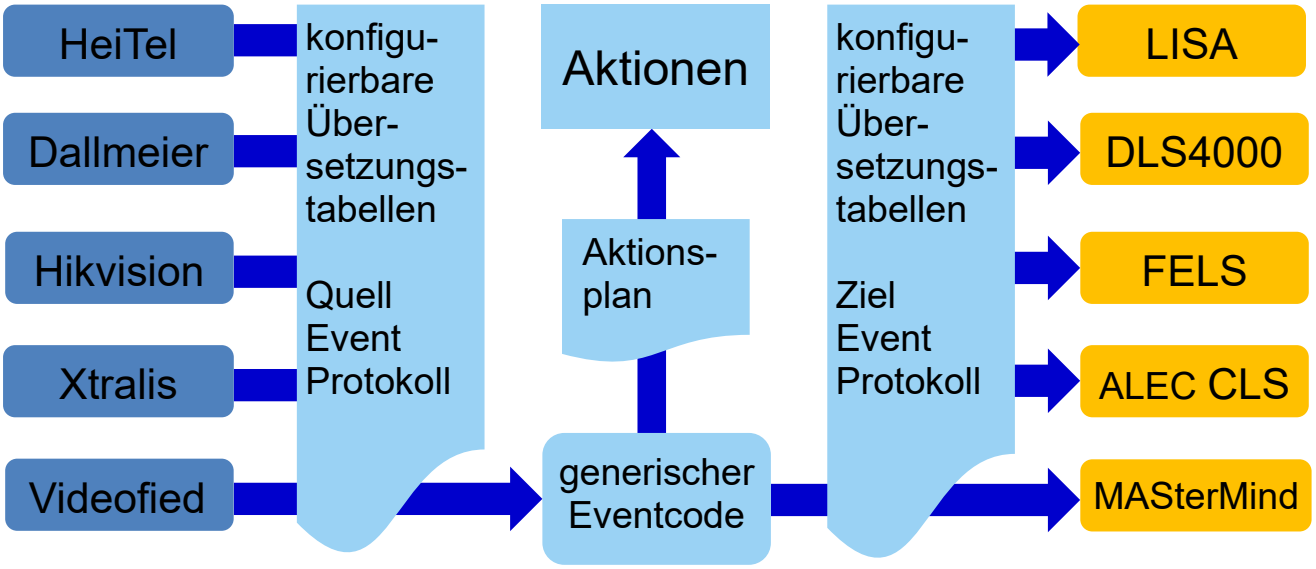
EventRules

Regeln für das Auswerten von Ereignissen konfigurieren



Status: Freigegeben

Accellence Technologies GmbH • Garbsener Landstr. 10 • 30419 Hannover
Tel. 0511 277-2400 • Fax 0511 277-2499 • E-Mail: info@accellence.de • Website: www.accellence.de

Inhalt

Syntaxhinweise

<x> Platzhalter, für den konkrete Werte eingesetzt werden müssen.
→ Verweis auf weitere Dokumente oder Textstellen
{F4} Bezeichnung einer Taste auf der PC-Tastatur
Diese Stelle muss noch bearbeitet werden

1. Einführung

Systeme verschiedener Hersteller melden Ereignisse auf ganz unterschiedliche Weise. Eine wichtige Leistung von EBÜS besteht darin, diese Meldungen zu normieren, damit sie einheitlich und unabhängig vom jeweiligen Typ des alarmauslösenden Geräts angezeigt, verarbeitet und weitergeleitet werden können.

Ab Version 2.1.3 sind die Algorithmen für das Auswerten von Ereignissen (engl.: Events) nicht mehr fest in den Programmcode von EBÜS eincompiliert, sondern können mittels scriptbasierter Regeln und editierbarer Eventcode-Tabellen frei konfiguriert werden.

Somit kann die Event-Verarbeitung von EBÜS in Minutenschnelle an neue Datenformate und Eventcodes angepasst und erweitert werden.

Sie können diese Anpassungen selbst vornehmen, Sie können aber gern auch unseren Support nutzen, falls Sie neue Ereignistypen verarbeiten möchten. In jedem Fall empfehlen wir, dass Sie solche Erweiterungen mit uns abstimmen, damit wir sie in unser Setup übernehmen, damit Ihre Änderungen ab dem nächsten Update in unserem Setup enthalten sind und Sie diese Änderungen nicht immer wieder manuell nachpflegen müssen.

Um Ihre ggf. selbst angepassten Scripts und Tabellen zu erhalten, überschreiben wir diese Daten nicht bei Updates. Um die Scripts und Tabellen aus dem letzten Update zu verwenden, klicken Sie in EBÜS auf Konfiguration → Event-Manger → Tabellen updaten.

2. Grundlagen

Sämtliche Informationen zu allen Ereignissen sind in dem Dateipfad und dem Inhalt der Dateien enthalten, die auf dem EBÜS-Server in der FTP-Verzeichnisstruktur eintreffen, die vom AlarmReceiverFTP überwacht wird. Auch der AlarmReceiverSMTP, der AlarmReceiverSIA und die AlarmReceiver für die verschiedenen herstellerspezifischen Protokolle legen dort ihre Daten ab, so dass sie von dort aus einheitlich weiterverarbeitet werden können.

Damit Alarmdaten richtig zugeordnet werden können, muss für jede **Bildquelle**, die Alarme empfangen soll, in **EBÜS_Config** auf der Registerkarte **Alarme** im Eingabefeld Verzeichnis für Alarmdaten: ein ***eindeutiges*** (ausschließlich für diese Bildquelle verwendetes) Unterverzeichnis konfiguriert werden. Nur dann können Alarmdaten, die in diesem Verzeichnis eintreffen, genau dieser Bildquelle **bq** im Schutzobjekt **so** zugeordnet werden.

Anhand **so** und **bq** kann EBÜS dann aus den mit EBÜS_Config angelegten Konfigurationsdaten u.a. den Bildquellentyp **bqt** ermitteln. Das ist der Name des Bildquellenadapters (***.bqa-Datei**), der für diese Bildquelle auf der Registerkarte **Verbindung** in der Liste **Typ der Bildquelle** ausgewählt wurde.

Neben den mit EBÜS_Config manuell konfigurierten Daten (<**so**>.so-Datei) können auch automatisch generierte Daten aus der Datei \EBÜS\Bilder\<**so**>\<**bq**>**BqaInfo.txt** verwendet werden, die von der BQA-Client-Anwendung bzw. der BQA-Datei bei Aufschaltungen auf eine bestimmte Bildquelle ermittelt und gespeichert werden. Dazu zählt z.B. die **camtable**, die zur Berechnung der Kameranummern bei Alarmen von Hikvision-kompatiblen Systemen benötigt wird.

Zum Auswerten von Ereignissen werden von EBÜS folgende Variablen bereitgestellt:

- **eventfilepath** enthält den FTP-Dateipfad incl. Dateinamen unterhalb **ftproot**
- **eventmessage** enthält den Inhalt (Text) einer Alarmdatei ***.msg, *.xml, …**

Bei Meldung eines Ereignisses durch einen Anruf in der Anrufliste (call) steht nur **eventfilepath** zur Auswertung zur Verfügung, beim Anzeigen von Meldungen nur **eventmessage**, bei Direktbenachrichtigung vom **AlarmReceiverFTP** mittels AMS_RCP-Kommando „notify alarm_img“ steht nur **eventfilepath** zur Auswertung zur Verfügung, bei „notify alarm_msg“ sowohl **eventfilepath** als auch **eventmessage**.

Die Scripts müssen deshalb so angelegt werden, dass sie wahlweise **eventfilepath** oder **eventmessage** auswerten, je nachdem, welche Informationen jeweils verfügbar sind.

Die Aufgabe der Scripts im EventManager besteht darin, auf Basis dieser Informationen die Werte für mindestens folgende Variablen zu berechnen:

- **eventprotocol** legt die Tabelle des zuständigen Quell-Protokolls fest, mit dem Eventcodes dieses Alarms normiert (d.h. in ein einheitliches Format übersetzt) werden
- **eventcode_src** enthält den Eventcode, nach dem in dieser Tabelle gesucht werden soll
- **eventtime** ist der Zeitpunkt, der aus den Eventdaten ermittelt wurde, im Format YYYYMMDDhhmmssttt (UTC in 17 Ziffern stellenwerttreu ohne Trennzeichen)
- **eventsrcid** gibt Auskunft, von welchem Gerät der Alarm kommt, z.B. die Seriennummer des Gerätes; kann alternativ auch in EBÜS_Config pro Bildquelle vorkonfiguriert werden
- **eventmessage** wird von den Scripts ggf. so umformatiert, dass der Meldungstext für Leitstellenmitarbeiter verständlich ist

Darüber hinaus können weitere Variablen angelegt werden, die weitere ergänzende Informationen zu dem Ereignis enthalten.

Die dabei verwendete **Scriptsprache** wird hier beschrieben → www.ebues.de/doc/AccParser.pdf

In den Scripts können u.a. die in diesem Text **fett und grün** markierten Variablennamen verwendet werden. In der **Live Event Protokollierung** des **Event-Managers** können durch Doppelklick auf ein Event alle Variablen gelesen werden, die bei diesem Event ermittelt wurden und zur weiteren Auswertung bereitstehen.

Zur Auswertung neu eintreffender Eventdaten wird zuerst das Script ausgeführt, das für diese Bildquelle in EBÜS_Config auf der Registerkarte **Alarme** konfiguriert wurde. Falls dort kein Script konfiguriert wurde, oder falls dieses Script kein Protokoll und keinen Eventcode ermitteln konnte, wird als nächstes das Script mit dem Namen des Bildquellentyps (**bqt**) ausgeführt. Falls auch so ein Script nicht vorhanden ist oder falls auch dieses Script keine gültigen Werte für **eventprotocol** und **eventcode_src** ermittelt, wird das Script **!default** ausgeführt. Falls auch dieses keinen Eventcode ermitteln kann, wird der Eventcode 1000011 (Allgemeiner Videoalarm, nicht näher spezifiziert) verwendet.

Im nächsten Schritt muss nun aus dem (herstellerspezifischen) **eventcode_src** ein normierter (generischer, herstellerübergreifender) **eventcode_gen** ermittelt werden.

Die Variable **eventprotocol** legt fest, welche Eventcode-Tabelle dabei zur Übersetzung der Original-Event-Codes **eventcode_src** aus dem Quell-Protokoll zum Eventcode **eventcode_gen** verwendet wird.

EBÜS schaut also in der **Tabelle des Quell-Protokolls eventprotocol** nach dem passenden Eintrag für **eventcode_src**, über diese Tabellen kann diese Zuordnung gesteuert werden.

Zu jedem Eventcode können in der Tabelle des Quell-Protokolls **eventprotocol** in der **Spalte „Formeln“** ggf. noch weitere Scripts hinterlegt werden, die –soweit erforderlich – eventcodeabhängige Berechnungen enthalten. Die **Spalte „Kommentar“** enthält Event-Beschreibungen des jeweiligen Herstellers oder Original-Protokolls, die in der Variablen **eventcomment_src** bereitgestellt werden.

Typische Variablen, deren Werte soweit möglich ermittelt werden sollten, sind

- **alarmcam** definiert die alarmauslösenden Kamera, damit EBÜS diese automatisch auswählen kann
- **event_location** kennzeichnet, aus welchem Bereich im überwachten Objekt der Alarm kam
- **eventinfo** ggf. weitere für die Leitstelle relevante Informationen zu dem Alarm

Spezielle Einstellungen je Bildquelle (z.B. die Zuordnung digitaler Eingänge zu verschiedenen Ereignisarten) können Sie in EBÜS_Config vornehmen → www.ebues.de/Konfiguration.pdf#page=20

Mit dem normierten Eventcode **eventcode_gen** schaut EBÜS dann in der Haupt-Tabelle des Event-Managers nach, welche Aktionen für dieses Event konfiguriert sind, und führt diese aus.

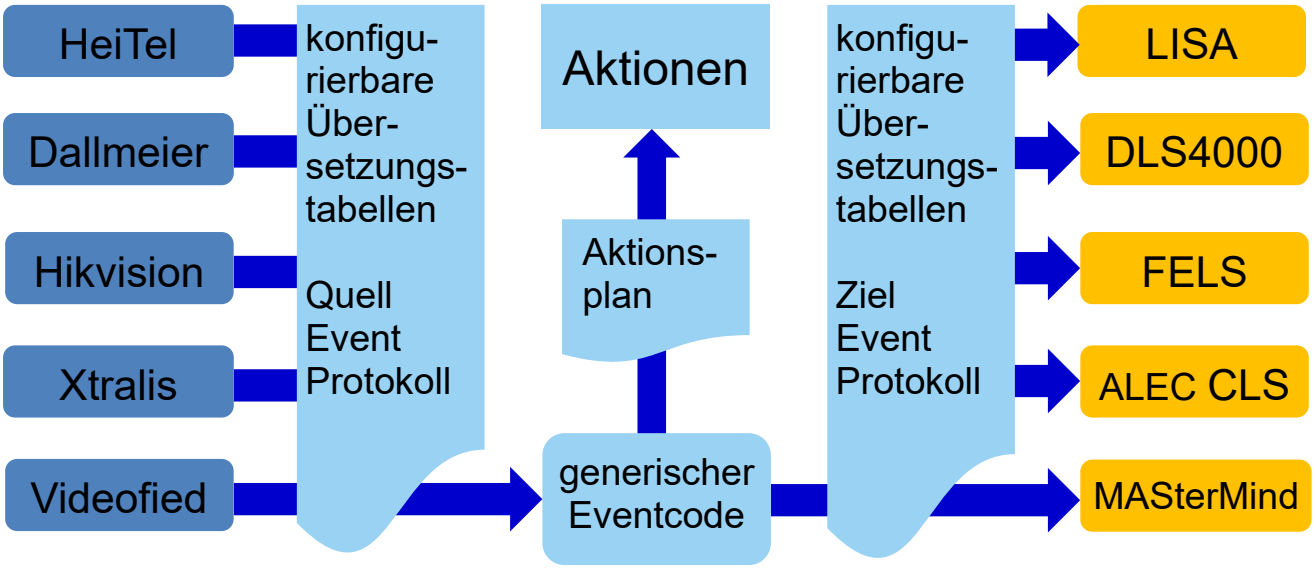
Anschließend kann der normierte Eventcode **eventcode_gen** ggf. noch in einen für das Zielsystem (z.B. LISA, MASterMind, DLS4000, FELS, AM/Win, …) passenden **eventcode_dst** übersetzt werden.

3. Verarbeitungsschema

Die Events der verschiedenen Hersteller und Protokolle werden mittels Tabellen für die **Quell-Protokolle** in einen **generischen** (normierten, herstellerübergreifenden) **Eventcode** übersetzt.

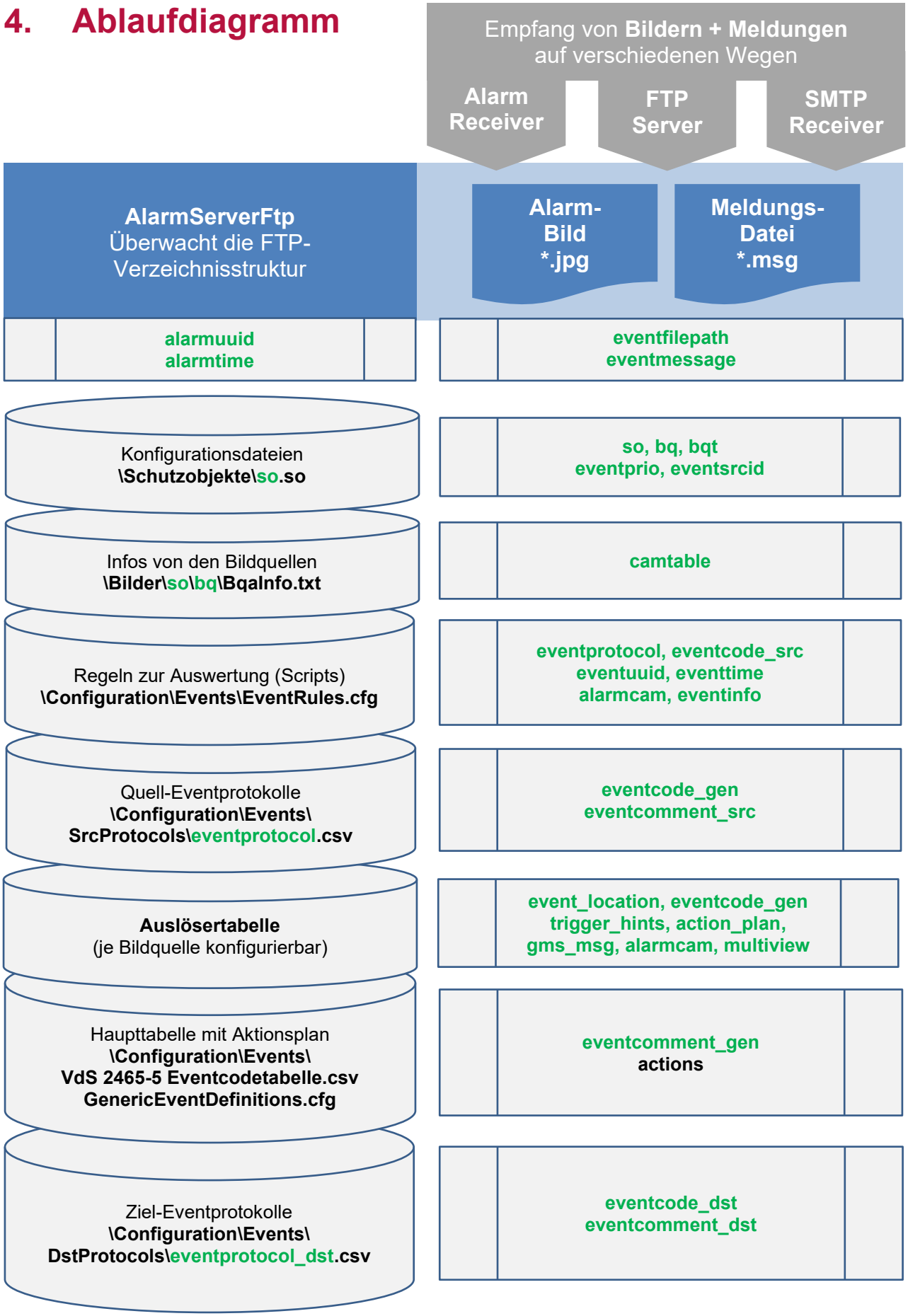
Somit können unabhängig davon, von welchem System ein Event ursprünglich kommt, die bei den verschiedenen Event-Arten erforderlichen Aktionen in einem gemeinsamen **Aktionsplan** konfiguriert werden.

Die generischen Eventcodes können anschließend in ein **Ziel-Event-Protokoll** übersetzt werden, damit sie vom jeweils verwendeten Management-System verstanden werden.



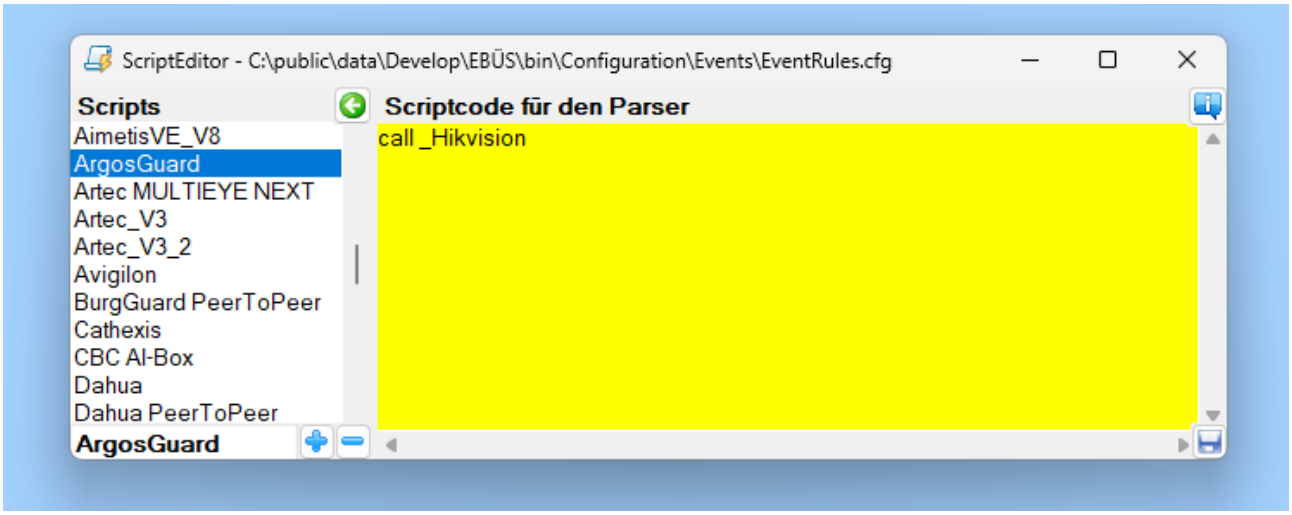
Der **Aktionsplan** legt fest, bei welchem **generischen Eventcode** welche **Aktionen** ausgeführt werden.
Alle **Übersetzungstabellen** und der **Aktionsplan** können im EBÜS **EventManager** bearbeitet werden.

4. Ablaufdiagramm



5. Bearbeiten der Regeln

Konfiguration → Event-Manager → Regeln zum Auswerten von Eventdaten öffnet den ScriptEditor, mit dem die Regeln zum Auswerten von Ereignissen interaktiv bearbeitet werden können:



Bei Auswahl eines Scripts wird rechts daneben der zugehörige Scriptcode angezeigt.
Der Scriptcode kann hier direkt bearbeitet werden. Änderungen sind nach dem Speichern sofort wirksam.
Damit erreichen wir extrem kurze Turn-Around-Zeiten bei Anpassungen, Tests und Inbetriebnahmen.

Bei jedem Ereignis wird zuerst das Script mit dem Namen des dafür konfigurierten Bildquellentyps **bqt** aufgerufen, z.B. ArgosGuard. Hier genügt beispielsweise der Eintrag „call _HikVision“. Damit werden bei diesem Bildquellentyp (und allen weiteren OEM-Varianten) die Regeln angewendet, die im Script _HikVision hinterlegt sind:

Sämtliche Informationen zu allen Ereignissen sind in dem Dateipfad und dem Dateiinhalt der Dateien enthalten, die in unserer FTP-Verzeichnisstruktur eintreffen.

- Der FTP-Dateipfad wird in der Variablen **eventfilepath** bereitgestellt,
- der Inhalt von *.msg-Dateien in der Variablen **eventmessage**

Auch alle Events, die von unseren AlarmReceivern (z.B. SIA, SMTP, …) empfangen werden, landen dort und können auf die gleiche Weise ausgewertet werden.

Mit einer einzigen Scriptcodezeile werden durch einen regulären Ausdruck (regex) aus dem Dateipfad (eventfilepath) sämtliche darin enthaltenen Informationen ausgelesen:

regex eventtime,eventprotocol,eventcode_src,alarmcam=%eventfilepath%([0-9]{17})##([\S]+)##([0-9]+)##([\S]+)##\.msg

Der Inhalt von Meldungen wird hier wahlweise als JSON-Datenstruktur oder (falls ersteres nicht zum Erfolg führte) als SMTP-Meldung (E-Mail) interpretiert:

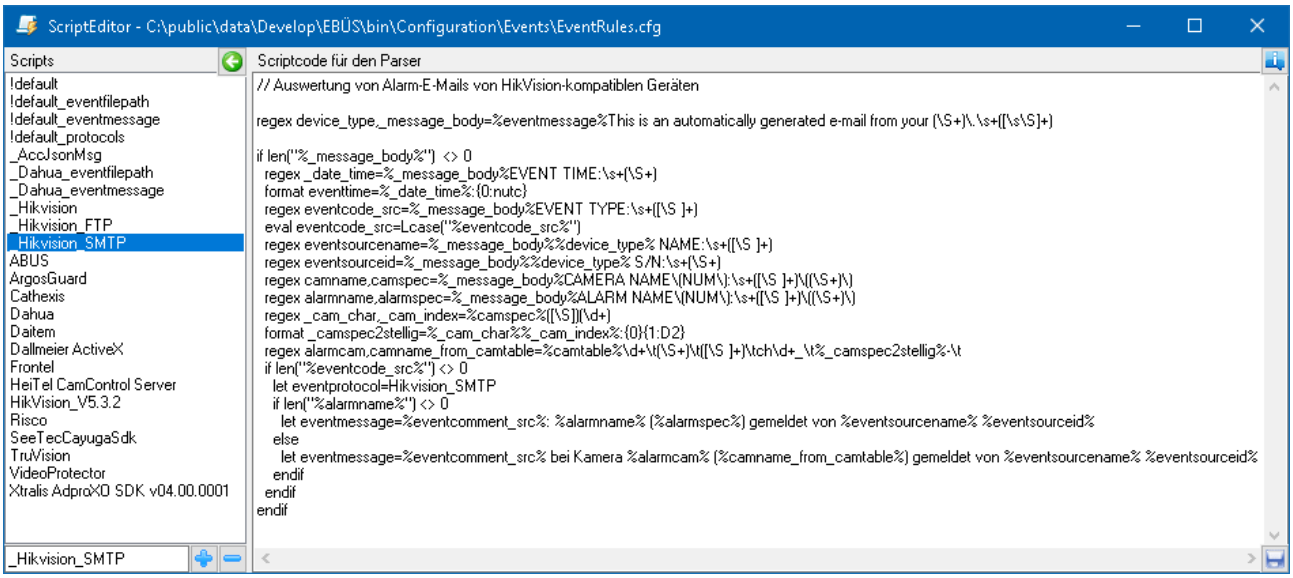
```
if len("%eventmessage%") <> 0
  call _AccJsonMsg

  // Falls es nicht die normierte Meldung vom AccAlarmReceiver ist, auf mögliche SMTP-Eventcodes prüfen:
  if len("%eventprotocol%") = 0
    call _Hikvision_SMTP
  endif
endif
```

Das Script zur Auswertung von Meldungen im JSON-Format, wie sie die AlarmReceiver liefern, sieht so aus:

Die Auswertung der E-Mails von HikVision ist schon etwas anspruchsvoller, weil

- sich die Parameternamen abhängig vom Recordertyp (**device_type** € {DVR|NVR}) unterscheiden
- die Kameranummer anhand einer Tabelle (**camtable**) umgerechnet werden muss

aber auch dies alles lässt sich in unserem ScriptEditor mit wenigen Zeilen regeln:



- Mit **+** können weitere Scripts hinzugefügt, mit **-** Scripts gelöscht werden.
- Ein gelber Hintergrund weist darauf hin, wenn ein Script bearbeitet, aber noch nicht gespeichert wurde.
- Das Diskettensymbol rechts unten speichert das aktuell bearbeitete Script.

Umrechnen der Lokalzeit in UTC

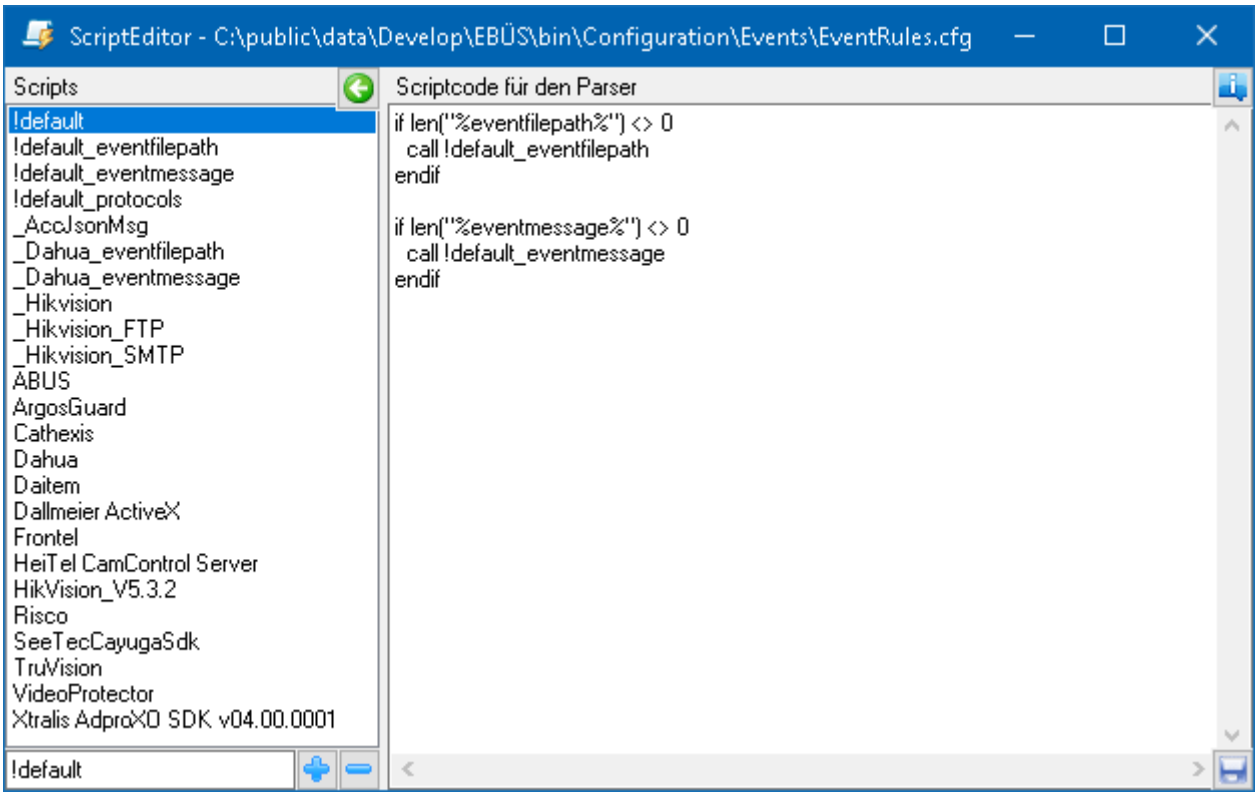
Viele Videosysteme liefern Zeitangaben nicht – wie von uns empfohlen – in zeitzonen- und sommerzeit-unabhängiger UTC, sondern als Lokalzeit. Dies kann mit

```
// Lokalzeit --> in UTC umrechnen:
format eventtime=%eventtime%:{0:Loc2Utc}
```

umgerechnet werden.

Default-Script

Falls für einen Bildquellentyp kein eigenes Script angelegt wurde, oder falls das bildquellenspezifische Script nicht die erforderlichen Parameter berechnen konnte, kommt das Script !default zum Einsatz:



Von hier aus wird dann zu allen anderen Auswertungen verzweigt, sobald ein bestimmtes Meldungsformat erkannt wurde.

Ein Doppelklick im Script-Editor auf einen Script-Namen hinter einem call öffnet das betreffende Script. Mit dem grünen Pfeil nach links über der Script-Liste kann zum vorherigen Script wieder zurückgekehrt werden.

6. Test neuer Regeln

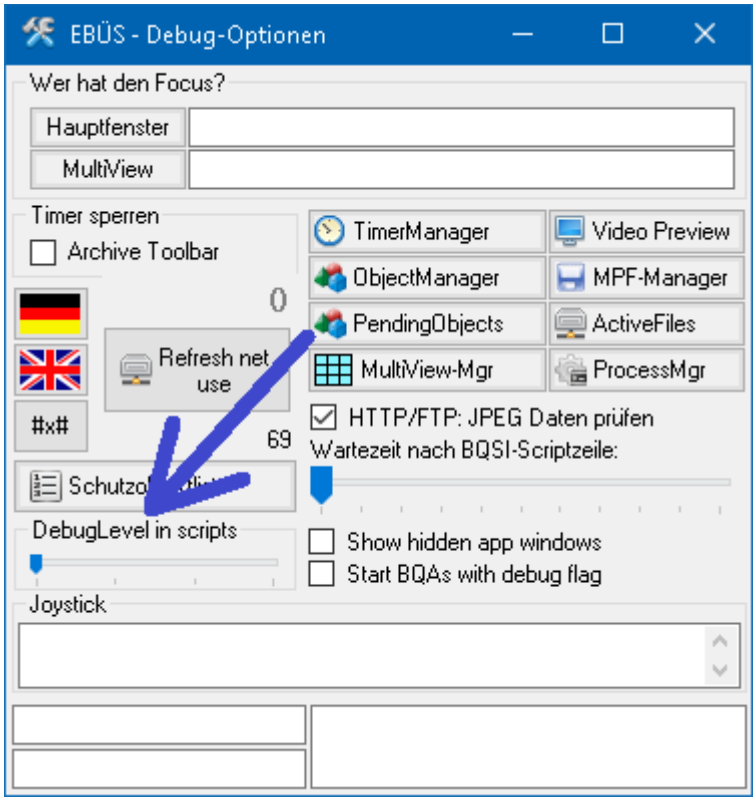
Zum Erstellen und Testen neuer Scripte das ZIP-Archiv → www.ebues.de/bin/Tools/ParserTest.zip entpacken und daraus die Anwendung ParserTest.exe starten.

Die Anleitung dazu liegt bereit unter → www.ebues.de/doc/AccParser.pdf

Das Format der Datei **MyScripts.txt**, in der ParserTest.exe die Scripts speichert, ist identisch zum Format der Datei \EBÜS\Configuration\Events**EventRules.cfg**, d.h. einzelne Scripts oder auch das komplette Scriptset können mit einem Texteditor wie beispielsweise Notepad per Copy&Paste zwischen EBÜS und ParserTest ausgetauscht werden.

- Jede Zeile in diesen Dateien enthält genau ein Script
- Der Name des Scripts steht am Anfang der Zeile vor dem Gleichheitszeichen

Wenn Scripts in EBÜS getestet werden sollen, kann mit den „Debug-Optionen zur Fehleranalyse“ der Debug-Level bei der Scriptausführung eingestellt werden:



Dieses Fenster wird mit Doppelklick auf das accellence-Logo im Fenster Konfiguration Videoarbeitsplatz geöffnet, sofern für den angemeldeten Benutzer in der Benutzerverwaltung von EBÜS das Benutzerrecht „Debug-Optionen zur Fehleranalyse nutzen“ freigegeben wurde.

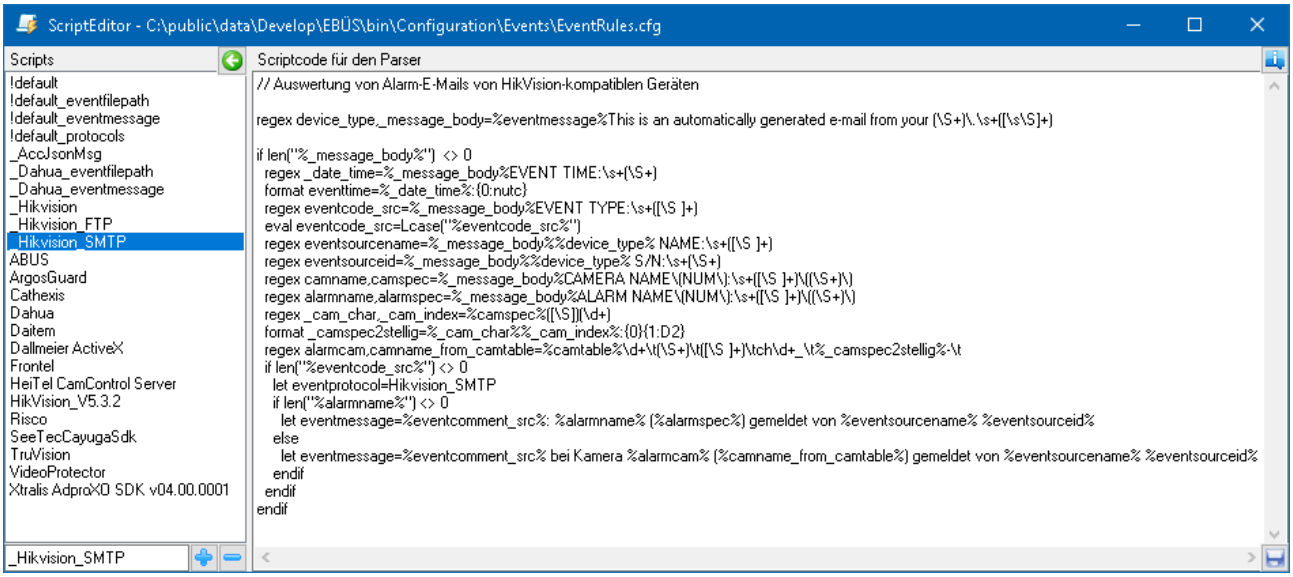
Nach jedem Anwendungsstart ist hier DebugLevel 0 eingestellt, der keine Ausgaben im Logbuch erzeugt.

- DebugLevel 1: nur Fehler
- DebugLevel 2: auch Warnungen
- DebugLevel 3: alle Informationen

Mit DebugLevel 3 kann die Scriptausführung im Logbuch Schritt für Schritt nachvollzogen werden.

7. Auswahl Eventcodetabellen

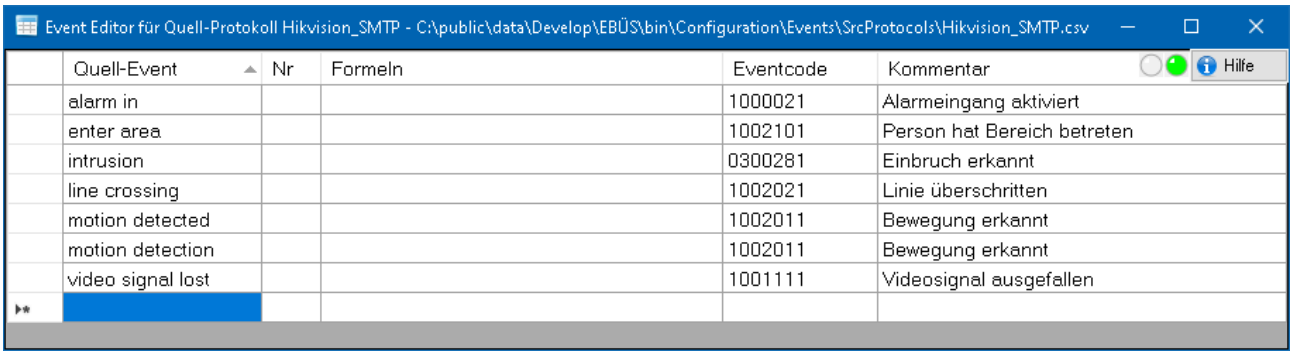
Mit Konfiguration → Event-Manager → Regeln zum Auswerten von Eventdaten können Sie bestehende Regeln anpassen und neue hinzufügen:



Mit der Zeile

let **eventprotocol**=Hikvision_SMTP

wird hier festgelegt, dass die **Eventcodetabelle** für das **Quell-Protokoll** „Hikvision_SMTP“ zum Übersetzen dieser Events verwendet werden soll:



Quell-Event	Nr	Formeln	Eventcode	Kommentar
alarm in			1000021	Alarmeingang aktiviert
enter area			1002101	Person hat Bereich betreten
intrusion			0300281	Einbruch erkannt
line crossing			1002021	Linie überschritten
motion detected			1002011	Bewegung erkannt
motion detection			1002011	Bewegung erkannt
video signal lost			1001111	Videosignal ausgefallen

Einige Videosysteme von Hikvision melden Bewegungen mit

motion detected
andere mit motion detection

Auch solche Besonderheiten können durch unsere Eventcode-Tabellen schnell angepasst werden.

Die **Quell-Events** in dieser Tabelle basieren auf den von uns bereits gesammelten Beispielen. Wenn Sie weitere Events in Alarm-E-Mails von Hikvision finden, können die in dieser Tabelle einfach ergänzt werden.

Bitte neue Events an support@accellence.de melden, damit wir die auch in unseren Setups für die nächsten Updates berücksichtigen können.

8. Eventspezifische Regeln

Darüber hinaus gibt es Fälle, in denen die Protokolldaten abhängig vom Eventcode eine unterschiedliche Bedeutung haben und demzufolge unterschiedlich interpretiert werden müssen, etwa beim FRONTEL-Protokoll → vgl. Abschnitt 2.7 aus dem Dokument ProtocolFullGI_1.2.1.pdf

Deshalb kann in den Quell-Protokoll-Tabellen zu jedem Event in der Spalte „Formeln“ weiterer Scriptcode konfiguriert werden, der dann nur bei Auftreten dieses bestimmten Events von cParser ausgeführt wird.

Die Datei \EBÜS\Configuration\Events\SrcProtocols\Frontel.csv zeigt, wie mit diesen Möglichkeiten auch eine komplexe Spezifikation wie FRONTEL durch reine Konfiguration implementiert und ggf. schnell an neue Versionen angepasst werden kann, ohne dass ein Compilerlauf oder Neustart der Anwendung nötig ist:

Event Editor für Quell-Protokoll Frontel - C:\public\data\Develop\EBÜS\bin\Configuration\Events\SrcProtocols\Frontel.csv

Quell-Event	Nr	Formeln	Eventcode	Kommentar
1	1	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0300281	Intrusion
2	2	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0300280	Intrusion end
3	3	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0300241	Tamper fault
4	4		0300240	Tamper fault end
5	5	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0300351	Panic button
6	6		0700291	5 successive wrong codes
7	7	regex peripheral,badge=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0300041	Duress code 1
8	8	regex peripheral,badge=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0300041	Duress code 2
9	9	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0600521	Supervision
10	10	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0600520	Supervision end
11	11		0600521	Radio jamming detection
12	12		0600520	Radio jamming detection end
13	13		0600521	Radio interference
14	14		0600520	Radio interference end
15	15		0600171	Panel low battery
16	16		0600170	Panel low battery end
17	17	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	0600471	Device low battery
18	18	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0600470	Device low battery end
19	19		0400200	AC power loss
20	20		0400191	AC power loss end
21	21		0400031	Phone line
22	22		0400030	Phone line end
23	23		0600041	Panel reboot
24	24	regex arming_profile,badge=%eventinfo%([0-9]+)-([0-9]+)	1006101	Panel armed
25	25	regex arming_profile,badge=%eventinfo%([0-9]+)-([0-9]+)	1006100	Panel disarmed
26	26		1006200	Periodic test
27	27		1099991	Alarm memo
28	28		1099991	Unused event
29	29		0100031	Alarm test
30	30		1099991	Remote enrollement request
31	31		1099991	Remote maintenance request
32	32	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0200071	Smoke detection
33	33	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0200070	Smoke detection end
34	34	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	0800011	Medical alert
35	35	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	1001121	Ethernet connection loss
36	36	regex peripheral=%eventinfo%([0-9]+)-[0-9]+\nlet alarmline=%peripheral%	1001120	Ethernet connection loss end
37	37	regex peripheral,generic_event=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1005031	Detection on the programmable input
38	38	regex peripheral,generic_event=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1005030	Detection on the programmable input end
39	39		1099991	Unused event
40	40		1099991	Connection request Frontel
41	41		0600521	GPRS / 2G3G jamming detection
42	42		0600520	GPRS / 2G3G jamming detection end
43	43	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Device bypass
44	44	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Device ejection
45	45	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Device ejection end
46	46	regex peripheral,wired_supervision_type=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Input Supervision Fault
47	47	regex peripheral,wired_supervision_type=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Input Supervision restoral
48	48	regex peripheral,wired_supervision_type=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Wirelink Supervision
49	49	regex peripheral,wired_supervision_type=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1099991	Wirelink Supervision end
50	50	regex battery_status=%eventinfo%([0-9]+)-[0-9]+	0600171	Battery Fault
51	51		0600170	Battery Fault end
52	52	regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)\nlet alarmline=%peripheral%	1004101	Device moved

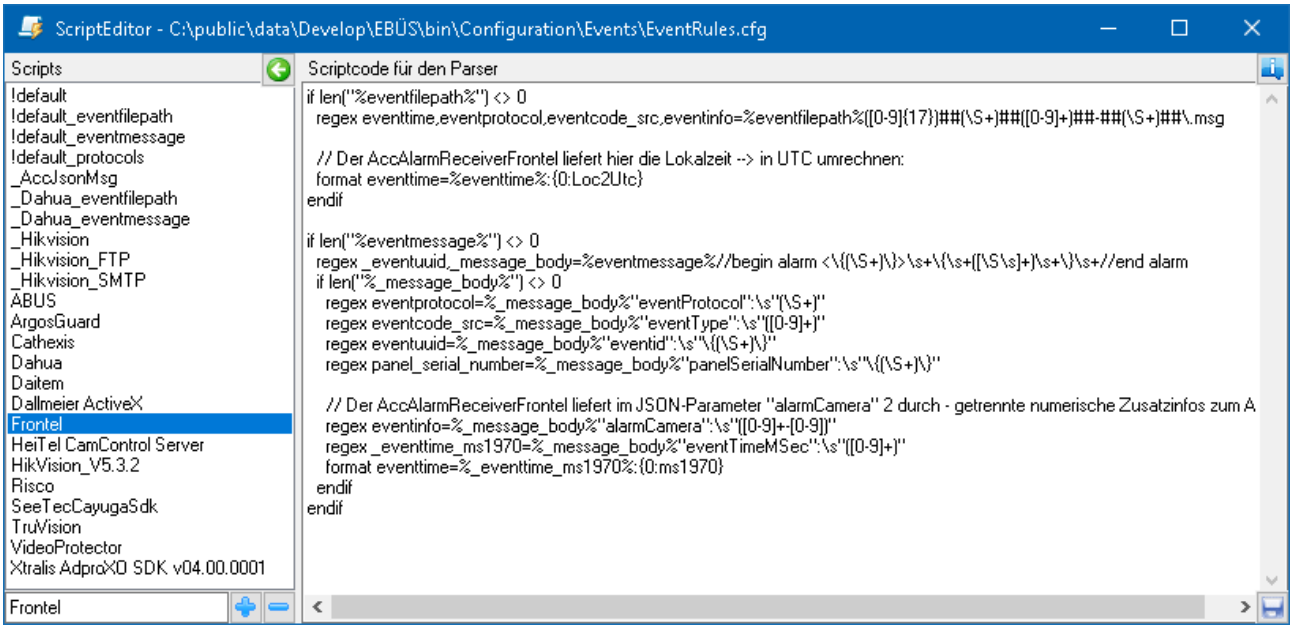
Diese Tabelle bildet 1:1 und nahezu klartextlesbar die vom Hersteller vorgegebene Spezifikation ab.

Die Regel aus Zeile 52 dieser Tabelle wird als Beispiel auf der nächsten Seite erläutert.

Mit dem ersten Teil der Regel aus Zeile 52 (vor dem Zeilenumbruch \n)

regex peripheral,detector=%eventinfo%([0-9]+)-([0-9]+)

werden die neuen Variablen **peripheral** und **detector** definiert. Ihre Werte werden mittels regulärem Ausdruck aus dem Inhalt der Variablen **eventinfo** ermittelt, die vom Script Frontel bereitgestellt wird:



Mit dem zweiten Teil der Regel aus Zeile 52 (nach dem Zeilenumbruch \n)

let alarmline=%peripheral%

wird der Wert **peripheral** als **alarmline** verwendet, die z.B. an das Management-System gemeldet wird.

Somit können auch solche Protokolle sehr effizient und gut wartbar implementiert werden.
Neue Protokollversionen (Events, Parameter) können jederzeit einfach in dieser Tabelle nachgepflegt werden.
Nützliche Zusatzinformationen können hier in ergänzend definierten Variablen bereitgestellt werden.

9. Bildquellenspezifische Regeln

Falls bei bestimmten Bildquellen eine spezielle Auswertung der Events nötig ist, kann in EBÜS_Config auf der Karteikarte Alarme ein individuelles Script zur Alarm-Auswertung für diese Bildquelle konfiguriert werden.

Eine spezifische Reaktion auf unterschiedliche Auslöser (Kameras, digitale Eingänge, Alarmlinien, Stati, …) kann auf der Karteikarte Auslöser konfiguriert werden → www.ebues.de/Konfiguration.pdf#page=20

10. Anwendungsbeispiel

Ein Kunde meldet, dass das Videosystem von Hersteller X in einer neuen Version das Event Y meldet, das wir bislang noch nicht kannten.

Solche Anpassungen brauchten früher viele Tage, weil dazu erst einmal das ggf. mehrere Jahre zurückliegende Projekt wiederaufgenommen und vergegenwärtigt werden musste: Build-Umgebung aufsetzen, Änderungen einchecken, testen, Setup bauen, installieren … so etwas kann nun in wenigen Minuten direkt auf dem Kundensystem erledigt werden.

Und auch die Dokumentation aller implementierten Regeln zur Event-Auswertung ist aufgrund der übersichtlichen Tabellen und klartextlesbaren Scripte automatisch erledigt und stets aktuell.

11. Referenzliste

Alle im **EventManager** von EBÜS üblicherweise verwendeten Variablen zum Nachschlagen:

Variablenname	*Inhalt*	*Hinweise*
1. Rohdaten aus der erkannten Datei in der FTP Verzeichnisstruktur		
eventfilepath	FTP-Datei-Pfad (incl. Dateiname)	wo diese Datei empfangen wurde
eventmessage	Meldungstext	der in dieser Datei steht
2. Daten, die der AlarmReceiverFTP liefert		
alarmuuid	Eindeutige Kennung für den Alarm	
alarmtime	Alarmzeit, die über das AMS_RCP-Kommando "alarm" gesetzt wurde	UTC stellenwerttreu
3. Daten, die aus den Konfigurationsdaten im Verzeichnis Schutzobjekte ermittelt werden		
so	Name des Schutzobjektes	aus dem dieser Alarm kommt
bq	Name der Bildquelle	von der dieser Alarm kommt
bqt	Bildquellentyp, entspricht dem Namen der BQA-Datei ohne Extension (Datei-Endung)	wird in EBÜS_Config aus der Liste Typ der Bildquelle ausgewählt
eventsrcid	Eindeutige Kennung für das Gerät, das dieses Event gesendet hat	wird in EBÜS_Config auf der Karteikarte Bildquellen → Alarme konfiguriert, kann durch Regeln (Scripts) übersteuert werden
eventprio	Priorität dieses Events Wertebereich -100…100	
event_location	Ort des Ereignisses im Schutzobjekt	wird in EBÜS_Config auf der Karteikarte Bildquellen → Auslöser konfiguriert oder mit Regeln ermittelt
action_plan	Maßnahmenplan je nach Auslöser	
trigger_hints	Hinweise des Errichters zu diesem Auslöser	
gms_msg	Text der bei diesem Auslöser an das GMS oder AMS gesendet werden soll	
4. Daten, die aus der BqaInfo.txt bereitgestellt werden		
camtable	Tabelle zur Zuordnung der Kameranummern	Wird bei Aufschaltungen vom Recorder abgefragt und gespeichert
5. Daten, die mit Regeln aus dem ScriptEditor ermittelt werden		
eventprotocol	Name des Protokolls, mit dem dieses Ereignis gemeldet wurde, z.B. HeiTel, Frontel, SIA, …	legt fest, welche Tabelle aus der Liste „Quell-Protokolle“ verwendet wird
eventcode_src	Eventcode, wie er von der Bildquelle geliefert wurde	
eventuuid	Eindeutige Kennung für ein Ereignis zur Zuordnung der Dateien dieses Events	
eventtime	Diese Zeit wird vom EventManager aus den Eventdaten ermittelt	// Lokalzeit → in UTC umrechnen: format **eventtime**=%**eventtime**%:{0:Loc2Utc}
alarmcam	Nummer der zum Alarm zugehörigen Kamera gemäß eindeutiger Bildquellenzählung. Falls die Nummer nicht eindeutig oder nicht verfügbar ist, wird der Kameraname angegeben.	
displaymessage	Meldungstext der angezeigt werden soll	
eventinfo	Zusatzinformationen, die die Eventquelle zu diesem Ereignis geliefert hat	
routinecallinterval	Soll-Abstand zwischen Routinerufen in Sekunden für die Routinerufauswertung	
6. Daten, die mit den Tabellen des EventManagers ermittelt werden		
eventcomment_src	Zum **eventcode_src** zugehöriger Eintrag aus der Spalte Kommentar aus der Tabelle „Quell-Protokoll **eventprotocol**“	Hersteller- oder protokollspezifische Beschreibung des Ereignisses
eventcode_gen	Generischer Eventcode gemäß VdS2465-5 mit Erweiterungen von accellence	Normierter Eventcode aus der Haupttabelle des EventManagers
eventcomment_gen	Hinweistext zu diesem Ereignis gemäß VdS2465-5 mit Ergänzungen accellence	Beschreibt in Klartext, worum es bei dieser Art von Ereignis geht
eventcode_dst	Eventcode für das Zielprotokoll (übersetzt mit der Tabelle „Ziel-Protokoll“	Wird an das Management-System gesendet
eventcomment_dst	Eintrag aus der Spalte „Kommentar“	aus der Tabelle „Ziel-Protokoll“

Bei Bedarf können mit den Regeln im ScriptEditor weitere Variablen definiert werden, die ergänzende Informationen enthalten.
Alle diese Variablen können im EventManager unter Live Event Protokollierung angesehen und u.a. an Management-Systeme weitergeleitet werden.

12. Support / Hotline

Haben Sie noch Fragen zu EBÜS?

Dann wenden Sie sich bitte

- telefonisch unter 0511 - 277.2490
- per E-Mail an support@accellence.de

an unsere Hotline. Wir sind Werktags von 9:00-17:00 Uhr zu erreichen.

Aktuelle Informationen zu EBÜS finden Sie stets unter → www.ebues.de.

Wir wünschen Ihnen viel Erfolg bei Ihrer Arbeit mit EBÜS und stehen für Ihre Wünsche und Fragen jederzeit gern zu Ihrer Verfügung.