|  |  |
| --- | --- |
| Projekt Middleware (Dispatcher) |  |
| Schnittstellenbeschreibung v0.1  Locsim – Interlocking-Dispatcher | Autor / Referenzperson  Sebastian Straube  desm@it-hotspot.de  Mobile: +41 79 4452 127 |

Inhaltsverzeichnis

[2 ToDo 3](#_Toc343206810)

[3 Kommunikation 4](#_Toc343206811)

[3.1 Ethernet 4](#_Toc343206812)

[3.2 Kommunikationsmodell 4](#_Toc343206813)

[3.3 Asynchrone Übertragung 4](#_Toc343206814)

[3.3.1 Libraries 4](#_Toc343206815)

[3.4 Kommunikationsprotokoll 5](#_Toc343206816)

[3.4.1 Synchronisation 5](#_Toc343206817)

[4 Dynamic Link Library (DLL) 6](#_Toc343206818)

[4.1 Funktionssicht 6](#_Toc343206819)

[4.2 Funktionen der DLL 6](#_Toc343206820)

[4.2.1 DLL laden (initialize meta infos) 6](#_Toc343206821)

[4.2.2 DLL ist geladen 7](#_Toc343206822)

[4.2.3 Simulation Transition (Start, Aufbau und Ende) 7](#_Toc343206823)

[4.2.4 Simulation gestartet 10](#_Toc343206824)

# ToDo

* Interface von Dispatcher nach Locsim definieren (alle sich ändernden Parameter)

# Kommunikation

## Ethernet

Die Verbindung zwischen dem Dispatcher und dem Locsim wird per Ethernet hergestellt. Aus modularen gründen und der Wiederverwendbarkeit, wird in der DLL ein Server und ein Proxy für die Datenübertragung bereitgestellt. Dadurch ist es für den Locsim und dem Dispatcher möglich, mit der gleichen DLL eine Verbindung über das Netzwerk herzustellen.

## Modell

## Asynchrone Übertragung

Die Daten werden vom Locsim per Poll abgeholt. Der Dispatcher soll diese Daten per IRQ oder „state Poll“ abholen.

## Libraries

Es stehen verschiedene Techniken zur Verfügung, die Daten über eine Ethernet Verbindung auszutauschen.

Es wurden folgende Techniken evaluiert.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protokollname** | **Beschreibung** | **Vorteil** | **Nachteil** |
| RAW ([TCP](http://www.codeproject.com/Articles/13071/Programming-Windows-TCP-Sockets-in-C-for-the-Begin)) | Eine „Winsocket“ Client-Server Übertragung. Das Übertragunsprotokoll soll lediglich ByteStreams empfangen und senden können, weil nur XML Daten transferiert werden sollen und keine RAW Datentypen. Dafür muss ein Magic Packet (Initialisierung) definiert werden. | * schnell * Abstraktion durch XML Struktur | * integration kompliziert * eigenes Übertragungsprotokoll nötig |
| HTTP (high level) & [Json](http://www.json.org/json-de.html) | verschiedene libraries sind u.A. boostASIO, libHTTP, libEvent, libOV | * High Level Integration * ermöglich Steuerung über Webinterface | * Overhead gross * keine permanente Verbindung * zwei Webserver |
| [Open Sound Protokoll](http://gruntthepeon.free.fr/oscpkt/html/oscpkt__demo_8cc-example.html) |  | * library Unterstützung sehr umfangreich | * proprietäre Implementierung |

## Protokoll

## Synchronisation

Die Synchronisation wird über ein Status Flag gesteuert. Sobald Daten im DLL Cache verändert werden, wird das Status Flag auf „dirty“ gesetzt. Ausserdem wird durch eine „lock\_id“ die Synchronisationsrichtung gesteuert.

# Dynamic Link Library (DLL)

Diese DLL wurde als „unmanaged“ Code in C++ geschrieben. Zum Laden der DLL wird kein MFC benötigt. Auf die Definition als COM Komponente wurde aus Vereinfachungsgründen verzichtet. Die Daten in der DLL werden in einem Cache gehalten und von dort weitergegeben oder abgeholt.

## Funktionssicht

define: locsim => stellwerk (cached in DLL)

get:stellwerk (cached in DLL) => locsim

## Funktionen der DLL

Der Zeitpunkt des Zugriffs auf bestimmte DLL Funktionen ist durch gewisse Anwendungsstatus des Locsim gegeben.

### DLL laden (initialize meta infos)

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Beim Laden der DLL library, wird der Netzwerkserver durch diese Funktion gestartet. | |
| **seit Version** | 0.1 | |
| **Signatur** | bool stw\_onloadLibrary() | |
| **Attribute** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Beim Entaden der DLL library, wird der Netzwerkserver durch diese Funktion geschlossen und gestoppt. | |
| **seit Version** | 0.1 | |
| **Signatur** | bool stw\_onunloadLibrary() | |
| **Attribute** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **~~Funktionsbeschreibung~~** | * ~~Es braucht eine zusätzliche Funktion stw\_reset(), ohne Parameter, welche beim Neuladen einer Strecke alle Daten in der Stellwerk-dll löscht.~~ | |
| **~~seit Version~~** |  | |
| **~~Signatur~~** | ~~int stw\_reset ()~~ | |
| **~~Attribute~~** |  |  |
| **~~Rückgabepointer~~** |  |  |
| **~~Rückgabewert (OK)~~** | ~~0~~ | ~~i.o.~~ |
| **~~Rückgabewert (ERROR)~~** | ~~err-stw\_000.000.000.000~~ |  |

### DLL ist geladen

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Versionskonflikt Behandlung der DLL | |
| **seit Version** | 0.1 | |
| **Signatur** | void stw\_get\_version() | |
| **Attribute** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

### Simulation Transition (Start, Aufbau und Ende)

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * markiert Simulation Start * impliziert das Event „Strecke neu laden“ * es werden mögliche „defines“ erwartet | |
| **seit Version** | 0.1 | |
| **Signatur** | void stw\_init() | |
| **Attributbeschreibung** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * markiert das Ende der Simulation * erwartet danach „gets“ Funktionen oder die „define\_trainposition“ Funktion | |
| **seit Version** | 0.1 | |
| **Signatur** | void stw\_ready() | |
| **Attributbeschreibung** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Definition von zu Hauptgleis parallelen Gleisabschnitten * Was ist das Hauptgleis, wie wird es definiert? (eingestellte Fahrstrasse? Regelgleis?) | |
| **seit Version** | 0.1 | |
| **Signatur** | int define\_track (int gleisId, double von, double bis, float abstand, char\* name) | |
| **Attributbeschreibung** | gleisId  von  bis  abstand  name | locsim-interne ID  als Meterangabe  als Meterangabe  Abstand von Hauptgleis(Meter) - pos. Rechts, neg. Links  Gleisnummer gemäss Sicherungsanlage, z.B. A1 (kann auch leer sein) |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Definition von Verbindungen zwischen parallelen Gleisabschnitten * weiche1\_id, weiche2\_id = ID der Weiche (eindeutige locsim-interne Nummerierung), wenn 0: keine Weiche (wenn z.B. connection2 in gleis2 übergeht) * pro Längsposition dürfen max. 20 verschiedene Gleise definiert sein (ID= 1…20) | |
| **seit Version** | 0.1 | |
| **Signatur** | int define\_track\_connection (int gleis1, int gleis2, double von, double bis, char\* name, int weiche1\_id, int weiche 2\_id) | |
| **Attributbeschreibung** | gleis1  gleis2  von  bis  name  weiche1\_id  weiche2\_id | von gleis  nach gleis  Positionsangabe der Gleise als Meterangabe  Positionsangabe der Gleise als Meterangabe  Gleisnummer gemäss Sicherungsanlage, z.B. A1 (kann auch leer sein)  ID der ersten Weiche  ID der zweiten Weiche |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * wird pro Signal aufgerufen | |
| **seit Version** | 0.1 | |
| **Signatur** | int define\_signal (int signal\_id, int gleis\_id, double position, int typ, float hoehe, float distanz, char\* name) | |
| **Attributbeschreibung** | signal\_id  gleis\_id  position  typ  hoehe  distanz  name | eindeutige Nummer (willkürlich)  interne Gleisnummer  Geoposition des Signals  der Typ des Signals  die Höhe des Signals  die Distanz des Signals zum Gleis  name des Signals |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * definiert eine Balise auf einem bestimmten Gleis an einer bestimmten Position | |
| **Signatur** | int define\_balise (int gleis\_id, double position, int balise\_id) | |
| **Attributbeschreibung** | gleis\_id  position  balise\_id | eindeutige Gleis ID  eindeutige Nummer (willkürlich)  Wirkungsrichtung, 1=vorwärts, -1=rückwärts |
| **Rückgabewert (Attribut)** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Ist in den locsim-Streckendaten bis jetzt nicht drin, könnte aber dazugefügt werden | |
| **seit Version** | 0.1 | |
| **Signatur** | int define\_isolierstoss (int gleis, double position) | |
| **Attributbeschreibung** |  |  |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

### Simulation gestartet

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Funktion: Locsim fragt DLL, welche events vom Stellwerk vorhanden sind (DLL cached) * jedes Event führt zum Aufruf einer der nachstehenden Funktionen * stw\_get\_event(3, typeList(1,1,2), idList(63, 32 , 765) ) | |
| **seit Version** | 0.1 | |
| **Signatur** | int get\_events(int\* number, int\*\* type\_list, int\*\* id\_list) | |
| **Attributbeschreibung** | number  type\_list  id\_list | Anzahl Events (Arraygrösse): 0=nichts  types = 1 (Signal), 2 (Balise), 3 (Weiche) as Array list  id = id from type as Array list |
| **Rückgabepointer** | int\* number | als Integer Pointer |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * gibt die Stellung eines bestimmten Signals zurück | |
| **seit Version** | 0.1 | |
| **Signatur** | int get\_signal (int signal\_id, int\* stellung) | |
| **Attributbeschreibung** | signal\_id  stellung | Stellung gemäss help\locsimmanualsignal-d.htm |
| **Rückgabepointer** | int\* stellung | als Integer Pointer |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * gibt eine Stellung und das Protokoll einer bestimmten Balise zurück | |
| **seit Version** | 0.1 | |
| **Signatur** | int get\_balise (int balise\_id, int\* stellung, char\*\* protokoll) | |
| **Attributbeschreibung** | balise\_id |  |
| stellung | gemäss locsimmanualsignal-d.htm, „Zugbeeinflussung durch Signale“, „v“,  ausser -3001…-7000; wenn=-9998 => protokoll |
| protokoll | gemäss help\zugsicherungen.txt, wenn stellung ungleich -9998: leer |
| **Rückgabepointer** | int stellung  char\*\* protokoll | ist rückgabewert, als Integer Pointer  ist rückgabewert, als char Doppelpointer (String Array) |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * gibt eine bestimmte Weiche zurück | |
| **seit Version** | 0.1 | |
| **Signatur** | int get\_weiche (int weiche\_id, int\* gleis\_id) | |
| **Attributbeschreibung** | weiche\_id  gleis\_id | ID gemäss define\_track\_connection  Gleisnummer der Stellung (im stumpfen Bereich) |
| **Rückgabewert (Attribut)** | int\* gleis\_id | ist rückgabewert, als Integer Pointer |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Übergibt pos1 – gleis1 – pos2 – gleis2 – pos3 - … des Zuges train an stellwerk.dll wenn die Zugspitze oder der Zugschluss einen Isolierstoss überfährt * Ein Track ist kein Gleis und umgekehrt, weil es kein Hauptgleis ist? * Was bedeutet in diesem Fall die Position? * Warum die Position + 1 als Anzahl Gleise? * Anzahl pos = Anzahl gleis +1 | |
| **seit Version** | 0.1 | |
| **Signatur** | int define\_trainposition(int train, int direction, double array positionList, int array gleisList) | |
| **Attributbeschreibung** | direction  train  posistionList  gleisList | 1=vorwärts, -1=rückwärts  0=simulierter Zug, 1 und weitere=andere Züge |
| **Rückgabepointer** |  |  |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Speicher Allocation and Deallocation muss in DLL und von Locsim aufgerufen warden, sonst tritt ein memory-leak auf. | |
| **seit Version** | 0.1 | |
| **Signatur** | void stw\_deallocate(void\*) | |
| **Attributbeschreibung** |  |  |
| **Rückgabepointer** | void\* | pointer auf memory |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |

|  |  |  |
| --- | --- | --- |
| **Funktionsbeschreibung** | * Speicher Allocation and Deallocation muss in DLL und von Locsim aufgerufen werden, sonst tritt ein memory-leak auf. | |
| **seit Version** | 0.1 | |
| **Signatur** | void stw\_deallocate\_array(void\*) | |
| **Attributbeschreibung** |  |  |
| **Rückgabepointer** | void\* | pointer auf memory |
| **Rückgabewert (OK)** | 0 | i.o. |
| **Rückgabewert (ERROR)** | err-stw\_000.000.000.000 |  |