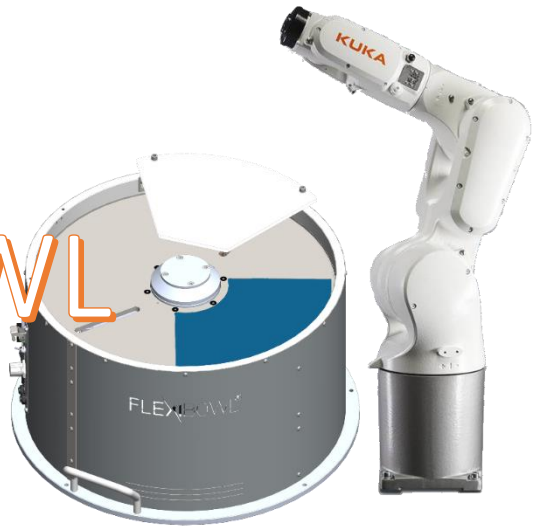
KUKA FLEXIBOWL PLUGIN



Questo Plugin è nato con l'idea di comunicare in maniera rapida e sicura con il flexibowl tramite i robot Kuka, madiante il software Work Visual 5.
Il Plugin necessita della licenza KUKA Ethernet KRL per il corretto funzionamento.

FlexiBowl®

KUKA
WorkVisual
SOLUTIONS FOR AUTOMATION

Work Visual

STEP 1:

Panoramica KUKA.Ethernet KRL

Funzioni KUKA.Ethernet KRL è un pacchetto tecnologico ricaricabile con le seguenti funzioni:

- Scambio di dati tramite l'EKI
- Ricezione di dati XML di un sistema esterno
- Invio di dati XML a un sistema esterno
- Ricezione di dati binari di un sistema esterno
- Invio di dati binari a un sistema esterno

Proprietà
- Controllo robot e sistema esterno come client o server
- Configurazione di collegamenti tramite il file di configurazione basato su XML
- Configurazione di "Messaggi evento"
- Controllo di collegamenti tramite un ping sul sistema esterno.
- Lettura e scrittura di dati dell'interprete Submit
- Lettura e scrittura di dati dell'interprete robot

Comunicazione I dati vengono trasferiti tramite il protocollo TCP/IP. L'utilizzo del protocollo
UDP/IP è possibile, ma non consigliato (protocollo di rete privo di collegamento, ad es. nessun riconoscimento della perdita di dati).

STEP 2:

Configurazione di un collegamento Ethernet

Panoramica Un collegamento Ethernet viene configurato tramite un file XML. Per ogni collegamento,
nella cartella C:\KRC\ROBOTER\Config\User\Common\EthernetKRL
del controllo robot deve essere definito un file di configurazione.
Il nome del file XML è al tempo stesso la chiave di accesso in KRL.
Esempio: …\EXT.XML —> EKI_INIT("EXT")

Struttura XML per caratteristiche di collegamento

Descrizione Nella sezione <EXTERNAL> … </EXTERNAL>, possono essere definite le impostazioni per il sistema esterno:

I file XML sono "case sensitive". Considerare le maiuscole/minuscole.

```
<ETHERNETKRL>
<CONFIGURATION>
<EXTERNAL></EXTERNAL>
<INTERNAL></INTERNAL>
</CONFIGURATION>
<RECEIVE>
<ELEMENTS></ELEMENTS>
</RECEIVE>
<SEND>
<ELEMENTS></ELEMENTS>
</SEND>
</ETHERNETKRL>
```

Work Visual

STEP 3:

Di seguito verrà indicato come realizzare il file di configurazione EthernetKRL, nominato ServerKrl.xml

```
ETHERNETKRL>
  <CONFIGURATION>
    <EXTERNAL>
      <TYPE>Client</TYPE>
    </EXTERNAL>
    <INTERNAL>
     <ENVIRONMENT>Submit</ENVIRONMENT>
      <IP>192.168.1.10</IP>
      <PORT>77775</PORT>
      <ALIVE Set_Flag="2"/>
      <Messages Display="disabled" Logging="error"/>
    </INTERNAL>
  </CONFIGURATION>
  <RECEIVE>
    <RAW>
      <ELEMENT Tag="Buffer" Type="STREAM" Set_Flag="1" Size="64" EOS="13,10" />
    </RAW>
  </RECEIVE>
  <SEND>
    <RAW>
      <ELEMENT Tag="Buffer" Type="STREAM" Set_Flag="1" Size="64" EOS="13,10"/>
    </RAW>
  </SEND>
</ETHERNETKRL>
```

Work Visual

STEP 4:

Di seguito verrà indicato il codice per la comunicazione con il Flexibowl tramite L'EthernetKRL. Tale script puo essere richiamato per il movimento del Flexibowl oppure messo in un task parallelo, e regolata l'esecuzione tramite un semaforo.
Il codice riceverà un comando da eseguire, e restituira una stringa con la risposta dal flexibowl.

```
&ACCESS  RVO
&COMMENT USER specified PLC program
DEF  FlbPlugin ( )
  DECL EKI_STATUS RET

  CHAR Bytes[100]
  CHAR TMP[100]
  CHAR command[100]
  int lenght
  char returnFlb[100]
  int returnok
  int movment
  int found
  CHAR moving [24]

LOOP

 ;INIZIALIZZO
 FOR i=(1) TO (128)
   Bytes[i]=0
 ENDFOR

 ;flag per avvio movimento
 wait for $FLAG[3]

 ;setto il comando che vogliamo inviare
 ;for example
 command="QX2"

 ;CREO LA MIA STRINGA (CHAR(0)+CHAR(7)+COMMAND+CHAR(13))
 lenght = StrLen(command[])
 TMP[]="0"
 returnok = stradd(Bytes[],TMP[])
 TMP[]="7"
 returnok = stradd(Bytes[],TMP[])
 TMP[]=command[]
 returnok = stradd(Bytes[],TMP[])
 TMP[]="13"

 returnok = stradd(Bytes[],TMP[])

 ;abilito la connessione
 RET=EKI_Init("ServerKrl")
 RET=EKI_Open("ServerKrl")
```

```
;in attesa di una comunicazione da un client
WAIT FOR $FLAG[2]
;invio il comando al flexibowl
RET = EKI_Send("ServerKrl",Bytes[])
;attendo al risposta del flexibowl
WAIT FOR $FLAG[1]
RET=EKI_GetString("ServerKrl","Buffer",Bytes[])

;analizzo il comando che ho mandato
movment = StrFind(1, command, "QX", #NOT_CASE_SENS)
returnok = StrFind(1, command, "%", #NOT_CASE_SENS)

if((movment>0)and(returnok>0)) then
  ;ho inviato un comando di movimento e mi ha risposto % quindi WAITMOVE
  moving=1;
  While (moving=="1")
    ;INIZIALIZZO
    FOR i=(1) TO (128)
      Bytes[i]=0
    ENDFOR

    ;CREO LA MIA STRINGA (CHAR(0)+CHAR(7)+COMMAND+CHAR(13))
    lenght = StrLen("RS")
    TMP[]="0"
    returnok = stradd(Bytes[],TMP[])
    TMP[]="7"
    returnok = stradd(Bytes[],TMP[])
    TMP[]="SC"
    returnok = stradd(Bytes[],TMP[])
    TMP[]="13"
    returnok = stradd(Bytes[],TMP[])
    ;invio il comando al flexibowl
    RET = EKI_Send("ServerKrl",Bytes[])

   ;attendo al risposta del flexibowl
   WAIT FOR $FLAG[1]

   RET=EKI_GetString("ServerKrl","Buffer",Bytes[])
   found = StrFind(1, Bytes[], "F")
   if(found>0) then
     moving=1
   else
     moving=0
   endif

  endwhile
  returnFlb="Done"
else
  returnFlb=Bytes[]
endif
  RET = EKI_ClearBuffer("ServerKrl",Bytes[])

ENDLOOP

END
```

Work Visual

STEP 5:

Lista dei comandi:

Action	Description
MOVE	Moves the feeder the current parameters.
MOVE-FLIP	Moves the feeder and activates Flip simultaneously
MOVE-BLOW-FLIP	Moves the feeder and activates Flip and blow simultaneously
MOVE-BLOW	Moves the feeder and activates Flip simultaneously
SHAKE	Shakes the feeder with the current parameters
LIGHT ON	Light on
LIGHT OFF	Light off
FLIP	Flip
BLOW	Blow
QUICK_EMPTING	Quick Emptying Option
RESET_ALARM	Reset Alarm and enable the motor

Command	Description
QX2	Move
QX3	Move - Flip
QX4	Move - Blow - Flip
QX5	Move - Blow
QX5	Shake
QX7	Light on
QX8	Light off
QX9	Blow
QX10	Flip
QX11	Quick Emptying Option
QX12	Reset Alarm