============================================================================

Allgemeine Informationen:

============================================================================

Als Protokoll für die Ereignisnachrichtenversendung wird UDP/IP über den Defaultport 50000 (es sind aber auch alle anderen zulässigen Portnummern möglich) verwendet. Für die initiale, per Broadcast gesendete Profilinformation kommt Port 50001 per Default als Zielport zur Verwendung. Die Einheiten selbst bestimmen ihre IP-Adresse über DHCP. Adressbereich für die Einheiten ist 192.168.254.0/24, wobei die Adresse 192.168.254.1 für den Master (PC) reserviert ist. Die maximale Länge einer Ereignisnachricht ist - mit Rücksicht auf Speicherplatzbeschränkungen bei den Einheiten - auf 1408 Byte beschränkt. Jeder Sender eines Ereignisses hat sicherzustellen, dass er nur so viele Ereignisse zu einem Empfänger sendet, wie dieser verarbeiten kann, ohne dass es zu einem Ereignisverlust kommt. D.h. insbesondere bei Downloads ist auf eine Bestätigung ("ack") der gesendeten Nachrichten zu warten, bevor die nächste Nachricht gesendet wird.

Um die XML-Auswertung insbesondere auf Einheitenseite einfach zu halten, sind Elementtyp-, Entity- und sonstige Deklarationen nicht zulässig. D.h. alles, was mit "<!" beginnt und keine DOCTYPE-Deklaration ist

(innerhalb deren allerdings auch keine Deklarationen zulässig sind), ist nicht zulässig.

Ebenso sind außer "<?xml...?>" keine Processing-Instructions zulässig.

Innerhalb von Attributwerten sind keine Entity-Referenzen (also "&#...;") zulässig.

============================================================================

Beispiele:

============================================================================

Auch wenn im nachfolgenden Leerzeichen und Zeilenendezeichen zur besseren Lesbarkeit verwendet werden, sind diese in den eigentlichen Nachrichten nicht zu verwenden.

Die Namespace-Angabe wird in den folgenden Beispielen weggelassen, d.h. korrekterweise müsste statt

<epm> ... </epm>

das folgende angegeben werden:

<epm xmlns="http://www.hamburg.krauth-online.de/XMLSchema/epm"> ... </epm>

Die Beispiele können wie folgt getestet werden, wenn "epm.xsd" diese Schemadefinition enthält und "example.xml" das Beispiel:

xmllint (minus)(minus)schema epm.xsd example.xml

- Beim (Neu-)Start einer Einheit sendet diese periodisch per Broadcast Kurzinformationen über die von ihr unterstützten Funktionseinheiten:

<epm uid="0" ack="yes" home="192.168.254.14:50000">

<unit name="power" />

</epm>

Über das Attribut "home" wird angegeben, an welche Adresse und welchen Port Anforderungen für die entsprechende Einheit zu senden sind.

Antwort des Masters, um die Broadcastaussendung zu beenden:

<epm uid="0"><unit><ack /></unit></epm>

Allgemein gilt, dass ein nicht angegebener Name einer Funktionseinheit ("unit") für alle Funktionseinheiten gilt, sofern dies sinnvoll ist. Sofern Einheiten nur eine Funktionseinheit unterstützen, kann der Name der Funktionseinheit problemlos weggelassen werden.

- Anforderung von Basisinformationen einer Einheit (hier: hypothetisches Netzteil mit seriellen Schnittstellen):

<epm><unit><get>info</get></unit></epm>

Antwort:

<epm><unit name="superpower"><info /></unit></epm>

Nähere Informationen für diese Einheit können dann mit

<epm><unit name="superpower"><get>info</get></unit></epm>

angefordert werden. Die Antwort lautet dann beispielsweise:

<epm><unit name="superpower">

<info>

<versions>

<software>1234</software>

<hardware>5</hardware>

<softwareinfo>V 1.4.2TEST (23.7.2004)</softwareinfo>

<hardwareinfo>superpower zero-energy-module</hardwareinfo>

</versions>

<capabilities>

<capability type="download" />

<capability type="power" />

<capability type="serial">

<subtype id="0">rs232</subtype>

<subtype id="1">rs232 rs485</subtype>

</capability>

</capabilities>

<description>Netzteil mit serieller Schnittstelle</description>

</info>

</unit></epm>

Ist in Info-Tag das Attribut "ready" auf einen Wert ungleich "yes" gesetzt, ist die Einheit nur eingeschränkt oder gar nicht betriebsbereit. Nähere Informationen können dann mit

<epm><unit name="superpower"><get>readystate</get></unit></epm>

angefordert werden. Eine mögliche Antwort wäre dann:

<epm><unit name="superpower">

<readystate>

<state>restricted</state>

<description>Temperatursensor 1 defekt</description>

</readystate>

</unit></epm>

Die Standardfähigkeiten "get", "ack", "info" sowie "readystate" müssen von jeder Einheit unterstützt werden und werden daher nicht in der Liste der Fähigkeiten aufgeführt.

- Für einen Drucker, der ein getrenntes Nachladen von Code und Zeichensätzen erlaubt, könnte die Antwort auf die allgemeine Informationsanfrage wie folgt lauten:

<epm>

<unit name="printer"><info /></unit>

<unit name="printerfonts"><info /></unit>

</epm>

Um jetzt Informationen über die Zeichensätze zu bekommen, können diese mit

<epm><unit name="printerfonts"><get>info</get></unit></epm>

angefordert werden. Die Antwort lautet dann:

<epm><unit name="printerfonts">

<info>

<versions>

<software>1234</software>

</versions>

<capabilities>

<capability type="download" />

<capabilities>

</info>

</unit></epm>

- Eine Touchscreen mit zusätzlicher Tastatur (also Input-Overkill) und Soundausgabe könnte auf

<epm><unit><get>info</get></unit></epm>

Die folgende Antwort geben:

<epm>

<unit name="touchdisplay"><info /></unit>

<unit name="touchsound"><info /></unit>

</epm>

Dann könnte

<epm><unit name="touchdisplay"><get>info</get></unit></epm>

folgende Antwort liefern:

<epm><unit name="touchdisplay">

<info>

<versions>

<software>1234</software>

<hardware>5</hardware>

</versions>

<capabilities>

<capability type="download" />

<capability type="touch" />

<capability type="keyboard" />

<capabilities>

</info>

</unit></epm>

und

<epm><unit name="touchsound"><get>info</get></unit></epm>

diese Antwort liefern:

<epm><unit name="touchsound">

<info>

<versions>

<software dependson="touchdisplay">1234</software>

</versions>

<capabilities>

<capability type="download" />

<capability type="sound" />

<capabilities>

</info>

</unit></epm>

Damit wären Code und Klänge getrennt nachladbar. Durch das Attribut "dependson" wird angegeben, dass beim Nachladen des Codes (also für "touchdisplay") die Downloaddaten für "touchsound" beeinflusst (d.h. in diesem Fall gelöscht) werden. Damit kann ein Abhängigkeitsbaum und eine Downloadreihenfolge bestimmt werden.

- Werden nur Informationen über den Druckercode benötigt, könnten diese wie folgt angefordert werden:

<epm><unit name="printer"><get>info</get></unit></epm>

Die Antwort lautet dann:

<epm><unit name="printer">

<info>

<versions>

<software>1234</software>

<hardware>5</hardware>

</versions>

<capabilities>

<capability type="download" />

<capability type="candevice" />

<capabilities>

</info>

</unit></epm>

- Einmalige Anforderung der Netzteiltemperaturinformationen:

<epm><unit><get>power:temperature</get></unit></epm>

Antwort:

<epm><unit name="power">

<power><temperature>45 46 45</temperature></power>

</unit></epm>

- Periodische Anforderung des Wegzählerstands, falls mehr als 10m gefahren wurden oder falls mehr als 2s vergangen sind. Außerdem sollen die Ereignisse mit einer Sequenznummer und einem (relativen) Zeitstempel versehen und an 192.168.64.98 Port 12345 geschickt werden. Die Wegzählerereignisnachrichten sollen mit der Kennung 5566 versehen werden. Eine Bestätigung der Abarbeitung der Anforderung soll erfolgen.

<epm dest="192.168.64.98:12345" uid="5566"

withseqno="yes" withreltime="yes" ds="10m" dt="2s" ack="yes">

<unit><get>wheel:wheelstate</get></unit>

</epm>

Antwort (an Absender der Anforderung):

<epm uid="5566"><unit name="wheel"><ack cmd="get" /></unit></epm>

Antwort (an 192.168.64.98 Port 12345):

<epm uid="5566" seqno="1" reltime="0ms">

<unit name="wheel">

<wheel>

<wheelstate distance="55m" beacon="no" dooropen="no"></wheelstate>

</wheel>

<unit>

</epm>

<epm uid="5566" seqno="2" reltime="123ms">

<unit name="wheel">

<wheel>

<wheelstate distance="65m" beacon="no" dooropen="no"></wheelstate>

</wheel>

</unit>

</epm>

...

- Beenden der obigen periodischen Aussendung:

<epm uid="5566" reply="never"><unit>

<get>wheel:wheelstate</get>

</unit></epm>

- Codedownload für das Netzteil (mit Bestätigung der Übertragung):

<epm uid="1" ack="yes">

<unit name="power">

<download state="start">

<data><hex>112233...</hex></data>

</download>

</unit>

</epm>

Antwort:

<epm uid="1"><unit name="power"><ack cmd="download" /></unit></epm>

oder im Fehlerfall:

<epm uid="1"><unit name="power">

<ack cmd="download" error="parameter" />

</unit></epm>

Nächstes Datenpaket:

<epm uid="2" ack="yes">

<unit name="power">

<download state="intermediate">

<data><hex>112233...</hex></data>

</download>

</unit>

</epm>

Antwort:

<epm uid="2"><unit name="power"><ack cmd="download" /></unit></epm>

Letztes Datenpaket:

<epm uid="576" ack="yes">

<unit name="power">

<download state="final">

<data><hex>112233...</hex></data>

</download>

</unit>

</epm>

Antwort:

<epm uid="576"><unit name="power"><ack cmd="download" /></unit"</epm>

- Der Aufbau der Download-Daten-Datei ist wie folgt:

4 Byte Dateikennung ("XMLB")

2 Byte Länge des Headers (Big-Endian)

Header im binXML-Format (siehe "binXML.c")

eigentliche Download-Daten

Aus Gründen der Einfachheit sollte genau die komplette Headerinformation im ersten ("start") Download-Paket übertragen werden, um eine Defragmentierung auf Einheitenseite zu vermeiden. Die Länge des Headers wird nur benötigt, um die Headerinformationen von den eigentlichen Download-Daten zu trennen. Diese Länge ist aber nicht Bestandteil der zur der Einheit übertragenen Header-Information. Für die XML-Schema-Definition des Download-Headers siehe "dataheader.xsd".

- Aussenden einer IBIS-Nachricht (als IBIS-Master) ohne Erwartung einer Antwort:

<epm><unit><ibis mode="master"><msg>hP1234</msg></ibis></unit></epm>

- Aussenden einer IBIS-Nachricht (als IBIS-Master) mit Erwartung einer Antwort in max. 500ms:

<epm><unit><ibis mode="master" timeout="500ms">

<msg>a0</msg>

</ibis></unit></epm>

Antwort (falls keine Antwort erfolgt, ist das "reply"-Element leer):

<epm><unit name="ibis"><ibis timeout="500ms">

<reply>a2</reply>

</ibis></unit></epm>

- Vorbereiten des IBIS-Slave-Betriebs mit automatischer Antwort "a2" auf die Master-Nachricht "a0":

<epm><unit><ibis mode="slave">

<msg>a0</msg><reply>a2</reply>

</ibis></unit></epm>

Bei Erhalt der Master-Nachricht "a0" wird dann folgendes Ereignis ausgelöst:

<epm><unit name="ibis"><ibis><msg>a0</msg></ibis></unit></epm>

- Aussendung einer Broadcast-Nachricht über den seriellen Bus:

<epm><unit><serbus dev="0"><msg>t5</msg></serbus></unit></epm>

- Aussendung einer Nachricht an Einheit 1 über den seriellen Bus, wobei der Antwort dann noch die eigentliche Nachricht angefügt werden soll:

<epm><unit>

<serbus dev="1" withmsg="yes"><msg>t5</msg></serbus>

</unit></epm>

Antwort:

<epm><unit name="serbus">

<serbus dev="1"><msg>t5</msg><reply>t6</reply></serbus>

</unit></epm>

- Aussendung von Zeichen über eine serielle Schnittstelle, wobei empfangene Zeichen als Ereignis zurückgemeldet werden sollen:

<epm uid="5" reply="onchange"><unit>

<serial id="0"

baud="9600" bits="8" parity="N" stopbits="1" type="rs232">

<data>Dies ist ein Test.</data>

</serial>

</unit></epm>

Weitere Zeichen können dann wie folgt gesendet werden:

<epm uid="5" reply="onchange"><unit>

<serial id="0"><data>Dies ist ein Test.</data></serial>

</unit></epm>

Sobald Zeichen empfangen wurden, werden diese wie folgt versendet:

<epm uid="5"><unit name="superpower"><serial id="0">

<data>Empfangene Zeichen</data>

</serial></unit></epm>

Ein Schließen der seriellen Schnittstelle erfolgt dann mit:

<epm uid="5" reply="never"><unit><serial id="0" /></unit></epm>

- Senden von Daten an den CAN-Drucker:

<epm><unit><candevice dev="12345678">

<obj id="0637"><hex>000100008000</hex></obj>

<obj id="0620"><hex>00010000</hex></obj>

<obj id="0610">Hallo Leute, 5 &lt; 10.</obj>

<obj id="0642"><hex>00</hex></obj>

</candevice></unit></epm>

- Abfrage der Druckersoftwareversion beim CAN-Drucker:

<epm><unit>

<candevice dev="12345678"><obj id="0107" /></candevice>

</unit></epm>

Antwort ist dann:

<epm><unit name="can"><candevice dev="12345678">

<obj id="0107"><hex>AABBCCDD</hex></obj>

</candevice></unit></epm>

- Um alle Nachrichten über einen zentralen Proxy laufen zu lassen, müsste dieser die Broadcastnachrichten von allen angeschlossenen Einheiten auswerten und daraus dann eigene simulierte Einheiten bilden. Um diese von den echten Einheiten zu unterscheiden, muss der Proxy das Attribut "isproxy" bei seinen generierten Broadcastnachrichten einfügen. Die Applikation kann damit die vom Proxy generierten Broadcastnachrichten eindeutig erkennen und nur auf diese reagieren.

Alternativ kann der Proxy auch auf die Aussendung der Broadcastnachrichten verzichten, wenn seitens der Applikation die Adresse und der Port des Proxies a priori bekannt sind. In diesem Fall simuliert der Proxy eine "Super"-Einheit, die den gesamten Funktionsumfang aller angeschlossenen echten Einheiten beinhaltet. Problematisch wird es hier allerdings dann, wenn mehrere gleichartige echte Einheiten angeschlossen sind, da die seitens der Applikation, die nur über den Proxy darauf zugreift, nicht mehr unterschieden werden können. Es sei denn, der Proxy führt eine eindeutige Umbenennung der Funktionseinheiten durch.

- Um Einheiten der gleichen Art, aber mit unterschiedlicher Hardware- ausprägung (z.B. Fahrerterminal mit ALGRA- oder Folientastatur) zu erkennen, kann folgende Einheitenbezeichnung gewählt werden:

"EINHEIT:AUSPRÄGUNG"

Also beispielsweise "display:algra" bzw. "display:simple".

- Der Aufbau der Download-Daten ist wie folgt:

2 Byte Länge des Headers (Big-Endian)

Header im binXML-Format (siehe "binXML.c")

eigentliche Download-Daten

Aus Gründen der Einfachheit sollte die komplette Headerinformation im ersten ("start") Download-Paket übertragen werden, um die Notwendigkeit einer Defragmentierung auf Einheitenseite zu vermeiden.

Für die XML-Schema-Definition des Download-Headers siehe "dataheader.xsd".