Unisens 2.0

Dokumentation

Jörg Ottenbacher, Malte Kirst

9. September 2010

Übersicht

Unisens ist ein universelles Datenformat für das Abspeichern, die Dokumentation und das Austauschen von Sensordaten. Unisens wurde speziell dafür entwickelt, semantisch zusammenhängende Sensordaten von unterschiedlichen Sensoren in einem einzigen Datensatz zu speichern, auch wenn die einzelnen Sensoren technisch unterschiedlich zu behandeln sind. Unisens fokussiert sich auf Vitaldaten, schränkt sich aber nicht darauf ein.

Ein Beispiel: Beim cardiovaskulären Langzeit-Monitoring fallen unterschiedliche Daten an: Es wird kontinuierlich ein zweikanaliges EKG aufgezeichnet und mit einer Abtastrate von 200 Hz gespeichert. Ein Algorithmus erkennt zur Laufzeit die Herzschläge und notiert diese Samplegenau zur EKG-Aufnahme. Das Langzeit-Blutdruckmessgerät misst etwa alle 30 Minuten den Blutdruck und speichert den systolischen und diastolischen Wert. Ein SpO_2-Sensor speichert den kontinuierlich gemessenen Sauerstoffgehalt des Blutes mit einer Abtastfrequenz von 256 Hz.

In diesem Beispiel wird schnell klar: Aufgrund der unterschiedlichen Abtastraten und der Mischung von kontinuierlichen und ereignisorientierten Daten ist eine konventionelle Speicherung in einer Datei nicht sinnvoll. Bei einem Aufsplitten in unterschiedliche Dateien geht schnell der semantische Zusammenhang verloren. Unisens speichert die Daten in unterschiedlichen Dateien innerhalb eines Datensatzes und stellt den Zusammenhang durch eine übergeordnete Meta-Datei her.

Begriffe

Um Missverständnisse zu vermeiden, wird an dieser Stelle die Bedeutung der in diesem Dokument verwendeten Begriffe beschrieben.

Datensatz (Dataset) Zusammenhängende Einheit aus einer Header-Datei und einer beliebigen Anzahl von Datendateien und optional einer Kontextinformationsdatei. Ein Datensatz bezieht sich immer auf eine Messung.

Header-Datei (Unisens-Datei) Die Header-Datei eines Datensatzes heißt immer unisens.xml. Sie ist eine XML-Datei, die Informationen über Inhalt und Format der Datendateien enthält.

Datendatei (Data Entry) Überbegriff für eine Datei, die Signal-, Ereignis- oder Einzelmesswertdaten enthält.

Signal (Signal) Kontinuierlich mit beliebiger, aber fester Samplerate aufgezeichnetes Messsignal (z.B. das EKG). Ein Signal kann mehrere Kanäle enthalten.

Ereignisse (Events) Zeitdiskrete Ereignisse, bestehend aus einem Zeitstempel und einem Ereignis (z.B. ein QRS-Trigger).

Einzelwerte (Values) Mischung aus Sinal und Ereignis: Zeitdiskrete Messwerte, die aber in mehreren Kanälen vorliegen können (z.B. Blutdruck). Jeder Einzelwert besteht aus dem Messwert und einem Zeitpunkt.

Kanal (Channel) Untergruppierung eines Signals oder eines Messwerts. Alle Kanäle eines Signals bzw. eines Messwerts haben dieselbe Quantisierung und dieselbe Abtastrate.

Datentyp (Datatype) Unterstützt werden folgende Datentypen: uint8, int8, uint16, int16, uint32, int32, float, double.

Dateiformat (File Format) Datendateien können in unterschiedlichen Dateiformaten abgelegt werden. Definierte Dateiformate sind CSV (Character Separareted Values), BIN (Binär-Format) und XML (Extensible Markup Language).

Kontextinformation (Context) Anwendungsspezifische Zusatzinformationen zum Umfeld einer Messung (z.B. Personendaten).

Proprietäre Daten (Custom Data) Anwendungsspezifische Daten, die zu einer Messung gehören, aber in Unisens nicht durch Signale, Ereignisse oder Einzelwerte abgedeckt sind.

Klasse (Content Class) Jede Datendatei wird genau einer Klasse zugeordnet. Die Klasse bezieht sich auf den Dateninhalt, die eigentliche Information.

Gruppe (Group) Ereignisse, Signale und Messwerte können zu einer Gruppe zusammengefasst werden. Eine Gruppe spiegelt den semantischen Zusammenhang wider.

Beschreibung

Eigenschaften und Funktionen

Freie Kennzeichnung von Inhalten (Art der Daten, Umgebung, Ursprung) in einem menschenlesbaren Format In den Datendateien werden nur die aufgezeichneten Signaldaten gespeichert. Alle weiteren Informationen zur Aufnahme und zum Aufnahmesystem oder einfache Kommentare zum Signal werden in der Header-Datei im menschenlesbaren und -editierbaren XML-Format gespeichert.

Unterstützung von Signaldaten, Ereignisdaten und Bereichsdaten Das Datenformat kann kontinuierliche aufgezeichnete Sensordaten und einzeln aufgezeichnete Ereignisse in einem Datensatz vereinen. Daten unterschiedlichen Typs werden in getrennten Dateien gespeichert, der Zugriff erfolgt aber über das gleiche Interface.

Beliebige Anzahl von Datendateien pro Datensatz Die Anzahl der Datendateien (Signale, Ereignisse und Einzelwerte) pro Datensatz ist nicht begrenzt. In Bereichen wie z. B. der Polysomnographie ist es nötig, sehr viele unterschiedliche Signale und Ereignisse parallel aufzuzeichnen. Hier kann jeder der benutzen Sensoren seine Daten in einer eigenen Datendatei abspeichern. In der Header-Datei werden alle Datendateien aufgelistet und in einen gemeinsamen Kontext gebracht.

Klassenbildung für Daten gleichen Typs Daten des gleichen Typs (z. B. EKG, Beschleunigung, EKG-Trigger) werden zu einer Klasse zusammengefasst. Damit wird beispielsweise einem EKG-Darstellungswerkzeug ermöglicht, die EKG-Daten eines Datensatzes zu identifizieren. Die Klasse wird im Attribut `contentClass` gespeichert. Die Klassennamen können frei gewählt werden, es wird jedoch mpfohlen, sich an die auf S. 7 stehenden Richtlinien zu halten.

Eine Datenklasse kann in einem Datensatz mehrfach vorhanden sein. Somit können in einem Datensatz bearbeitete und unbearbeitete Daten eines Signal enthalten sein (z. B. gefilterte EKG-Daten und EKG-Rohdaten oder automatisch erstellte Triggerliste und Referenztriggerliste)

Eindeutige Identifizierbarkeit einer Datendatei innerhalb eines Datensatzes Über die `entryId` ist eine Datendatei eindeutig innerhalb des Datensatzes identifizierbar.

Unterstützung von Daten mit verschiedener Abtastraten und Abtasttiefen Innerhalb eines Datensatzes können Daten mit unterschiedlicher Abtastrate und Abtasttiefe gespeichert werden. Die Daten werden in getrennten Dateien abgelegt.

Mehrkanalige Aufnahmen möglich Signale und Einzelwerte können mehrere Kanäle besitzen, diese müssen jedoch in Abtastastrate und Abtasttiefe übereinstimmen.

Keine Einschränkung der Größe der Daten durch die Spezifikation Die Größe der Daten ist lediglich durch das Filesystem und den Datenträger begrenzt. Übersteigt die Anzahl der gespeicherten Samples 2^{63}, kann die Referenzimplementierung die Datei nicht mehr vollständig lesen.

Möglichkeit der Beschreibung von Daten Alle Signal-, Ereignis- und Bereichsdaten sowie alle Kanäle können mit einem Kommentar versehen und damit verbal beschrieben werden.

Unterstützung verschiedener Dateiformate für Datendateien Für Datendateien können unterschiedliche Dateiformate verwendet werden (Binär, XML, CSV). Für jedes Dateiformat ist eine Referenzimplementierung in Java vorhanden, die Daten lesen und schreiben

Samplegenaue Angabe von Ereignissen und Bereichen Alle Zeitpunkte können samplegenau angegeben werden. Zu jeder Datendatei wird in `sampleRate` eine Samplerate angegeben, die als Zeitbasis für diese Datendatei gilt. Die einzelnen Datendateien innerhalb eines Datensatzes können unterschiedliche Sampleraten besitzen.

Samplegenaue Unterstützung von Signalen größer und kleiner 1 Hz Die Samplerate kann sowohl ganzzahlig oder als Dezimalzahl angegeben sein. Dezimaltrenner ist in jedem Fall ein Punkt.

Umgang mit proprietären Daten Innerhalb des Datensatzes können proprietäre Daten abgelegt und gekennzeichnet werden.

Gruppierungsfunktion Datendateien können zu inhaltlichen Gruppen zusammengefasst werden. So kann z. B. ein EKG mit der zugehörigen Annotation gruppiert werden.

Unterstützung von aufnahmespezifischen Kontextinformationen Aufnahmespezifische Kontextinformationen wie z. B. Patientendaten bei der Aufzeichnung von Vitalsignalen können als XML-Datei im Datensatz abgelegt werden, inklusive der Information über das zugehörige XML-Schema. Das Interface unterstützt das Parsen dieser Kontext-Datei und das Laden der zugehörigen Schema-Datei.

Einschränkungen

Einschränkungen durch das Filesystem Dateinamen und damit auch die Namen von Entries sind durch das Filesystem eingeschränkt.

Einschränkungen durch Ressourcen Der Umfang eines Unisens-Datensatzes wird sowohl durch die Größe des Datenträgers als auch durch das Betriebssystem vorgegebene maximale Dateigröße beschränkt.

Einschränkungen durch Datentypen Eine maximale Datei- oder Datensatzlänge ist nicht spezifiziert, jedoch durch Datentypen gegeben. Mit dem Erreichen von 2^{63} Samples ist die adressierbare Länge eines Datensatzes erschöpft.

Datenformat

Modell

Ein Unisens-Datensatz lässt sich als Container auffassen, der verschiedene Dateneinträge (Entries) sowie einen Abschnitt mit Kontextinformationen enthalten kann. Abbildung 1 verdeutlicht diese Struktur und listet die möglichen Entries auf.

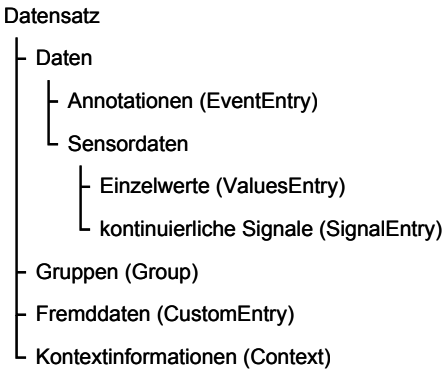

Abbildung 1: Aufbau eines Unisens-Datensatzes

Ein Daten-Entry enthält die Messdaten eines Sensors oder eine Liste von Ereignissen. Jedes Daten-Entry hat eine eigene Zeitbasis (Abtastrate) und bekommt eine Inhalts-Klasse (ContentClass) zugewiesen (z. B. EKG, Pleth, Blutdruck etc.). Die ContentClass ist frei wählbar, sollte aber möglichst aus einer vorgegeben Liste passend ausgewählt werden. Sie hilft bei der automatischen Identifikation von Datensätzen. Sensormessdaten können als quasikontinuierliches Signal mit fester Abtastrate aufgezeichnet (`signalEntry`) werden, so wie es beispielsweise bei einem EKG der Fall wäre. Genauso können aber auch zeitdiskrete Werte erfasste werden (`valueEntry`) , wie z. B. bei einer oszillometrischen Langzeit-Blutdrucküberwachung. Sensordaten werden mit allen wichtigen Parametern des Sensors gespeichert (Informationen über A/D-Wandler, Abtastrate, Quantisierung etc.) und können mehrere Kanäle enthalten. Ein Daten-Entry mit Annotationen (`eventEntry`) entspricht einer Liste von Ereignissen, die in Textform beschrieben werden können und über einen Zeitstempel und einen Ereignistyp verfügen.

Alle Daten-Entries können beliebig zu Gruppen zusammengefasst werden. Damit kann z. B. einem EKG eine zugehörige QRS-Triggerliste oder einer Aktivitätsaufnahme die Liste der durchgeführten Aktivitäten zugeordnet werden.

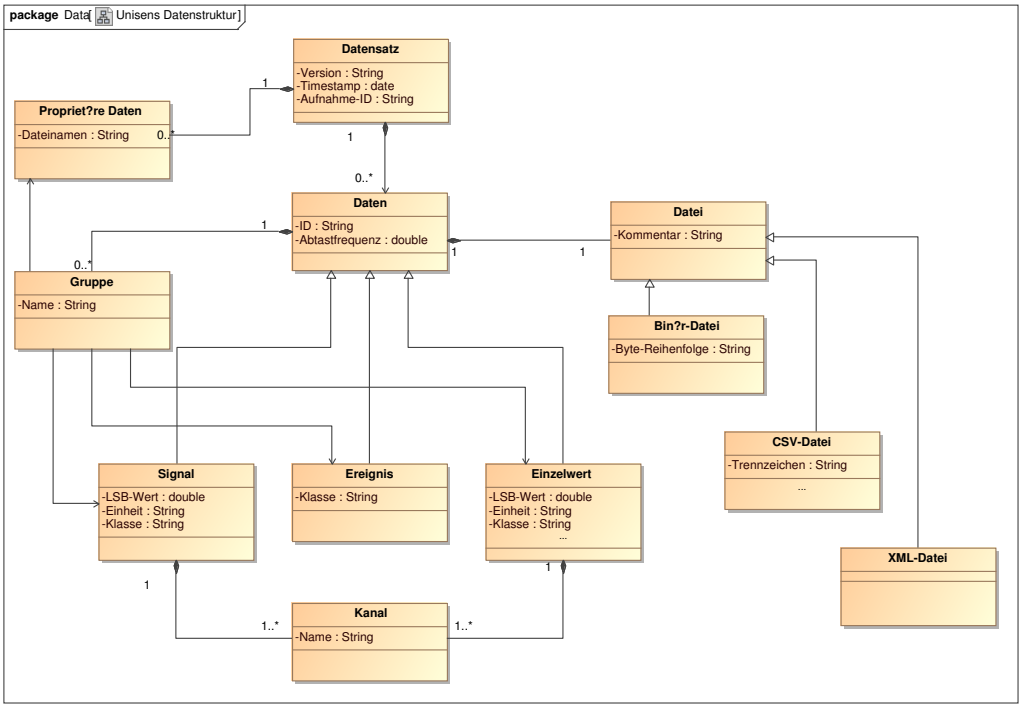

Abbildung 2: Struktur eines Unisens-Datensatzes

Aufbau

Ein Unisens-Datensatz besteht aus einem Ordner im Dateisystem. Alle Meta-Informationen zum gesamten Datensatz bzw. zu jedem einzelnen Entry stehen in einer Header-Datei. Für diese Header-Datei wurde das XML-Format gewählt, die Struktur ist in einem XML-

Schema vorgegeben. Der Name der Header-Datei ist immer unisens.xml. Der Name des Datensatzes entspricht dem Ordner-Namen.

Die Header-Datei, das Kernstück eines Unisens-Datensatzes, verweist auf alle zugehörigen Entries, die in eigenen Dateien abgelegt sind. Ein Entry wird dabei über seine `entryId` als Primärschlüssel identifiziert, die dem Dateinamen entspricht. In Abbildung 3 ist die Dateistruktur eines Datensatzes gezeigt.

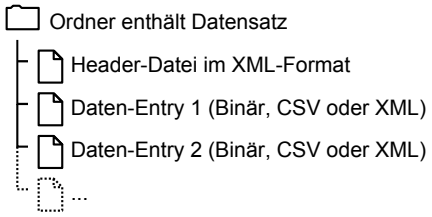

Abbildung 3: Repräsentation eines Unisens-Datensatzes im Dateisystem

Für jedes Daten-Entry existiert eine eigene Datei, welche in verschiedenen Formaten vorliegen kann. Spezifiziert sind die Formate XML und CSV sowie Binärdateien. Der Anwender kann sich beim Speichern der Daten für ein Dateiformat entscheiden. Für kontinuierlich aufgezeichnete Sensordaten bietet sich ein Binärformat an, da dieses auch bei langen Datensätzen noch von akzeptabler Größe bleibt. Für Binärdateien werden unterschiedliche Datentypen unterstützt, unter anderem 32-Bit Integer-Werte (Int32) und Fließkommazahlen (Double). Bei Annotationen ist das CSV-Format empfehlenswert, da sich diese Dateien auch ohne die Unisens-Bibliothek komfortabel lesen und bearbeiten lassen.

Gruppen werden innerhalb der Header-Datei als XML gespeichert. Die Gruppierung erfolgt über eine Liste der EntryIds der zu gruppierenden Einträge. In einer Beschreibung kann die Gruppe textuell umschrieben werden.

Die modulare Organisation der Daten-Entries in einzelnen Dateien ermöglicht es, einen Datensatz jederzeit schnell und einfach durch neue Entries zu erweitern oder bestehende Entries zu entfernen. Damit ist es möglich, zusätzliche Signale zu speichern, die durch die Bearbeitung der Sensorrohdaten entstehen.

Dateiformate

Bereits spezifiziert und in der Referenzimplementierung umgesetzt ist ein binäres Dateiformat sowie ein CSV-Dateiformat. Die Unterstützung von XML-Dateien ist vorgesehen, aber noch nicht abschließend implementiert. Jedes andere Dateiformat kann als `customEntry` mit einem `customFileFormat` eingebunden werden. Tabelle 1 listet die Dateiformate auf.

Tabelle 1: Dateiformate

Name	Dateiendung	Attribute
`binFileFormat`	`bin`	Byte-Reihenfolge, Datentyp
`csvFileFormat`	`csv`	Trennzeichen, Dezimaltrennzeichen
`xmlFileFormat`	`xml`	Dezimaltrennzeichen
`customFileFormat`	beliebig	Dateityp

Datentypen

Unisens kann mit den gängigsten Datentypen umgehen. In Tabelle 2 sind die Datentypen aufgelistet.

Tabelle 2: Datentypen

Datentyp	Größe (Byte)	Wertebereich
`DOUBLE`	8	$4.9 \cdot 10^{-324} \ldots 1.7976931348623157 \cdot 10^{308}$
`FLOAT`	4	$1.4 \cdot 10^{-45} \ldots 3.4028235 \cdot 10^{38}$
`INT32`	4	$-2147483648 \ldots 2147483647$
`INT16`	2	$-32768 \ldots 32767$
`INT8`	1	$-128 \ldots 127$
`UINT32`	4	$0 \ldots 4294967295$
`UINT16`	2	$0 \ldots 65535$
`UINT8`	1	$0 \ldots 255$

Aufbau der Metadatei

Die Headerdatei `unisens.xml` ist im menschenlesbaren XML-Format geschrieben. Die verwendete XML-Schema-Definition (XSD) ist unter [KO08] einzusehen.

Empfehlungen

Klassen

Für die Werte des Attributes `ContentClass` sind grundsätzlich frei wählbar, es wird jedoch dringend empfohlen, sich an die in Tabelle 3 aufgelisteten Klassenbezeichner zu halten, sofern dieses möglich ist.

Tabelle 3: Empfehlung für die Anwendung des Attributes `contentClass`

Klasse	Beschreibung
ECG	EKG-Signal
TRIGGER	Trigger-Liste (EKG-Annotation)
ACC	Beschleunigungssignal
IMP	Impedanzsignal
RAW	Rohdaten
BLOODPRESSURE	Blutdruck
PLETH	Pleth-Signal
RESP	Atmung
MARKER	Patienten-Marker

Dateinamen

Die IDs von Datendateien sollten nur aus den Buchstaben `A-Z`, den Ziffern `0-9` sowie aus den Zeichen `_` und `-` bestehen. Die Länge der IDs ist durch das Dateisystem vorgegeben – bei einem FAT16-System sind dies lediglich acht Zeichen.

Anwendung

Unisens-Dateien können direkt auf dem Embedded System des aufzeichnenden Sensors geschrieben werden. Ebenso können bereits vorhandene Daten nach Unisens konvertiert werden. Die vorhande Java-Bibliothek erlaubt die Nutzung aus Java-Programmen sowie die direkte Einbindung in MATLAB.

Um den Funktionsumfang der Unisens-Library vollständig nutzen zu können, bestehen folgende Anforderungen:

- Java 5 oder höher
- Unisens Interface `org.unisens.jar`
- Referenzimplementierung der Unisens Library `org.unisens.ri.jar`

Verzichtet man auf den Komfort der Bibliothek, können Unisens-Dateien auf nahezu jedem System erstellt werden. Durch die menschenlesbare Header-Datei im XML-Format ist ein editieren per Hand möglich, ebenso können CSV- und Binär-Dateien mit Standardwerkzeugen erstellt werden.

Embedded Systems

Soll Unisens in einem Embedded System zum Speichern von Sensordaten verwendet werden, ist es zweckmäßig, die `unisens.xml`-Datei aus einer Vorlage zu generieren, in der nur noch der Timestamp für den Beginn der Messung und ggf. der Kommentar geändert wird. Die Daten des Sensors können direkt als Binärdatei im Ordner des Datensatzes gespeichert werden. Die Eigenschaften des verwendeten AD-Wandlers sowie das Datenformat der Binärdatei sind Systemabhängig und können entsprechend in der XML-Vorlage berücksichtigt werden.

Java

Die Java-Referenzimplementierung erlaubt die direkte Nutzung von Unisens innerhalb einer Java-Anwendung. Die API ist in englischer Sprache dokumentiert und auf [KO08] verfügbar. Ebenso wird dort ein leicht verständliches Beispiel vorgehalten, in dem das Erstellen, Lesen, Bearbeiten und Speichern von Unisens-Daten ausführlich beschrieben wird.

Matlab

Alle MATLAB-Versionen, die Java 5 oder höher verwenden, können Unisens uneingeschränkt verwenden. Dazu muss lediglich das Unisens-Interface und dessen Implementierung dem Java-Pfad von MATLAB hinzugefügt werden. Eine Reihe von zur Verfügung gestellten Standardfunktionen erleichtert den Umgang in MATLAB. In [KO08] ist die Verwendung von Unisens in MATLAB ausführlich erklärt.

Erweiterbarkeit

CustomAttributes

Die einfachste Art der Erweiterung ist das Erstellen eigener Attribute (`customAttributes`). Mit Hilfe von Key-Value-Paaren können beliebige Informationen im Datensatz gespeichert werden. Sie können wahlweise global für den gesamten Datensatz spezifiziert werden oder nur für einzelne Datendateien.

`customAttributes` sind Bestandteil von Unisens und in der Referenzimplementierung enthalten.

Context

Das Speichern von Kontextinformationen kann auch in einer separaten XML-Datei mit selbst vorgegebener Struktur erfolgen (`context.xml`). In diesem Fall muss ein entsprechendes XML-Schema mitgeliefert werden, das diese Datenstruktur beschreibt. Aus der Headerdatei wird über das Element `context` darauf verwiesen.

CustomEntries

Sollen Dateien innerhalb eines Datensatzes gespeichert werden, die sich nicht als eines der vorgesehenen Datentypen darstellen lassen, kann ein `customEntry` angelegt werden. Innerhalb eines `customEntry` wird auf den Namen der Datei verwiesen, zusätzlich können bestimmte Informationen (z.B. Datentyp, Kommentar u.ä.) als Attribut gespeichert werden. Ein Beispiel für ein `customEntry` wäre ein Foto, dass das Messobjekt zeigt und somit durchaus ein wichtiger Bestandteil der Messung sein kann, sich aber nicht anders in das Datenformat integrieren lässt. Ebenso können Anwendungen Zusatzinformationen zu einem Datensatz in einem `customEntry` speichern.

Eigene Implementierung

Mit einer eigenen Implementierung der Library hat man Möglichkeiten, Unisens kompatibel zu erweitern. Solange das Interface korrekt implementiert wird, ist gewährleistet, dass alle Unisens-Daten korrekt geöffnet werden können. Eigene Erweiterungen wie z.B. andere Lese-Funktionen oder Signalverarbeitung wie zum Beispiel Up- und Downsampling können so problemlos eingefügt werden.

Andere Datenformate

Im medizinischen Umfeld gibt es bereits etablierte Standards für das Speichern und Austauschen von Daten. Als prominenteste Vertreter seien hier das DICOM-Format für den Austausch digitaler Bilder oder HL7 bzw. GDT für den weiteren Austausch von Information und Daten genannt [NEM08, Hea08]. Für die Übertragung von Daten sind verschieden Protokolle empfehlenswert, etwa VITAL (ISO 11073) oder IMEX / MSD [SFWS07, Int05].

Will man die Daten speichern, bearbeiten, umsamplen oder annotieren, ist sowohl das WFDB-Format als auch das European Data Format (EDF) und dessen Ableger etabliert [Moo08, KO03, KVR+92]. Zudem existieren bereits viele öffentlich zugängliche Daten in

diesen Formaten. Aufgrund dessen ist es angestrebt, einfache Konvertierungsmethoden zur Verfügung zu stellen, die vorhandene Daten verlustfrei in das Unisens-Format überführen.

Dateien anderer Datenformate können als `customEntry` einem Unisens-Datensatz hinzugefügt werden.

Lizenz

Die Unisens-Library und das Unisens-Interface stehen unter der GNU Lesser General Public License (LGPL). Der Lizenztext wird mit dem Code ausgeliefert und ist auf [FSF07] einsehbar.

Literatur

[FSF07] Free Software Foundation, Inc.: *GNU Lesser General Public License*. 2007. URL `http://www.gnu.org/licenses/lgpl-3.0.txt`. – Zugriffsdatum 2008-03-18

[Hea08] Health Level Seven, Inc. *HL7 – Health Level 7*. http://www.hl7.org/. 2008

[Int05] International Electrotechnical Commission (IEC). *Micro System Data Format Specification*. 2005

[KO03] Kemp, B. ; Olivan, J.: European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data. In: *Clin Neurophysiol* 114 (2003), Sep, S. 1755–1761

[KO08] Kirst, Malte ; Ottenbacher, Jörg. *Unisens*. http://www.unisens.org. 2008

[KVR+92] Kemp, B. ; Värri, A. ; Rosa, A.C. ; Nielsen, K.D. ; Gade, J.: A simple format for exchange of digitized polygraphic recordings. In: *Electroencephalogr Clin Neurophysiol* 82 (1992), May, S. 391–393

[Moo08] Moody, George B.: *WFDB Programmer's Guide : Tenth Edition (revised and with additions for WFDB library version 10.4.5)*. Harvard-MIT Division of Health Sciences and Technology, 2008

[NEM08] NEMA. *DICOM – Digital Imaging and Communications in Medicine*. http://dicom.nema.org/. 2008

[SFWS07] Schmitt, L. ; Falck, T. ; Wartena, F. ; Simons, D.: Novel ISO/IEEE 11073 Standards for Personal Telehealth Systems Interoperability. In: *High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability, 2007. HCMDSS-MDPnP. Joint Workshop on*, 2007, S. 146–148

Kontakt

FZI Forschungszentrum Informatik
Embedded Systems and Sensors Engineering (ESS)
Haid-und-Neu-Str. 10–14
76131 Karlsruhe

Dipl.-Ing. Malte Kirst
kirst@fzi.de

Universität Karlsruhe
Institut für Technik der Informationsverarbeitung (ITIV)
Engesserstr. 5
76131 Karlsruhe

Dipl.-Ing. Jörg Ottenbacher
ottenbacher@itiv.uni-karlsruhe.de

http://www.unisens.org