Traffic Data Acquisition Daemon (TDAD)

Software zur Verkehrsdatenerfassung
Revision 1.04 / August 2008

EETS GmbH
Electronic Equipment for Traffic Systems

EETS GmbH
Chaltenbodenstrasse 4
CH-8834 Schindellegi
Switzerland

Traffic Data Acquisition Daemon (TDAD)

Revision 1.04

Inhaltsverzeichnis

1 Übersicht

Die Software *Traffic Data Acquisition Daemon* dient der Erfassung und Aggregation von Verkehrsdaten. Der *TDAD* ist für die Betriebssysteme LINUX und Microsoft Windows erhältlich. Das vorliegende Dokument beschreibt jedoch ausschliesslich die Konfiguration und den Betrieb von *TDAD* unter LINUX.

2 Zielpublikum

Das vorliegende Dokument richtet sich an System Administratoren und beschreibt die Konfiguration von *TDAD*. Der *TDAD* ist normalerweise Bestandteil eines *VIPER* Gesamtsystems, welches bereits vor der Auslieferung durch die EETS GmbH vollständig konfiguriert wird.

3 Zusammenfassung

3.1 Erfassung der Daten

Mit Hilfe in der Fahrbahn eingelegter induktiver Schleifen und den entsprechenden Detektoren werden einzelne Fahrzeuge erfasst. Je nach Detektor lassen sich Eigenschaften wie Geschwindigkeit, Länge, Belegunsgrad und Fahrzeugtyp erfassen.

Über einen seriellen EIA-485 Bus welcher die Detektoren mit dem Erfassungsrechner verbindet erhält der *TDAD* die Daten der Fahrzeuge. Dabei wird jedes **einzelne** Fahrzeug mit seinen Kenndaten erfasst.

3.2 Aggregation der Daten

Die von den Detektoren erfassten Daten werden vom *TDAD* in die entsprechenden Fahrzeugklassen aufgeteilt und über ein frei wählbares Intervall kumuliert. Dies geschieht für jede Messstelle einzeln. Am Ende jedes Messintervalls werden weitere Daten, wie z.B. der Belegungsgrad der Messstelle errechnet.

3.3 Übergabe der Daten an Drittsysteme

Die vom *TDAD* aggregierten Daten sowie Statusinformationen werden am Ende des Messintervalls den Drittsystemen (Signalsteuerung, Archivierung, usw.) über eine TCP/IP Schnittstelle zur Verfügung gestellt. Dafür wird das Protokoll *TDAP* verwendet.

4 Optionen der Kommandozeile

Das Verhalten von *TDAD* lässt sich mit Parametern, welche auf der Kommandozeile aufzuführen sind, beeinflussen. *TDAD* wird wie folgt aufgerufen:

```
tdad [-V] [-b] [-f <Konfigurationsdatei>]
```

Dabei sind alle Parameter in eckigen Klammern (`[]`) optional. Nicht aufgeführte Parameter erhalten einen Standardwert. Die Parameter haben folgende Bedeutung:

Parameter	Bedeutung	Standardwert
`-V`	Zeigt die Software Version, der *TDAD* wird **nicht** gestartet.	-
`-b`	Ausführung als Daemon im Hintergrund (background). Dies ist die normale Betriebsart.	*TDAD* wird im Vordergrund ausgeführt und kann mit CTRL-C beendet werden.
`-f <Datei>`	Der Name der zu verwendenden Konfigurationsdatei. Die Datei ist mit dem gesamten, absoluten Pfad aufzuführen.	Es wird die Datei `/usr/local/etc/tdad.conf` verwendet.

Der *TDAD* wird für den normalen Betrieb immer mit dem Parameter `-b` gestartet. Auf LINUX Systemen protokolliert *TDAD* mit Hilfe von *syslog* und der facility *daemon*.

5 Konfiguration

Der *TDAD* wird mit einer einzigen Textdatei konfiguriert, welche beim Starten gelesen und verarbeitet wird. Wird die Option `-f` der Kommandozeile nicht verwendet, so lädt *TDAD* die Konfigurationsdatei `/usr/local/etc/tdad.conf`.

Die Konfigurationsdatei besitzt dieselbe Syntax wie .ini Dateien auf MS Windows Betriebssystemen. Kommentare sind immer einzeilig und beginnen mit einem Strichpunkt; alle folgenden Zeichen bis zum Zeilenumbruch werden als Kommentar behandelt.

Die Konfigurationsdatei ist in Mehrere Abschnitte aufgeteilt. Ein Abschnitt besteht aus einem Namen in eckigen Klammern (`[]`). Gross- und Kleinschreibung wird für Abschnittsnamen nicht unterschieden.

Jeder Abschnitt enthält einen oder mehrere Schlüssel, denen ein Wert zugewiesen ist. Ein Schlüssel hat das Format `Schluessel = Wert`, wobei Gross- und Kleinschreibung für den Schlüsselnamen nicht unterschieden werden. Enthält der Wert des Schlüssels den Strichpunkt, so ist der gesamte Wert in doppelte Anführungszeichen (`""`) zu setzen. Die im folgenden **`fett`** ausgezeichneten Schlüssel müssen vorhanden sein.

5.1 Globale Konfiguration

Die globale Konfiguration befindet sich im Abschnitt `[global]`. Dieser Abschnitt enthält Grundeinstellungen des *TDAD* und muss immer vorhanden sein.

Die gültigen Schlüssel für den Abschnitt `[global]` sind der folgenden Tabelle zu entnehmen:

Schlüssel	Bedeutung und Wert
`interval`	Das Aggregationsintervall für die Daten in Sekunden. Typische Werte sind vielfache von 15 Sekunden (15, 30, 60, 180, ...). Nach Ablauf des Aggregationsintervalls übermittelt *TDAD* die Daten den Clients.
`device`	Die Bezeichnung der seriellen Schnittstelle (EIA-232), über welche *TDAD* mit den Detektoren kommuniziert. Für LINUX System werden hier typischerweise Werte wie `/dev/ttyS0` (erste serielle Schnittstelle) bzw. `/dev/ttyS1` (zweite serielle Schnittstelle) verwendet. Windows Systeme verwenden Werte wie `COM1` (erste serielle Schnittstelle) bzw. `COM2` (zweite serielle Schnittstelle).
`baudrate`	Die Baudrate mit der der *TDAD* mit den Detektoren über die serielle Schnittstelle kommuniziert. Typische Werte sind `4800`, `9600` oder `19200`.
`timeout`	Die maximale Zeit, in welcher ein Detektor auf die Anfrage des *TDAD* antworten kann. Diese Zeit wird in Millisekunden festgelegt und beträgt typischerweise `100`. Antwortet ein Detektor nicht fristgerecht kommt es zu einem Timeout und der *TDAD* fährt mit dem nächsten Detektoren weiter.

Eine Konfigurationsdatei besitzt immer **genau einen** Abschnitt `[global]`.

5.2 Detektor spezifische Konfiguration

Jeder logischen Detektor wird mit einem eigenen Abschnitt `[detector]` beschrieben und konfiguriert. Dabei ist zwischen einem physikalischen Detektor (Einschub am Rack) und einem logischen Detektor zu unterscheiden; ein physikalischer Detektor kann mehrere logische Detektoren umfassen (z.B. besitzt ein physikalischer Detektor mit 2 Doppelschleifen 2, ein solcher mit 4 Einzelschleifen 4 logische Detektoren). Je nach den örtlichen Gegebenheiten (z.B. Anzahl Fahrspuren) wird nur eine Teilmenge der logischen Detektoren eines physikalischen Detektors konfiguriert. Die nicht verwendeten logischen Detektoren sind in diesem Fall auf dem Einschub zu deaktivieren.

Die gültigen Schlüssel für den Abschnitt `[detector]` sind der folgenden Tabelle zu entnehmen:

Schlüssel	Bedeutung und Wert
`id`	Die Adresse auf dem EIA-485 Bus unter welchem der Detektor erreichbar ist. Diese Adresse ist identisch mit der auf dem Einschub konfigurierten Adresse. Jeder logische Detektor hat eine eindeutige Adresse zwischen `1` und `254`.
`manufacturer`	Der Hersteller des physikalischen Detektors (Einschub). Die möglichen Werte sind der Tabelle unten zu entnehmen. Der Wert wird beim starten von *TDAD* zur Prüfung des tatsächlich vorhandenen physikalischen Detektors verwendet und muss mit diesem übereinstimmen.
`type`	Die Art des verwendeten physikalischen Detektors (Einschub). Die möglichen Werte sind der Tabelle unten zu entnehmen. Der Wert wird beim starten von *TDAD* zur Prüfung des tatsächlich vorhandenen physikalischen Detektors verwendet und muss mit diesem übereinstimmen.
`class`	Die Möglichkeiten des physikalischen Detektors zur Klassifizierung der Fahrzeuge gemäss TLS*. Es können die Werte `2` (2 Fahrzeugklassen), `51` (5+1 Fahrzeugklassen) und `81` (8+1 Fahrzeugklassen) verwendet werden. Der Wert wird beim starten von *TDAD* zur Prüfung des tatsächlich vorhandenen physikalischen Detektors verwendet und muss mit diesem übereinstimmen.
`teltype`	Die Möglichkeiten des physikalisch Detektors zur Ermittlung detaillierter Werte für ein einzelnes Fahrzeug. Die möglichen Werte sind der Tabelle unten zu entnehmen. Der Wert wird beim starten von *TDAD* zur Prüfung des tatsächlich vorhandenen physikalischen Detektors verwendet. Unterstützt der Detektor einen höheren Wert als den in der Konfiguration festgelegten, so wird automatisch der höhere Wert von *TDAD* verwendet, da dieser genauere Daten liefert. Unterstützt der Detektor nur kleinere Werte, so bricht *TDAD* mit einer entsprechenden Fehlermeldung ab.

Es ist grundsätzlich möglich, eine beliebige Anzahl logischer Detektoren zu konfigurieren; *TDAD* besitzt diesbezüglich keine Beschränkung. Aus Gründen der für die Kommunikation auf dem EIA-485 benötigten Zeit ist die Anzahl jedoch einzuschränken um die Umlaufzeit (d.h. die Zeit welche benötigt wird um alle logischen Detektoren einmal abzufragen) zu begrenzen. Ist der Einsatz von mehr als **16** logischen Detektoren in einem System geplant, sind genauere Abklärungen erforderlich.

Die folgende Tabelle enthält die von *TDAD* unterstützten Detektoren.

Hersteller	Detektor	Wert `manufacturer`	Wert `type`	Wert `teltype`	Anzahl logischer Detektoren
Weiss GmbH	RD2002	`WEISS`	`RD2002`	`1,2,3`	1
Weiss GmbH	MC2014SL	`WEISS`	`MC2014`	`1,2,3`	4
Weiss GmbH	MC2024	`WEISS`	`MC2024`	`1,2,3`	2
Weiss GmbH	MC2024[1]	`WEISS`	`MC2024L`	`1,2`	2

Sowohl der Hersteller als auch der Typ des Detektors werden von *TDAD* zur Ermittlung der korrekten Bestückung des Rack's verwendet. Fehlt ein Detektor Einschub oder ist ein falscher Detektor Einschub bestückt, so wird dies vom *TDAD* erkannt und als Fehler protokolliert; *TDAD* lässt sich in diesem Fall **nicht** starten. Für spezielle Anwendungen bzw. Tests kann der Wert `generic` für die Schlüssel `manufacturer` und `type` verwendet werden. Der Einsatz von `generic` führt zu einer ungenauen Erkennung der verwendeten Detektor Einschübe und darf deshalb **nicht für in Betrieb befindliche Anlagen** verwendet werden!

5.3 UDP Client spezifische Konfiguration

TDAD stellt den UDP Clients die aggregierten Daten nach Ablauf des Messintervalls zur Verfügung. Jeder UDP Client wird mit einem eigenen Abschnitt `[pushclient]` konfiguriert. Die Daten werden gemäss der Protokoll Spezifikation *TDAP* (Traffic Data Acquisition Protocol) übertragen.

Die gültigen Schlüssel für den Abschnitt `[pushclient]` sind der folgenden Tabelle zu entnehmen.

Schlüssel	Bedeutung und Wert
`ip`	Die IP Adresse des Clients, an welchen die Daten übertragen werden. Die Adresse ist im Format `aaa.bbb.ccc.ddd` einzugeben. Sollen die Daten auf dem Messsystem selbst zur Verfügung gestellt werden, ist die IP Adresse `127.0.0.1` (localhost) zu verwenden.
`port`	Der UDP Port auf welchem der Client die Daten erwartet. Es können Werte zwischen `1` und `65535` verwendet werden. Zur Vermeidung von Problemen mit anderen auf dem Client bereits laufenden Diensten sind vorzugsweise nur Portnummern von `1024` bis `65535` zu verwenden.

Die Anzahl der UDP Clients ist grundsätzlich nicht beschränkt. Damit ist es auf einfache Weise möglich, die Daten gleichzeitig verschiedenen Systemen (Steuerung, Archiv Datenbank, Internet, etc.) zur Verfügung zu stellen.

1 Es handelt sich um Detektoren des Typs MC2024 mit einer Firmware Version **vor** v1.34. Die Firmware Version kann mittels Handterminal und Eingabe von 'X' ermittelt werden.

5.4 Beispiel

Das folgenden Beispiel enthält die vollständige Konfiguration eines einfachen Messsystems.

```
;-------------------------------------------------------------------------
; Traffic Data Acquisition Daemon (TDAD) configuration file
;
; Copyright 2005, EETS GmbH
;-------------------------------------------------------------------------
;
; DO NOT MODIFY THE FOLLOWING DATA!
; $LastChangedRevision: 44 $
; $LastChangedDate: 2005-05-30 16:37:12 +0200 (Mon, 30 May 2005) $
; $LastChangedBy: felix $
;-------------------------------------------------------------------------

[global]                ; global parameters
interval= 15            ; aggregation interval in seconds
device= /dev/ttyS0      ; device connecting to DE bus
baudrate= 9600          ; baudrate of DE bus device
timeout= 100            ; DE bus response timeout in ms

[detector]              ; detector specific parameters
id= 1                   ; DE number
manufacturer= WEISS     ; manufacturer
type= MC2024            ; type or model
class= 2                ; vehicle classificaion capabilites
teltype= 3              ; type of telegrams to retrieve data

[pushclient]            ; parameters specific to push client
ip= 192.168.1.21        ; IP address to where data will be sent
port= 8834              ; port that is to receive the data

[pushclient]            ; parameters specific to push client
ip= 192.168.2.34        ; IP address to where data will be sent
port= 8834              ; port that is to receive the data
```

Alle Daten werden über ein Zeitintervall von 15 Sekunden aggregiert. Das System besitzt einen logischen Detektor und sendet die Daten an zwei UDP Clients.

6 Liste der Revisionen

Revision	Datum	Kommentar
1.00	02.06.2005	Erstfassung
1.01	11.07.2005	Erweiterung um Weiss MC2024 Detektor mit Firmware Version vor v1.34 (MC2024L Type)
1.02	21.07.2005	Kapitel Zielpublikum eingefügt
1.03	15.08.2005	Ergänzung syslog facility
1.04	21.08.2008	Entfernung der nicht mehr benötigten globalen Parameter für die Anzahl Detektoren bzw. UDP Clients