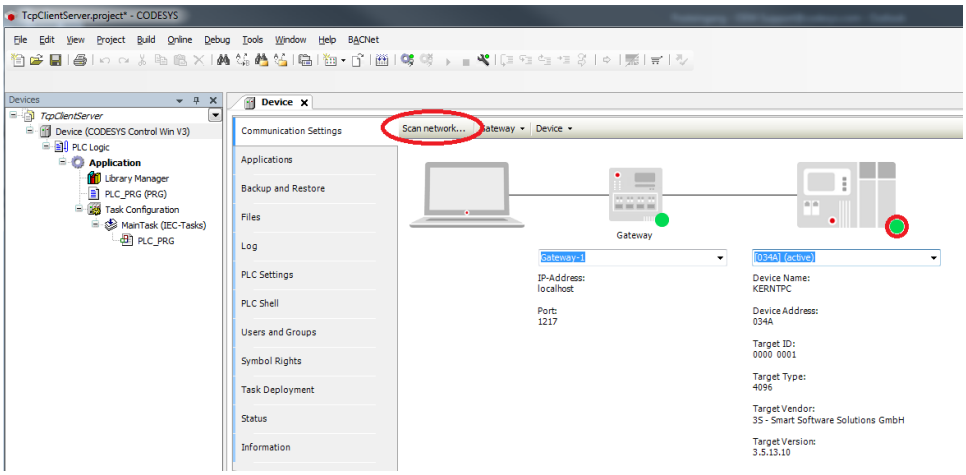
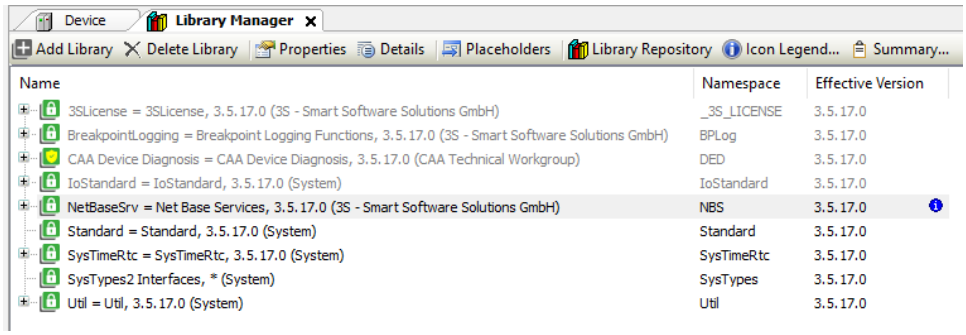
TCP: Example for Server and Client

- Create a "Standard project" and select *CODESYS Control Win V3* as the device.
- Define the target system by means of the *Network scan*.



As of SP16:

- Open the *Library Manager* and add the following libraries:
  Net Base Services
  SysTypes2 interfaces
  SysTimeRtc
  Util



- *Create a global variable list named gvlSetting and define the following variables:*

Declaration

```
VAR_GLOBAL CONSTANT
        gc_wMaxTelegram         : INT := 15;
        gc_uiPort                 : UINT := 50001;
END_VAR
VAR_GLOBAL
        gc_stIpAddr                 : String(19) := '192.168.99.109';
END_VAR
```

Adapt the IP address and the port to your system.

- *Create a new POU named TcpServer and adapt it as follows:*

Declaration

```
VAR CONSTANT
        c_wMaxTelegram        : INT := 15;
END_VAR
VAR
        ipAddress                : NBS.IPv4Address;
        fbTcpConnection        : NBS.TCP_Connection;
        fbTcpServer                : NBS.TCP_Server;
        fbTcpRead                : NBS.TCP_Read;
        fbTcpWrite                : NBS.TCP_Write;


        abyRx                        : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;
        abyTx                        : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;

        iIndex                        : INT;
        xRead                        : BOOL := TRUE;
        xWrite                        : BOOL := TRUE;
        xAckTelegram        : BOOL;
        xBlockAck                : BOOL;
        udiRead                        : DINT;
END_VAR
```

Implemen
tation

```
IF fbTcpRead.xReady THEN
        IF (fbTcpRead.udiCount = (gvlSetting.gc_wMaxTelegram + 1)) THEN
                IF ((abyRx[0] = 87) AND  (abyRx[1] = 68) AND (abyRx[2] = 58) AND (abyRx[3] = 32))
THEN // 'WD: ' = Watchdog-Telegram
                        FOR iIndex := 0 TO c_wMaxTelegram DO
                                abyTx[iIndex] := 0;
                        END_FOR
                        // 'ACK: ' = Acknowledge-Telegram
                        abyTx[0] := 65;
                        abyTx[1] := 67;
                        abyTx[2] := 75;
                        abyTx[3] := 58;
                        abyTx[4] := 32;
                        // Receive-Counter
                        abyTx[5] := abyRx[4];
                        abyTx[6] := abyRx[5];
                        abyTx[7] := abyRx[6];
                        abyTx[8] := abyRx[7];
                        xWrite := TRUE;
                END_IF
                xWrite := TRUE;
        END_IF
ELSIF fbTcpRead.xError THEN
        xRead := FALSE;
END_IF


fbTcpWrite(xExecute := xWrite AND NOT xBlockAck, itfConnection := fbTcpConnection, udiSize :=
SIZEOF(abyTx), pData := ADR(abyTx),  udiTimeOut := 0);
IF fbTcpWrite.xDone OR fbTcpWrite.xError THEN
        xWrite := FALSE;
END_IF
```

- Create a new POU named *TcpClient* and adapt it as follows:

Declaration

```
VAR CONSTANT
        c_tInterval               : TIME := T#1S;
        c_udiInterval        : UDINT := 3 * TIME_TO_UDINT(c_tInterval)/1000;
END_VAR
VAR
        ipAddress                : NBS.IPv4Address;
        fbTcpClient               : NBS.TCP_Client;
        fbTcpRead                : NBS.TCP_Read;
        fbTcpWrite                : NBS.TCP_Write;

        abyTx                        : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;
        abyRx                        : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;

        fbBlink                        : BLINK := (TIMELOW := c_tInterval, TIMEHIGH :=
c_tInterval);
        xBlink                        : BOOL; // Memory of the last state of PLC_PRG.fbBlink

        udiVal                        : UDINT;
        pudiVal                        : POINTER TO BYTE;
        iIndex                        : INT;
        xConnect                : BOOL;

        xRead                        : BOOL;
        xMissingAck                : BOOL;
        udiResult                : UDINT;
        udiLastAck                : UDINT;
        udiNow                        : UDINT;
        udiRead                        : UDINT;
        eRErrorID                : NBS.ERROR;
        eWErrorID                : NBS.ERROR;
END_VAR
```

Implementation

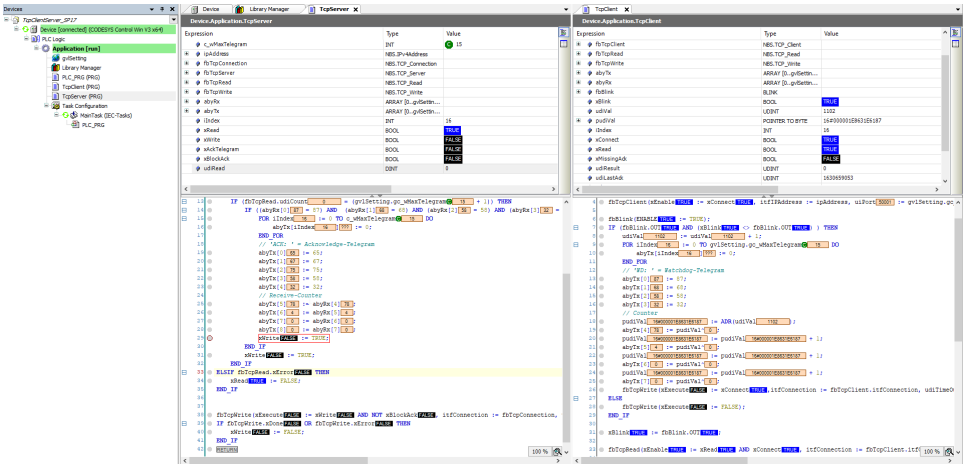
```
IF NOT fbTcpClient.xActive THEN
	ipAddress.SetInitialValue(ipAddress := gvlSetting.gc_stIpAddr);
END_IF
fbTcpClient(xEnable := xConnect, itfIPAddress := ipAddress, uiPort := gvlSetting.gc_uiPort,
udiTimeOut := 0);

fbBlink(ENABLE := TRUE);
IF (fbBlink.OUT AND (xBlink <> fbBlink.OUT) ) THEN
	udiVal := udiVal + 1;
	FOR iIndex := 0 TO gvlSetting.gc_wMaxTelegram DO
		abyTx[iIndex] := 0;
	END_FOR
	// 'WD: ' = Watchdog-Telegram
	abyTx[0] := 87;
	abyTx[1] := 68;
	abyTx[2] := 58;
	abyTx[3] := 32;
	// Counter
	pudiVal := ADR(udiVal);
	abyTx[4] := pudiVal^;
	pudiVal := pudiVal + 1;
	abyTx[5] := pudiVal^;
	pudiVal := pudiVal + 1;
	abyTx[6] := pudiVal^;
	pudiVal := pudiVal + 1;
	abyTx[7] := pudiVal^;
	fbTcpWrite(xExecute := xConnect,itfConnection := fbTcpClient.itfConnection, udiTimeOut :=
0, udiSize := SIZEOF(abyTx), pData := ADR(abyTx));
ELSE
	fbTcpWrite(xExecute := FALSE);
END_IF

xBlink := fbBlink.OUT;

fbTcpRead(xEnable := xRead AND xConnect, itfConnection := fbTcpClient.itfConnection, udiSize :=
SIZEOF(abyRx), pData := ADR(abyRx), udiCount => udiRead);

IF fbTcpRead.xReady THEN
	IF (fbTcpRead.udiCount = (gvlSetting.gc_wMaxTelegram + 1)) THEN
		IF ((abyRx[0] = 65) AND  (abyRx[1] = 67) AND (abyRx[2] = 75) AND (abyRx[3] = 58)
AND (abyRx[4] = 32)) THEN // 'WD: ' = Watchdog-Telegram
			udiLastAck := SysTimeRtc.SysTimeRtcGet(udiResult);
		END_IF
	END_IF
ELSIF fbTcpRead.xError THEN
	fbTcpRead(xEnable := FALSE);
END_IF


IF NOT fbTcpClient.xActive AND NOT fbTcpClient.xBusy AND NOT fbTcpClient.xDone THEN
	xConnect := TRUE;
ELSIF fbTcpClient.xDone THEN
	xConnect := FALSE;
END_IF

udiNow := SysTimeRtc.SysTimeRtcGet(udiResult);

IF (udiNow > (udiLastAck + c_udiInterval)) THEN
	xMissingAck := TRUE;
ELSE
	xMissingAck := FALSE;
END_IF
```

- Adapt the POU *PLC_PRG* as follows:

Implemen
tation

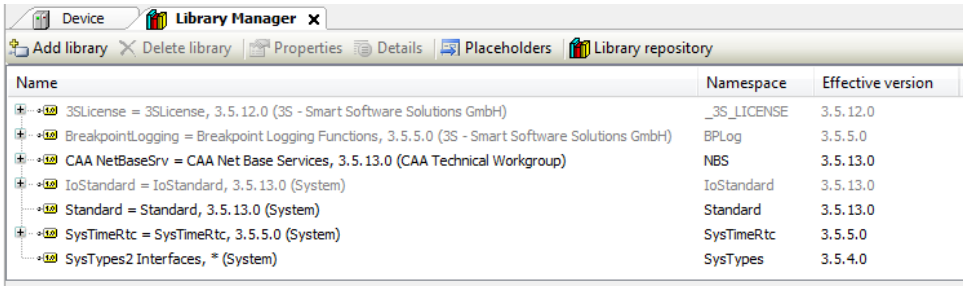
```
TcpServer();
TcpClient();
```

- Load the project to the controller and start it. Set The variable *TcpClient.xRead* to *TRUE*.



Up to SP16:

- Open the *Library Manager* and add the following libraries:
  CAA Net Base Services
  SysTypes2 interfaces
  SysTimeRtc
  Util



- Create a global variable named *gvlSetting*.

  Adapt *gvlSetting* as follows:

Declaration

```
{attribute 'qualified_only'}
VAR_GLOBAL CONSTANT
    gc_uiPort        : UINT := 50001;
    gc_stIpAddr      : NBS.IP_ADDR := (sAddr := '192.168.99.74');

    gc_wMaxTelegram  : INT := 15; // Length o the telegram
END_VAR
```

Adapt the IP address and the port to your system.

- Create a new POU named *TcpServer* and adapt it as follows:

Declaration

```
VAR CONSTANT
    c_wMaxTelegram  : INT := 15;
END_VAR
VAR
    fbTcpConnection : NBS.TCP_Connection;
    fbTcpServer     : NBS.TCP_Server;
    fbTcpRead       : NBS.TCP_Read;
    fbTcpWrite      : NBS.TCP_Write;

    abyRx           : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;
    abyTx           : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;

    iIndex          : INT;
    xRead           : BOOL := TRUE;
    xWrite          : BOOL := TRUE;
    xAckTelegram    : BOOL;
    xBlockAck       : BOOL;
    udiRead         : UDINT;
END_VAR
```

Implemen
tation

```
fbTcpServer(xEnable := TRUE, ipAddr := gvlSetting.gc_stIpAddr, uiPort := gvlSetting.gc_uiPort);
fbTcpConnection(xEnable := fbTcpServer.xBusy, hServer := fbTcpServer.hServer);

fbTcpRead(xEnable := fbTcpConnection.xActive, hConnection := fbTcpConnection.hConnection, szSize :
= SIZEOF(abyRx), pData := ADR(abyRx), szCount => udiRead);

IF fbTcpRead.xReady THEN
    IF (fbTcpRead.szCount = (gvlSetting.gc_wMaxTelegram + 1)) THEN
        IF ((abyRx[0] = 87) AND  (abyRx[1] = 68) AND (abyRx[2] = 58) AND (abyRx[3] = 32)) THEN //
'WD: ' = Watchdog-Telegram
            FOR iIndex := 0 TO c_wMaxTelegram DO
                abyTx[iIndex] := 0;
            END_FOR
            // 'ACK: ' = Acknowledge-Telegram
            abyTx[0] := 65;
            abyTx[1] := 67;
            abyTx[2] := 75;
            abyTx[3] := 58;
            abyTx[4] := 32;
            // Receive-Counter
            abyTx[5] := abyRx[4];
            abyTx[6] := abyRx[5];
            abyTx[7] := abyRx[6];
            abyTx[8] := abyRx[7];
            xWrite := TRUE;
        END_IF
        xWrite := TRUE;
    END_IF
ELSIF fbTcpRead.xError THEN
    xRead := FALSE;
END_IF

fbTcpWrite(xExecute := xWrite AND NOT xBlockAck, hConnection := fbTcpConnection.hConnection,
szSize := SIZEOF(abyTx), pData := ADR(abyTx),  udiTimeOut := 0);
IF fbTcpWrite.xDone OR fbTcpWrite.xError THEN
    xWrite := FALSE;
END_IF
```

- Create a new POU named *TcpClient* and adapt it as follows:

Declaration

```
VAR CONSTANT
    c_tInterval   : TIME := T#1S;
    c_udiInterval : UDINT := 3 * TIME_TO_UDINT(c_tInterval)/1000;
END_VAR
VAR
    fbTcpClient   : NBS.TCP_Client;
    fbTcpRead     : NBS.TCP_Read;
    fbTcpWrite    : NBS.TCP_Write;

    abyTx         : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;
    abyRx         : ARRAY [0..gvlSetting.gc_wMaxTelegram] OF BYTE;

    fbBlink       : BLINK := (TIMELOW := c_tInterval, TIMEHIGH := c_tInterval);
    xBlink        : BOOL; // Memory of the last state of PLC_PRG.fbBlink

    udiVal        : UDINT;
    pudiVal       : POINTER TO BYTE;
    iIndex        : INT;
    xConnect      : BOOL;

    xRead         : BOOL := TRUE;
    xMissingAck   : BOOL;
    udiResult     : UDINT;
    udiLastAck    : UDINT;
    udiNow        : UDINT;
    udiRead       : UDINT;
END_VAR
```

Implemen
tation

```
fbTcpClient(xEnable := xConnect, ipAddr := gvlSetting.gc_stIpAddr, uiPort := gvlSetting.
gc_uiPort, udiTimeOut := 0);

fbBlink(ENABLE := TRUE);
IF (fbBlink.OUT AND (xBlink <> fbBlink.OUT) ) THEN
    udiVal := udiVal + 1;
    FOR iIndex := 0 TO gvlSetting.gc_wMaxTelegram DO
        abyTx[iIndex] := 0;
    END_FOR
    // 'WD: ' = Watchdog-Telegram
    abyTx[0] := 87;
    abyTx[1] := 68;
    abyTx[2] := 58;
    abyTx[3] := 32;
    // Counter
    pudiVal := ADR(udiVal);
    abyTx[4] := pudiVal^;
    pudiVal := pudiVal + 1;
    abyTx[5] := pudiVal^;
    pudiVal := pudiVal + 1;
    abyTx[6] := pudiVal^;
    pudiVal := pudiVal + 1;
    abyTx[7] := pudiVal^;
    fbTcpWrite(xExecute := xConnect,hConnection := fbTcpClient.hConnection, udiTimeOut := 0,
szSize := SIZEOF(abyTx), pData := ADR(abyTx));
ELSE
    fbTcpWrite(xExecute := FALSE);
END_IF

xBlink := fbBlink.OUT;

fbTcpRead(xEnable := xRead AND xConnect, hConnection := fbTcpClient.hConnection, szSize := SIZEOF
(abyRx), pData := ADR(abyRx), szCount => udiRead);

IF fbTcpRead.xReady THEN
    IF (fbTcpRead.szCount = (gvlSetting.gc_wMaxTelegram + 1)) THEN
        IF ((abyRx[0] = 65) AND  (abyRx[1] = 67) AND (abyRx[2] = 75) AND (abyRx[3] = 58) AND
(abyRx[4] = 32)) THEN // 'WD: ' = Watchdog-Telegram
            udiLastAck := SysTimeRtc.SysTimeRtcGet(udiResult);
        END_IF
    END_IF
ELSIF fbTcpRead.xError THEN
    fbTcpRead(xEnable := FALSE);
END_IF

IF NOT fbTcpClient.xActive AND NOT fbTcpClient.xBusy AND NOT fbTcpClient.xDone THEN
    xConnect := TRUE;
ELSIF fbTcpClient.xDone THEN
    xConnect := FALSE;
END_IF

udiNow := SysTimeRtc.SysTimeRtcGet(udiResult);

IF (udiNow > (udiLastAck + c_udiInterval)) THEN
    xMissingAck := TRUE;
ELSE
    xMissingAck := FALSE;
END_IF
```

- Adapt the POU *PLC_PRG* as follows:

Implementation

```
TcpServer();
TcpClient();
```

- Load the project to the controller and start it.