VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ŘÍZENÍ INTELIGENTNÍHO DOMU S VYUŽITÍM SOFTWAROVÉ IMPLEMENTACE PLC AUTOMATU

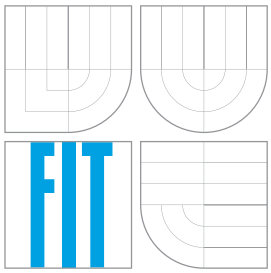
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVEL MICHAL

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ŘÍZENÍ INTELIGENTNÍHO DOMU S VYUŽITÍM SOFTWAROVÉ IMPLEMENTACE PLC AUTOMATU

INTELLIGENT BUILDING MANAGEMENT USING SOFTWARE IMPLEMENTATION OF PLC

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVEL MICHAL

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ALEŠ MARVAN

BRNO 2013

Abstrakt

Tato práce se zabývá vývojem modulu SoftPLC aplikace řídicí jednotky založené na operačním systému Linux pro firmu Elko EP. Řídicí jednotka pro sběrnici IMB je určena pro řízení inteligentních domů a je konfigurovatelná přes protokol SOAP pomocí programu iNELS Designer & Manager 3.

Abstract

This work deals with the development of an application module SoftPLC, which is running on the control unit based on the Linux operating system for the company Elko EP. The control unit for IMB is designed for intelligent building management and it is configurable via the SOAP protocol using iNELS Designer & Manager 3.

Klíčová slova

IMB, iNELS, Elko EP, inteligentní budova, Linux, POCO, řídicí jednotka, SOAP, Soft PLC, C++

Keywords

IMB, iNELS, Elko EP, intelligent building, Linux, POCO, control unit, SOAP, Soft PLC, C++

Citace

Pavel Michal: Řízení inteligentního domu s využitím softwarové implementace PLC automatu, diplomová práce, Brno, FIT VUT v Brně, 2013

Řízení inteligentního domu s využitím softwarové implementace PLC automatu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Aleše Marvana. Další informace mi poskytli Jaromír Přikryl, DiS. a Ing. Jan Sršeň. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Michal
21. května 2013

Poděkování

Rád bych poděkoval Ing. Aleši Marvanovi za konzultace a vedení práce, dále Jaromíru Přikrylovi, DiS. a Ing. Janu Sršňovi za poskytnuté informace o systému Inels firmy Elko EP, s.r.o.

Obsah

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2012/2013

Zadání diplomové práce

Řešitel: **Michal Pavel, Bc.**
Obor: Počítačové a vestavěné systémy
Téma: **Řízení inteligentního domu s využitím softwarové implementace PLC automatu**
Intelligent Building Management Software Implementation Using PLC
Kategorie: Vestavěné systémy

Pokyny:

1. Seznamte se s problematikou vývoje softwarových PLC a inteligentní elektroinstalací, která se v využívá v moderních domech.
2. Seznamte se existujícími prvky, moduly a softwarem inteligentní elektroinstalace firmy Elko EP, s.r.o.
3. Navrhněte softwarové PLC, které bude konfigurovatelné parametrizačním SW firmy Elko EP a které bude ovládat aplikaci VarStore této firmy.
4. Vámi navrženou aplikaci implementujte.
5. Otestujte vytvořenou aplikaci s Vámi vybranými prvky inteligentní elektroinstalace společnosti Elko EP. Zhodnoťte dosažené výsledky.

Literatura:

- Papazoglou, Michael P., Web services: principles and technology, ISBN 978-0-321-15555-9
- Šmejkal Ladislav, PLC a automatizace, Praha, BEN, 1999, ISBN 80-86056-58-9
- Poco libraries online documentation, http://pocoproject.org/docs-1.5.0/
- iNELS - Inteligentní elektroinstalace, Technický katalog, Elko EP, 2012, http://www.elkoep.cz/downloads/promotion_materials/iNELS_SHS_02.pdf

Při obhajobě semestrální části diplomového projektu je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Marvan Aleš, Ing.**, UITS FIT VUT
Datum zadání: 17. září 2012
Datum odevzdání: 22. května 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav inteligentních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

Kapitola 1

Úvod

Inteligentní budovy jsou v současné době módním pojmem. V článcích časopisů nejrůznějšího zaměření jsou vyzdvihovány výhody moderního bydlení v inteligentním domě. V komerčních objektech se s inteligentním řízení budov setkáváme již delší dobu a nyní začíná být standardem automatizace i v rezidenční sféře. Potřeba řídit svůj dům většinou přichází s novostavbou nebo rekonstrukcí objektu, kde máme k dispozicí více zdrojů elektřiny a tepla, chceme zabezpečit majetek proti nezvaným hostům nebo zvýšit komfort bydlení automatickými žaluziemi a ovládáním světel z kteréhokoliv místa. Tato práce se zabývá vývojem modulu SoftPLC aplikace řídicí jednotky založené na operačním systému Linux pro firmu Elko EP. Společnost Elko EP se zabývá vývojem a prodejem inteligentní elektroinstalace.

V první kapitole jsou uvedeny systémy automatizace budov, které jsou v současné době na trhu. Dále pokračuje seznámení s programovatelnými automaty PLC, které jsou využívány v automatizaci obecně a jejich novým odvětvím, takzvané Soft PLC. Na rozvoji Soft PLC má také podíl OS Linux, na konci druhé kapitoly jsou uvedeny hlavní důvody jeho rozšíření. V další kapitole následuje představení systému iNELS, který využívá společnost Elko EP. Ve čtvrté kapitole je přiblížen protokol SOAP, který je použit při implementaci modulu SoftPLC. V páté kapitole jsou shrnuty požadavky na vyvíjený řídicí systém, dále je tento systém popsán a je navržena implementace modulu SoftPLC. V kapitole Implementace jsou popsány použité datové struktury, ve kterých je uložena konfigurace modulu a v kapitole testování se nachází popis dvou testů, kterými byla ověřena funkčnost vyvíjeného modulu. V závěru jsou shrnuty dosažené výsledky.

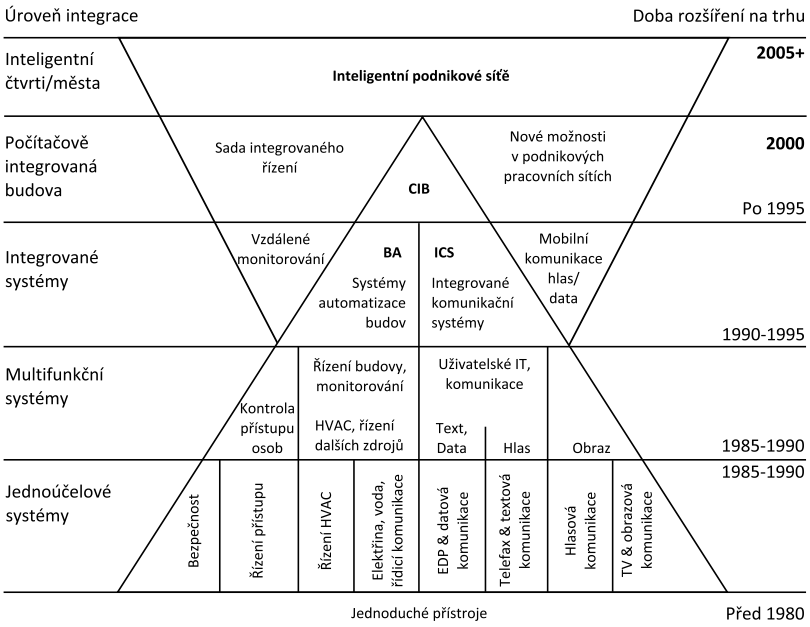
V rámci semestrálního projektu vznikla kapitola Automatizace inteligentních budov včetně problematiky PLC a využití operačního systému Linux, poté kapitoly o systému Inels a návrhu modulu SoftPLC. Diplomová práce na semestrální projekt navázala doplněním informací o normě IEC 61131 a systému Beremiz. Dále byla doplněna kapitola Návrh a přidány kapitoly o protokolu SOAP, v další kapitole byla popsána Implementace a následně Testování.

Do příloh je zařazena fotografie vývojové verze jednotky CU3, ukázka konfigurační aplikace IDM3, popis obsahu přiloženého CD a schéma SOAP rozhraní a datového modelu programu IDM3 a modulu SoftPLC.

Kapitola 2

Automatizace inteligentních budov

Za takzvané inteligentní domy jsou označovány objekty obsahující technické vybavení (zdroje tepla, elektrické energie, ovládání žaluzií, osvětlení, elektronický zabezpečovací systém apod.), která jsou řízena integrovaným řízením komunikujícím přes chytrou elektroinstalaci (drátová nebo bezdrátová sběrnice) s tímto technickým vybavením. Kromě řízení s cílem poskytnout komfort uživatelům nabízí inteligentní domy monitorování aktuálního stavu domu (například teplota, zapnuté spotřebiče, aktuální spotřeba energií) [17]. Na obrázku 2.1 je zachycen vývoj inteligentních budov spolu s využitím komunikačních a informačních sítí v těchto budovách.



Obrázek 2.1: Vývoj inteligentních budov a integrace sítí, převzato z [9]

2.1 Struktura řídicího systému

Informace pro tuto sekci byly čerpány z článku [10]. Možnosti řízení inteligentní budovy jsou určeny jak samotným řídicím systémem, tak použitým typem sběrnice, na které komunikují senzory, aktory a řídicí logika. Senzory snímají fyzikální veličinu nebo děj a převádějí ji

na informaci, kterou odesílají na sběrnici. Nejjednodušším příkladem může být tlačítko vypínače. Na informace ze senzorů reagují přímo aktory nebo podle zadaných pravidel ovlivňují činnost aktorů řídicí jednotky. Aktory vyhodnocují přijaté zprávy od senzorů, případně řídicích jednotek a reagují na ně vykonáním činnosti (mechanické). Aktory se rozlišují podle činností na spínací, stmívací, žaluziové, aktory topení apod. Řídicí systémy lze rozdělit na tři kategorie:

- jednoduché systémy s pevnou komunikací,
- proprietární (centralizované) systémy,
- otevřené (distribuované) systémy.

Podle toho, jak jsou zpracovávány informace ze senzorů, lze zpracování dat rozdělit na centralizované, decentralizované a distribuované. U centralizovaného zpracování probíhá vyhodnocení na jednom řídicím uzlu. Výhodou je konfigurace a správa systému v jednom místě. Výpadek řídicího uzlu však představuje problém pro celou budovu. Příkladem může být individuální regulace místnosti s centrálním systémem ovládání objektu. Decentralizované zpracování je význačné velkým počtem řídicích jednotek, které mohou plnit stejné funkce – jednotlivé okruhy totiž využívají vlastní výpočetní systémy, které nejsou vzájemně přímo propojeny (společná sběrnice). Na rozdíl od distribuovaného nelze v decentralizovaném systému přenést řízení z jedné jednotky na jinou. Výhodou je vysoká odolnost proti poruchám, při výpadku řídicí jednotky není činnost zbytku systému omezena. Řídicí jednotky pro decentralizované zpracování mohou být jednodušší a méně výkonné (levnější) než centralizované. Typické použití v inteligentních budovách decentralizovaného řízení je lokální regulace – např. kotle nebo individuální regulace místnosti a vzájemné sdílení dat na sběrnici, ke kterým mají přístup také uživatelé a dispečeři. Do decentralizovaných systémů patří technologie LON (Local Operating Network), EIB (European Installation Bus), z kterého po sloučení s EHS a BatiBus vychází KNX (Konnex). Dále sem patří LCN, Luxmate pro řízení osvětlení a xComfort. Příkladem původně českých standardů jsou firmy Amit nebo Elsaco. Při distribuovaném zpracování jsou výpočetní systémy okruhů vzájemně propojeny. V současné době se jedná o nejdokonalejší typ řízení. Mezi jeho přednosti oproti centralizovanému zpracování patří:

- distribuci komunikačních a datových zdrojů je možné přizpůsobit organizační struktuře budovy,
- dobře navržené distribuované zpracování je levnější, umožňuje řídit funkce budovy na úměrně výkonných a přiměřeně nákladných výpočetních prostředcích,
- distribuované zpracování umožňuje lepší škálovatelnost výpočetních prostředků.

U distribuovaných systémů je důležitá především komunikace. Jako výpočetní jednotky lze použít počítače, PLC nebo mikrokontroléry, přičemž činnost celého systému je ovlivněna fungováním a vzájemnou komunikací všech jeho částí. Pro každé použití distribuovaných systémů vznikají specifické požadavky na výkonnost použité komunikační sítě, na rychlost odezvy, množství přenášených dat atd.

2.1.1 Jednoduché řídicí systémy

Jednoduché řídicí systémy s pevnou konfigurací dovolují realizovat funkce např. sluneční a větrné automatiky s možností lokálního ovládání uživatelem. Jsou složeny z jednoúčelových jednotek s pevně danou funkcí. Může se jednat o stínicí zařízení budovy, osvětlení a stmívání, vytápění, klimatizaci, větrání apod. Jednoduché řídicí systémy jsou primárně určeny pro rezidenční sféru, případně pro malé komerční objekty s nenáročnými požadavky na řídicí systém.

2.1.2 Proprietární (centralizované) řídicí systémy

Proprietární (uzavřený protokol podléhající licenci) řídicí systémy jsou určeny pro střední až velké komerční objekty. Tyto centralizované systémy poskytují výhodný poměr cena/výkon a nabízí široké možnosti nastavení a lze tak jejich chování přizpůsobit konkrétnímu objektu. Dále jsou uvedeny nejznámější proprietární řešení využívaná v ČR a Evropě.

Moeller xComfort/Nikobus

xComfort je bezdrátový systém domovní automatizace, který pracuje na frekvenci 838 MHz (vyhrazeno pro elektroinstalaci budov). Dosah vysílače je jeden strop a dvě stěny, pokud není přijímač v dosahu, uplatní se tzv. rating a je předáván ostatními prvky k cíli. Nikobus je částečně decentralizovaný (distribuovaný) řídicí systém. V tomto systému jsou zasílány pouze příkazy typu on-off, může obsahovat maximálně 256 senzorů a tři typy inteligentních jednotek – spínací, stmívací nebo roletové jednotky. Všechny výstupy (zásuvky, spínané/stmívané osvětlení, rolety) jsou připojeny přímo na tyto jednotky. Napájecí napětí sběrnice Nikobus je 9 V. Sběrnice je stejně jako ostatní oddělena od sítě nn 230 V. Pro konfiguraci Nikobusu postačí malý šroubovák, není potřeba počítač. Každému sběrnicovému tlačítku lze bez komplikovaných programovacích technik přiřadit jednu nebo více funkcí.

ABB Ego-n

Moduly tohoto systému se připojují pomocí čtyřžilového kabelu, přičemž dva vodiče jsou vyhrazeny pro komunikaci a dva pro napájení. Topologie sběrnice je lineární a v systému se rozlišují dva typy sběrnic: *primární sběrnice* – připojení vstupů, výstupů, řídicího modulu a napájení. Řídicí modul zabezpečuje komunikaci jak mezi prvky primární sběrnice, tak mezi primární a sekundární sběrnicí a dalšími řídicími jednotkami. Na každou primární sběrnici může být připojeno maximálně 64 prvků a délka sběrnice nesmí přesáhnout 700 m. *Sekundární sběrnice* slouží pro propojení řídicích jednotek a rozhraní (Ethernet, GSM, RF modul atd.). Na sekundární sběrnici se může nacházet nejvýše osm jednotek a délka nesmí přesáhnout 2000 m. Celkově se tedy může na sběrnici nacházet 512 prvků, to omezuje použití pouze na rezidenční sféru.

iNELS

Centralizovaný systém domovní automatizace, který vyvinula firma Teco Kolín ve spolupráci se společností Elko EP. Systém využívá sběrnici CIB a řídicí jednotku typu PLC, která bude spolu se systémem popsána v následující kapitole.

2.1.3 Otevřené (decentralizované) volně programovatelné systémy

Pro použití otevřených systémů není na rozdíl od proprietárních pro použití vyžadována licence. To umožňuje výrobu specifických modulů na přání zákazníka, na které výrobce nemyslel. Poskytují tak největší volnost a nejširší možnosti při realizaci představ a požadavků na vlastnosti a chování řízeného systému. Nejvíce používané otevřené protokoly jsou LON, DALI, Open Therm, EnOcean, SMI a BACnet a KNX .

LonWorks

Decentralizovaný sběrnicový systém napodobující nervový systém. Každý uzel sběrnice může být buď senzor nebo akční člen a obsahuje univerzální čip – „neuron“ pro připojením na sběrnici. Uzly si mohou vyměňovat informace navzájem mezi sebou. Neuronový čip obsahuje tři osmibitové mikrokontroléry, paměti, časovací jednotku, vstupní a výstupní část a modul pro připojení na sběrnici. Použitá zařízení obsahují vlastní distribuovanou inteligenci. Systém LonWorks vychází ze síťového modelu ISO/OSI a je nejvíce rozšířen v USA.

DALI

DALI (Digital Addressable Lighting Interface) je otevřený standard pro řízení intenzity osvětlení. Komunikace je obousměrná – je možné intenzitu svítidla jak nastavit, tak přečíst aktuální úroveň. Na sběrnici, která se skládá ze dvou vodičů maximální délky 300 m, může být umístěno až 64 stmívacích modulů. Je určena jako doplňující sběrnice pro jiné systémy (po připojení přes vhodné rozhraní) [6].

OpenTherm

Jedná se o otevřený, na výrobci nezávislý řídicí systém a též protokol pro komunikaci systémů vytápění, klimatizace a vzduchotechniky (HVAC). Komunikace je obousměrná, typu master/slave, probíhá na dvou vodičích a je nezávislá na polaritě. Slave jednotka (bojler) poskytuje master jednotce (termostat) data i napájení. Existují standardní rozhraní pro připojení OpenTherm sběrnicím BACnet, LONWorks, Modbus, Ethernet.

EnOcean

Technologie firmy EnOcean je založena na bezdrátových modulech, které se obejdou bez přívodu napájení či baterií. Energii pro provoz získávají změnou teploty, tlaku, osvětlení nebo vibrací [5].

SMI

SMI (Standard Motor Interface) je specializovaná sběrnice pro řízení motorů. Důvodem pro vývoj protokolu SMI byla nespokojenost se zavedeným způsobem řízení pohonů v budovách. Příklady nevýhod zavedených systémů: absence inteligentního řízení (pohyb na přesnou pozici), žádná zpětná vazba od pohonů o jejich poloze a stavu, složitá kabeláž. Hlavní využití sběrnice SMI je v budovách pro inteligentní řízení pohonů rolet, žaluzií, markýz, slunečních ochran, zakrývání zimních zahrad, bazénů a požárních ochran. SMI pohony mohou být řízeny samostatnými řídicími jednotkami, PLC nebo je lze připojit do systémů jako KNX nebo LON. SMI jednotky jsou připojeny pětižilovým kabelem, na třech vodičích

je napájení pro motor a na zbylých dvou probíhá komunikace. Délka sběrnice je omezena na 350 m.

BACnet

BACnet (Building Automation and Control Network) je standardizovaný otevřený a pružný komunikační protokol pro automatizaci a řídicí systémy budov, kde si řízení a systémy mohou navzájem vyměňovat informace. BACnet lze propojit s jakýmkoliv standardizovaným protokolem a lze ho tak využít pro libovolné úlohy řízení technických zařízení budov nebo pro propojení různých nekompatibilních systémů a protokolů řízení budov.

KNX/EIB

Standardizovaný otevřený protokol pro komunikaci mezi snímači, akčními členy, regulačními a řídicími moduly. Výhodou KNX/EIB je nezávislost na jednom výrobci a tedy velký výběr komponent, nevýhodou je zatím vysoká cena. Prvky na sběrnici fungují v režimu distribuované inteligence. Počátky systému EIB (European Installation Bus) sahají do přelomu 80-90. let minulého století. Mezi zakládajícími členy asociace patřily společnosti Merten, Siemens, Jung, Gira a Berker. Protokol KNX/EIB vznikl v roce 2003 spojením systémů EIB, EHS a BatiBus. V současné době je KNX/EIB nejdůležitějším standardem pro řízení technických zařízení budov v Evropě, kde pokrývá 80 % trhu.

2.2 PLC

PLC (Programmable Logic Controller) je programovatelný automat určený pro řízení a sledování průmyslových a výrobních procesů. Na PLC jsou kladeny požadavky maximální spolehlivosti a zvýšená mechanická odolnost pro použití v průmyslovém prostředí. Programovatelné automaty jsou také základem řízení technických zařízení budov. Další text o PLC a Soft PLC byl převzat z [18].

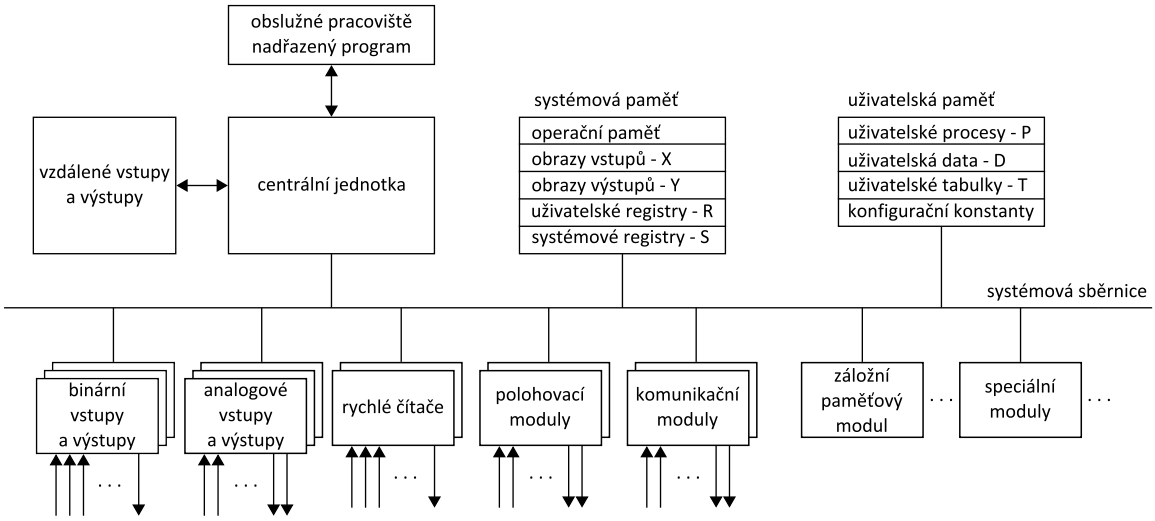
PLC jsou v automatizační technice využívány už asi 40 let. Programovatelné automaty vznikly jako náhrada za pevnou reléovou logiku. Tomu odpovídal i způsob programování pomocí jazyku kontaktních (reléových) schémat. Dnes je pro programování možné použít jazyk logických schémat, jazyk kontaktních (reléových) schémat, jazyk mnemokódů (podobné assembleru) nebo jazyk strukturovaného textu (obdoba vyšších jazyků, například C). Programování PLC je standardizováno normou IEC 61131. Provedení programovatelných automatů může být kompaktní, kdy je jeho konfigurace pevně daná nebo modulární provedení, u kterého lze zvyšovat celkovou funkčnost přidáváním rozšiřujících modulů.

2.2.1 Struktura PLC

Vnitřní struktura PLC je znázorněna na obrázku 2.2. Skládá se z centrální jednotky (procesor), paměti (systémová, uživatelská), analogových a binárních vstupů/výstupů, čítačů, polohovacích modulů a modulů pro komunikaci na různých rozhraních. Binární vstupy/výstupy mohou používat stejnosměrné nebo střídavé napětí a jsou galvanicky oddělené od napětí automatu. Hodí se pro různé spínané kontakty typu tlačítko nebo dotekové snímače polohy.

Analogové vstupy/výstupy je možné použít pro spojité snímače a převodníky, například teplotní čidla. Rychlé čítače a polohovací moduly jsou určeny pro rychlé čítání, měření sledů impulzů, měření polohy a pohybu přírůstkovými i absolutními snímači, pro nastavování polohy a řízení pohybu po naprogramované dráze. Lze je využít především pro řešení úloh

geometrického charakteru, podobně jako systémy CNC. Další moduly mohou implementovat různá rozhraní standardních protokolů, existují také počítačové moduly pro řešení náročných úloh a moduly s PID a fuzzy regulátory.



Obrázek 2.2: Schéma vnitřní struktury PLC, převzato z[18]

2.2.2 Soft PLC

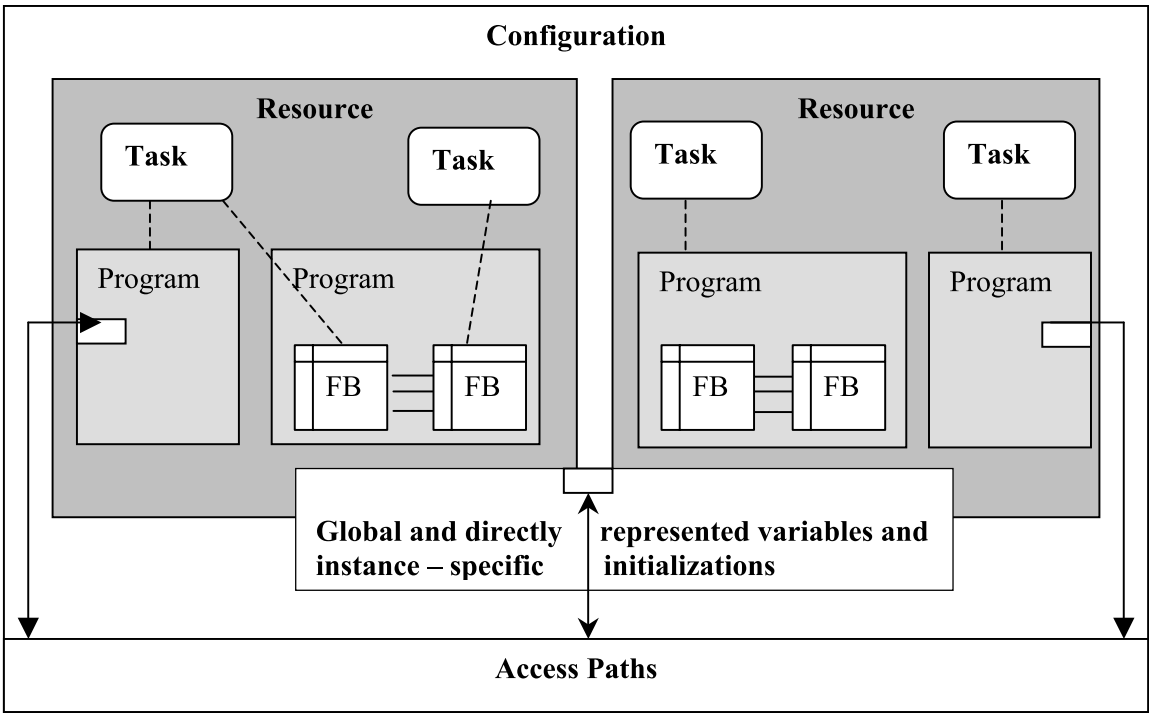
Softwarová PLC jsou poměrně mladou kategorií řídicích systémů. Jedná se o centrální počítač (PC, průmyslové PC případně jiný specializovaný počítač), ke kterému jsou připojeny přes rychlou a spolehlivou průmyslovou sběrnici distribuované vstupní moduly. Tento centrální počítač musí zajistit spolehlivé a dostatečně rychlé odezvy na vnější události, což vyžaduje speciální operační systém reálného času. Centrální PC se stará o řídicí funkce PLC a funkce vývojového systému. Programátor má k dispozici tradiční programovací jazyk pro PLC a také standardní služby operačního systému (integrovaná komunikační rozhraní, vyšší programovací jazyky, simulační a výpočetní nástroje atd.). Díky tomu je v Soft PLC možná integrace řízení, vizualizace s operátorským rozhraním, archivace provozních dat a různých optimalizací a diagnostik.

2.2.3 Norma IEC 61131

Norma definuje požadavky na programovatelné logické automaty, jejich periferie a komunikační rozhraní. Tato norma také popisuje možnosti programování PLC a je definován software model PLC (obrázek 2.3).

Základním prvkem je aplikace na PLC konfigurace (**Configuration**), která je závislá na konkrétním HW automatu – CPU, mapování vstupů a výstupů do paměťového prostoru a operačním systému. Součástí konfigurace je definice zdrojů (**Resource**). Zdroj je zařízení (obvykle CPU) vykonávající programy zapsané v některém z jazyků definovaných normou. V rámci zdrojů se definují úlohy (**Task**), které řídí vykonávání programů (**Program**) a funkčních bloků (**Function Block – FB**). Programy a funkční bloky mohou být vykonávány periodicky nebo na základě události (např. změna proměnné).

Programy, funkční bloky a funkce jsou souhrnně označeny jako programové organizační jednotky (Program Organization Units – POU). Funkce se dělí na standardní a uživatelské.



Obrázek 2.3: IEC software model, převzato z [15]

Funkční bloky mají představovat hardwarové řešení specializované funkce jako například regulační smyčka nebo PID regulátor. Program je složen z funkcí a funkčních bloků. Všechny tyto POU mohou být zadány v libovolném, normou definovaném jazyce. Tyto jazyky lze rozdělit na textové a grafické. Do textových patří:

- IL (Instruction List) – jazyk podobný assembleru,
- ST (Structured Text) – jazyk podobný vyšším jazykům typu C a Pascal.

Grafické jazyky jsou reprezentovány:

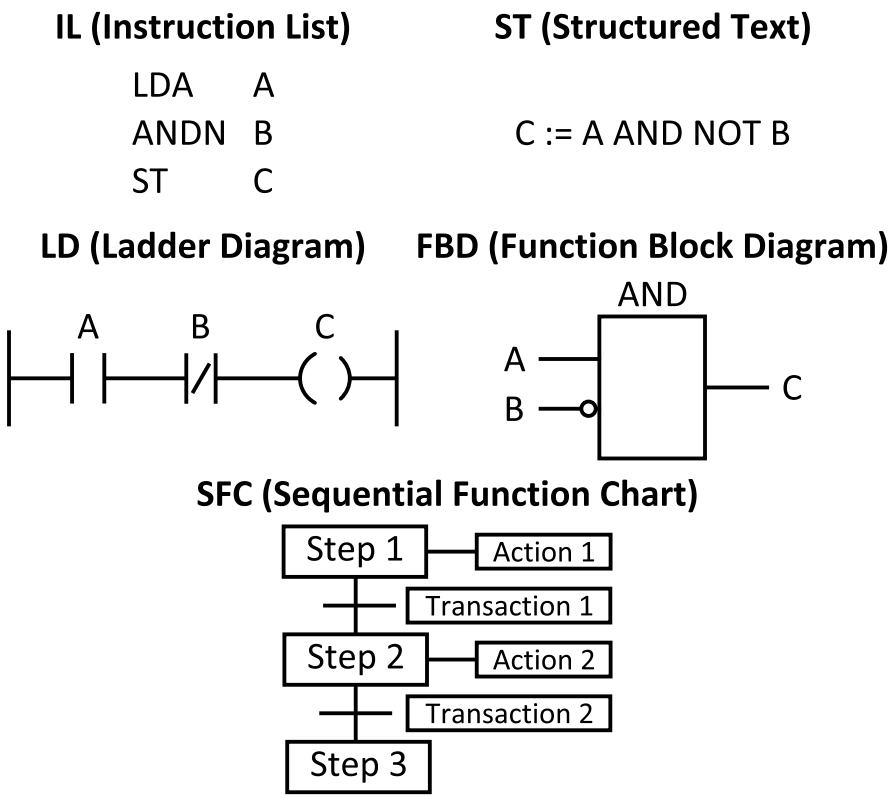
- LD (Ladder Diagram) – jazykem kontaktních schémat (reléová logika),
- FBD (Function Block Diagram) – jazykem provázaných grafických bloků, blízký schématům elektrických obvodů,
- SFC (Sequential Function Chart) – jazykem podobným Petriho sítím.

Příklady jazyků jsou uvedeny na obrázku 2.4. Více o programování podle normy IEC 61131 lze nalézt v [12].

2.3 Využití open source v automatizaci

2.3.1 Linux pro automatizaci

Operační systém Linux je úspěšně využíván především v oblastech serverových aplikací a vestavěných systémů, kam patří i oblast síťové infrastruktury. OS Linux nabízí několik



Obrázek 2.4: Jazyky definované normou IEC 61131, převzato z [12]

výhod při nasazení ve vestavěných systémech [14]. Hlavní výhodou je otevřený a prověřený kód. To umožňuje doplňovat funkcionalitu jádra podle našich potřeb, díky široké uživatelské základně je kód prověřený a případné chyby jsou obvykle brzy odstraněny. Za použití systému není nutné platit licenční poplatky. Další výhodou, kvůli které je volen právě Linux, je podpora síťových protokolů a množství dostupných ovladačů. Vývoj řídicí jednotky inteligentního domu založeného na operačním systému Linux je popsán v [14].

2.3.2 Beremiz

Projekt Beremiz [2] vznikl sloučením několika open source projektů, které dohromady tvoří kompletní vývojové prostředí pro vývoj aplikací podle normy IEC 61131. Součástí jsou tyto nástroje:

- PLCOpen Editor – IDE pro tvorbu programu v jazycích definovaných normou IEC 61131. Programy jsou ukládány a načítány ve formátu XML (PLCOpen TC6-XML),
- MatPLC's IEC compiler – překládá jazyky ST/IL/SFC do ANSI-C,
- CanFestival – implementace vysokoúrovňového protokolu CANOpen na sběrnici CAN, definuje využití 11 b identifikátoru a 8 B dat,
- SVGUI – nástroj pro tvorbu HMI (Human Machine Interface) rozhraní, používá SVG a umožnuje generování webových ovládacích rozhraní.

Kapitola 3

iNELS

Jelikož se práce zabývá implementací modulu řídicí aplikace pro systém iNELS společnosti Elko EP, v následující kapitole se s tímto systémem a sběrnicí CIB seznámíme. iNELS je systém inteligentní elektroinstalace, který vznikl za spolupráce firem Teco a Elko EP. Tento systém využívá speciální proprietární centralizovanou sběrnici CIB a řídicí PLC jednotku. Příklad možného použití systému je na obrázku 3.1. Zdrojem informací byl článek [11].

3.1 Sběrnice CIB

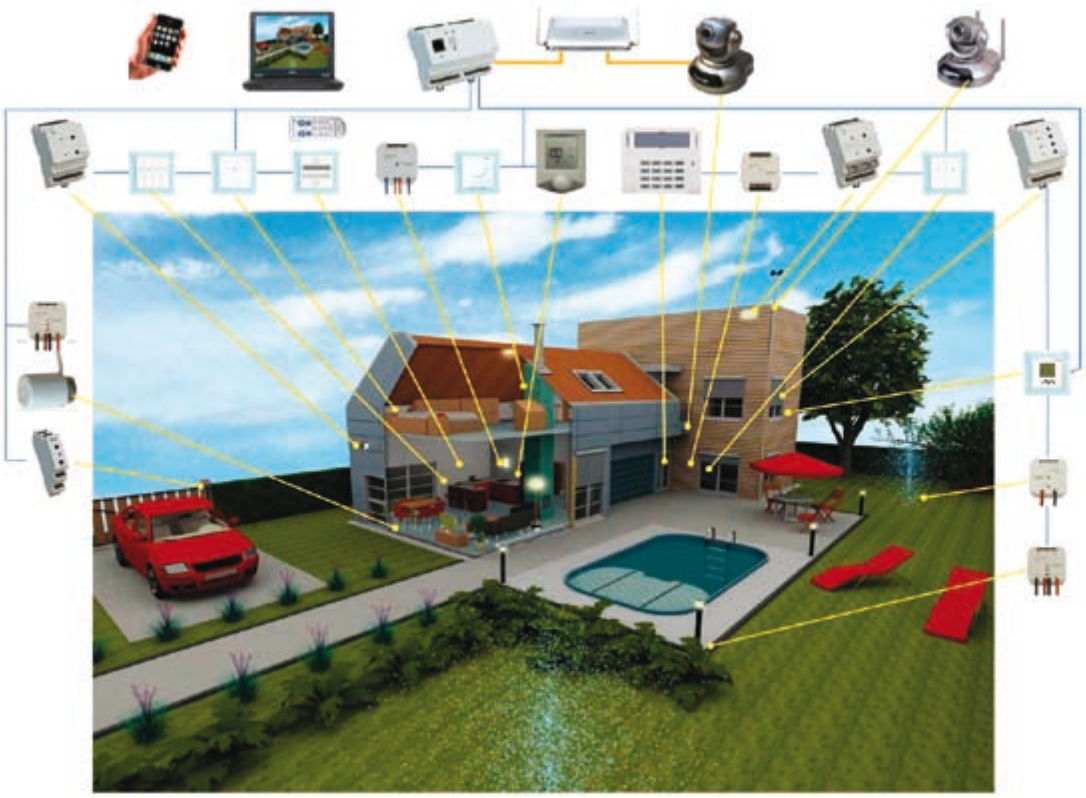
CIB (Common Installation Bus) je dvouvodičová sběrnice sdružující napájení i data. Použití dvouvodičového kabelu zjednodušuje spolu s libovolnou (mimo kruhovou) topologií instalaci systému. Je třeba dbát pouze na polaritu. Komunikace na sběrnici je typu master/slave. Jedna master jednotka obsahuje dvě větve, přičemž na každou větev lze umístit až 32 jednotek. Počet připojitelných jednotek i rozlehlost systému lze zvýšit připojením dalších master jednotek. Metalickým kabelem je možné připojit jednotku do vzdálenosti 300 m od řídicí jednotky, optickým až do 1700 m.

Sběrnice je napájena 24 V DC, lze použít i vyšší napětí 27,2 V pro dobíjení akumulátorů. Systém tak může udržovat komunikační a zabezpečovací funkce, i když dojde k výpadku napájení. Důležitý parametr sběrnice je odezva. CIB garantuje odezvu do 150 ms i při zapojení maximálního počtu jednotek. Přenosová rychlost sběrnice je 19,2 kb/s. Díky tomu není problém použít sběrnici pro řízení osvětlení, kde je vyžadována okamžitá odezva při stisku vypínače. Adresace jednotek je pevně daná, každá jednotka má z výroby přidělenou vlastní unikátní 16 b adresu. Systém je odolný vůči výpadku nebo odpojení jedné i více jednotek. Odpojení jednotek je detekováno a lze na něj reagovat. Další zajímavou funkcí sběrnice je možnost aktualizace firmware připojených jednotek.

3.2 Centrální jednotka

Na výběr je ze dvou typů centrálních jednotek: CU2-01M a Tecomat Foxtrot (CP-10xx). Jednotka CU2-01M je určena do rozsahu 192 jednotek na sběrnici a konfigurace probíhá pomocí programu IDM (iNELS Designer & Manager). Na jednotce CU2-01M se nachází dvě větve sběrnice CIB, navíc lze rozšířit externími moduly počet připojitelných jednotek čtyřmi dalšími větvemi. Centrální jednotka obsahuje ethernetový port, který je určen pro parametrizaci a vizualizaci stavu systému. Dále je obsažena sériová linka pro komunikaci s GSM modulem a čtyři diskrétní vstupy.

Tecomat Foxtrot je volně programovatelný modulární PLC, který se programuje v prostředí Mosaic, umožňuje připojit až 288 jednotek na sběrnici CIB. Tato jednotka obsahuje jednu větev CIB a umožňuje doplnit sběrnici o dalších osm větví. Tecomat Foxtrot nabízí také dvě volně programovatelné sériové linky, na které lze připojit zařízení dalších dodavatelů. Systém je možné připojit do jiných sítí například LON, KNX, Profibus-DP, BACnet.



Obrázek 3.1: Ukázka použití systému iNELS, převzato z[11]

3.3 Další sortiment

Firma Elko EP dále nabízí systém bezdrátového ovládání a modulů iNELS RF Control, který je vhodný především pro domy po rekonstrukci nebo tam, kde je problematické natáhnout kabeláž. Pro správu audiovizuálního obsahu v domě je možné využít iNELS multimedia, který dovoluje streamování satelitní televize po IP a k dispozici je síťové úložiště multimédií, přístupné kdekoliv v budově přes ethernetovou síť. Více informací, včetně všech modulů je možné nalézt v katalogu [4].

Kapitola 4

SOAP

SOAP (Simple Object Access Protocol) je komunikační protokol využívající XML (eXtensible Markup Language) a HTTP (Hypertext Transfer Protocol) pro výměnu dat mezi aplikacemi po síti. Tyto distribuované aplikace dokonce mohou běžet na různých operačních systémech a mohou být napsány v rozdílných programovacích jazycích. Použití formátu XML pro obsah zpráv umožňuje o proti binárně kódovaným zprávám jednodušší čtení člověkem a volnější strukturu zprávy (nezáleží například na pořadí hodnot, některé jsou nepovinné atd.). Další výhodou oproti binárním protokolům je možná integrace SOAPu s web serverem a tím pádem odpadá potřeba konfigurace firewallu. Na druhou stranu, XML zprávy jsou oproti binárním zprávám delší a jejich zpracování trvá déle.

Ačkoliv lze použít i jiný transportní protokol, díky HTTP je možné využívat SOAP nejen v rámci LAN (Local Area Network) ale i napříč Internetem. SOAP také obsahuje mechanismy pro oznámení a detekci SOAP služeb UDDI (Universal Description, Discovery and Integration) a WSDL (Web Services Description Language), který popisuje schéma objektů a zpráv. Informace o protokolu SOAP byly čerpány z knihy [13].

Podobně jako protokol HTTP je i SOAP nestavovým protokolem – následující zpráva je nezávislá na předchozích. Základní struktura v SOAP zprávy vypadá následovně:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope>
  <SOAP-ENV:Header>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body xmlns:MujNamespace="http://www.firma.com/SOAPschema">
    <MujNamespace:ZiskejData>
      <Parametr1>Hodnota1</Parametr1>
      <Parametr2>Hodnota2</Parametr2>
    </MujNamespace:ZiskejData>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Server po přijetí takovéto zprávy zavolá funkci ZiskejData s parametry Parametr1 s hodnotou Hodnota1 a Parametr2 s hodnotou Hodnota2. Poté klientovi odešle odpověď s návratovou hodnotou, případně chybovou hláškou. Hodnotami mohou být objekty. Skutečná SOAP zpráva bude obsahovat v hlavičce odkazy na použitá schémata objektů a bude mít definovaný namespace. Pokud je přenášenou hodnotou objekt, musí nejprve proběhnout jeho serializace (jeho převedení na XML). Opakem serializace je deserializace, kdy se z XML popisu vytvoří příslušný objekt.

4.1 Přenos binárních dat

Při použití SOAP protokolu se můžeme setkat s požadavkem posílat přes něj binární data. Může se jednat o webovou službu s možností uploadu souboru, posílání multimediálních dat nebo přenos obecných dat při distribuovaném výpočtu. První a nejjednodušší možností, jak přes SOAP poslat binární data je použít datový typ base64Binary. Toto však není příliš efektivní, blok dat se musí překódovat, navíc tento přístup může být problematický pro XML parsery, které musí token s blokem dat držet v paměti. Množství přenášených dat pomocí base64Binary je omezeno a hodí se spíše pro malé objemy dat.

Další možnost je použít standard MIME (Multipurpose Internet Mail Extensions), který se používá pro kódování příloh v emailu a binárních dat na WWW stránkách. MIME zvětšuje počet přenášených dat o 30 %, ale lze přenést data libovolné délky. Do místa, kde by měla být v SOAP zprávě vložena binární data se vloží reference na data, která budou ve formátu MIME následovat za SOAP zprávou. Použití MIME v protokolu SOAP je specifikováno jako SwA (SOAP with attachments). DIME (Direct Internet Message Encapsulation) je vylepšením standardu MIME, který spoléhá na oddělení binárních dat textovými značkami. Toto může být problém, pokud se v příloze objeví stejná značka. DIME proto používá namísto textových značek offsety. Umožňuje stejně jako SwA přenášet více binárních souborů.

Součástí specifikace SOAP 1.2 je také MTOM (Message Transmission Optimization Mechanism). MTOM využívá SwA spolu s XOP (XML-binary Optimized Packaging), který definuje serializaci SOAP zpráv s binárními daty, tak aby byla zachována struktura XML dokumentu. Informace o přenosu binárních dat byly čerpány z [8].

Knihovna gSOAP umožňuje využít všechny popisované metody přenosu binárních dat, u DIME a MTOM dokonce podporuje streamování obsahu.

Kapitola 5

Návrh

Cílem práce bylo vytvořit modul SoftPLC, část aplikace Inels CU3 pro řízení inteligentních budov společnosti Elko EP. Tato aplikace byla vyvíjena na FIT VUT ve spolupráci a na základě požadavků firmy Elko EP.

5.1 Požadavky

Záměrem řešení nového řídicího systému bylo oprostit se od stávajícího řídicího systému a začít využívat moderní technologie, které umožňují rozšiřovat řídicí systém pomocí open source projektů a dalších typů sběrnic. Nová řídicí jednotka tedy přechází od PLC k vestavěnému systému s operačním systémem Linux. Pro tuto jednotku je potřeba vytvořit aplikaci, která se bude starat o samotné řízení budovy. Aplikace musí být schopná pracovat s digitálními (binárními) a analogovými vstupy; dalším typem vstupů jsou též čtečky přístupových karet a klávesnice EZS (elektronický zabezpečovací systém). Výstupy jsou taktéž digitální a analogové.

Samostatné části, které bude tato aplikace řídit jsou vytápění/klimatizace, EZS a GSM subsystém. Dále je potřeba správa čítačů, časovačů, časově závislých funkcí (stmívání analog. výstupu, generování impulzu, zpoždění hodnoty výstupu, periodická inkrementace, dekrementace výstupu) a časových programů pro opakované vykonávání zadaných funkcí v požadovanou dobu.

V aplikaci musí být možné spravovat propojení komponent typu senzor-aktor-akce, kdy se při události na senzoru (stisk tlačítka) provede akce na aktoru (rozsvícení žárovky). Tato propojení musí být vhodně modelována tak, aby bylo možné reagovat i na vnitřní události řídicího systému, jako například odeslání SMS zprávy při poklesu napájecího napětí. Řídicí jednotka by si měla uchovat aktuální konfiguraci i při výpadku napájení. Společně s vývojem aplikace řídicí jednotky probíhá paralelně vývoj nové verze konfiguračního programu iNELS Designer & Manager – IDM3, který bude s řídicí jednotkou komunikovat přes rozhraní SOAP. V IDM3, grafické uživatelské aplikaci designér vytvoří konfiguraci, kterou poté přes síť nahraje do SoftPLC. SoftPLC bude vyhodnocovat vstupní události a na základě propojení událostí s akcemi jsou tyto akce vykonávány.

U digitálních vstupů musí být v řídicí jednotce implementován mechanismus detekce krátkého a dlouhého stisku. Analogové vstupy by měly umožňovat filtraci vstupu (zpožděné načtení hodnoty), přímé ovládání analogového výstupu (např. ovládání svitu žárovky otočným potenciometrem) a také by mělo být možné vytvářet podmínky, které při splnění budou vyvolávat akci na výstupu. K analogovým vstupům typu teploměr musí jít přiřadit

akce při překročení vysoké a nízké teploty. Čtečky karet mohou vyvolávat akci při načtení karty. Digitální a analogové výstupy je možné ovládat samostatně nebo ve skupinách a lze přiřadit akce při změně výstupu.

Na jednu událost vstupu musí být možné přiřadit více akcí na více výstupech, případem použití je rozsvícení více žárovek, které nejsou ve stejné skupině, jedním tlačítkem. Vytápění/klimatizace se řídí podle nastavených časových plánů. Pro EZS musí být možnost vytvoření skupin vstupů, při jejichž změně je vyhlášen alarm. Musí být také dostupný kalendář událostí, který umožňuje definovat opakované (vybrané dny v týdnu, měsíci nebo každý den vybraného měsíce) vyvolání akcí v zadaný čas. Dále by také měla existovat možnost při zvolené události poslat SMS zprávu. Aplikace musí umožňovat aktualizaci firmware v připojených jednotkách na sběrnici. V řídicím systému se počítá s využitím různých sběrnic, používaných pro inteligentní elektroinstalace, proto bude použitá sběrnice označována souhrnně IMB (Inels Master Bus). Konkrétní volba sběrnice závisí na implementaci modulu IM Worker.

5.2 Cílová platforma

Současně s vývojem softwaru CU3 probíhá také vývoj hardwaru řídicí jednotky. Nová řídicí jednotka, taktéž nazvaná CU3, je založena na architektuře ARM926EJ-S, jejímž základem je 32-bit RISC procesor s MMU (Memory Management Unit). MMU se stará o překlad adres v systému virtuální paměti a umožňuje běh operačních systémů jako je Symbian OS, Windows CE a Linux [7]. Jednotka CU3 obsahuje procesor AM1707 od firmy Texas Instruments. Procesor v závislosti na napájecím napětí může pracovat na 375 nebo 456 MHz. CU3 dále obsahuje 64 MB SDR RAM a 256 MB NAND flash.

Jednotka CU3 je vyrobena jako modul na DIN lištu ve velikosti 6M. Uvnitř tohoto modulu se nachází tři desky plošných spojů: první, nejspodnější zajišťuje napájení celého modulu, komunikaci na sběrnici IMB a obsluhu vstupů a výstupů modulu. Prostřední deska obsahuje procesor AM1707, SDRAM a NAND. IMB sběrnice je procesoru AM1707 zpřístupněna přes UART. Na vrchní desce je umístěn OLED displej, informační LED diody, ethernetový port s oddělovacím transformátorem a bezdrátový modul pro komunikaci s Inels RF Control. Jak již bylo zmíněno, na jednotce CU3 běží operační systém Linux, ve kterém jsou spuštěny všechny tři softwarové moduly CU3 – IM Worker, VarStore a SoftPLC. Fotografie vývojové verze jednotky CU3 se nachází v příloze B.1.

5.3 Konfigurační program IDM3

IDM3 je program s grafickým uživatelským rozhraním sloužící pro konfiguraci celého řídicího systému CU3 a je vyvíjen v jazyce C#. Program je rozdělen na dvě části: menu a plochu, kde je zobrazen půdorys domu s rozmístěnými prvky (světla, vypínače atd). Screenshot aplikace je uveden v příloze C.1. V menu lze vytvořit nový projekt, načíst/uložit projekt na disk nebo ústřednu (jednotka CU3) a nastavit IP adresu ústředny, na které běží SoftPLC. Dále jsou v menu záložky, kde lze definovat zařízení připojená na IMB sběrnici, přidávat podlaží řízené budovy, přidávat prvky systému (světla, vypínače, pohybové detektory, ...), dále lze definovat skupiny prvků, které se hodí zejména pro ovládání více světel zároveň.

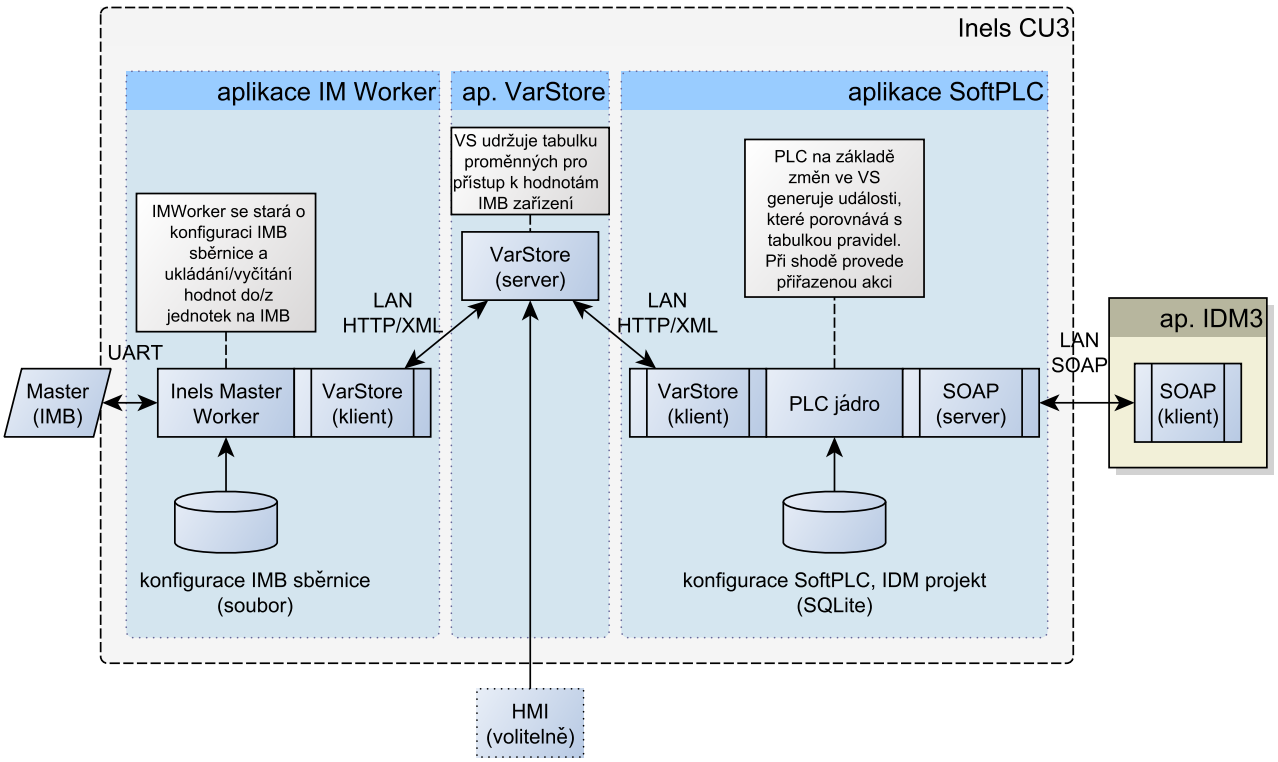
V další záložce se definuje propojení jednotlivých prvků (po stisku kterého vypínače se má rozsvítit jaké světlo atd.). Poslední záložka obsahuje nastavení systémových zařízení

(virtuální zařízení typu čítač a časovač, skupiny zařízení, alarmy a časové programy), správu uživatelů pro alarmy a nastavení časových programů pro vytápění/chlazení. Na zobrazeném půdorysu jsou rozmístěny prvky systému, u nichž lze definovat připojení na vstupy a výstupy IMB zařízení, přidávat je do skupin, zobrazit si jejich propojení a provádět simulaci celého systému. Pokud je v simulaci program připojen na jednotku CU3, všechny změny v simulaci jsou prováděny v reálném čase také fyzicky na jednotkách systému Inels.

Pro komunikaci s modulem SoftPLC je použit protokol SOAP, jenž umožňuje přenos parametrů a volání funkcí na vzdáleném systému. Pro přenos se využívají XML zprávy, do kterých jsou serializovány C#/C++ objekty. SOAP klient odešle serveru zprávu s názvem funkce, která má být na serveru vykonána a seznam vstupních parametrů. Na straně serveru je vykonána zvolená funkce a výsledek je vrácen klientovi. Rozhraní mezi programem IDM3 a modulem SoftPLC je popsáno dále (kapitola Implementace) a v příloze D se nachází kompletní schéma.

5.4 Architektura řídicí aplikace

Architektura řídicí aplikace je znázorněna na obrázku 5.1. Aplikace je rozdělena na moduly IM Worker, VarStore a SoftPLC. IM Worker zajišťuje komunikaci s moduly na sběrnici IMB, modul VarStore reprezentuje stav systému – jsou zde uvedeny hodnoty vstupů a výstupů jednotlivých zařízení na sběrnici IMB a také konfigurace IMB jednotek, jako například offsety teplotních senzorů, nastavení rozsahu analogových výstupů apod.



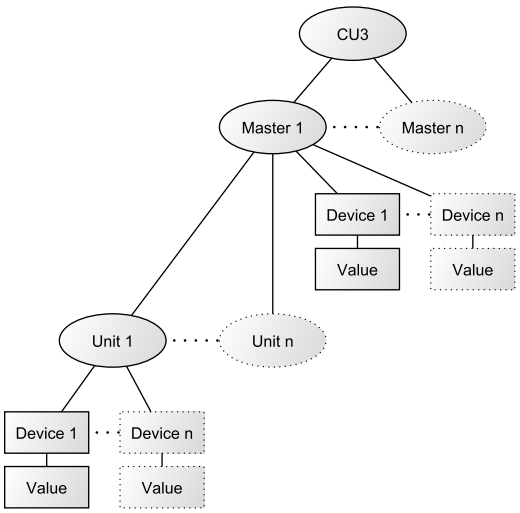
Obrázek 5.1: Schéma aplikace CU3 a komunikace mezi moduly

VarStore slouží jako cache hodnot vstupů a výstupů zařízení na sběrnici IMB. Zároveň

tento modul odstiňuje vlastní řízení (SoftPLC) od použité sběrnice. Jednotlivé moduly řídicí aplikace jsou vyvíjeny jako samostatně spustitelné aplikace, které si vyměňují data přes VarStore pomocí HTTP/XML. Toto řešení usnadňuje vývoj, který tak může probíhat na každém modulu odděleně. Implementační jazyk je C++, komunikaci pomocí protokolu SOAP zajišťuje knihovna gSOAP a pro zjednodušení a zefektivnění vývoje aplikace byla vybrána C++ knihovna POCO.

5.4.1 IM Worker

Jednotka CU3 může obsahovat až šest masterů IMB sběrnice, přičemž ke každému masteru může být připojeno až 32 jednotek. Master i ostatní jednotky obsahují tzv. devices. Device je jedna z několika funkcí jednotky a je adresovatelná pomocí logické adresy ve VarStore. Například jednotka LM3-11B (stmívací jednokanálový aktor) obsahuje jeden binární vstup, analogový vstup pro připojení teploměru a jeden analogový výstup pro spínání RLC zátěže. Ve VarStore budou pro tuto jednotku vytvořeny tři proměnné. Na obrázku 5.2 je uvedeno logické schéma sběrnice IMB.

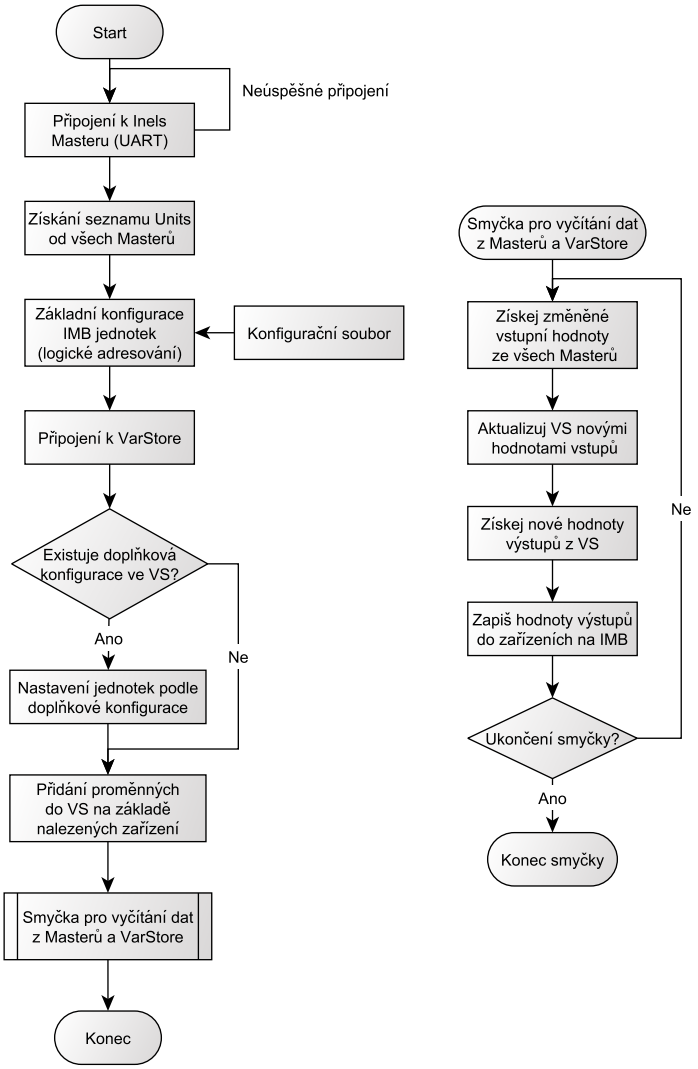


Obrázek 5.2: Logické schéma IMB sběrnice

Modul se po startu nejprve připojí na Inels Master přes UART. Inels Master je jeden z masterů, který řídí komunikaci na sběrnici. Poté se načtou informace o tom, jaké jednotky jsou připojeny. Proběhne jejich základní konfigurace, která představuje přiřazení logických adres jednotkám za účelem odstínění používání HW adres z výroby. HW adresy jsou důležité pro komunikaci na sběrnici, ale pokud by na ně byla vázaná konfigurace SoftPLC, nemohla by být stejná konfigurace použita ve více instalacích. Pomocí logických adres přistupuje SoftPLC do VarStore, logické adresy jsou poté v IM Workeru překládány na unikátní HW adresy.

Po konfiguraci logických adres se modul IM Worker připojí k modulu VarStore a zkontroluje, zda existuje konfigurace pro jednotky na IMB a případně je podle této konfigurace nastaví. V dalším kroku je přidána pro každou logickou adresu proměnná do VarStore. Po přidání proměnných již následuje smyčka vyčítání vstupních hodnot z jednotek na IMB sběrnici a tyto hodnoty jsou následně uloženy ve VarStore. Změněné výstupní hodnoty z VarStore jsou načteny do IM Workeru a následně zapsány do jednotek na IMB. Smyčka

by se měla opakovat asi po 200 ms, aby se stihly vyčíst a zapsat hodnoty na IMB a zároveň byla reakce řídicího systému na vzniklé události co nejrychlejší. Na obrázku 5.3 se nachází vývojový diagram modulu IM Worker.



Obrázek 5.3: Vývojový diagram modulu IM Worker

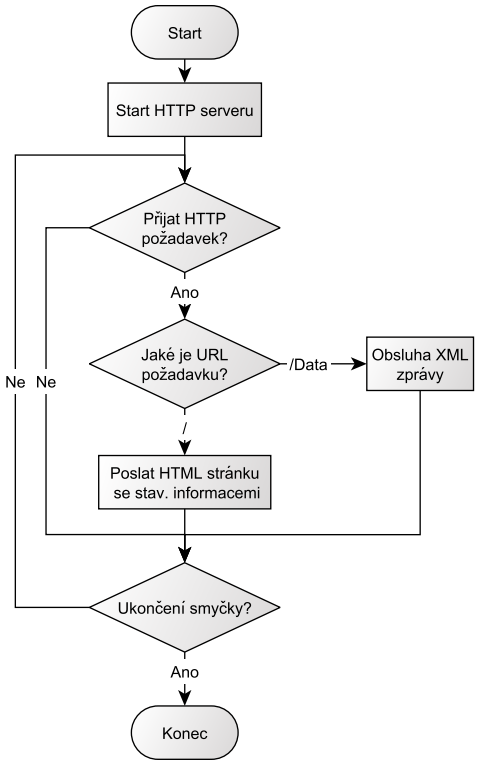
5.4.2 VarStore

Modul VarStore udržuje informaci o stavu řízeného systému a zpřístupňuje IMB jednotky dalším modulům. Pro přehledné zobrazení tohoto stavu lze použít grafické aplikace, které se připojí přímo k VarStore, tzv. HMI (Human Machine Interface) aplikace. HMI je koncept využívaný především pro vizualizaci stavu řízení průmyslových procesů, kdy jsou měřené veličiny přehledně zobrazeny v grafické uživatelské aplikaci. SoftPLC, IM Worker a případné další moduly se k VarStore připojují přes tzv. VarStore klienta. VS klient je HTTP klient, který parsuje XML zprávy z VarStore. VS klient implementuje následující metody:

- `getList()` – v odpovědi se přenáší všechny proměnné,

- `getChangedList()` – v odpovědi přijdou pouze změněné hodnoty proměnných,
- `getValue()` – VS pošle pouze hodnotu jedné zvolené proměnné,
- `setValue()` – nastavení hodnoty proměnné ve VS,
- `addDeviceValue()` – přidání nové proměnné do VS,
- `setConfiguration()` – uložení konfigurace IMB jednotek do VS,
- `getConfiguration()` – načtení konfigurace z VS.

VarStore tedy funguje jako HTTP server s XML zprávami. Jako HTTP server také umožňuje zobrazit stavové informace v HTML (HyperText Markup Language). Proměnné ve VarStore jsou adresovány 32 b adresou, která je složena z 8 bitových identifikátorů (seřazeno podle MSB) worker, master, unit a device. Proměnná reprezentuje hodnotu device – jedna z několika funkcí každé IMB jednotky. Proměnné jsou ve VS uložené jako dynamické pole bytů s tím, že interpretace hodnot závisí na aplikaci, která hodnoty používá. Pokud bude potřeba přenášet příkazy ze SoftPLC do IM Workeru (například restart celého systému), lze do VS přidat proměnnou se speciální adresou (různou od logických adres IMB sběrnice) a příkazy předávat přes ni. Na obrázku 5.4 se nachází vývojový diagram modulu VarStore.



Obrázek 5.4: Vývojový diagram modulu VarStore

5.4.3 SoftPLC

Modul SoftPLC je rozdělen do několika vláken. Vlákna pro gSOAP server a HTTP server obsluhují SOAP a HTTP požadavky. Oba servery jsou implementovány jako konkurentní, pro každý požadavek je vyhrazeno vlastní vlákno. Hlavní vlákno, tzv. PLCEngine nejprve vytvoří jedno vlákno pro inkrementaci časovačů a kontrolu časových programů. Poté je načtena konfigurace z databáze SQLite, pokud existuje. Následně vlákno PLCEngine vstupuje do smyčky, ve které s periodou 100 ms kontroluje události systémových zařízení a zařízení umístěných na sběrnici IMB. Systémové události jsou ukládány do seznamu a vykonávány synchronně s hlavní smyčkou, aby se předešlo rekurzivnímu výskytu událostí.

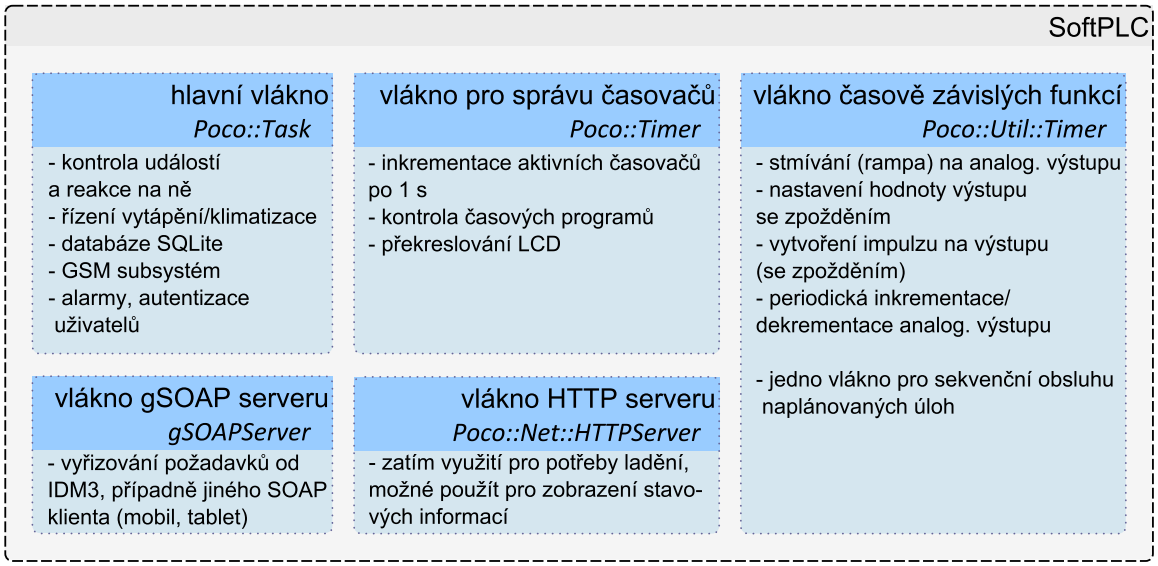
Po obsluze systémových událostí jsou získány z VS změněné hodnoty proměnných. Na základě hodnoty proměnné a typu zařízení, ke kterému proměnná patří, je odvozena událost. Obsluha události probíhá stejně i pro systémová zařízení. V tabulce WireConnections se vyhledává zařízení, na kterém nastala událost (senzor) a při shodě se dále porovnává typ události. Pokud se položka z WireConnections shoduje i v typu události, jsou zkontrolovány podmínky vykonání přiřazených funkcí a případně jsou tyto funkce vykonány na zařízení označeném jako aktor. Hlavní smyčka pro kontrolu událostí je volána každých 100 ms.

Pokud je senzor vázán k vytápění/klimatizaci, dochází k vyhodnocení výstupu na základě časových plánů vytápění/klimatizace. Systém může obsahovat obecně n časovačů, které mohou vyvolat systémovou událost. Aby v řídicím systému nemuselo běžet n vláken časovačů zároveň, je vyhrazeno právě jedno vlákno pro aktualizaci všech povolených časovačů za určitý počet jednotek (1 s). V tomto vlákně je také obsluhován LCD displej, zejména aktualizace data a času a dochází také ke kontrole časových programů – pokud je časový program aktivován nebo deaktivován je vytvořena systémová událost.

Časově závislé funkce mohou být vyvolány na aktoru na základě události a odpovídajícího záznamu ve WireConnections. Časově závislé funkce jsou obsluhovány pomocí samostatného vlákna, ve kterém se sekvenčně plánují. Na obrázku 5.5 je zobrazeno rozložení vláken v modulu SoftPLC. Třída SoftPLC by měla obsahovat tyto struktury: tabulku propojení (WireConnections) ve tvaru (ID senzoru, událost senzoru, ID aktoru, akce na aktoru), tabulku čítačů, tabulku časovačů, seznam skupin vstupů/výstupů, tabulku uživatelů, definice vytápění/klimatizace včetně tabulky časových programů. Pro trvalé uchování konfigurace SoftPLC je použita databáze SQLite, která je podporovaná v knihovně POCO. Vývojový diagram modulu SoftPLC se nachází na obrázku 5.6.

5.5 Knihovna gSOAP

gSOAP je open source sada nástrojů pro vývoj SOAP/XML webových služeb a převod datových struktur C/C++ do XML. Knihovna také umožňuje efektivní přenos binárních dat pomocí protokolů MTOM, MIME a DIME, knihovnu lze použít na různých platformách, včetně těch vestavěných a je kompatibilní s jinými SOAP implementacemi jako například WCF, .NET a Apache Axis. gSOAP obsahuje HTTP/HTTPS server nebo je možné provést integraci s existujícím web serverem. Namísto parsování XML je použit kompilovaný vygenerovaný kód, jenž zrychluje zpracování XML. Knihovna zvládá přenos objektů a paměti ukazatelů mezi aplikacemi. U přenášených objektů lze využít polymorfismu (přes ukazatel na nadřazenou třídu se mohou serializovat instance odvozených tříd). Informace o knihovně byly čerpány z [1].



Obrázek 5.5: Schéma modulu SoftPLC – rozvržení vláken

5.5.1 Omezení gSOAP

Z STL (Standard Template Library) knihovna gSOAP podporuje std::string, std::deque, std::list, std::vector a std::set. Uživatelsky definované šablony (templates) nejsou podporovány. Dále není podporována vícenásobná dědičnost, třídy mohou mít nejvýše jednoho předka. Ačkoliv metody nejsou serializovány, nelze použít abstraktní metody, protože musí existovat možnost vytvořit instanci třídy při deserializaci. Kromě textových řetězců (pole znaků typu `char*`) nelze přímo bez udání velikosti serializovat ukazatel na blok paměti. Při serializaci ukazatele se totiž serializuje pouze první položka. Stejně tak ukazatele typu `void*` lze serializovat až po doplnění patřičných údajů do struktury nebo třídy, která je používá.

5.6 Knihovna POCO

POCO je modulární multiplatformní ANSI C++ knihovna s minimálním počtem závislostí, vhodná pro vestavěné systémy (spolupracuje s knihovnami μClibc, glibc). Obsahuje funkce pro zpracování textu a XML, práci s procesy a vlákny, komunikaci po síti (HTTP server, podpora SSL, SMTP, FTP klient, obecný TCP server), funkce pro logování a práci s databází [3].

5.7 Zabezpečení

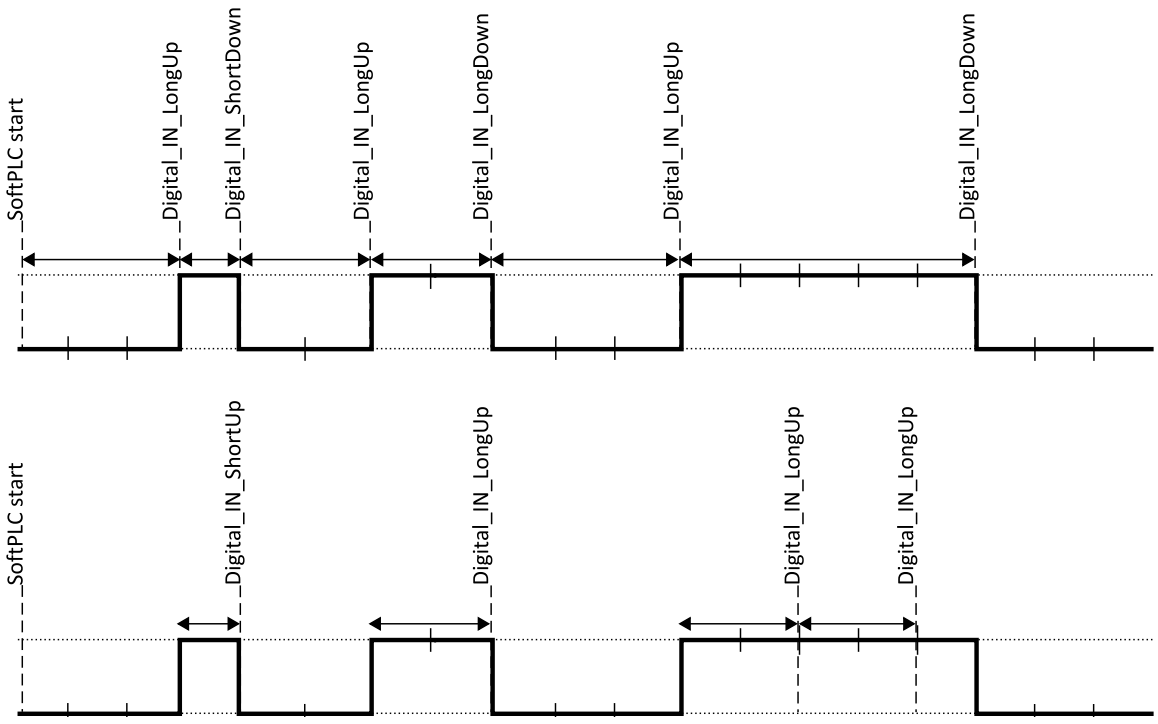
Ačkoliv není v návrhu zabezpečení komunikace řídicího systému CU3 uvažováno, v případě nasazení systému u zákazníka je zabezpečení dat nezbytné. V případě výpadku napájení by měla být celá IMB sběrnice zálohována z baterií a řídicí jednotka má k dispozici informaci o výpadku napájení i slabých bateriích. Komunikace mezi moduly (IM Worker, SoftcPLC a VarStore) probíhá pomocí protokolu HTTP bez použité autentizace nebo šifrování. Zabezpečení komunikace mezi moduly však není potřeba, pokud všechny moduly běží na jedné řídicí jednotce. Naopak nezbytné je zabezpečení SOAP rozhraní u modulu SoftPLC.

V případě nešifrované komunikace může útočník pozměnit nebo dokonce znemožnit činnost řídicího systému. Může vytvořit vlastní nebo zachytit a upravit přenášené SOAP zprávy. V knihovně gSOAP lze využít HTTP autentizaci, která však nezaručuje identitu a integritu zpráv a je náchylná vůči útoku typu man-in-the-middle. Dále je v gSOAP možné použít šifrovaný transportní protokol HTTPS (k tomu potřeba jedna z knihoven OpenSSL/GnuTLS), dokonce s podporou HW akcelerace. Další možností je použít WS-Security plugin [1].

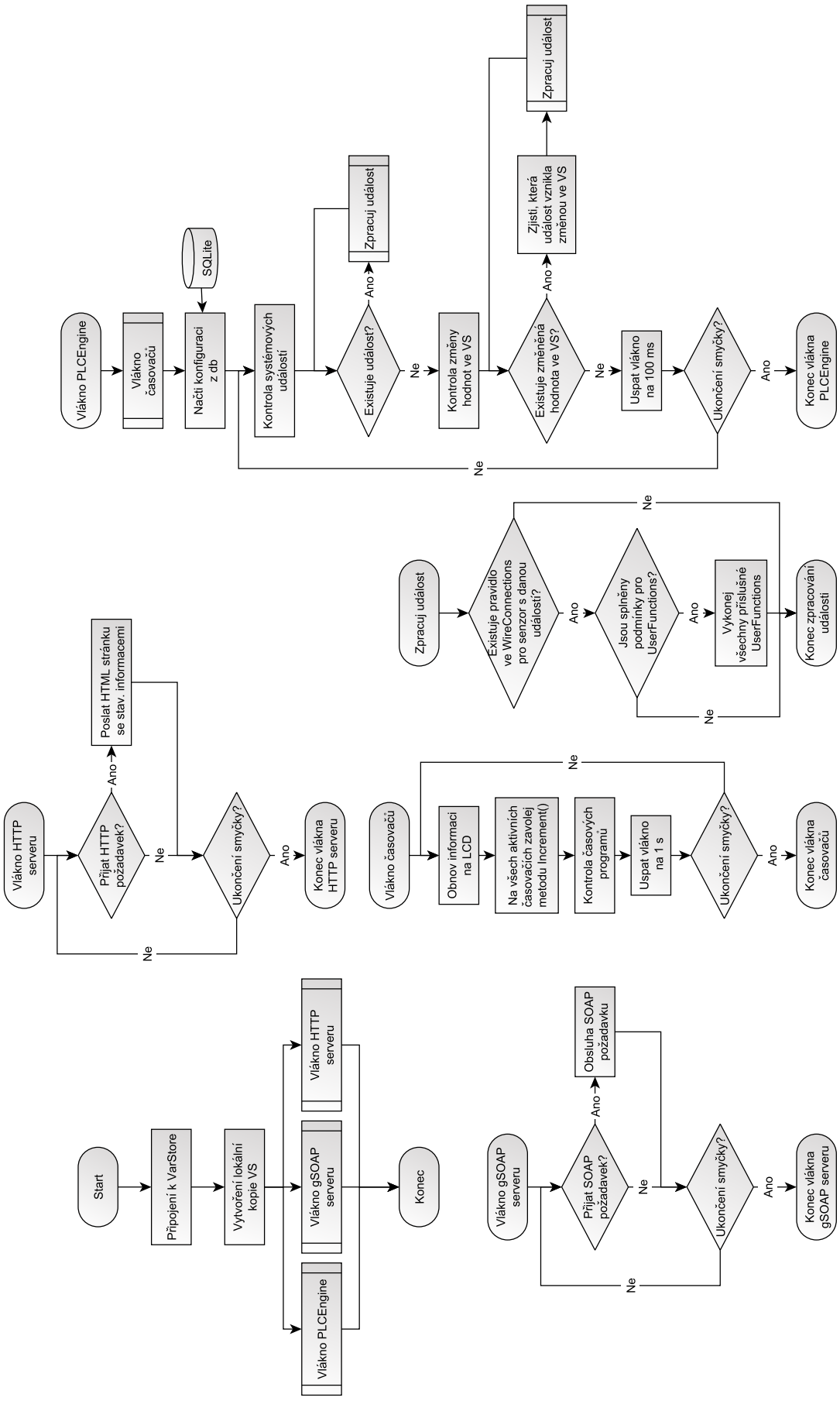
5.8 Detekce délky stisku na digitálním vstupu

Program IDM3 dovoluje konfiguraci s těmito událostmi na digitálním vstupu: krátké povolení/stisk vypínače (Digital_IN_ShortUp/Down) a dlouhé povolení/stisk vypínače (Digital_IN_LongUp/Down).

V modulu SoftPLC je tedy potřeba tyto události detekovat podle aktuálních hodnot digitálních vstupů. SoftPLC zpracovává události na fyzických zařízeních na základě změny hodnoty. U digitálních vstupů pak máme informaci pouze o nástupných nebo sestupných hranách a délka stisku je určena dobou, která uběhla od poslední hrany. V případě, že měříme dobu od uplynutí libovolné hrany, dostáváme informaci o tom, jak dlouho bylo tlačítko stisknuto i povoleno, jak je uvedeno na horní části obrázku 5.7. Pokud nepotřebujeme informaci o povolení vypínače, může být vhodnější použít způsob detekce, který je se nachází v dolní části obrázku 5.7. V tomto případě je měřena doba, která uplynula pouze od nástupné hrany. Při čtení hodnoty digitálního vstupu v každém cyklu smyčky SoftPLC je možné detekovat dlouhý stisk jako několikanásobný. V SoftPLC je využit první přístup.



Obrázek 5.7: Možnosti detekce stisku vypínače na digitálním vstupu



Obrázek 5.6: Vývojový diagram modulu SoftPLC

Kapitola 6

Implementace

Cílem práce bylo vytvořit modul SoftPLC řídicí aplikace CU3 v jazyce C++. Ačkoliv je cílová platforma ARM9 s operačním systémem Linux, aplikace CU3 je vyvíjena jako multiplatformní. Nejprve začala vznikat aplikace IDM3 pro konfiguraci řídicího systému. Tato aplikace má definovaný datový model řízené budovy pomocí rozhraní tříd (interface v C#) a protože do modulu SoftPLC může být tento model nejen uložen ale i v něm upravován a zpátky načten do IDM3, je důležité, aby bylo SoftPLC přizpůsobeno tomuto rozhraní. V SoftPLC je tento datový model implementován pomocí tříd, které jsou definovány v konfiguračním souboru knihovny gSOAP.

V datovém modelu je použita u několika pojmů jiná sémantika, než byla dosud v tomto dokumentu používána. Dříve uváděná událost, která může být vyvolána změnou hodnoty na IMB nebo systémových zařízeních, bude při popisu datových struktur označována jako akce. Vztahy mezi prvky systému jsou popsány tabulkou WireConnection. Zařízení, na kterém se objeví *akce* (dříve událost), je označené jako *aktor* (dříve senzor) a zařízení, na kterém se mají vykonat uživatelské funkce je označené jako *konzument* (dříve aktor).

V datovém modelu je použit C# polymorfní datový typ `object`, který umožňuje pracovat s IMB zařízeními (vstupní, výstupní) a systémovými jednotným způsobem. Ačkoliv v C++ takový datový typ nenajdeme, v knihovně gSOAP je polymorfismus podporován a pomocí ukazatele rodičovskou třídu můžeme deserializovat i instance tříd odvozených. Při deserializaci objektů s ukazateli musíme dát pozor při vytváření kopií těchto objektů. Deserializovaný objekt máme k dispozici pouze v době obsluhy SOAP metody, ale v našem případě potřebujeme s objekty přijatými přes SOAP pracovat i v jiných částech programu. gSOAP server si spravuje paměť potřebnou pro serializaci a deserializaci sám, proto je vytvořený objekt dostupný pouze v době běhu funkce obsluhující SOAP požadavek.

Standardně se v C++ vytváří tzv. mělká kopie objektu, kdy se u atributů typu ukazatel kopíruje pouze jeho hodnota, ale nikoliv paměť, na kterou ukazatel ukazuje. Kopie vytvořená mělkým kopírováním sdílí paměť s původní instancí objektu. Pokud potřebujeme, aby byla kopie nezávislá na svém vzoru včetně této paměti, musíme vytvořit tzv. hlubokou kopii objektu. Hluboké kopírování lze implementovat pomocí explicitní definice kopírovacího konstruktoru, operátoru přiřazení pro kopírování-přiřazení a destruktoru pro uvolnění alokované paměti [16]. Hluboké kopírování je potřeba implementovat pro všechny datové struktury, které posílá IDM3 přes SOAP a které je potřeba používat i mimo obsluhu SOAP požadavku.

Pro kontrolu činnosti SoftPLC je použito logování výpisů programu do textové konzole (`stdout`) a do souboru. O logování se stará třída Poco::Logger, která umožňuje formátování logovacích zpráv včetně nastavení priority. Díky prioritě logovaných zpráv lze na jednom

místě změnit úroveň podrobností u vypisovaných zpráv. Tak je možné vypisovat pouze nezbytné informace ve verzi, která má být nasazena u zákazníka a ve vývojové verzi naopak zobrazovat informace podrobně.

Mimo vypisování zpráv do logu podává SoftPLC informace o svém běhu pomocí OLED displeje. Vývojový modul CU3 obsahuje displej DD-128128FC-6A, který má rozlišení 128×128 bodů a umožňuje zobrazit až 65 tisíc barev. Displej je ovládán pomocí rozhraní SPI. Na displeji je zatím zobrazováno pouze datum, čas a informace, zda je IDM3 připojen. V budoucnu se dá očekávat využití displeje pro zobrazení dalších stavových informací či nastavení některých parametrů. Zápis na displej je prováděn ve vlákně Timer, které spravuje časovače a časové programy. To vše se děje ve smyčce s periodou 1 s.

6.1 Datové struktury

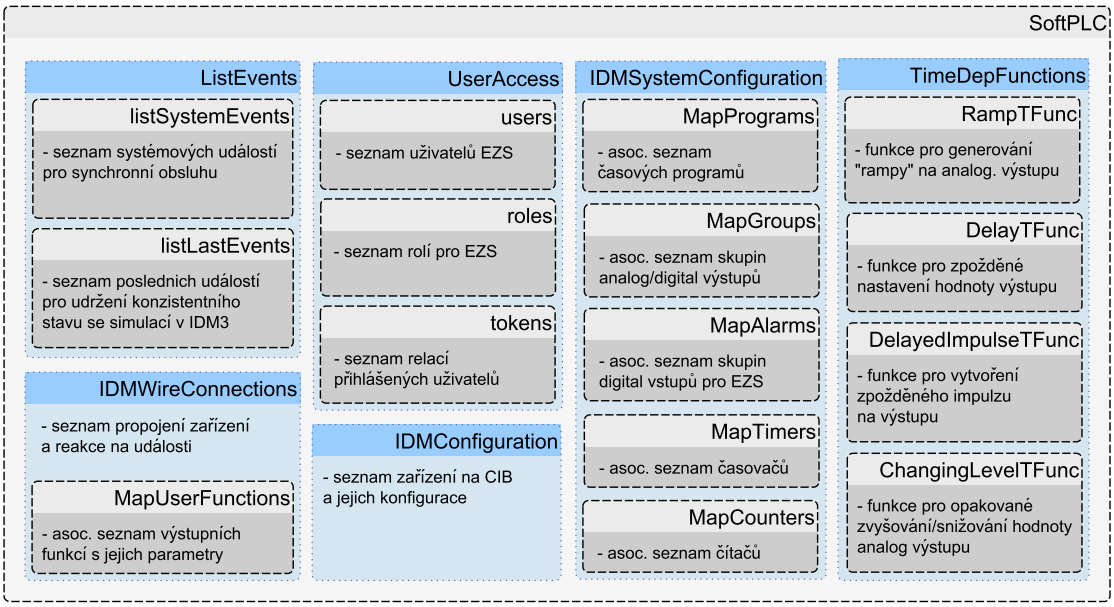
Aby mohl modul SoftPLC fungovat, potřebuje nějakou konfiguraci. Ta se nahrává pomocí SOAP rozhraní a je primárně vytvářena pomocí programu IDM3, ale je možné využít i jiného klienta, například ovládání některých prvků za použití mobilního telefonu. V SoftPLC je konfigurace uložena v několika strukturách, které jsou po vzniku události procházeny. Struktury konfigurace jsou implementovány pomocí seznamů typu `std::list` a asociativních seznamů typu `std::map`.

Typ `std::list` bývá implementován jako obousměrně vázaný seznam s konstantní časovou složitostí přidávání a odebírání prvků a hodí se pro uchovávání struktur, které budou procházeny sekvenčně. Datový typ `std::map` se hodí pro struktury, ve kterých potřebujeme vyhledávat podle klíče (v použitém datovém modelu se jedná například o atribut UnicateDeviceKey). Vyhledávání v `std::map` má logaritmickou časovou složitost a je typicky implementován jako binární vyhledávací strom [16].

S programem IDM3 jsou struktury vždy vyměňovány jako `std::list` (navíc knihovna gSOAP přenos typu `std::map` také nepodporuje), proto musí být asociativní seznamy při příjmu převedeny na `std::map` a při odesílání na `std::list`. Všechny datové struktury jsou doplněny o binární semafory (mutexy) pro vzájemné vyloučení přístupu z více vláken. Na obrázku 6.1 jsou zobrazeny struktury uchovávající konfiguraci SoftPLC. Použité datové typy jsou uvedeny v příloze D.

6.1.1 ListEvents

Struktura ListEvents obsahuje dva seznamy událostí: listSystemEvents obsahuje dvojice typu `std::pair<std::string, enum ActionsEnum>`, kde první prvek je klíč zařízení (atribut UnicateDeviceKey) a druhý je prvek z enumerátoru všech akcí. Systémové události jsou procházeny na začátku smyčky kontrolující události a podle klíče UnicateDeviceKey je procházena struktura WireConnections pro obsluhu událostí. Druhým seznamem v ListEvents je listLastEvents, který obsahuje objekty typu `ObjectAction`. `ObjectAction` obsahuje atributy Action (prvek z enumerátoru ActionsEnum) a InteractiveObject, což je kopie objektu, na kterém se objevila událost. listLastEvents slouží k přenášení událostí, které nastanou v SoftPLC a jsou důležité v IDM3 pro udržení konzistentního stavu simulace. Počet položek v listLastEvents je omezen na 100.



Obrázek 6.1: Datové struktury konfigurace SoftPLC

6.1.2 IDMWireConnections

IDMWireConnections je seznam pravidel, které definují, jak se má reagovat na vzniklé události. Prvky seznamu jsou typu `WireConnection` a obsahují atributy Actor – kopie objektu, na kterém se objevila akce, Consumer – kopie objektu, na kterém se mají vykonat výstupní funkce. Výstupní funkce jsou zároveň třetím atributem – seznam Functions. U každé výstupní funkce je definována akce, která ji spouští a seznam podmínek, které se musí před spuštěním výstupní funkce ověřit. Díky těmto podmínkám lze omezit provádění výstupních funkcí na základě testování hodnot na zařízeních. Výstupní funkce obsahují také klíč pro vyhledání seznamu parametrů, například zpoždění a délka impulzu na výstupu. K IDMWireConnections patří ještě asociativní seznam MapUserFunctions obsahuje prvky typu `UserFunction`, který obsahuje zmíněné seznamy s parametry a typ výstupní funkce.

6.1.3 UserAcess

Struktura UserAccess zahrnuje informace o přihlašování uživatelů do EZS. Seznam users s prvky typu `User` obsahuje informace o uživatelích, zejména identifikační číslo přístupové karty, ID uživatele, heslo a úroveň oprávnění. V seznamu roles jsou pomocí prvky typu `Role` definovány úrovně oprávnění. Ověření úrovně oprávnění se ve starším systému Inels2 používá například při ovládání domu pomocí SMS zpráv. Seznam tokens udržuje relace (sessions) přihlášených uživatelů a prvky seznamu jsou typu `Token`.

6.1.4 IDMConfiguration

Ve struktuře IDMConfiguration se nachází seznam zařízení na sběrnici IMB a také jejich doplňková konfigurace jako například teplotní offsety, invertování binárního výstupu nebo rozsah hodnot výstupu analogového. Zařízení jsou organizována v hierarchii, která je zobrazena na obrázku 5.2 a dělí se na vstupní, výstupní a ostatní zařízení. Základní třídou

je `DeviceInterface`. Vstupními a výstupními zařízeními mohou být zařízení analogové a digitální, do ostatních patří jednotky RFID, GSM a klávesnice EZS.

6.1.5 IDMSystemConfiguration

Systémová zařízení jsou abstraktní zařízení, která fyzicky na IMB sběrnici nenajdeme, ale mohou také vyvolávat události a pomocí struktury WireConnections je lze provazovat s fyzickými zařízeními. V IDM3 jsou systémová zařízení umístěna v jednom seznamu, v SoftPLC jsou tyto zařízení rozdělena do jednotlivých asociativních seznamů pro rychlejší vyhledávání. Systémová zařízení jsou odvozena z třídy `SystemDevice`.

Asociativní seznam MapPrograms udržuje časové programy. Ty slouží pro opakované vykonávání výstupních funkcí. Programy mohou být těchto typů:

- `PeriodicalProgram` – vykonávání výstupních funkcí ve vybraných dnech týdne, měsíce nebo v každém dni vybraného měsíce. Tento program lze také využít pro simulaci přítomnosti osob v řízené budově díky atributu DispersionTime, který definuje interval, ve kterém může událost libovolně nastat.
- `ContinuousProgram` – platnost programu je definována dvěma kalendářními daty a reakce na události lze přiřadit k začátku a konci platnosti programu.
- `HeatCoolProgram` – nastavení intenzity vytápění/klimatizace v několika krocích pro jednotlivé dny v týdnu v časových intervalech.
- `TwoStateProgram` – program simulující elektronické spínací hodiny. Pro jednotlivé dny lze v časových intervalech nastavit vypnutí a zapnutí časového programu, u kterého lze nastavit reakci na jeho aktivování a deaktivování.

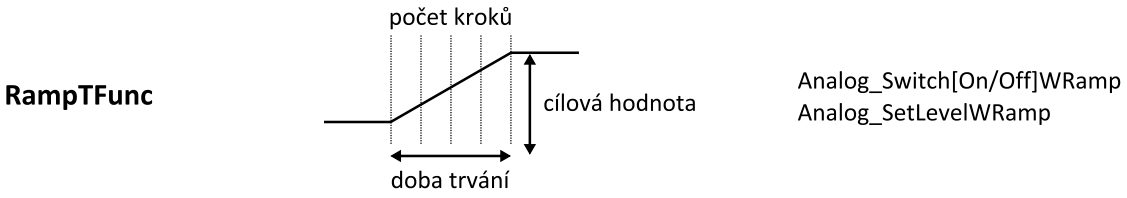
MapGroups je asociativní seznam, ve kterém se nachází skupiny analogových či digitálních výstupů nebo analogových vstupů – teploměrů. MapAlarms obsahuje skupiny digitálních vstupů, které při změně vyvolají alarm, pokud je aktivní. Systémové časovače jsou udržovány v MapTimers a čítače v MapCounters.

6.1.6 TimeDepFunctions

Většina výstupních funkcí je na výstupním zařízení provedena okamžitě. Některé výstupní funkce však mění hodnotu výstupního zařízení v čase. Příkladem může být plynulé rozsvícení žárovky, kdy se během několika sekund mění hodnota na výstupním zařízení plynule v intervalu 0-100 %. Těchto funkcí může samozřejmě běžet několik zároveň na různých zařízeních, a aby nebylo potřeba složitě uchovávat stav, ve kterém se zrovna tyto funkce nachází, je naplánován jejich běh ve zvláštním vlákně.

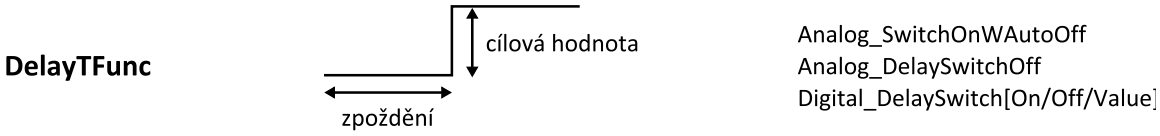
Tyto funkce jsou označeny jako časově závislé funkce a podle výstupních funkcí měnících se v čase, byly vytvořeny čtyři typy, které pokrývají všechny tyto funkce. Pokud je jedno zařízení ovládáno výstupní funkcí, nemůže být zároveň ovládáno jinou časově závislou funkcí. Požadavek na ovládání zařízení druhou časově závislou funkcí je zahozen. Časově závislé funkce jsou plánovány pomocí třídy Poco::Util::Timer a jejich instance jsou uchovány v asociativním seznamu typu `std::map`, kde klíčem je IMB adresa ovládaného zařízení. Instance časově závislých funkcí jsou uvolňovány s koncem programu nebo při nahrazení jinou instancí po dokončení běhu. Na obrázcích časově závislých funkcí jsou zobrazeny jejich parametry, průběhy a výstupní funkce, které danou časovou funkci používají.

Funkce RampTFunc plynule mění hodnotu analogového výstupu ve tvaru tzv. rampy, která může být klesající nebo stoupající. Parametry funkce jsou počet kroků (hodnota se mění skokově), cílová hodnota, které se má dosáhnout z aktuální hodnoty a doba, ze kterou se má dosáhnout cílové hodnoty. Funkce RampTFunc se nachází na obrázku 6.2.



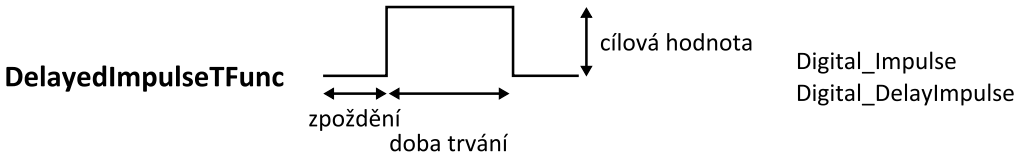
Obrázek 6.2: Časově závislá funkce RampTFunc

DelayTFunc je možné použít pro digitální i analogové výstupy. Funkce nastavuje požadovanou hodnotu výstupu se zpožděním. Funkce je zobrazena na obrázku 6.3.



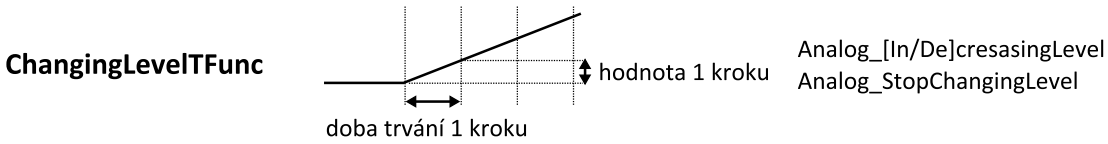
Obrázek 6.3: Časově závislá funkce DelayTFunc

Funkce DelayedImpulseTFunc umožňuje vytvářet impulzy na digitálních zařízeních, které mohou být zároveň zpožděné. Parametry jsou zpoždění impulzu, délka impulzu a hodnota na konci impulzu. Parametry funkce a související výstupní funkce jsou uvedeny na obrázku 6.4.



Obrázek 6.4: Časově závislá funkce DelayedImpulseTFunc

ChangingLevelTFunc je plynule přidává nebo ubírá hodnotu na analogovém výstupu. Pokud funkce není zastavena pomoci `Analog_StopChangingLevel`, pokračuje s přidáváním nebo ubíráním hodnoty dokud nedosáhne maxima nebo minima. Funkce je zobrazena na obrázku 6.5.



Obrázek 6.5: Časově závislá funkce ChangingLevelTFunc

6.2 Rozhraní SOAP

Server rozhraní SOAP je implementováno pomocí knihovny gSOAP. Knihovna vyžaduje popis přenášených datových struktur a implementaci metod, které budou pomocí SOAP volány na klientovi a vykonávány na serveru. V konfiguračním souboru knihovny gSOAP umístěným v `Apps/SoftPLC/idm.h` jsou tyto struktury a metody deklarovány. V tomto souboru však nemohou být implementace metod definovaných tříd – ty se nacházejí v souboru `Apps/SoftPLC/idm-functions.cpp`. Na základě konfiguračního souboru `idm.h` jsou knihovnou gSOAP vygenerovány zdrojové kódy, které zajišťují serializaci a deserializaci objektů do XML. Kompilovaný kód přináší zrychlení při parsování XML oproti univerzálním XML parserům typu SAX nebo DOM.

Mimo serializaci objektů byl také požadavek na aplikaci pro přenos binárních dat, například pro aktualizaci firmware IMB jednotek z IDM3. Po doplnění přenosu binárních dat lze uložit celý projekt z IDM3 včetně plánků řízené budovy, které jsou reprezentovány obrázky (bitmapami). Pro přenos binárních dat byl zvolen protokol MTOM, protože jako jediný fungoval v kombinaci gSOAP-C# (ještě byly testovány MIME a DIME). C# klient (v našem případě IDM3) může MTOM využívat po instalaci doplňku VisualStudia WSE3 a před odesíláním binárních dat je potřeba nastavit atribut instance SOAP klient třídy na `RequireMtom = true`. Po odeslání binárních dat se tento atribut musí vrátit zpět na `false`. gSOAP také z konfiguračního souboru vygeneruje soubory .wsdl a .xsd, které lze v C# využít pro vygenerování klientské SOAP třídy. Úplný seznam všech SOAP metod se nachází v příloze D.

Knihovnu gSOAP je možné ještě více přizpůsobit pro použití ve vestavěných zařízeních tím, že se odeberou funkce nepotřebné na těchto zařízeních. Odebráním nepotřebných funkcí se především sníží paměťové nároky. Pro použití odlehčené verze musí být knihovna přeložena s parametrem `-DWITH_LEAN`. Jedná se především o odstranění funkcí pro logování, HTTP autentizaci, serializace typu `time_t` a `LONG64`, podpory UDP a odstranění timeoutů [1].

Kapitola 7

Testování

Testování modulu SoftPLC je rozděleno na dvě části – testování SOAP komunikace a testování vlastní funkčnosti, tedy vykonávání výstupních funkcí na základě událostí, správné vytváření událostí a správná změna hodnot ve VarStore. Podle zadání diplomové práce mělo proběhnout testování modulu SoftPLC s jednotkami systému Inels. Bohužel do termínu odevzdání této práce nebyl hotový hardwarový modul řídicí jednotky CU3, který umožňuje připojení jednotek na sběrnici IMB. Pro ověření funkčnosti modulu SoftPLC bylo tedy využito simulace činnosti jednotek. SoftPLC je během testování spouštěno v programu `valgrind` s parametry `--leak-check=full --show-reachable=yes ./softPLC` pro detekci chyb spojených se špatným přístupem do paměti.

7.1 SOAP komunikace s IDM3

Test komunikace mezi SoftPLC a IDM3 přes SOAP je založen na testovacím projektu pro program IDM3. Pomocí tohoto projektu lze testovat:

- správnou implementaci SOAP metod, zda sedí typy a názvy parametrů funkcí a návratové hodnoty,
- zda probíhá správně serializace a deserializace objektů na straně serveru i klienta,
- správné naplnění a uvolnění datových struktur, převod `std::list` na `std::map`, alokace/uvolňování paměti při hlubokém kopírování objektů,
- přenos binárních dat, ukládání a načítaní z databáze SQLite,
- v simulaci správné vyvolání a obsluhu událostí,
- chyby v implementaci datového modelu.

Pro testování byl mimo modul SoftPLC, program IDM3 a valgrind použit ještě program Fiddler2. Fiddler2 je HTTP proxy server určený pro ladění webových aplikací. Žádosti a odpovědi procházející přes proxy server lze prohlížet v různých zobrazeních – jako text, surová data, XML, hexadecimální hodnoty. Pro testování je využit režim zobrazení XML a jsou porovnávány přenášená data směrem do a z SoftPLC. Například při zavolání metody SaveConfiguration dojde v IDM k serializaci seznamu Configuration, tedy převedení do XML reprezentace, která je odeslána SoftPLC. SoftPLC přijaté XML deserializuje a vytvoří

z něj seznam IDMConfiguration. Pokud by došlo k problému s deserializací, například kvůli špatné implementaci datového modelu, bude seznam IDMConfiguration prázdný.

Testovací projekt obsahuje dvě podlaží, každé z nich má načtený obrázek jako půdorys. V podlaží označeném jako přízemí jsou rozmístěny všechny prvky systému, podlaží s názvem první patro je ponecháno prázdné. Ve Správci zařízení je přidáno sedm IMB jednotek a jsou připojeny do centrální jednotky CU3. Prvky systému jako vypínače a zářivky jsou rozmístěny na půdorysu podle jejich umístění v řízené budově a připojeny na vstupy a výstupy IMB jednotek. V plánu přízemí se nachází čtyři vypínače, tři zářivky, tři světla, dva detektory otevřených dveří a jeden detektor otevřeného okna. Dále se v plánu nachází dvě skupiny zařízení, ve kterých se nachází světla a zářivky, a skupina alarmu, kde jsou detektory dveří a okna.

Pro uložení konfigurace na ústřednu jsou použity následující SOAP metody v tomto pořadí:

- `SaveFile()` – uložení plánů podlaží (bitmap) a souřadnic prvků rozmístěných v plánech, binární data jsou uložena pomocí typu `xsd__base64Binary`, pro načtení dat je použit formát MTOM a SOAP metoda `LoadFile()`. Při srovnání vyměňovaných SOAP zpráv je možné si všimnout o přibližně 30 % delší zprávy využívající typ `xsd__base64Binary` oproti zprávě ve formátu MTOM.
- `SaveUserFunctions()` – uložení seznamu výstupních funkcí včetně jejich typu a parametrů, načtení `LoadUserFunctions()`,
- `SaveWires()` – uložení seznamu propojení událostí, senzorů s aktory a výstupními funkcemi, načtení `LoadWires()`,
- `SaveConfiguration()` – uložení seznamu jednotek na sběrnici IMB včetně jejich konfigurace, načtení `LoadConfiguration()`,
- `SaveSystemConfiguration()` – uložení seznamu systémových zařízení, načtení pomocí `LoadSystemConfiguration()`,
- `SaveGroups()` – uložení seznamu skupin analogových vstupů/výstupů a digitálních výstupů, načtení `LoadGroups()`,
- `SavePrograms()` – uložení seznamu časových programů, načtení `LoadPrograms()`.

Obsah zaznamenaných SOAP zpráv ve formátu XML s ukládanými i načítanými strukturami jsou dostupné na přiloženém CD v adresáři `tests`.

7.1.1 Ukázka serializace/deserializace

Ukázka serializace a deserializace objektu `DigitalOutputRO` je na obrázku 7.1.

7.2 Ověření funkčnosti modulu SoftPLC

V IDM3 zatím není možné přiřadit logické adresy jednotlivým funkcím (devices) IMB jednotek pro přístup do VarStore, proto byl pro test funkčnosti modulu SotfPLC vytvořen SOAP klient s testovací konfigurací a změnou hodnot ve VarStore pro simulaci změn hodnot na jednotkách IMB. Test se skládá ze dvou částí. Obě části testu se nachází na přiloženém CD v `Apps/SoftPLC/tests.cpp`. Pro účely testování je smyčka pro kontrolu událostí volána

```
<Device xsi:type="DigitalOutputRO" Type="DigitalOutputRO" IsUsed="NotSpecified"
ExportForVisualization="false" IsInverted="NotUsed">
  <UnicateDeviceKey>DigitalOutputRO_3</UnicateDeviceKey>
  <Value xsi:type="xsd:boolean">true</Value>
  <Name>RO2</Name>
  <Description />
  <Address>255</Address>
</Device>

<Device IsUsed="NotSpecified" ExportForVisualization="false" IsInverted="NotUsed"
xsi:type="DigitalOutputRO" Type="DigitalOutputRO">
  <UnicateDeviceKey>DigitalOutputRO_3</UnicateDeviceKey>
  <Value xsi:type="xsd:boolean">true</Value>
  <Description></Description>
  <Name>RO2</Name>
  <Address>255</Address>
</Device>
```

Obrázek 7.1: Ukázka serializace (první část, IDM3) a deserializace (druhá část, SoftPLC) objektu `DigitalOutputRO`

s periodou 2,5 s a limit pro rozlišení krátkého a dlouhého stisku na digitálních vstupech je 7,5 s.

První část je založena na obecném testu, jehož inspirací byla předchozí verze testovacího projektu v IDM3. V seznamu Configurations se nachází dva vypínače DI1, DI2 a jedno tlačítko TL1, dále tři zářivky (RO1, RO2, RO3) a tři světla (RO4, RO5, RO6). Vypínače jsou typu `DigitalInputDC`, tlačítko `DeviceTL` a zářivky se světly `DigitalOutputRO`. Jako systémové zařízení vystupuje časovač Timer0. Seznam Groups obsahuje dvě skupiny typu `Digital`. První skupina je označena jako „obyvák“ obsahuje zářivky RO1, RO2, RO3, druhá skupina má název „chodba“ a obsahuje žárovky RO4, RO5, RO6. V seznamu UserFunctions se nachází pět výstupních funkcí bez parametrů. Každá funkce má přiřazen unikátní název a v závorce je uvedeno jaká výstupní funkce z `enum FunctionsEnum` se má vykonat. Jsou definovány tyto funkce: PrepniRele(Digital_SwitchValue), StiskTlacitka(Button_Click), TimerStart(Timer_Start), TimerStop(Timer_Stop), PrepniSkupinu(GroupL_Switch). Dále je definována tabulka WireConnections 7.1:

Actor	**Consumer**	**Action**	**Functions**
DI1	RO1	Digital_IN_ShortDown	PrepniRele
DI2	skupina chodba	Digital_IN_LongDown	PrepniSkupinu
DI2	Timer0	Digital_IN_ShortDown	TimerStart
DI2	Timer0	Digital_IN_ShortUp	TimerStop
Timer0	skupina chodba	Timer_Elapsed	PrepniRele
TL1	RO4	Button_Click	PrepniSkupinu

Tabulka 7.1: Tabulka WireConnections v první části testu

Druhá část testu je zaměřena na ověření časově závislých funkcí. Pro tento test jsou přidány další dva vypínače DI4, DI5 typu `DigitalInputDC` a dvě světla RO7, RO8 typu `AnalogOutputAO`. Pro tyto dvě světla je vytvořena skupina „Svetla“. Do seznamu UserFunctions jsou přidány následující výstupní funkce: Ramp(Analog_SetLevelWRamp), OnAu-

toOff(Analog_SwitchOnWAutoOff), Impulse(GroupL_DelayImpulse), Decrease(Analog_DecreasingLevel). Obsah seznamu WireConnections je uveden v tabulce 7.2.

Actor	**Consumer**	**Action**	**Functions**
DI4	RO7	Digital_IN_LongDown	Ramp
DI4	RO7	Digital_IN_LongUp	Decrease
DI5	skupina Obyvak	Digital_IN_LongDown	Impulse
DI5	RO7	Digital_IN_LongUp	OnAutoOff

Tabulka 7.2: Tabulka WireConnections v druhé části testu

Nejprve je pro každé zařízení typu vypínač nebo žárovka vytvořena proměnná ve VarStore a všechny hodnoty proměnných jsou inicializovány na nulu. Poté se pomocí SOAP přenese do SoftPLC výše popsaná konfigurace. Po načtení konfigurace v SoftPLC začne test měnit hodnoty ve VarStore a tím simulovat události na IMB zařízeních. V tabulce 7.3 jsou uvedeny změny hodnot ve VarStore.

Čas[s]	**Senzor**	**Hodnota**	**Událost**
0	DI1	1	Digital_IN_ShortUp
3	DI1	0	Digital_IN_ShortDown
3	TL1	1	Button_Click
3	DI2	1	Digital_IN_ShortUp
13	DI2	0	Digital_IN_LongDown
23	DI2	1	Digital_IN_LongUp
26	DI2	0	Digital_IN_ShortDown
34	DI2	1	Digital_IN_LongUp
37	DI2	0	Digital_IN_ShortDown
41	DI2	1	Digital_IN_ShortUp
53	DI4	1	Digital_IN_LongUp
65	DI4	1	Digital_IN_LongUp
77	DI4	0	Digital_IN_LongDown
89	DI5	1	Digital_IN_LongUp
101	DI5	0	Digital_IN_LongDown

Tabulka 7.3: Simulace změny vstupů na IMB jednotkách

7.2.1 Ukázka testu funkčnosti

Na obrázku 7.2 je uvedena ukázka testu. Ve výpisu je u každého řádku vypisováno datum a čas, typ zprávy (I – information, W – warning). Číslo za typem zprávy označuje vlákno. Na této ukázce vidíme, že se změnila hodnota proměnné ve VarStore u proměnné s adresou 2 na hodnotu 0. Podle typu zařízení a uplynulé doby od poslední hrany na digitálním vstupu bylo rozhodnuto, že se stala událost typu Digital_IN_LongDown. Po zjištění typu události je procházena tabulka WireConnections, ve které se nejprve hledá podle unikátního klíče zařízení (UnicateDeviceKey) a poté podle typu události.

V případě shody obou položek jsou vykonávány výstupní funkce. Na výpisu vidíme, že se našlo pravidlo, které se shoduje s nastalou událostí a je vykonána funkce GroupL_Switch na skupině Obyvak. Poté existují ještě další dvě pravidla ve WireConnections, která mají

shodu v položce Actor ale už ne v Action. V dalším cyklu smyčky pro obsluhu událostí vidíme změnu tří proměnných ve VarStore, u kterých byla změněna hodnota v předchozím kroku. Na těchto zařízeních byly vyvolány události ale v seznamu WireConnections nejsou události na těchto zařízeních obsluhovány. Výpis z celého testu je z důvodu jeho rozsahu umístěn pouze na CD v `tests/test_funkcnosti_log`.

```
13-05-19 23:04:28 I~2 ### new iteration checkIMBDevicesAndSysEvents ###

13-05-19 23:04:28 I~2 values changed in VS:
13-05-19 23:04:28 I~2  IMB dev[addr: 2, val: 00]
13-05-19 23:04:28 I~2 Current value: 0 lastTstamp: 1368997458737588 tstamp diff: 10086837 [uS]
13-05-19 23:04:28 I~2 Event determined: Digital_IN_LongDown on iface DI2
13-05-19 23:04:28 I~2 Rule match (actor->consumer unique dev keys): DI2->Obyvak
13-05-19 23:04:28 I~2 Action match: DI2: Digital_IN_LongDown
13-05-19 23:04:28 I~2 UserFunction execution:
13-05-19 23:04:28 I~2 GroupL_Switch on the group: Obyvak
13-05-19 23:04:28 I~2 DigitalOutput on the IMB address: 4 - setValue (0)
13-05-19 23:04:28 I~2 DigitalOutput on the IMB address: 5 - setValue (1)
13-05-19 23:04:28 I~2 DigitalOutput on the IMB address: 6 - setValue (1)
13-05-19 23:04:28 I~2 Rule match (actor->consumer unique dev keys): DI2->Chodba
13-05-19 23:04:28 I~2 Rule match (actor->consumer unique dev keys): DI2->Timer1
13-05-19 23:04:28 I~2
13-05-19 23:04:31 I~2 ### new iteration checkIMBDevicesAndSysEvents ###

13-05-19 23:04:31 I~2 values changed in VS:
13-05-19 23:04:31 I~2  IMB dev[addr: 4, val: 00]
13-05-19 23:04:31 I~2 Event determined: Digital_OUT_SwitchOff on iface RO1
13-05-19 23:04:31 W 2 The event Digital_OUT_SwitchOff is not registered on the device RO1.
13-05-19 23:04:31 W 2 The actor RO1 is not registered in WireConnections.
13-05-19 23:04:31 I~2
13-05-19 23:04:31 I~2  IMB dev[addr: 5, val: 01]
13-05-19 23:04:31 I~2 Event determined: Digital_OUT_SwitchOn on iface RO2
13-05-19 23:04:31 W 2 The event Digital_OUT_SwitchOn is not registered on the device RO2.
13-05-19 23:04:31 W 2 The actor RO2 is not registered in WireConnections.
13-05-19 23:04:31 I~2
13-05-19 23:04:31 I~2  IMB dev[addr: 6, val: 01]
13-05-19 23:04:31 I~2 Event determined: Digital_OUT_SwitchOn on iface RO3
13-05-19 23:04:31 W 2 The event Digital_OUT_SwitchOn is not registered on the device RO3.
13-05-19 23:04:31 W 2 The actor RO3 is not registered in WireConnections.
```

Obrázek 7.2: Ukázka z testu funkčnosti

Kapitola 8

Závěr

V práci jsem představil současné trendy v oblasti řízení inteligentních budov, používané standardy a sběrnice. Zmínil jsem, jak fungují PLC a Soft PLC i důvody použití Linuxu v oblasti vestavěných systémů. Dále jsem uvedl technologie, které využívá firma Elko EP, především sběrnici CIB a použité řídicí jednotky. Čtenáře jsem seznámil s protokolem SOAP a možností přenosu binárních dat pomocí tohoto protokolu. V Návrhu jsem formuloval požadavky na celý řídicí systém CU3 a zejména na vyvíjený modul SoftPLC, u kterého jsem následně navrhl jeho implementaci. V kapitole Implementace jsem popsal vytvořenou aplikaci SoftPLC a její datové struktury. Implementaci modulu jsem následně testoval s programem IDM3 a samostatným SOAP klientem, pomocí kterého byla testována funkčnost modulu pomocí simulace IMB jednotek.

Výsledkem této práce je implementovaný modul SoftPLC řídicí aplikace CU3, který lze konfigurovat programem IDM3. SoftPLC umožňuje reagovat na změny digitálních a analogových vstupů prvků systému Inels a na události systémových zařízení. Podle zadaných pravidel a pomocí výstupních funkcí lze tak řídit budovu s inteligentní elektroinstalací.

Jako práce navazující na tento projekt se nabízí díky modulárnímu řešení mnoho možností. Například je možné implementovat moduly IM Worker pro další typy sběrnic jako je CAN nebo Ethernet. K modulu VarStore by bylo vhodné vytvořit aplikaci typu HMI, která bude přehledně zobrazovat stavové informace řídicího systému. Protože nebyl zatím celý systém CU3 testován s fyzickými jednotkami Inels, je možné že odesílání hodnot z VarStore do ostatních modulů přes XML bude tvořit úzké hrdlo systému a bude potřeba implementovat toto rozhraní jiným způsobem. Samotný modul SoftPLC by bylo možné rozšířit o spouštění uživatelských skriptů, které mohou využívat Linuxové řídicí jednotky nad rámec konfiguračního programu IDM3. Případně by bylo užitečné nahradit modul SoftPLC modulem, který by se programoval podle normy IEC 61131, podobně jak je tomu u projektu Beremiz.

Literatura

[1] The gSOAP Toolkit for SOAP Web Services and XML-Based Applications. 2012, [Online], [cit. 2013-01-08].
URL http://www.cs.fsu.edu/~engelen/soap.html

[2] Beremiz. 2013, [Online], [cit. 2013-04-09].
URL http://www.beremiz.org/

[3] POCO C++ Libraries. Applied Informatics Software Engineering GmbH, 2006-2012, [Online], [cit. 2013-01-07].
URL http://pocoproject.org

[4] iNELS - Technický katalog. Elko EP, 2012, [Online], [cit. 2013-01-07].
URL
http://www.elkoep.cz/downloads/promotion_materials/iNELS_SHS_02.pdf

[5] EnOcean, Energy Harwesting Wireless Sensor Modules. EnOcean GmbH, 2013, [Online], [cit. 2013-01-08].
URL http://enocean.com

[6] DALI AG. ZVEI Division Luminaires, 2012, [Online], [cit. 2013-01-08].
URL http://dali-ag.org

[7] ARM Ltd.: ARM926EJ-S: Technical Reference Manual. 2008, [Online], [cit. 2013-04-27].
URL http://infocenter.arm.com/help/topic/com.arm.doc.ddi0198e/DDI0198E_arm926ejs_r0p5_trm.pdf

[8] Cho, P.: *Attachments in SOAP Messages*. Oracle, 2005, [Online], [cit. 2013-05-14].
URL http://www.oracle.com/technetwork/middleware/ias/ws-attachment-pcho-130995.pdf

[9] Croome, D.: *Intelligent Buildings: Design, Management And Operation*. Inst of Civil Engineers Pub, 2004, ISBN 9780727732668.

[10] Garlík, B.: Inteligentní budovy. *Elektro*, 2012: s. 8–12, ISSN 1210-0889.

[11] Klaban, J.: Inels a sběrnice CIB – moderní systém inteligentní elektroinstalace. *Automa*, 2008: s. 70–71, ISSN 1210-9592.

[12] Kovář, J.; Prokopová, Z.; Šmejkal, L.: Programování dle normy IEC 61 131. 2010, [Online], [cit. 2013-03-21].
URL
http://www.spszl.cz/soubory/plc/programovani_dle_normy_iec61131.pdf

[13] Mueller, J.: *Special edition using SOAP*. Special Edition Using Series, QUE Corporation, 2001, ISBN 9780789725660.

[14] Novotný, T.; Marvan, A.; Hájek, J.; aj.: Vestavěný Linux pro inteligentní dům. In *Sborník příspěvků z 41.konference EurOpen.CZ*, EurOpen.CZ, 2012, ISBN 978-80-86583-24-2, s. 15–30.

[15] Plaza, I.; Medrano, C.; Blesa, A.: Analysis and implementation of the IEC 61131-3 software model under POSIX Real-Time operating systems. *Microprocessors and Microsystems*, ročník 30, č. 8, 2006: s. 497 – 508, ISSN 0141-9331, [Online], [cit. 2013-04-21].
URL http://www.sciencedirect.com/science/article/pii/S014193310600086X

[16] Stroustrup, B.: *The C++ programming language*. Addison-Wesley, třetí vydání, 1997, ISBN 9780201889543.

[17] Šmejkal, L.: Co jsou chytré domy? *Perspektivy bydlení (speciální příloha časopisů Elektro a Automa)*, 2012: str. 2.

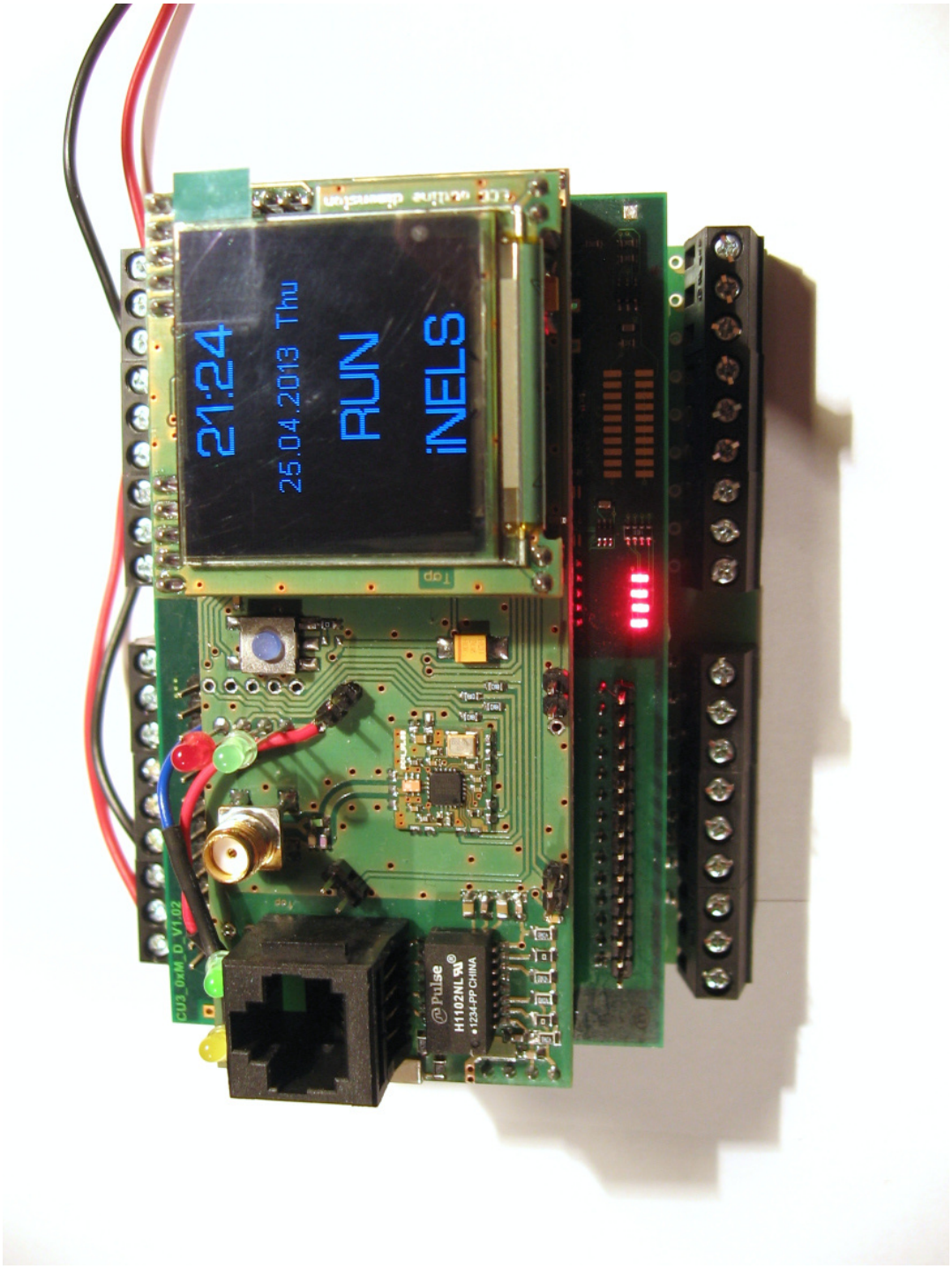
[18] Šmejkal, L.; Martinásková, M.: *PLC a automatizace*. BEN, první vydání, 1999, ISBN 80-86056-58-9, 223 s.

Příloha A

Obsah CD

Soubor	Popis
CU3/	adresář s řídicí aplikací CU3, která se skládá z modulů IMWorker, VarStore a SoftPLC
tests/	adresář s výstupy testů
doc.pdf	technická zpráva
doc.zip	zdrojové soubory technické zprávy
README	uživatelská příručka aplikace CU3

Příloha B



Obrázek B.1: Vývojová verze jednotky CU3

Příloha C

Obrázek C.1: Ukázka programu IDM3

Příloha D

Diagram SOAP rozhraní

Příloha je umístěna na přiloženém listu ve formátu A3.

Příloha D: Diagram SOAP rozhraní