Development of .NET software module for communication with PLC

Janos Orgovanyi [a]

[a] *Delta-Tech Mérnöki Iroda Kft., Ipari Park 5, Balassagyarmat, 2660, Hungary, janos.orgovanyi@deltatech.hu*

Abstract

Today, large companies are striving for productivity, which is why in most places they already replace human resources with robots, since the robots work based on a specific program. As a result, they are less likely to produce scrap products than humans. Although we are still a long way from eliminating human intervention from factory areas, multinational companies are already striving for this. The topic of this article is the development of a software whose task is to communicate with automated systems and to control the system. Overall, automation is important for productivity, that is, for companies to strive for productivity. However, it must be taken into account that, in many ways, it is inappropriate to exclude human intervention from production. To some extent, the module I wrote also contributes to the displacement of the human factor, but at the same time it also requires the intervention of a user.

Keywords: *programming; PLC; .NET*

.NET szoftver modul fejlesztése PLC-vel kommunikációhoz

Orgovanyi Janos [a]

[a] *Delta-Tech Mérnöki Iroda Kft., Ipari Park 5, Balassagyarmat, 2660, Hungary, janos.orgovanyi@deltatech.hu*

Absztrakt

Napjainkban a nagyvállalatok a termelékenységre törekednek ezért a legtöbb helyen már az emberi erőforrásokat robotokkal helyettesítik, mivel a robotok egy meghatározott program alapján dolgoznak. Ezáltal kisebb eséllyel gyártanak selejtes termékeket, mint az emberek. Bár azért attól még talán messze állunk, hogy a gyárak területeiről az emberi beavatkozásokat eltüntessük, de a multinacionális cégek már erre törekednek. A cikk témája egy olyan szoftver fejlesztése, amelynek a feladata az automatizált rendszerekkel való kommunikáció, illetve a rendszer vezérlése. Összességében fontos az automatizálás a termelékenység szempontjából, vagyis az, hogy a cégek termelékenységre törekednek. Azonban figyelembe kell venni, sok szempontból nem megfelelő, hogyha az emberi beavatkozásokat kiakarják szorítani a termelésből. Az emberi tényező kiszorításához valamilyen szinten az általam írt modul is hozzájárul, ugyanakkor igényli egy felhasználó beavatkozását is.

Kulcsszavak: programozás; PLC; .NET

1. Bevezető

Az információs technológia rohamos előrehaladása mélyreható hatást gyakorol mind a kis, mind a nagyvállalatok gazdasági helyzetére. Ebben az új gazdasági paradigmában elengedhetetlen az automatizáció bevezetése ahhoz, hogy a vállalatok fenntartsák termelékenységüket és versenyképességüket. Kiemelkedő számú tanulmány rámutat arra, hogy az IT és automatizációs megoldások alkalmazása jelentősen hozzájárulhat a vállalatok hatékonyságának javításához és piaci pozíciójuk megerősítéséhez (Badawi U., et al, 2020)

A globális gazdaságban való részvétel a vállalatokat olyan komplex kihívások elé állítja, amelyekre az informatikai szakértők nyújthatnak megoldásokat. A vállalatok ma már nem csupán egy országban működnek, hanem szerte a világon létrehozzák leányvállalataikat, akár egy országon belül is több régióban. Ezáltal kialakul az adatok diverzifikációja és integrációja, amelyeknek kezelésében a jól képzett informatikusok elengedhetetlen szerepet töltenek be. Az informatikusok segítségével a vállalatok olyan komplex rendszereket hozhatnak létre, amelyek lehetővé teszik a hatékony adatmegosztást és integrációt, így optimálisan kezelhetik globális tevékenységeiket (Fernandez D. & Aman A., 2018).

Az informatikusok szerepe azonban túlmutat az adatok kezelésén. Egyidőben több helyről érkező adatok bevitele és naprakészen tartása a rendszerben kihívást jelent, amelynek megoldására az informatikusoknak kell megfelelő mechanizmusokat és alkalmazásokat kifejleszteniük. Az informatikai szakemberek kreativitása és szaktudása révén a vállalatok hatékonyabban működhetnek és versenyképesebbek lehetnek a dinamikus üzleti környezetben (Badawi U., et al, 2020)

Ezen felül, az informatikusok által megvalósított automatizációs megoldások és fejlesztések segítségével a vállalatok nem csak hatékonyabban működhetnek, hanem gyakran új lehetőségeket is felfedezhetnek az üzleti tevékenységük fejlesztésére és a piaci részesedésük növelésére (Vishnoi S., 2019).

Mindezen érvek alapján látható, hogy az informatika és az automatizáció elengedhetetlen eszközeivé váltak a vállalatok hatékony működésének és fejlődésének. Az informatikusok szakértelme és innovációja kulcsfontosságú ahhoz, hogy a vállalatok a gyorsan változó üzleti környezetben versenyképesek maradjanak és kiaknázzák az informatikai fejlődésben rejlő lehetőségeket.

2. A modul kifejlesztésére vonatkozó igények ismertetése

Mivel több ipari cég is gyártósorok tervezésével és gyártásával foglalkozik, így sok PLC – vel kell egyszerre kommunikálni, melyek nem biztos, hogy egy gyártótól származnak.

A modullal kapcsolatosan az alábbi fő feladatok kerültek megfogalamzásra:

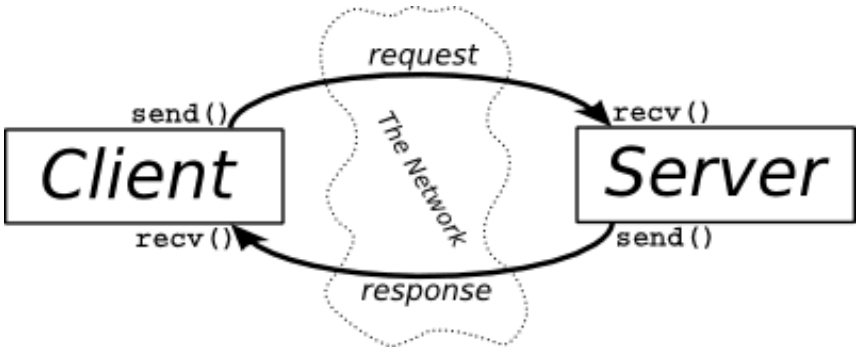
- Kapcsolódni a PLC-hez (TCP/IP kommunikációval)
- A Kapcsolat létrejöttének ellenőrzése (FINS Protokollal)
- „.csv1” kiterjesztésű fájl beolvasása, amiben a lekérendő memóriacímek szerepelnek
- A Különböző memória területek értékeinek lekérése

- A memóriacímekből jövő információk feldolgozása és tárolása
- A kiválasztott memóriacímterület értékének módosítása

A modul a .NET keretrendszerben a „Visual Studio” szoftverrel Visual Basic programnyelven kerül implementálásra.

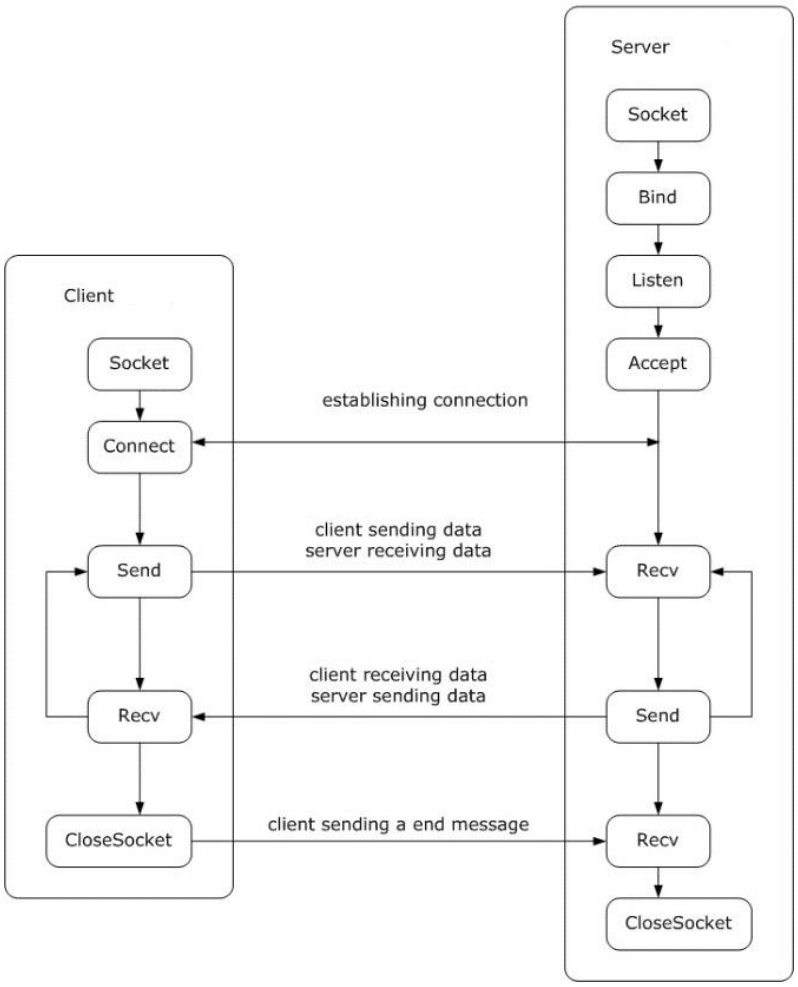
3. A modul terve

Már magának a tervezésnek a folyamata sem volt egyszerű feladat, mivel a hálózati közeget is figyelembe kellett venni a tervezés során. A hálózaton a megvalósítás végére egy szerver kliens kommunikációnak kell majd folynia, ez az 1.ábrán látható.



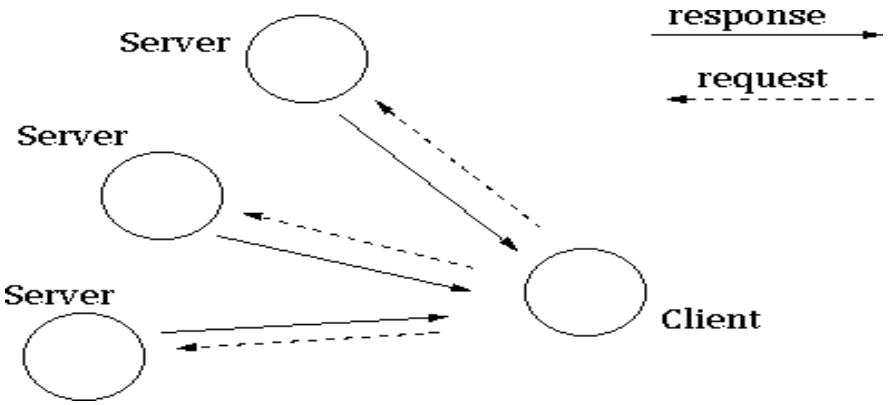
1. ábra: Szerver-kliens kapcsolat

Ebben az esetben a szerver feladatát a PLC látja majd el, ezért az elkészített tervből a szerver program elkészítésére nem lesz szükség a fejlesztés folyamán. A TCP kapcsolat felépítése az 2. ábrán látható.



2. ábra: Szerver-kliens kapcsolat

Az előzetes terv alapján az üzenet küldés modellje egyszerűsített szisztéma alapján valósult meg. Azonban a későbbiek folyamán az üzenet feldolgozásának folyamatát újra kellett tervezni, mert a választ fel kell dolgozni annak érdekében, hogy a kliens akár több PLC-hez is csatlakozhasson, ahogy azt a 3. ábra is szemlélteti.



3. ábra: Szerver-kliens kapcsolat

4. A modul vázának elkészítése

A modul fejlesztése során az első lépés egy TCP/IP protokollon alapuló kommunikáció létrehozása volt, amely egy chat programhoz hasonló. Ennek érdekében két programot kellett írni, amelyek a hálózaton kommunikáltak. A gépek lokális hálózatot használtak a teszteléshez, és a fejlesztés során a hatékony és gyors kommunikációt kellett összeegyeztetni a TCP/IP protokoll szabályaival. Az így létrejött programokkal szemben a kód tisztasága és hatékonysága is elvárás volt, amelyhez kutatómunkát is kellett végezni. A programok fejlesztése a felhasználói felülettel kezdődött, majd a vezérlők kódjának megírásával folytatódott. A tesztelés során azonban probléma merült fel, amikor a szerverprogram nem kapott IP-címet, és kiderült, hogy a forráskódok felcserélődtek. Ezeket a hibákat kijavítva elkészült a működőképes modul, amelynek további optimalizációra van szüksége, például a több szálra bontás lehetőségének kezelésére a fő szál terhelésének csökkentése érdekében. A szerver és kliens programok felhasználói felületei különböző fázisokban készültek el, a teszteléshez és a későbbi fejlesztésekhez szükség volt rájuk. Az ábrák bemutatják a fejlesztett felületeket, amelyek változhatnak a modul jövőbeni feladatai alapján.

5. Implementáció

5.1. A szerver működésének megvalósítása

A modul fejlesztése során először a TCP/IP protokollon alapuló kommunikációt kellett megvalósítani, majd a párhuzamos végrehajtást kellett beépíteni a modul futása során. A szerverprogramon belül a párhuzamosítás a nagyobb terhelés lekezelése miatt volt szükséges, és a Visual Basic nyelv beépített "BackgroundWorker" komponensével lett megoldva. A szerverprogram további optimalizációjára kiderült, hogy a PLC sajátos jellegének megfelelően szükséges, és a kliensprogramot is fejleszteni kellett, hogy üzeneteket küldjön a PLC-nek. Az elküldött és fogadott üzenetek konverziójára a hexadecimális és bináris számrendszerek között a fejlesztés során figyeltek.

5.2. A PLC-vel való kommunikáció megvalósítása

A modul fejlesztése során először a TCP/IP protokollon alapuló kommunikációt kellett megvalósítani, majd a párhuzamos végrehajtást kellett beépíteni a modul futása során. A szerverprogramon belül a párhuzamosítás a nagyobb terhelés lekezelése miatt volt szükséges, és a Visual Basic nyelv beépített "BackgroundWorker" komponensével lett megoldva. A

szerverprogram további optimalizációjára kiderült, hogy a PLC sajátos jellegének megfelelően szükséges, és a kliensprogramot is fejleszteni kellett, hogy üzeneteket küldjön a PLC-nek. Az elküldött és fogadott üzenetek konverziójára a hexadecimális és bináris számrendszerek között a fejlesztés során figyeltek.

5.3. A PLC -vel való kommunikáció megvalósítása

A PLC érkezése után meg kellett találni, hogyan csatlakozhat hozzá a modul, és hogyan lehet a hálózaton hexadecimális utasításokkal kommunikálni vele, amihez az utasítást byte-okká kellett konvertálni. A fejlesztés során a modult automatikus csatlakozásra és kapcsolatellenőrzésre kellett fejleszteni, amelyhez az ip címet és port számot is be kellett építeni, valamint a hibakezelést is megoldani. A kapcsolati üzenet elküldése és ellenőrzése során felmerülő problémákat egy beágyazott Try Catch blokk megoldotta, majd az üzenetvárás időzítését kellett optimalizálni a helyes válaszellenőrzés érdekében.

5.4. Egy fájl beolvasása

Az eddig fejlesztett program már képes automatikusan építeni és ellenőrizni a kapcsolatot a PLC-vel, valamint a memória adatokat is lekérni tőle. A feladat az, hogy a modul csak a szükséges terület értékeit kérje le a PLC-től, és ezeket egy üzenetben továbbítsa. Az adatokat egy ".csv" fájlból kellett kinyerni, majd a kezdő és végpontokat meghatározva a felhasználó által kiválasztott területet tölteni egy dinamikus méretű tömbbe. Az utolsó lépés a tömb feltöltése volt a PLC válaszának megfelelő értékekkel a tárolás céljából.

5.5. A kód objektumokra bontása

Az első lépés a kód optimalizálása volt a működés hatékonyságának ellenőrzése érdekében. A második lépésben a kódot függvényekbe és szubrutinokba kellett átrendezni, majd egy új modulba helyezni ezeket, amely jelentősen tisztította a fő ablak alatti kódterületet. A kódrendezés során felmerült hibákat a változók publikussá tételével és kapcsolatvizsgálattal kellett kezelni, így a modul új felépítése átláthatóvá tette a kódot és hatékonyabbá tette a működést. Ezen felül, az objektumokhoz kapcsolódó CIO-terület is rendezésre került a cím és bitpozíció különbségek figyelembevételével.

5.6. A modul futásának több szálra helyezése

Az első lépés a kód optimalizálása volt a működés hatékonyságának ellenőrzése érdekében. A második lépésben a kódot függvényekbe és szubrutinokba kellett átrendezni, majd egy új modulba helyezni ezeket, amely jelentősen tisztította a fő ablak alatti kódterületet. A kódrendezés során felmerült hibákat a változók publikussá tételével és kapcsolatvizsgálattal kellett kezelni, így a modul új felépítése átláthatóvá tette a kódot és hatékonyabbá tette a működést. Ezen felül, az objektumokhoz kapcsolódó CIO-terület is rendezésre került a cím és bitpozíció különbségek figyelembevételével.

Már régóta foglalkozni kellett volna külön szál futtatásával a modulban, vagyis bizonyos feladatok elvégzését egy mellékszálon tegye meg a modul, hogy a fő szál futása ne legyen akadályozva.

Így feladat a modul tekintetében az, hogy bizonyos feladatokat áthelyezzünk külön szálra. Az előzőekben megírt Objektum lekérést optimalizáljuk, amennyire csak lehet, mert mennél automatikusabb és hatékonyabb a modul a rábízott feladatok elvégzésében annál jobb.

Akkor hát az első feladat az volt, hogy a Timer függvényben lévő kódokat át legyenek helyezve egy külön szálra. Bár a függvényben is nagyon jól ellátta feladatát a kód, amellyel az egyes területek értékeit kérte le és dolgozta fel. A probléma az volt igazából, hogy minden folyamat a fő szálon, a Form-on futott keresztül és néha mikor egy folyamaton tovább tartott dolgozni akkor semmi üzenetet nem kapott a felhasználó, semmire nem tudott kattintani. Mivel a program egy feladattal foglalkozott ezzel azt a hatást keltette mintha lefagyott volna. Tehát mivel a lekérés egy folyamatos dolog, amit a program automatikusan tud csinálni, ezért azt át kellet tenni egy másik szálra. A feladat igazából annyi volt, hogy létre kellett hozni egy függvényt Sub-ot. melynek törzsébe át kellett helyezni a Timer-ben lévő kódokat, majd pedig a következő kódrészlettel, mely egy deklarálás, létre kellett hozni a külön szálat.

```
Public Req_Thread As New System.Threading.Thread(AddressOf Req_Process)
```

Ugyan erre a szára át lett helyezve az a folyamat is, amelyik az újra kapcsolódással foglalkozott így a fő szálon már szinte nem maradt semmi. Csak azoknak a folyamatoknak a vezérlői maradtak a fő szálon, amelyekkel át lehet írni az egyes címterületek értékeit. A Fő szálon továbbá még egy folyamat kapott helyet ez pedig nem más, mit az a folyamat, amely megjeleníti a felhasználó számára sz objektumok adatait.

5.7. A modul végleges formába „öntése”

Az utolsó lépésben a modul hibáit és hiányosságait kellett felmérni, mivel a folyamatos írás közben néhány alapvető funkció és paraméter vizsgálat elmaradt. Ezeket pótolni és optimalizálni kellett, valamint ki kellett vizsgálni, mely vezérlők és függvények szükségesek valóban a programban. A funkcióvizsgálat során kiderült, hogy egyes vezérlők feleslegessé váltak, például egy ComboBox, melyre új megoldás található, hogy az értékek megfelelően kerüljenek hozzárendelésre az objektumokhoz. Ezzel párhuzamosan folyt a hibák kijavítása és a funkciók fejlesztése, mivel a modul még a tesztelés és finomhangolás fázisában van.

6. Konklúzió

A fejlesztés során néhány váratlan helyzet és kihívás adódott, különösen a gyors és erőforrás-takarékos tervezés terén. A kódolási folyamat során az adatok konvertálása, különösen a FINS protokollhoz és TCP/IP kommunikációhoz való alakítás okozott problémákat. A modul működése számos teszt során ellenőrzésre került, beleértve a kapcsolatfelépítést, fájlkezelést, objektumok kezelését és üzenetküldést. Az elvárásoknak megfelelően a modul hatékonyan működik és a tesztek során számos fontos funkciója sikeresen teljesült, beleértve a kapcsolattartást és az adatlekérést. Bár még vannak apróbb hibák és optimalizációs lehetőségek, a modul önállóan is használható és cégen belül hasznos. Az elvárt funkcionalitás, például a kapcsolatfelvétel és adatlekérdezés, megvalósult. A modul fejlesztése sikeresnek tekinthető, és bár lehetnek apróbb hibák, azok az ellenőrzés során kiderülnek, így a modul felhasználás előtt áttekinthető és továbbfejleszthető.

Irodalomjegyzék

Badawi, U. A., Tayfour, M. F., Alghamdi, F. A., & Aseri, A. M. (2020). Impacts of marketing automation on business performance. *Journal of Theoretical and Applied Information Technology, 98*(11), 1957-1969.

Fernandez, D., & Aman, A. (2018). Impacts of robotic process automation on global accounting services. *Asian Journal of Accounting & Governance, 9*.

Vishnoi, S. K., Tripathi, A., & Bagga, T. (2019). Intelligent automation, planning & implementation: A review of constraints. *International Journal on Emerging Technologies, 10*(1a), 174-178.

Rövid szakmai életrajz

2018 óta szoftverfejlesztőként dolgozom a Delta-Tech Mérnöki Iroda Kft.-nél. A szakterületem traceability rendszerek tervezése és kivitelezése, amelyeket Visual Basic és SQL (MSSQL) nyelven végzek.