1) Einleitung:

Das in dieser Anleitung beschriebene Projekt, bestehend aus Hardwarekomponenten und Softwarekomponenten für die Homematic CCU (CCU1 wird nicht unterstützt, getestet wurden: CCU2 & Raspberrymatic) ermöglicht den direkten Versand einer SMS (bspw. für eine Alamierung) durch die CCU. Der Versand basiert damit ausschließlich auf lokalen Komponenten und ist weder von einem aktiven Ethernet, einer Internetanbindung oder von Drittanbietern abhängig.
Das System ist also gerade für Alarmmeldungen prädestiniert.
Für die CCU2 werden die benötigten Kerneltreiber für die USB-Serial-Adapter Unterstützung automatisiert geladen.
Hinweis: Bei einem Wechsel zwischen CCU2 und Raspberry muss das AddOn nicht deinstalliert und neu installiert werden, da es auf die Hardware dynamisch reagiert.

Das auf der CCU zu installierende Addon-Paket arbeitet als Hintergrundprozess auf der CCU und stellt die Kommunikation zur CCU-Weboberfläche her. Hierbei bietet es folgende Möglichkeiten:
a) Regelmäßige Überwachung des GSM-Modems und der Verbindung (inklusive Ermittlung der Signalstärke) zum Mobilfunkprovider
b) Message-Queue
c) Versand der einzelnen Messages an multiple Empfänger
d) Optional: Automatisiertes Starten des GSM-Modems bei Nutzung von Raspberrymatic (s. Punkt 3)

Der Informationsaustausch zwischen dem Addon und der CCU erfolgt komplett über Homematic-Systemvariablen, so dass die Funktionalität des SMS-Versandes voll kompatibel zur Homematic-Welt ist.

Ab Version 2.0 bietet der Messenger auch die Möglichkeit des SMS-Empfangs, um so per SMS Homematic-Steuerungsbefehle auszulösen. Weitere Infos unter 3a vi und 3b 8.

2) Benötigte Hardware-Komponenten:

a. 1 x Siemens TC35-Modul mit serieller Anbindung:
SainSmart-TC35-Modul bei Amazon
Bitte unbedingt diese neuere Version des TC35-Moduls nutzen, da hier der Start des GSM-Modems automatisiert erfolgt.
Bitte kontrollieren, ob der auf dem Board enthaltene Jumper zwischen den PINs mit der Beschriftung IGT und AUTO gesetzt ist. Nur dann startet das enthaltene GSM-Modem automatisch

Hinweis: Es können mit hoher Wahrscheinlichkeit auch andere GSM-Modems genutzt werden, da die Befehle die von diesem AddOn an das GSM-Modul gesendet werden generalisiert sind und somit auch zu anderen Modulen kompatibel sind. Eine Garantie kann mangels Test natürlich nicht gegeben werden.

b. 1 x Seriell-USB-Adapter
Ugreen-Adapter bei Amazon
Bitte möglichst diesen Adapter verwenden (Profilic-Chip). Mit einem anderen Adapter hatte ich das Problem, das die Kommunikation teilweise gestört war und fehlerhafte Zeichen auf der seriellen Schnittstelle auftauchten.
Das AddOn beinhaltet für die CCU2 Treiber für USB-Serial-Adapter mit FTDI und Profilic-Chipsatz. Unter Raspberrymatic stehen diese standardmäßig zur Verfügung.

c. 1 x Netzteil (5V / 2A) passend für SainSmart-TC35-Modul
Wichtig: Das Netzteil muss mindestens 2A aufweisen, da ansonsten der SMS-Versand nicht gewährleistet ist.

d. 1 x SIM-Karte (am besten eine PrePaid-Karte, da diese ja lediglich für den SMS-Versand dient).

3) Konfiguration der CCU:

a. Systemvariablen :

Diese werden von dem AddOn beim Start automatisiert bzw. bei neueren Versionen des AddOns bei Bedarf aktualisiert.
Das AddOn kann so konfiguriert werden, das entweder Systemvariablen in deutscher oder englischer Sprache und mit deutschen oder englischen Inhalten (bspw. für Statusmeldungen) generiert werden.
Anbei eine Aufstellung der erzeugten Systemvariablen (in Klammern = englische Sprachversion):

i. *Messenger_Modus(Messenger_DaemonMode)* => als Logikwert

=> Info: Gibt den Status des Hintergrundprozesses an.
Nach Start des Messengers wird diese Systemvariable auf „aktiv“ gesetzt. Durch Setzen auf „inaktiv“ wird der Messenger beendet.

ii. *Messenger_LetzterPruefStatus*(*Messenger_LastCheckState)* => als Werteliste

=> Info: Diese Variable beinhaltet das Ergebnis der letzten Prüfung des GSM-Modems und der Verbindung zum Mobilfunkprovider

iii. *Messenger_LetzterSendeStatus(Messenger_LastSendState)* => als Werteliste

=> Info: Diese Variable beinhaltet das Ergebnis des jeweils letzten SMS-Versandes.

iv. *Messenger_LetzteSignalstaerke*(*Messenger_LastSignalStrength)* => als Werteliste

=> Info: Diese Variable beinhaltet die Signalstärke beim letzten SMS-Versand bzw. bei der letzten Prüfung des GSM-Modems.

v. *Messenger_Warteschlange(Messenger_Queue)* => als Zeichenkette

=> Info: In dieser Variable sind die zum Versand gewünschten SMS-Messages einzutragen. Mehrere Einzelmessages sind mit einem Doppelpunkt zu trennen. Sobald eine Message hinterlegt wird, wird der SMS-Versand gestartet.

vi. *Messenger_SMSBefehle(Messenger_SMSCommands)* => als Werteliste

=> Info: Sobald ein SMS-Text mit identischem Inhalt zu einem Wert innerhalb dieser Werteliste eingeht, wird diese Systemvariable auf diesen Wert gesetzt.

Beispiel:
Die Werteliste beinhaltet folgende Einträge „Befehl1“ und „Befehl2“. Geht nun per SMS mit dem Text „Befehl1“, so wird die Systemvariable auf den Wert 0 (also „Befehl1“) gesetzt, geht eine Textnachricht mit dem Text „Befehl2“ ein, so wird die Systemvariable auf den Wert 1 (also „Befehl2“) gesetzt.
Somit ist es also möglich per SMS-Nachricht Homematic-Events zu triggern.

Anmerkungen:
1.) Der SMS-Text muss von der Schreibweise (also auch Groß-/Kleinschreibung) her absolut identisch zu einem der Werte in der Werteliste sein. Sonst wird die SMS ignoriert und bei aktiviertem Logging lediglich ein entsprechender Fehler protokolliert.

2.) Alle eingehenden SMS werden nach dem Auslesen automatisch aus dem SMS-Speicher gelöst, da üblicherweise nur 20 „Slots“ zur Verfügung stehen.

3.) Die SMS und somit das Setzen der Systemvariable erfolgt in der Reihenfolge in der die SMS empfangen werden.

vii. *Messenger_SMSEmfangText(Messenger_SMSReceiveText)* => als Zeichenkette

=> Info: Letzter Textinhalt einer empfangenen SMS

viii. Messenger_SMSReceiveSenderID(Messenger_SMSEmfangAbsenderID) => als Zeichenkette

=> Info: In dieser Variable wird die Absenderkennung bei SMS-Empfang abgelegt

b. Konfiguration/Start/Stop des Hintergrundprozesses:
Der im AddOn enthaltene Hintergrundprozess wird bspw. über ein Script, wie unten dargestellt, gestartet. Beendet wird der Hintergrundprozess indem die Systemvariable „Messenger_DaemonModus“(„Messenger_DaemonMode“) auf „inaktiv“ gesetzt wird.

Beispielscript:

```
var stderr;
var stdout;
system.Exec("/etc/config/addons/messenger/messenger.tcl 1 0815
+49xxxxxxxx:+49xxxxxxxx 5 /dev/ttyUSB0 1 ger 5
+49xxxxxxxx:+49xxxxxxxx &");
```

Infos zu den Übergabeparametern:
1. Parameter:
Debug-Mode (0 = Kein Logging / 1 = Logging nur Fehler / 2 = Komplettes Logging) => Default = 1

2. Parameter:
SIM-Pin => Falls die SIM-Karte eine Pin zur Freischaltung benötigt, dann die Pin angeben, sonst nur ein x. Ich empfehle die Pin zu deaktivieren.

3. Parameter:
Empfänger-Mobilnummern. Multiple Nummern sind durch einen Doppelpunkt zu trennen. Aufbau: +49xxxxxxxx:+49xxxxxxxx

4. Parameter:
Anzahl der maximalen Sendeversuche. Kommt es beim Versand zu Problemen/Fehlern, so kann hier die Anzahl der Versuche festgelegt werden => Default = 5

5. Parameter: Device-Pfad zum ComPort des GSM-Modems (sofern keine anderen USB-Devices an der CCU angeschlossen sind, handelt es sich um /DEV/TTYUSB0)

6. Parameter:
Frequenz des Modemchecks in Stunden (hierbei wird geprüft, ob das Modem ansprechbar ist und ob eine Verbindung zum Provider aufgebaut wird) => Default: 1 Stunde

7. Parameter:
Sprache der Systemvariablen und deren Inhalt.
ger = Deutsch
eng = Englisch

8. Parameter:
Frequenz des SMS-Empfangscheck in Sekunden
=> 0 = deaktiviert
=> Minimaler Wert ist 5 Sekunden. Dies aus Sicherheitsgründen, da die Kommunikation mit dem GSM-Modem seriell erfolgt und das Auslesen und Löschen von SMS entsprechende Zeit benötigt.

9. Parameter (Optional):
Erlaubte SMS-Absendernummern für SMS-Empfang (mehrere Nummern müssen mit : getrennt werden).
Wird dieser Parameter nicht angegeben, so gelten alle Absendernummern als berechtigt.

c. Installation des beigefügten Addons auf der CCU:
Das AddOn (Datei: messenger.tar.gz) wird wie üblich über die Systemsteuerung/Zusatzsoftware der CCU installiert:

4) Sonstiges zur Nutzung / Wichtige Hinweise

a) Versenden von SMS-Messages:
Beschreiben der Systemvariable „Messenger_Warteschlange“(„Messenger_Queue“) mit dem gewünschten Text. Ein Doppelpunkt dient zur Trennung mehrerer Einzel-Messages die separat gesendet werden sollen. Alle Nachrichten werden immer an alle angegebenen Empfänger-Mobilnummern versendet!

c) Logfile:
Das Logfile befindet sich unter /etc/config/addons/messenger/messenger.log

d) CUxD:
Falls CUxD parallel genutzt werden soll, darf das entsprechende USB-Device des GSM-Modems nicht von CUxD „connected“ werden. Bitte CUxD daher nach folgenden Regeln konfigurieren:
a) Falls neben dem GSM-Modem **keine** weiteres USB-Device angeschlossen sind: In der CUxD-Konfiguration den Parameter TTYPARAM=NONE eingetragen.
b) Falls neben dem GSM-Modem weitere USB-Devices angeschlossen sind: In der CUxD-Konfiguration den Parameter TTYPARAM= entsprechend der für das jeweilige Device benötigten Konfiguration eintragen. Für das GSM-Modem darf **kein** entsprechender Eintrag existieren!