datenbankstruktur

Um die Datenbankstruktur anzulegen muss zunächst die Eingabeaufforderung geöffnet werden. Anschließend müssen folgende Kommandos eingegeben werden:

1. In das Verzeichnis `<Inst.Vz>mysql\bin` wechseln:
   Beispiel:
   ```
   >C:
   >cd apachefriends\xampp\mysql\bin
   ```

2. Als „root“ in die MySQL-Konsole einloggen:
   Dazu muss im o.g. Verzeichnis folgender Befehl ausgeführt werden::

   ```
   mysql -u root
   ```

3. Datenbank und Datenbankstruktur anlegen:
   Mit dem Folgenden Kommando wird eine Backup-Datei der leeren Datenbank mit der benötigten Struktur wiederhergestellt.

   ```
   \. <Inst.Vz>htdocs\backup\PCINVENTAR_LEER.bak
   ```

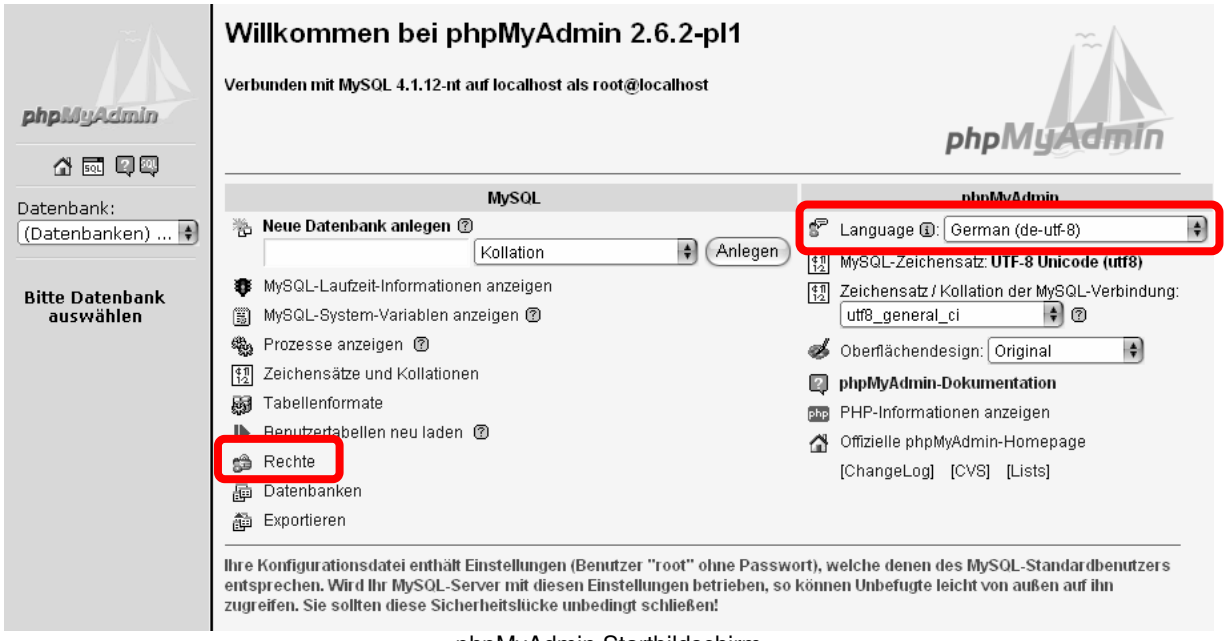
   Zusätzlich wird auch ein Benutzer zum späteren Navigieren auf der Weboberfläche angelegt:

Nick:	*admin*
Passwort:	*admin!?*

Datenbankbenutzer erzeugen

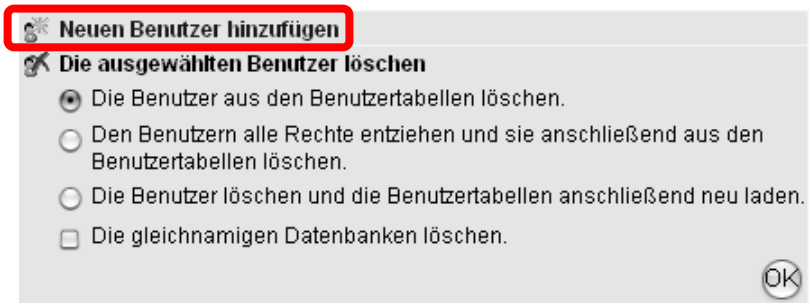
1. Webserver muss gestartet sein
2. mit Webbrowser auf http://localhost/phpmyadmin/ zugreifen und folgende Schritte durchführen:

I. Zunächst sollte die Sprache auf „Deutsch“ umgestellt werden (Markierung)
Am phpMyAdmin Startbildschirm dann den Punkt Rechte auswählen (Markierung).



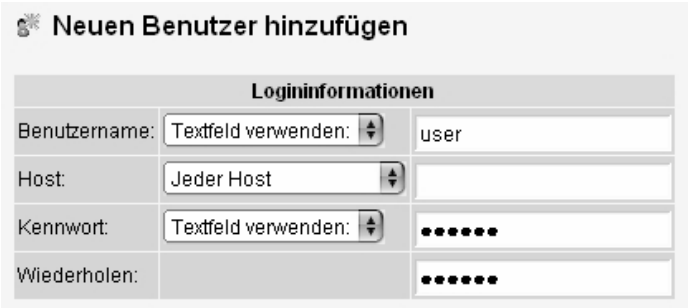
phpMyAdmin Startbildschirm

II. auf „Neuen Benutzer hinzufügen“ (Markierung) klicken



Neuen Benutzer hinzufügen

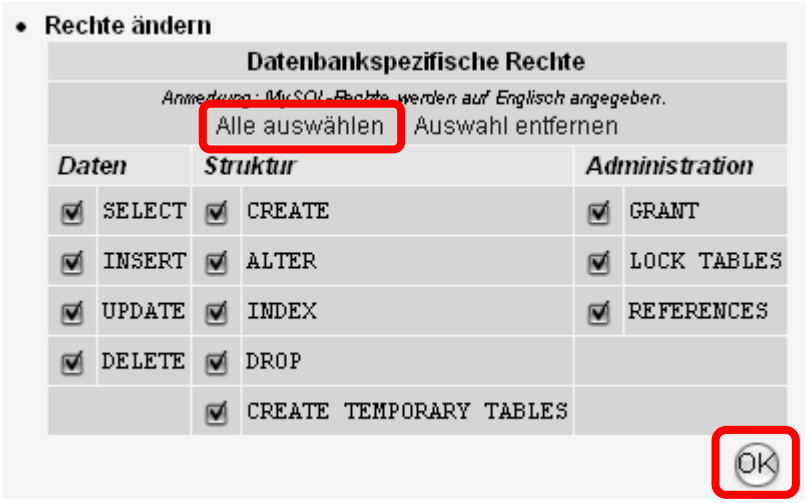
III. im nächsten Bildschirmfenster folgende Eingaben tätigen und mit „OK“ (ganz unten) bestätigen
→Benutzername: user
→Host leerlassen
→Kennwort: user!?



Eingabemaske für neuen Benzuter

IV. Datenbankspezifische Rechte zur Datenbank pcinventar hinzufügen. Hierzu muss im Eingabefeld (Markierung) der Name der Datenbank „pcinventar“ eingetragen und mit „OK“ bestätigt werden.

V. Nun müssen mit Klick auf „Alle auswählen“ die Rechte erteilt und schließlich mit „OK bestätigt werden.



VI. Standardmäßig ist für den MySQL-Administrator „root“ kein Passwort zugewiesen. Um diese Sicherheitslücke zu schließen muss zunächst dem User „root“ ein Passwort zugewiesen werden.
Dazu erneut den phpMyAdmin Startbildschirm und anschließend den Menüpunkt „Rechte“ aufrufen. (siehe I.)
In der erscheinenden Benutzerübersicht beim „root“ User auf „Rechte ändern“ (Button rechts) klicken.

	Benutzer	Host	Kennwort	Globale Rechte	Grant	
☐	Jeder	%	Nein	USAGE	Nein	
☐	Sebastian	localhost	Ja	USAGE	Nein	
☐	asdf	%	Ja	USAGE	Nein	
☐	asfd	%	Ja	USAGE	Nein	
☐	pma	localhost	Nein	ALL PRIVILEGES	Ja	
☐	root	%	Nein	USAGE	Nein	
☐	root	localhost	Nein	ALL PRIVILEGES	Ja	
☐	user	%	Ja	USAGE	Nein	

Benzutzerübersicht

VII. Im folgenden Bildschirm den Radiobutton vor „Kennwort“ aktivieren und ein beliebiges Kennwort (z.B root!?) eintragen. Mit „OK“ bestätigen. Der Browser kann nun geschlossen werden.

VIII. Nach dem einsetzen eines neuen Passwortes für „root“ muss auch phpMyAdmin informiert werden.
Das geschieht über die Datei „`config.inc.php`" zu finden als `<Inst.Vz>phpmyadmin\config.inc.php`. Dort also folgende Zeile editieren:

```
Zeile 83:   $cfg['Servers'][$i]['auth_type']      = 'config';
            ändern auf:
            $cfg['Servers'][$i]['auth_type']      = 'http';
```

IX. So wird zuerst das 'root' Passwort vom MySQL Server abgefragt, bevor phpMyAdmin zugreifen darf. Beim nächsten Start von phpMyAdmin mit http://localhost/phpmyadmin/ sieht das dann so aus:

D.1.6 Konfigurationen

Aktivieren der PHP SOAP-Extension

Um sie zu aktivieren muss das Semikolon vor dem folgenden Eintrag in der php.ini Datei entfernt werden.
Achtung: Bei der Installation von XAMPP werden mindestens drei Konfigurationsdateien mit dem Namen php.ini erzeugt. Die Änderung muss in der php.ini Datei in dem Verzeichnis `<Inst.Vz>apache\bin` vorgenommen werden.

```
597 ;extension=php_radius.dll
598 ;extension=php_rar.dll
599 ;extension=php_shmop.dll
600 ;extension=php_snmp.dll
601 extension=php_soap.dll
602 ;extension=php_sockets.dll
603 ;extension=php_stats.dll
604 ;extension=php_sybase_ct.dll
605 ;extension=php_threads.dll
```

Um die Änderung zu übernehmen muss der Server mit `<Inst.Vz>xampp_restart.exe` neu gestartet werden.

Server IP Adressen Anpassen

SoapServer.php

Verzeichnis: `<Inst.Vz>htdocs\soap\SoapServer.php`
Der Variable sip muss die IP Adresse des Servers zugewiesen werden. Dazu dass Skript öffnen und in Zeile 16 zwischen die beiden Anführungszeichen die IP eintragen.
Beispiel:

Zeile 16: `$sip = "172.31.11.21";`

admin_backup.php

Verzeichnis: `<Inst.Vz>htdocs\content\admin_backup.php`
Hier muss der Installationspfad zu den beiden Dateien *mysql.exe* und *mysqldump.exe* angepasst werden. Achtung! ALLE Backslashs müssen doppelt ausgeführt werden.

Zeile 50:
```
$command="<Inst.Vz>\\mysql\\bin\\mysqldump -u user -puser!? pcinventar > $pfad";
```
Zeile 66:
```
$command="<Inst.Vz>\\mysql\\bin\\mysql -u user -puser!? pcinventar < $pfad";
```

TagDataMsg.wsdl

Verzeichnis: `<Inst.Vz>htdocs\soap\TagDataMsg.wsdl`
Die angegebenen IP Adressen müssen mit der des Servers übereinstimmen:

Beispiel:

Zeile 2: `xmlns:tns=' http://172.31.11.21/TagDataMsg.wsdl '`
Zeile 3: `targetNamespace=http://172.31.11.21/TagDataMsg.wsdl`
Zeile 46: `<soap:address location='http://172.31.11.21/soap/SoapServer.php'/>`

D.1.7 Webanwendung starten

Um auf die Webanwendung zugreifen zu können muss nun am Server in einem Browser http://localhost aufgerufen werden.
Greift man darauf von einem Client zu, muss http://<ServerIP> aufgrufen werden.
Zur Verwaltung der Weboberfläche steht ein Administratoraccout (Nick: admin, Passwort: admin!?), zur Verfügung.

D.2 Transportschicht

Das Transportschicht-Programm wurde „RFID-invent“ getauft.
Es dient zu folgenden Zwecken:

- Senden und Empfangen von RFID-Reader-Daten
- Senden und Empfangen von SOAP-Daten
- User Interface für Interaktion mit Personen
- Ausführen eines Skripts zur Automatisierung der Transportschicht

In den folgenden Kapiteln wird erläuert, wie das Programm installiert und deinstalliert werden kann und welche Schritte notwendig sind, um das Programm weiterentwickeln zu können.

D.2.1 Installation von RFID-invent

1.) Installer ausführen: *ProjektCD*:\Transportschicht\Installer\RFID-invent 1_0 (Installer).msi
2.) RFID-invent Setup: Auf den Button <Weiter> klicken.
3.) RFID-invent Setup: Das Verzeichnis, in das das Programm installiert werden soll, wählen.
4.) RFID-invent Setup: Auf den Button <Weiter> klicken.
5.) RFID-invent Setup: Auf den Button <Installieren> klicken.
6.) Nach kurzer Zeit sollte ein Fenster mit der Frage, ob das MS Soap Toolkit installiert werden soll, erscheinen. Hier auf <Ja> klicken.
7.) SOAP Toolkit Setup: Auf den Button <Next> klicken.
8.) SOAP Toolkit Setup: Den Lizenzvereinbarungen zustimmen und auf <Next> klicken.
9.) SOAP Toolkit Setup: Name und Firma angeben und wider auf <Next> klicken.
10.) SOAP Toolkit Setup: Die zu installierenden Teile wählen. Es werden nur die C++ Support Files und der XML-Parser benötigt. Es schadet aber nicht, die anderen Teile auch zu installieren.
11.) SOAP Toolkit Setup: Den gewünschten Pfad wählen und auf <Install> klicken.
12.) SOAP Toolkit Setup: Auf <Finish> drücken.
13.) RFID-invent Setup: gewünschte Aktionen auswählen und auf <Ferigstellen> klicken

Zusammenfassung:
RFID-invent 1_0 (Installer).msi ausführen und den Anweisungen folgen.

D.2.2 Konfiguration von RFID-invent

Startskript anpassen:
Im Programmverzeichnis (meistens C:\Programme\HTL Hollabrunn\RFID-invent) befindet sich die Datei start.rfs. Diese Datei kann mit einem Texteditor bearbeitet werden. Hier können die Einstellungen, die beim Start geladen werden sollen, eingestellt werden (näheres siehe Kapitel 7.5 „Skript“).
Normalerweise muss in der Datei nur der Eintrag des WSDL-Pfades geändert werden.

WSDL-Datei:
Im Startscript den Eintrag „- WSDL *„Pfad der WSDL-Datei“* “ angeben
oder
Im RFID-invent UserInterface →Abfrageeinstellungen →WSDL-Pfad →*„Pfad der WSDL-Datei“* ändern

„Pfad der WSDL-Datei“ kann ein lokaler Pfad - z.B.: „C:\WSDL-Dateien\test.wsdl“ (absolut) oder „tagdatamsg.wsdl“ (relativ; in diesem Fall: im Programmverzeichnis) - aber auch ein Pfad auf einem Server z.B.: „http://172.31.11.21/SOAP/tagdatamsg.wsdl“ sein.

Falls die WSDL-Datei des Servers nicht geladen werden kann, empfiehlt es sich die lokale Kopie zu verwenden. Diese wird bei der Installation in den Programmordner kopiert → tagdatamsg.wsdl.
Damit man sich zum Server verbinden kann, müssen die IP-Adressen in der WSDL-Datei geändert werden.

COM-Port:
Der COM-Port kann im RS232-Bereich eingestellt werden. Ist der COM-Port nicht bekannt, so steht kann man den Autoset-Button betätigen.

D.2.3 Deinstallation von RFID-invent

1.) Installer ausführen: *ProjektCD*:\Transportschicht\Installer\RFID-invent 1_0 (Installer).msi
2.) RFID-invent Setup: Auf den Button <Weiter> klicken.
3.) RFID-invent Setup: Auf das Symbol <Entfernen> klicken.
4.) RFID-invent Setup: Auf den Button <Entfernen> klicken.
5.) RFID-invent Setup: Auf den Button <Fertigstellen> klicken.
6.) Nach kurzer Zeit sollte ein Fenster mit der Frage, ob das MS Soap Toolkit installiert werden soll, erscheinen. Hier auf <Ja> klicken.
7.) SOAP Toolkit Setup: Auf den Button <Next> klicken.
8.) SOAP Toolkit Setup: Auf das Symbol <Remove> klicken.
9.) SOAP Toolkit Setup: Auf das Button <Remove> klicken.
10.) SOAP Toolkit Setup: Auf den Button <Finish> klicken

Zusammenfassung:
RFID-invent 1_0 (Installer).msi ausführen und den Anweisungen folgen.

oder

mit Windows
Start → Einstellungen → Systemsteuerung → Software → RFID-invent → entfernen

D.2.4 Entwicklungsumgebung installieren

Das Programm wurde mit der ANSI-C Entwicklungsumgebung Labwindows CVI 8.0 realisiert. Die Transportschicht greift auf die SOAP-Schnittstell zu, für die das Microsoft SOAP-Toolkit 3.0 installiert werden muss.

CVI installieren:
1. ZIP-Datei öffnen: *ProjektCD*:\Transportschicht\Entwicklung\CVI 8.0.zip
2. Installer ausführen: CVI 8.0.zip\Setup.exe
3. den Anweisungen folgen

MS SOAP-Toolkit installieren:
1. Installer ausführen: *ProjektCD*:\Transportschicht\Entwicklung\soapsdk.exe
2. den Anweisungen folgen

Projekt laden:
1. Projektordner: *ProjektCD*:\Transportschicht\Entwicklung\Source auf ein wiederbeschreibbares Medium (Festplatte od. Flash-Speicher) kopieren
2. Doppelklick aud „RFID.prj“)

Hinweis:
Die CVI-Demoversion ist 30 Tage lauffähig, jedoch haben die damit kompilierten Programme ab dem 8. Tag nach der Installation nur eine Laufzeit von 10 Minuten.
Die Neuninstallation von CVI nach den 30 Testtagen gelang nur, wenn vor der Neuninstallation CVI 7 installiert und deinstalliert wurde.

D.2.5 CVI-Programm für andere Rechner lauffähig machen

CVI mit geöffneten Projekt:
Menüleiste → Build → Configuration → Release
Menüleiste → Build → Target Type → Executable
Menüleiste → Build → Target Settings → Run-time support → Full run-time Engine
→ Embed project .UIRs → *auswählen*
Menüleiste → Build → Target Settings → Add files to executable → *alle auswählen*
Ein Installer-Creator Ihrer Wahl (z.B.:Advance Installer 3.9):
„cvirte.dll" nach *Windows-Ordner*/System32/ kopieren
„cviauto.dll" nach *Windows-Ordner*/System32/ kopieren
„cvirte"-Ordner nach *Windows-Ordner*/System32/ kopieren
„mesa.dll" in den Programmordner kopieren

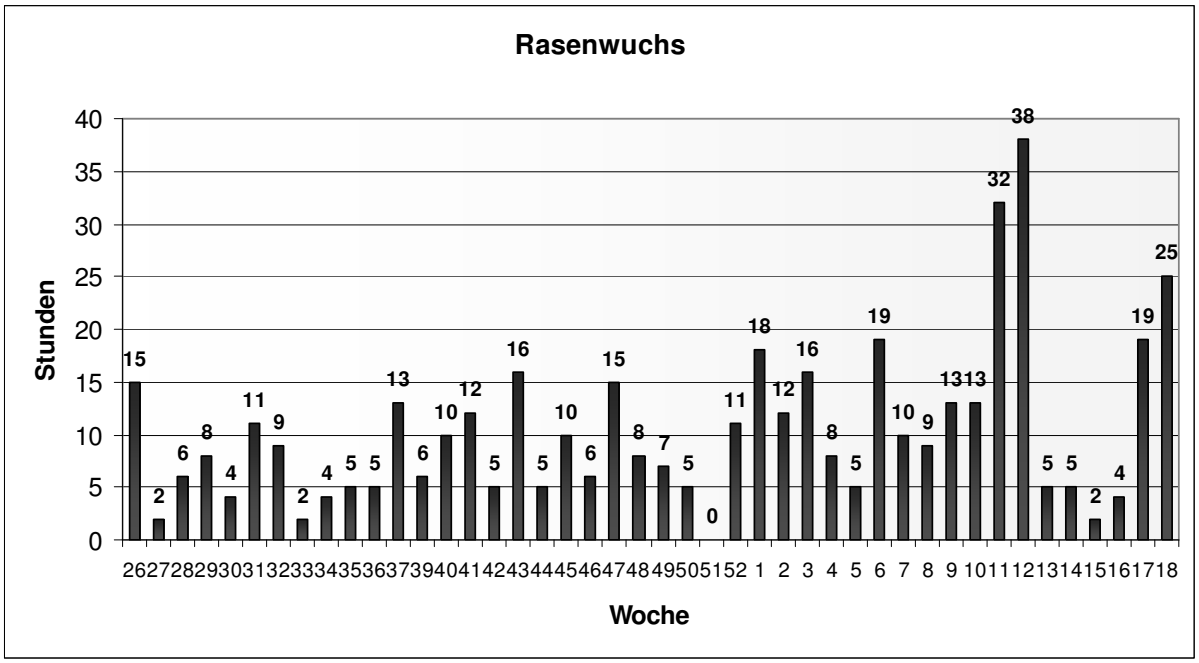
Falls Funktionen des SOAP-Toolkits verfügbar sein sollen, so muss dieses auch installiert werden. Es empfiehlt sich daher, das SOAP-Toolkit-Setup gleich in den Installer des Programms zu integrieren.

Hinweis:
Falls alle oben aufgelisteten Dateien kopiert wurden und das Programm trotzdem nicht ordnungsgemäß funktioniert, so empfiehlt sich folgende Vorgehensweise zur Fehlerbehebung:

1. *Die Programme „Filemon" und „Regmon" herunterladen*
   a. *Filemon überwacht Dateizugriffe*
   b. *Regmon überwacht Registry-Zugriffel*
2. *„Filemon" starten und den Filter auf das zu untersuchende Programm anwenden*
3. *Die Dateizugriffe des CVI-Programms beobachten (besonders wenn die gewünschte Funktion ausgeführt werden sollte; z.B.: beim Betätigen eines Buttons)*
4. *Die Datei, auf die zugegriffen wird, mit der Version der Datei auf der Entwicklungsumgebung austauschen.*
5. *Analoge Vorgehensweise mit „Regmon"*

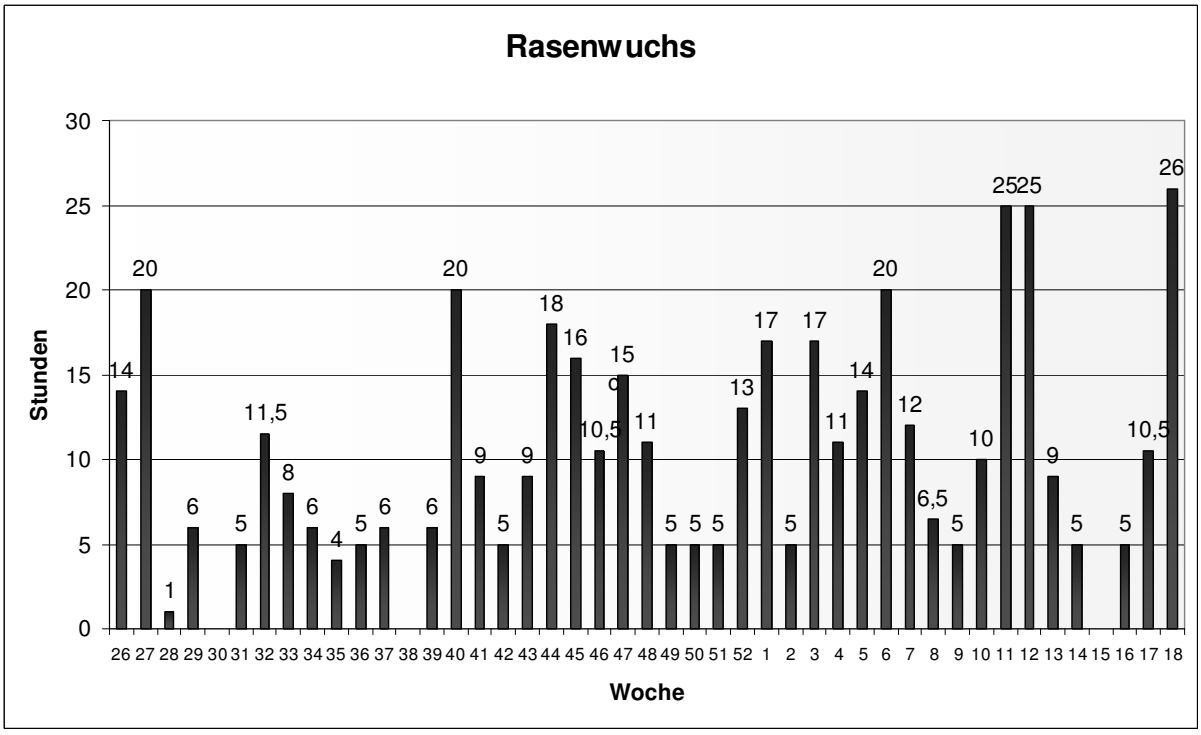
E Zeitaufstellung

E.1 Sebastian Glaßner



Gesamt: 453 Stunden

E.2 Mathias Wallner-Haas



Gesamt: 457 Stunden

F Inhalt der ProjektCD

├──Allgemeines	Allgemeines zum Projektverlauf
│ ├──Preisangebote	Preisangebote für RFID-Geräte
│ └──RFID-Infos	Nützliche Datenblätter zu RFID
│ └──Praesentationen	Nützliche Präsentationen zu RFID-Anwendungsgebieten
├──Dokumentation	Dokumetation der Diplomarbeit
│ ├──Bilder	Bilder, die in der Dokumentation verwendet wurden
│ └──TA Besprechungen	Dokumentation der Besprechungen mit Telekom Austria
├──Hilfssoftware	Software, die für das Projekt hilfreich war
├──Homepage	Statische Projekthomepage
│ ├──downloads	Dateien, die man von der Homepage herunterladen kann
│ ├──pics	Bilder, die auf der Homepage angezeigt werden
│ └──style	Styleelemte der Homepage (Banner, Stylesheets, usw.)
├──Praesentation	Projektpräsentation
│ └──Kick-off Praesentation	Die 3 Folien der Kick-off Präsentation
├──TI-CD	Texas Instruments MFR-CD
│ ├──Documentation	Datenblätter und Beschreibungen der API-Funktionen
│ └──Software	Software für den Reader
│ ├──Demo	Demoprogramm für die Readeransteuerung
│ └──Firmware	Firmware, die auf den Reader geladen werden kann
├──Transportschicht	Material zum Client-Programm (Transportschicht)
│ ├──Entwicklung	für die Weiterentwicklung notwendige Dateien
│ │ └──Source	Source-Projekt: UserInterfaces, C-Codes
│ │ └──Style	Icons und andere Grafiken
│ ├──Installer	Installationsdateien für die Transportschicht
│ │ └──Release	Dateien die vom Installationsgenerator verwendet werden
│ │ ├──all	Ergebnis der generierten Installation
│ │ └──files	Dateien, die in den Installer inkludiert werden
│ └──Übungen	Übungen haben zur Entwicklung des Projekts beigetragen
│ ├──C++	C++ Tutorial
│ ├──CVI	CVI-Beispielprogramm zur Webservice-Ansteuerung
│ ├──RS232	RS232-Programme (Mittschnitt, Virtuelle COM-Ports,...)
│ └──SOAP	SOAP Beispielprogramme
└──Webanwendung	Material zum Datenbankserver (SOAP-Server)
├──htdocs	Root-Verzeichnis der Webanwendung
│ ├──backup	Verzeichnis für Backups
│ ├──content	Seiteninhalte
│ ├──log	Verzeichnis für Log-Dateien
│ ├──soap	Verzeichnis des SoapServer Skripts und der WSDL Datei
│ ├──template	Seitenstrukturen
│ └──thumbs	Bilder
├──Manuals	Manuals zu HTML,PHP und MySQL
│ ├──PHPundMySQL_Manual(DSP)	PHP&MySQL
│ ├──selfhtml811	HTML
│ └──selfphp1_2	PHP Funktions Referenz
└──Software_Entwicklungsumgebung	Installations-File des Webservers (XAMPP)

G GNU Free Documentation License

Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document „free“ in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it,

with or without modifying it, either commercially or non commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of „copyleft“, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The „Document“, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as „you“. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A „Modified Version“ of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A „Secondary Section“ is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The „Invariant Sections“ are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The „Cover Texts“ are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A „Transparent“ copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not „Transparent“ is called „Opaque“.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The „Title Page“ means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, „Title Page“ means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section „Entitled XYZ“ means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as „Acknowledgements“, „Dedications“, „Endorsements“, or „History“.) To „Preserve the Title“ of such a section when you modify the Document means that it remains a section „Entitled XYZ“ according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled „History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled „History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the „History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled „Acknowledgements" or „Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled „Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled „Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled „Endorsements“, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled „History“ in the various original documents, forming one section Entitled „History“; likewise combine any sections Entitled „Acknowledgements“, and any sections Entitled „Dedications“. You must delete all sections Entitled „Endorsements“.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an „aggregate“ if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled „Acknowledgements“, „Dedications“, or „History“, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License „or any later version“ applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents
To use this License in a document you have written, include a copy of

the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME.
>
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled „GNU Free Documentation License“.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the „with...Texts.“ line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.