

```

//-----
#pragma hdrstop

#include "UnitDMSockets.h"
#include "UnitFormMain.h"
#include "UnitUtils.h"
#include "UnitThreadWorking.h"
//-----
#pragma package(smart_init)
#pragma classgroup "Vcl.Controls.TControl"
#pragma resource "*.dfm"
TdmSockets *dmSockets;
//-----
__fastcall TdmSockets::TdmSockets(TComponent* Owner)
: TDataModule(Owner)
{
}
//-----
void __fastcall TdmSockets::csOut1Connect(TObject *Sender, TCustomWinSocket *Socket)
{
    int intReqId = 0;
    if(Sender == csOut1)
    {
        intReqId = 1;
    }
    else if(Sender == csOut2)
    {
        intReqId = 2;
    }
    else if(Sender == csOut3)
    {
        intReqId = 3;
    }
    else if(Sender == csOut4)
    {
        intReqId = 4;
    }

    AddToLog(Socket, "CNC[" + IntToStr(intReqId) + "]");
    if(csOut1->Active && csOut2->Active && csOut3->Active && csOut4->Active)
    {
        formMain->buttonSync->Enabled = false;
    }
}
//-----
void __fastcall TdmSockets::csOut1Disconnect(TObject *Sender, TCustomWinSocket *Socket)
{
    int intReqId = 0;
    if(Sender == csOut1)
    {
        intReqId = 1;
    }
    else if(Sender == csOut2)
    {
        intReqId = 2;
    }
    else if(Sender == csOut3)
    {
        intReqId = 3;
    }
    else if(Sender == csOut4)
    {
        intReqId = 4;
    }

    AddToLog(Socket, "DSC[" + IntToStr(intReqId) + "]");
    if(!csOut1->Active && !csOut2->Active && !csOut3->Active && !csOut4->Active)
    {
        formMain->buttonSync->Enabled = true;
        Screen->Cursor = crDefault;
    }
}

```

```

//-----
void __fastcall TdmSockets::csOut1Error(TObject *Sender, TCustomWinSocket *Socket,
    TErrorEvent ErrorEvent, int &ErrorCode)
{
    int intReqId = 0;
    if(Sender == csOut1)
    {
        intReqId = 1;
    }
    else if(Sender == csOut2)
    {
        intReqId = 2;
    }
    else if(Sender == csOut3)
    {
        intReqId = 3;
    }
    else if(Sender == csOut4)
    {
        intReqId = 4;
    }

    AddToLog(Socket, "ERR[" + IntToStr(intReqId) + "]");
    if(!csOut1->Active && !csOut2->Active && !csOut3->Active && !csOut4->Active)
    {
        formMain->buttonSync->Enabled = true;
        Screen->Cursor = crDefault;
    }

    ErrorCode = 0;
}
//-----
void __fastcall TdmSockets::csOut1Read(TObject *Sender, TCustomWinSocket *Socket)
{
    int intReqId = 0;
    if(Sender == csOut1)
    {
        intReqId = 1;
    }
    else if(Sender == csOut2)
    {
        intReqId = 2;
    }
    else if(Sender == csOut3)
    {
        intReqId = 3;
    }
    else if(Sender == csOut4)
    {
        intReqId = 4;
    }

    unsigned long ulTime;

    int intBytesReceived = Socket->ReceiveBuf(&ulTime, 4);
    // if(intBytesReceived != 4)
    // {
    //     // Грешен отговор
    //     String str = "Error: " + IntToStr(intBytesReceived) + " bytes received";
    //     AddToLog(str);
    //     return;
    // }

    ulTime = ntohl(ulTime);

    String strTime;
    strTime.printf(L"%0X", ulTime);
    AddToLog(Socket, "RPL[" + IntToStr(intReqId) + "]::[" + strTime + "]");

    intPendingReq--;

    // СИМУЛАЦИЯ НА ОБРАБОТКА
    // многозадачно обслужване за избягване на сериализацията
    // на паралелните клонове
    //

```

```

std::thread threadWorking(DoWork, Socket, intReqId);
threadWorking.detach();          // развързване на дъщерната нишка от основната
////////////////////////////////////
}
//-----
void __fastcall TdmSockets::SendBatchOfReq(void)
{
    /* TODO : Формиране на пакет заявки */
    if(csOut1->Active || csOut2->Active || csOut3->Active || csOut4->Active)
    {
        String str = "There are pending requests. Cannot send batch.";
        AddToLog(str);
        return;
    }

    csOut1->Open(); intPendingReq++;
    csOut2->Open(); intPendingReq++;
    csOut3->Open(); intPendingReq++;
    csOut4->Open(); intPendingReq++;
}
//-----

```