ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta Elektrotechnická

Katedra měření

Využití řídicí jednotky motor-generátor v prostředí inteligentních budov

Use of Gen-set controller in the Environment of Intelligent Buildings

Diplomová práce

Studijní program: Inteligentní budovy

Studijní obor: Inteligentní budovy

Vedoucí práce: Ing. et Ing. Jakub Jiříček (ComAp a.s.)

Bc. Yauhen Minko

Praha 2015

Čestné prohlášení

Prohlašuji, že jsem diplomovou práci na téma „*Využití řídicí jednotky motor-generátor v prostředí inteligentních budov“* vypracoval samostatně s použitím odborné literatury a pramenů, uvedených na seznamu a za přispění konzultací u vedoucího práce Ing. et Ing. Jakuba Jiříčka ze společnosti ComAp a.s.

V Praze, dne 06.05.2015 Bc. Yauhen Minko

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická
Katedra měření

Akademický rok 2014-15

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Yauhen Minko**
Studijní program:	**Inteligentní budovy**
Název tématu česky:	**Využití řídicí jednotky soustroji motor-generátor v prostředí inteligentních budov**
Název tématu anglicky:	**Use of Gen-set Controller in the Environment of Intelligent Buildings**

Pokyny pro vypracování:

Seznamte se s řídicími jednotkami pro soustrojí motor-generátor a pro spínací prvky v silových elektrických rozvodech od výrobce ComAp a.s. Stručně popište jejich funkci a typické využití. Prostudujte problematiku týkající se Building Management Systému (BMS). Naprogramujte aplikaci, která bude simulovat interakci mezi výše uvedenými řídicími jednotkami a BMS. Aplikace bude využita pro prezentační účely a pro laboratorní úlohy předmětu „Senzory a sítě". Navrhněte úlohy tak, aby se uživatel nebo student mohl seznámit s:

1) navázáním komunikace pomocí SNMP protokolu s řídicí jednotkou;
2) základními funkcemi řídicí jednotky a synchronizací generátorů;
3) interakcí řídicí jednotky s inteligentní budovou.

Seznam odborné literatury:

[1] Manuály k jednotlivým komponentám společnosti ComAp a.s.
[2] Mauro, D., Schmidt, J.: Essential SNMP, Beijing: O'Reilly
[3] Herman, M., Hansemann, T., Hübner, Ch.: Automatizované systémy budov, Praha: Grada

Vedoucí diplomové práce:	Ing. et Ing. Jakub Jiříček (ComAp a.s.)
Datum zadání diplomové práce:	15. ledna 2015
Platnost zadání do[1]:	31. srpna 2016

[redacted]
Doc. Ing. Jan Holub, Ph.D.
vedoucí katedry

L. S.

[redacted]
Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 15. 1. 2015

[1] Platnost zadání je omezena na dobu tří následujících semestrů.

Abstrakt

Cílem této práce je ukázka interakce mezi modelem inteligentní budovy a řídicí jednotkou *InteliSys*^{NTC} od firmy ComAp a.s. vestavené do *Starter Kitu*. Model budovy je realizován v samostatně naprogramované aplikaci v jazyce C#. *Starter Kit* představuje kufr s implementovanou simulací gen-setu, elektrické sítě a zátěží. Hlavní pozornost je věnována projektování aplikace a popisu rozhraní SNMP, přes které se probíhá komunikace. Práce obsahuje stručný popis samotné řídicí jednotky a její využití. Rovněž je prostudována problematika řízení systémů budov. Výsledkem je souprava, kterou lze použit jak pro prezentační účely na odborných veletrzích, tak i pro laboratorní úlohy pro studenty.

V závěru budou shrnuty poznatky získané při tvorbě tohoto diplomového projektu.

Klíčová slova

BMS, ComAp, InteliSys, gen-set, SNMP, Visual Studio, C#

Abstract

The aim of this work is a demonstration of the interaction between model of “smart” building and control unit *InteliSys*^{NTC} from ComAp a.s. embedded to *Starter Kit*. The building model is realized in own C# application. *Starter Kit* represents case with implemented simulation of gen-set, power grid and auxiliary loads. Main attention is dedicated to application development process and SNMP interface description. The work briefly describes the control unit and its applications. Building management issues are also studied. The result is a kit, which can be applied both for presentation purposes at the exhibitions as well as for laboratory exercises for students.

The conclusion will summarize the knowledge obtained in creating this master project.

Key words

BMS, ComAp, InteliSys, gen-set, SNMP, Visual Studio, C#

Obsah

1. Předmluva

V našem životě se každý den setkáváme s projevy elektřiny. Díky ní můžeme rozsvítit žárovky, použít výtah, jet tramvají. Proto, aby všechna tato i jiná zařízení fungovala správně, je nutné umět regulovat a měřit základní parametry elektrické sítě: činný a jalový výkon, frekvenci, napětí a proud. Měření a regulace (MaR) představuje poměrně složitý systém, který zahrnuje nejen samotnou regulaci a měření. Při návrhu MaR je nutné brát v potaz specifika fungování jednotlivých systémů jako například elektrárna, nemocnice, továrna a zohlednit i rozvody elektrické energie. Jedná se v podstatě o odpovědi na otázky „Co se stane, když…" a „Proč měřená veličina je právě taková? ". Řešit tyto záležitosti rychle a efektivně v dnešní době nám pomáhají počítače a případně řídicí systémy. S jejích pomoci se navrhují složité systémy, jako například systém řízení budovy (*Building Management Systém, dále - BMS*). Systémy řízení budov kontrolují např. ventilaci, topení, klimatizaci (*HVAC*), záložní zdroje elektřiny, požární a zabezpečovací systémy. Příkladem inteligentní budovy se systémem řízení uvedeným v této práci je Národní Technická Knihovna v Praze. Problematice BMS a hlavně řízení spotřeby a výroby elektřiny v budovách je věnována první podkapitola teoretické část této práce.

Druhá polovina teoretické části pojednává o možnostech řízení soustrojí motor-generátor (gen-set) v rámci napájení budov elektrickou energii. Čtenář se bude moci stručně seznámit s vybranou řídicí jednotkou *InteliSys*^{NTC} od společnosti ComAp a její funkčností. Jednotka neboli kontrolér se zabývá ovládáním a monitoringem elektrických a mechanických veličin gen-setu a také regulací spínacích prvků v silových elektrických rozvodech. Bude popsána ukázková souprava *Starter Kit,* do níž je kontrolér vestaven. Souprava bude simulovat reálnou budovu se systémem napájení a zátěží. Také bude uvedena charakteristika softwaru *InteliMonitor,* kterým lze ovládat a parametrizovat řídicí jednotku.

V praktické části této práce se řeší programování aplikace v C# , která bude sledovat změny, provedené uživatelem v počítačovém modelu („softwarová" budova) a posílat data přes komunikační protokol SNMP do řídicí jednotky v *Starter Kitu*, anebo naopak na základě zmáčknutého přepínače soupravy dokáže změnit počítačový model, což se odrazí na uživatelském rozhraní. Aplikace bude obsahovat jednoduchou SCADA – Supervisory Control And Data Acquisition, systém pro dispečerské řízení a sběr dat. Následně bude rozepsána struktura aplikace, dodržení návrhových vzorů při programování a použitých pomocných knihovnách.

Výsledek práce by se mohl využít jako ukázka schopností kontrolérů společnosti ComAp a.s. na odborných veletrzích a u laboratorní úlohy na Fakultě elektrotechnické ČVUT, která je také stručně popsána v praktické části. Plná verze úlohy se nachází v přílohách k této práci. Student nebo uživatel by se seznámil se základním modelem napájení budov, specifikou ovládání zdrojů elektřiny a událostmi, které mohou ovlivnit zásobování elektrickou energií. Sloužilo by to k propojení studia a praxi. Pro studenty by bylo přínosem s hlediska rozšíření přehledu o způsobech regulace a dostupných výrobkách na trhu.

2. Teoretická část

2.1 Systémy řízení budov

2.1.1 Úvod

Moderní „chytré“ budovy jsou složitou a vícestrannou záležitostí. Musíme dodržet vyhovující stupeň ochrany budovy před nežádoucím vniknutím, sledovat a řídit komfortní mikroklimat pro osoby, kteří tam pracují, bydlí nebo odpočívají. Rovněž je nutno vyhovět požadavkům na co nejnižší spotřebu tepla, vody a elektřiny z ekologických a ekonomických důvodů.

Inteligentní budovy, kde se provádí automatizace, lze rozdělit na dvě velké části. Jednak jsou to rodinné výstavby, kde hlavně dbáme na hospodárnost a úspory energií, bezpečnost a komfort. Druhá část je představená tzv. „účelovými budovami“. Pod tímto pojmem rozumíme budovy, které plní určité funkční zaměření. Do kategorie účelových budov lze zařadit například obchodní střediska, nemocnice, letiště a vojenské objekty. Požadavky na tento typ budov ovlivňuje několik činitelů, například chování uživatelů. Často v obchodních centrech, kancelářských budovách a letištních terminálech značnou část účelových ploch zabírají firmy, jejíž počet pracovníků může se měnit s časem třeba kvůli restrukturalizaci. To znamená, že vznikne potřeba přeplánovat kanceláře nebo obchod. Tím se změní podmínky klimatizace nebo osvětlení. Proto koncepce účelové budovy musí vyhovovat případným změnám v budoucnu.

Účelové budovy se dá chápat i jako komerční produkt. Každý komerční produkt by měl směřovat k co nejvyšší kvalitu spolu s rozumnou cenou jak investičních, tak i provozních nákladů. Proto účelové výstavby jsou zpravidla vybavené rozsáhlými řídicím a regulačním systémem, který kromě třeba optimalizace spotřeby energie navíc sníží počet pracovníků obsluhy budovy. Počet akčních členů snímačů a regulátorů u těchto systémů může být obrovský. Vzniká tím pádem snaha spojit systémy navzájem prostřednictvím komunikačních sítí a datových sběrnic do jednoho celku a tento celek jednoduše ovládat z dispečerského pracoviště s nainstalovaným příslušným softwarem. (1 str. 11)

Jako shrnutí lze konstatovat, že při navrhování účelových budov se klade důraz na:

- Flexibilitu užití
- Hospodárnost provozu
- Komfort uživatelů
- Jednoduchost řízení systémů budovy

V této práci bych se více věnoval účelovým budovám, protože výsledkem mé práce je aplikace, jejíž účel a hlavně elementy jsou pro tento typ budov určeny.

2.1.2 Definice a popis

Systém řízení budovy (*Building Management System*, dále jen *BMS*) je určen k automatizaci procesů a operací, které v budově probíhají. Na BMS lze se dívat ze dvou pohledů: softwarového a hardwarového. Softwarem je speciální program nainstalovány na dispečerském pracovišti, které je připojeno ke komunikační sítí. Program hodnotí data z několika systémů budovy a na základě těchto dat může posílat povely ke změně činnosti ostatních systémů. Dalšími funkcemi jsou vizualizace dat pro uživatele, ukládání historie provedených změn, naměřených veličin a případných chybových hlášeních, sdílení informací o spotřebě energie, vody nebo plynu nadřazenému zúčtovacímu systému. Funkcí řízení spotřeby elektřiny neboli *Energy Managementu* je věnována podkapitola Energy Management.

Hardwarová část je představená akčními členy (servopohony, řízené spínače), snímači (průtokoměry, elektroměry, senzory pohybu a námrazy) a regulátory (PLC, dispečerské počítače). Snímače a akční členy často bývají vestavené do příslušného zařízení, zatímco regulátory se nacházejí v bezprostřední blízkosti, většinou v rozvodné skříni. Zpravidla regulační prvky můžou plnit svoje funkce autonomně bez přítomnosti nadřazeného systému a již na své úrovni zajišťovat úsporný a bezpečný provoz monitorovaného zařízení.

Většina BMS je postavena na hierarchickém principu. Rozlišují se tři úrovně automatizace:

- Vyšší (*Management Level*) – reprezentuje softwarové zaměření. Úroveň obsahuje databáze, SCADA systémy a další počítačové prostředky pro interakci mezi obsluhou (dispečery, administrátory) a samotným systémem budovy.
- Střední (*Automation Level*) – úroveň řízení funkčních procesů, kterou lze vnímat jako mezivrstvu. Sem řadíme PLC kontroléry, I/O moduly a komunikační zařízení, které mají určité funkce MaR již implementované.
- Nižší (*Field Level*) – snímače, detektory a výkonné prvky. Představuje hardwarový pohled (2)

V budovách, zejména účelových, předpokládá se velký stupeň mezisystémové informatické integrace. Základním předpokladem takové integrace je správně navržené komunikační propojení. Protože v budově mezi automatizovanými místy (střední úroveň) a nadřazeným počítačem (vyšší úroveň) za krátkou dobu mohou předávat tisíce různých informací, kladou se na komunikační rozhraní nároky na velkou přenosovou schopnost. V závislosti na typu přenášených dat se může vyžadovat od rozhraní zvýšená spolehlivost.

Například pokud se ve výrobní firmě nadřazený systém kvůli ztrátě komunikačního paketu nedozví, že byla zaznamenána porucha stroje a nenahlásí to včas obsluze, mohou se zastavit další vázané stroje, což může vést k vážné finanční škodě a (nebo) k úrazům nacházejících se v budově lidi.

S rostoucím počtem výrobců elementů pro automatizaci se zvyšuje i nabídka měřících a regulačních zařízení. Aby tyto zařízení, případně celé systémy, byli kompatibilní a mezi sebou integrovatelné, vznikla snaha přijmout otevřené a společné komunikační protokoly. Nemalý význam měl i aspekt závislosti zákazníků čili provozovatelů budov na dodavatelích. Když provozovatel měl zvolený nebo již instalovaný systém od jednoho výrobce, pak se musel při rekonstrukci z důvodu zajištěné kompatibility opět obrátit na toho samého výrobce nebo dodavatele jeho výrobků. Proto třeba největší evropské firmy-výrobci se dohodli na kompatibilní sběrnici KNX/EIB. Jako další obecné používané standardy v automatizaci budov lze uvést BACnet, LON nebo LCN.

Podle druhu technických zařízení v budově provádíme úpravu a monitoring parametrů, funkcí (1 stránky 25-26):

Vytápění, chlazení, ventilace (*HVAC*, *Heating,Ventilation,Air-conditioning*)

- Větrání v závislosti na kvalitě vzduchu v místnosti
- Požadované hodnoty pokojové teploty v závislosti na počtu osob
- Vytápění nebo chlazení: vliv otevřených oken
- Teplota v místnosti v případě ruční regulace termostatem

Osvětlení

- Spuštění osvětlení na základě přítomnosti osob
- Konstantní hodnota osvětlení udržována pomocí snímače jasu
- Regulace světla nastavením lamel žaluzií dle intenzity slunečního světla
- Světelné scény

Bezpečnost

- Vyznačení únikových cest při požáru
- Pomocí detektoru kouře měříme míru škodlivin. V případě překročení maximální přípustné hodnoty, řídicí systém vydá pokyn k odvětrání elektricky nastavitelnými okny
- Vyhlášením poplachu se sepne osvětlení

- Individualizovaná kontrola vstupu osob v přístupovém prostoru buď na základě karet, nebo snímáním biometrických údajů

Multimedia

- Spřažení serverů audio a video pro individuální rozmístění
- Ovládání vybavení místnosti systémem Personal Digital Assistent, mobilním telefonem anebo PC

S pohledu finančního pořizování BMS vypadá zcela zajímavé i když je potřeba předem ocenit, jak inteligentní řídicí funkce přispějí ke snížení provozních nákladů. V průměru cena automatizační a regulační techniky dosahuje 1,25 % celkových investičních nákladů na stavbu. Roční energetické provozní náklady představují částku 2 % až 4 %. Pokud podle konzervativních odhadů uspoříme díky automatizaci 10 % energie za rok, vrátí se nám investice do automatizační techniky za 4 roky (1 str. 19). O řešeních vedoucích k úsporám elektrické energie říká více následující podkapitola.

Kromě finančních výhod přináší implementace systému řízení pro majitele budov další plusy jako flexibilita, možnost dálkového ovládání a jednodušší hledání a řešení případně vzniklých problémů.

2.1.3 Energy Management

Jednou z nejvýznamnějších úloh automatizace budov je zajistit takový provoz, který bude z energetického hlediska šetrný. Touto problematikou se zabývá obor *Energy Management*. Jedná se o souhrn požadavků pro výrobu, spotřebu a distribuci elektrické a tepelné energie v rámci budovy. *Energy Management* se může provádět na více úrovních, tak jak je uvedeno v kapitole 2.1.2. Na automatizační úrovni je řídicí systém schopen regulovat autonomně. Na úrovni manažérské lze přepínat nebo upravovat režimy regulace a provozu podle aktuálních požadavků a podmínek.

Příkladem aplikace *Energy Managementu* tepla může být noční snížení pokojové teploty místností v kancelářských prostorách v zimním období, když jsou zaměstnanci nepřítomní. Další možností snížení nákladů je instalace výměníků tepla mezi odpadní a čistou vodou.

Mezi funkce *Energy Managementu* v oblasti kontroly spotřeby elektrické energie obvykle řadíme zapínání/vypínání spotřebičů podle časových úseků. Přizpůsobením provozní lhůty zařízení podle potřebného času dokážeme snížit náklady na elektřinu.

Při připojení objektu (obchodní centrum, továrna atd.) k elektrické síti uzavírá se smlouva mezi odběratelem (majitelem objektu) a dodavatelem (ČEZ, E.ON, RWE), která se pak každý rok prodlužuje. Smlouva určuje množství a cenu odebrané energie za rok a také odběrové maximum výkonu za určitý časový úsek. Za nedodržení podmínek smlouvy se stanovují vysoké pokuty. *Energy Management* v budově pomáhá sledovat spotřebu elektřiny a omezuje případně vzniklé špičky. Funkce eliminace špiček se zabývá prognózou průměrného odebíraného výkonu za krátký časový úsek (většinou 15 min). Pokud vypočtená hodnota je větší než smluvní, řídicí jednotka pošle povel na odpojení jednoho nebo několika spotřebičů podle jejich priority. V případě, že je odpojení spotřebiče nepřijatelné, lze uvažovat o využití záložního zdroje (viz funkce peak shaving).

V dalších podkapitolách bude kladen důraz primárně na management elektrické energie (viz podkapitola 2.2.3.3).

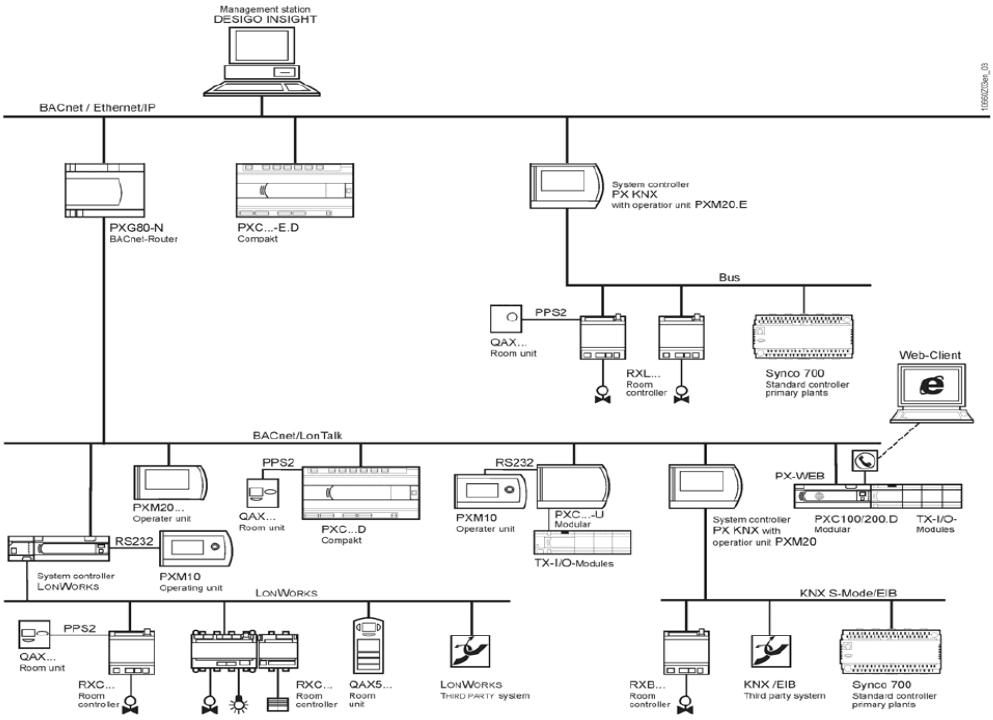
2.1.4 Národní Technická Knihovna jako příklad inteligentní budovy

Národní Technická Knihovna (NTK) byla postavena v Praze v roce 2009 a je výstižným příkladem moderní „inteligentní“ budovy. Dohromady má devět poschodí – šest nadzemních a tři podzemní, ve kterých jsou umístěné depozitáře, technické místnosti a parkoviště. V objektu jsou soustředěné fondy NTK, pobočky Městské knihovny pro Prahu 6, VŠCHT a také Ústřední knihovny ČVUT. Knihovna nabízí všem uchazečům zhruba hodinové komentované exkurze o historii knihovny, službách a použitých technických řešeních. O poslední bod jsem měl největší zájem, proto jsem se na exkurzi zaregistroval.

Obr. 1 Budova NTK v Praze. Zdroj (3)

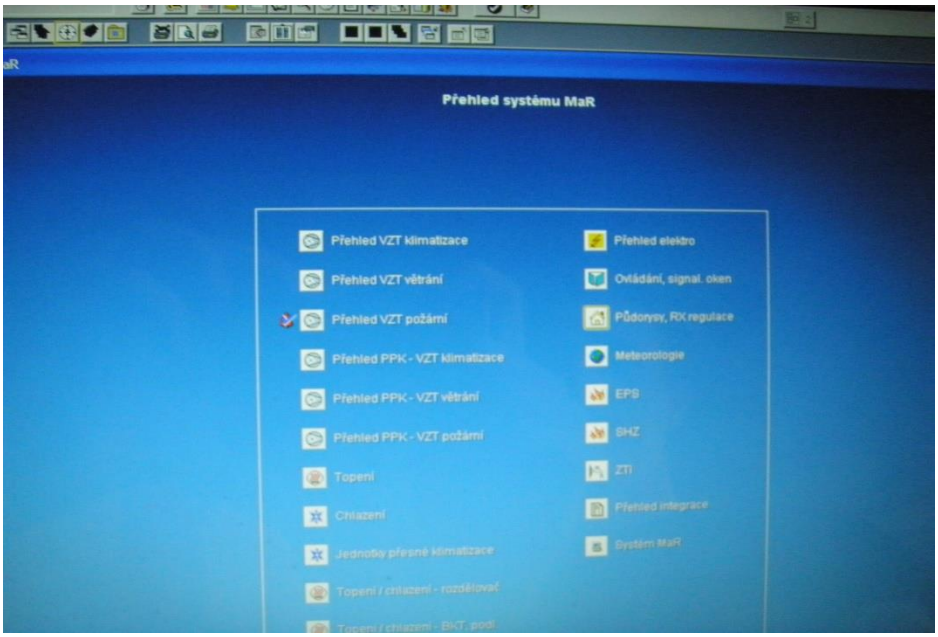
Jak jsem se dozvěděl na prohlídce od pracovníků, pro řídicí systém v budově NTK je použito řešení *DESIGO* od společnosti Siemens. Tento produkt je postaven na základě programovatelných kontrolérů a stanic dispečinku pro většinu aplikací ve sféře automatizace budov. *DESIGO* podporuje otevřené protokoly komunikace, což ulehčuje připojení rozmanitých zařízení na bázi otevřených rozhraní:

- *BACNet* – úroveň automatizace
- *LonWorks* a *KNX* – automatizace místností a integrace dodatečných systémů
- *M-bus, Modbus, OPC* atd. – univerzální integrace „third-party“ zařízení

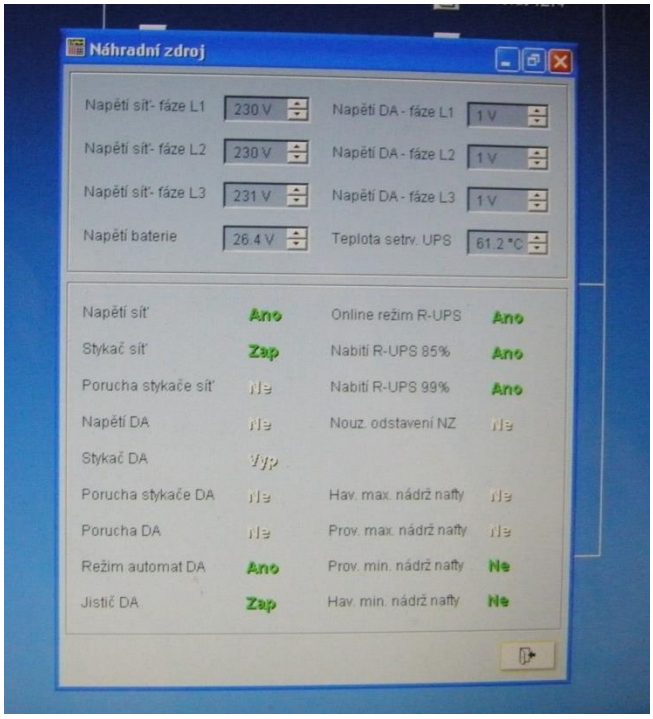


Obr.2 Topologie Siemens DESIGO. Zdroj: (2)

Ovládá se HVAC, elektrospotřebiče, EPS, meteorologie atd. Z bezpečnostních důvodů do ní nebyl zahrnut kamerový systém, přístup ke kterému má jen ostraha (příp. ostatní se souhlasem ředitele).



Obr.3 Systém MaR v NTK. Zdroj: vlastní



Obr.4 Okno měřených veličin systému napájení NTK. Zdroj: vlastní

Obsluha napájení budovy pouze monitoruje, protože regulace je přednastavena automaticky. V budově jsou k dispozici záložní zdroje pro servery a další elektrospotřebiče. Jedná se o dieselagregát Phoenix-Zeppelin o výkonu 400 kW s řídicím systémem vlastní výroby této firmy a rotační UPS se setrvačníkem, který může při výpadku trvajícím několik desítek vteřin vyrábět elektřinu, než se rozběhne samotný dieselagregát. Řídicí jednotka toho dieselagregátu musí být v automatickém režimu.

Obr.5 Rotační UPS Phoenix-Zeppelin 250i. Zdroj: (4)

Rotační UPS o výkonu 250 kVA (model *UPS 250i*) je rovněž dodána firmou Phoenix-Zeppelin. Podle technické specifikace kdyby došlo k výpadku napájení, dokáže udržet 100% zátěž (250 kVA) během 15 sekund, 25% zátěže zvládne 56 sekund. Když UPS je napájená a zátěž se zvyšuje, tak nějakou dobu může působit jako pomocný zdroj pro síť (viz tabulka).

Typ přetížení	**Velikost zátěže**
Stále	Až 105%
10 minut	125%
2 minuty	150%
30 vteřin	200%
10 milisekund	>200%

Tabulka 1: Zatěžovací charakteristika UPS 250i Zdroj: (4)

Zásobování Národní Technické Knihovny elektřinou v případě poklesu síťového výkonu realizováno následujícím způsobem. Elektřinou z gen-setu se napájí pouze nejdůležitější systémy: požárně bezpečnostní zařízení, chlazení a napájení serverovny, přičemž všechny mají stejnou prioritu zásobování. *DESIGO* je za těchto podmínek vypnutá. Jak jsem se dozvěděl, za celou dobu jak funguje NTK se to stalo pouze dvakrát.

Gen-set podléhá revizi každého půl roku, zátěžové zkoušky se konají jednou za rok. Avšak toto zařízení se spouští často jako ukázka na exkurzi.

Celý systém funguje na základě *TCP/IP* protokolu na úrovni řízení a protokolů *BACNet* a *M-bus* na úrovní automatizační. Sledují se například stav dieselagregátu, teplota v místnostech atd. Všechna monitorovací zařízení za výjimkou ovládačů pohonů požárních čerpadel jsou dodané firmou Siemens.

Obr.6 Požární čerpadla v NTK. Zdroj: vlastní

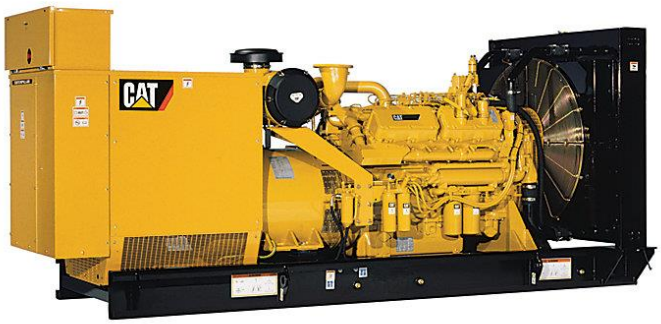
2.2 Řízení gen-setu

2.2.1 Gen-set

Jak již bylo zmíněno v předchozích podkapitolách, systémy řízení budov integrují monitoring a kontrolu podsystémů, které slouží ke komfortu uživatelů a provedení účelové činnosti. Mezi takové podsystémy řadíme HVAC, poplachový systém anebo osvětlení. Avšak každý z nich pro zajištění odpovídající funkčnosti potřebuje být napájen elektrickou energii. Proto systém řízení budovy by měl řešit otázku napájení elektřinou z vnitřních nebo vnějších zdrojů.

Výrobu elektřiny můžeme uskutečnit různými způsoby (spalování biomasy, jaderná energie, využití vodospádu u hydroelektráren). Například, při spalování nafty nebo jiného druhu látky, spalovací motor pohání generátor případně sadu generátorů, které vyrábějí elektrickou energii. Pro soustrojí spalovací motor – generátor se využívá termín **gen-set**, na který čtenář ještě v této práci narazí.

Kromě spalovacího motoru (naftový, benzinový, plynový) a generátoru (stejnosměrný, střídavý), které tvoří kostru zařízení, gen-setu se běžně doplňují nádrží na palivo, chladícím, mazacím, **vlastním měřícím a kontrolním systémem**.



Obr. 7. Caterpillar 3412C 900 kVA. Zdroj: (5)

Gen-sety se využívají jako zdroj elektrické energie v aplikacích, kde elektrická síť není přítomná nebo je dost „slabá“, čili nedovoluje přenést dostatečný výkon. Také můžou plnit funkce záložního zdroje pro případy výpadku sítě. Gen-sety nacházejí uplatnění například v dopravě (lokomotivy, lodní generátory), vojenské sféře (ponorky, protiraketová vojska) a

komunikaci. Vyrábějí se v mobilním a stacionárním provedení a širokém výkonovém rozmezí. Nejvýznamnějšími výrobci gen-setů jsou firmy Caterpillar, Cummins a FG Wilson.

Provoz gen-setu v budovách přináší kromě výhod i určité starosti. Musejí pravidelně procházet technickou kontrolou, mít dostatečné množství paliva a obsluhovat se zaškolenými pracovníky. Kromě toho, gen-set není jenom zdrojem elektřiny, ale ještě i hluku, což se většinou řeší tlumícím krytím. Také je potřeba zajistit odvod spalin motoru gen-setu ven z budovy.

Gen-sety se provozují v různých režimech. Základní rozdělení rozlišuje tzv. „ostrovní“ a „paralelní“ režim. V ostrovním režimu generátor nebo několik generátorů zásobují elektřinou zátěž samostatně bez sítě. V paralelním režimu předpokládáme připojení generátoru nebo několika generátorů na tvrdou síť. Paralelní provoz se sítí nebo použití více generátorů vyžaduje synchronizaci, což vede k požadavku pro řídicí jednotku gen-setu kontrolovat splnění určitých podmínek. Mezi tyto podmínky patří:

- stejná amplituda napětí generátorů a/nebo sítě
- stejná frekvence (měří se na výstupu z elektrického generátoru anebo počítá se snímáním otáček hřídele)
- stejný sled fází
- stejný fázový posuv

Řídicí jednotka rovněž monitoruje i klíčové veličiny spalovacího motoru gen-setu. Jsou to:

- množství paliva v tanku
- teplota oleje
- spotřeba oleje
- tlak oleje
- teplota chladicí kapaliny
- otáčky
- další parametry

V praxi se lze setkat se situací, když elektrická síť není dostatečně dimenzována. U takovýchto sítí mohou nastat výpadky (*blackout*). Tento případ je aktuální nejen pro rozvíjející se státy (Indie, Čína atd.) ale i pro státy EU, stačí si vzpomenout výpadek v Itálii v roce 2003.

Tehdy bez elektřiny zůstalo 56 milionů lidí. Nejelo metro, nefungovali semafory, výtahy, osvětlení.

Nicméně, provoz některých organizací vyžaduje trvale napájení, jinak následky můžou být fatální. Jsou to nemocnice, orgány státní správy, vojenské objekty, letiště atd. Proto jejich zásobování elektřinou je zajištěno jak z primárního zdroje energie (síť), tak i ze sekundárního. Sekundární systém obsahuje gen-set(y), UPS *(uninterruptible power supply*) anebo jejích kombinaci. Kombinace gen-setu a UPS je určena k spolehlivějšímu napájení objektu, protože UPS pokrývá zátěž, když síť je už „mrtvá“ a gen-set se ještě nerozběhl. Primární a sekundární systémy takových objektů mají nadřazenou řídicí jednotku, která dokáže obstarat případy výpadku sítě příp. jiných zdrojů a zajistit zabezpečení objektu elektrickou energií.

Funguje to následujícím způsobem. Kontrolér (řídicí jednotka) sleduje parametry síťového napětí jako frekvence, napětí fází, účiník. Pokud tyto parametry sítě opustí dovolené limity, kontrolér rozepne síťový stykač a spustí gen-set. Poté, co se rozběhne a napětí generátoru se ustálí v dovolených mezích, sepne stykač generátoru. Pokud jsou parametry sítě opět v pořádku a síť je prohlášena kontrolérem za „zdravou“, gen-set, který zároveň stále napájí zátěž, se synchronizuje se sítí. V tomto momentu je gen-set v paralelním režimu se sítí. Vzápětí kontrolér začne odlehčovat gen-set a síť si tedy přebírá postupně veškerou zátěž. Následně je gen-set odepnut od zátěže a po určité době vypnut.

2.2.2 ComAp – český výrobce řídicích systémů

Firma ComAp a.s. se specializuje na výrobě řídicích elektronických systémů pro energetiku a pohony. Nabízí souhrn hardwarových a softwarových prostředku k ovládání zdrojů elektřiny a spolehlivému zásobování objektů elektrickou energií. Mezi hlavní kategorie výrobků patří:

- ATS (Automatic Transfer Switch) kontroléry
- Řídicí systémy pro motorgenerátory
- Řídicí systémy pro lodní motory
- Řešení pro ovládání a ochranu duálních motorů
- Elektronické potenciometry

Firma byla založena v roce 1991 v Česku. Momentálně má sedm zahraničních poboček a zaměstnává přes 250 lidí. Díky široké distribuční síti vyváží do více než 105 zemi.

Obr. 8. Firemní logo. Zdroj: (6 str. 1)

Ve své práci jsem komunikoval s řídicí jednotkou *InteliSys*^{NTC} vestavenou do demonstračního kufru *StarterKit.* Popisu těchto zařízení jsou věnované dvě následující podkapitoly.

2.2.3 Řídicí jednotka InteliSys^{NTC}

2.2.3.1 Obecný popis

Kontrolér *InteliSys*^{NTC} patří mezi „premium“ výrobky firmy. Jedná se o řídicí jednotku gen-setu a přilehlých spínacích prvků. Řídicí jednotka je přeprogramována pro paralelní běh gen-setu se sítí nebo pro ostrovní režim. Je vhodná i pro aplikace s kogeneračními jednotkami. Kontrolér podporuje paralelní chod až 32 generátorů a ukládá do historie hodnoty jejích výkonu a případné chyby. Lze nastavit službu zasílaní emailů nebo SMS v případě, že gen-setu dochází palivo, uniká olej atd. Pomocí technologie *AirGate* od ComAp je možné se vzdáleně připojit ke kontroléru přes Internet i v případě když nemá přiřazenou statickou IP-adresu a sledovat množství dodané energie, spotřebu paliva, stav gen-setu a na základě daných informací generovat reporty podle časového rozmezí.

K zobrazení měřených veličin nabízejí se odpojitelné obrazovky *InteliVision 5*, *InteliVision 8* a *InteliVision 17* (dotyková).

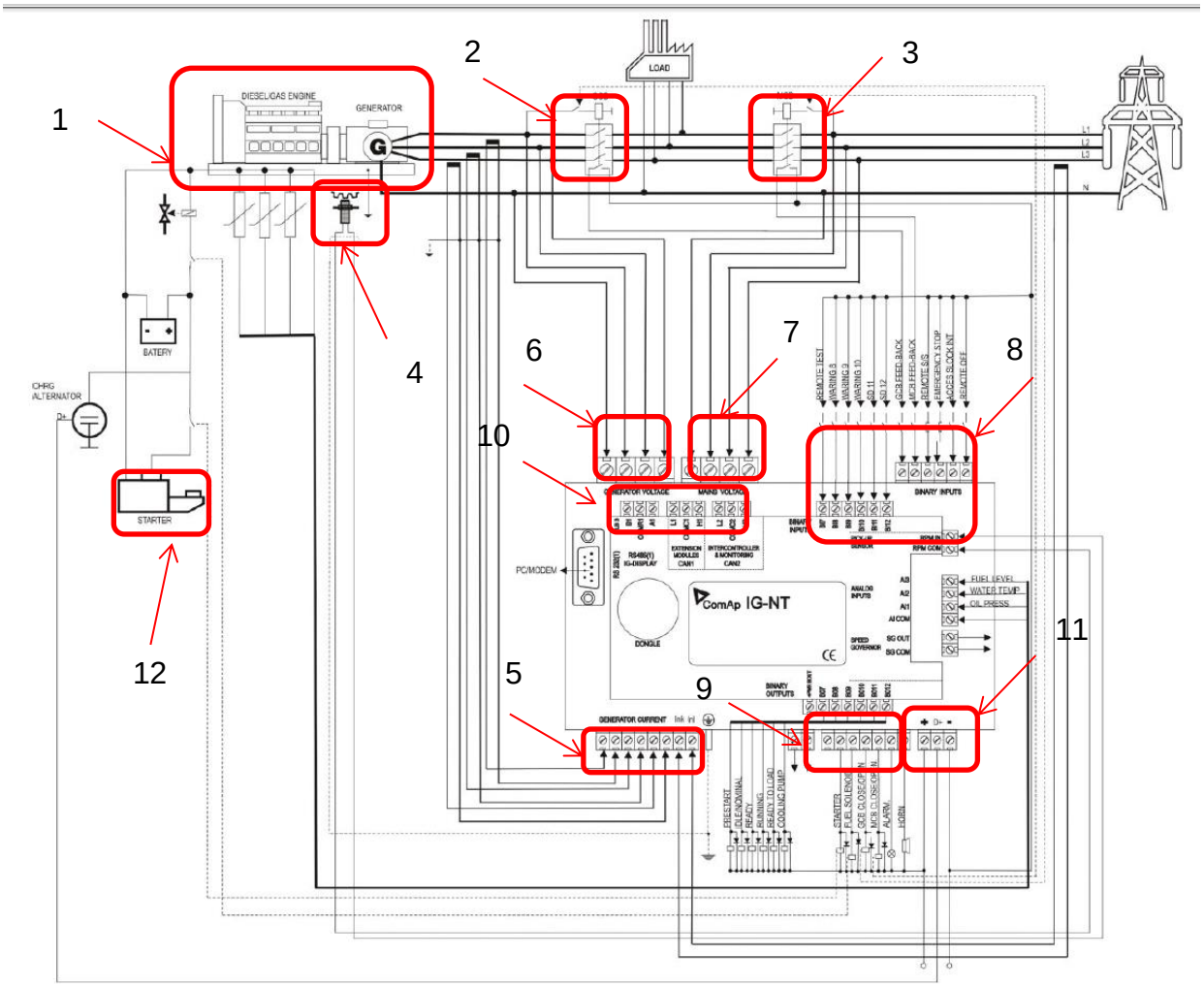
Obr.9 InteliSys^{NTC} Base Box. Zdroj: (7 str. 1)

U generátoru (sítě) *InteliSys*^{NTC} monitoruje a reguluje veličiny:

- napětí U V
- proud I A
- frekvence f Hz
- zdánlivý výkon S kVA

- činný výkon P kW
- jalový výkon Q kVAr
- účiník cosφ -

Z kontrolérů dané třídy a řady dalších (typy *InteliMains*^{NTC}, *InteliGen*^{NTC} atd.) a také doplňkových modulů včetně výrobků třetích stran lze postavit škálovatelný řídicí systém, odpovídající vysoké míře informativnosti. Příklad takového systému je uveden na obrázku Obr. 10.



Obr. 10. Schéma použití InteliSys^{NTC}. Zdroj: (7 str. 70)

Na obrázku jsou označené prvky:

1. Gen-set
2. Třífázový stykač generátoru (*Generator Circut Breaker, dále - GCB*)

3. Třífázový stykač sítě (*Mains Circut Breaker, dále - MCB*)
4. Indukční čidlo otáček
5. Měřící svorky transformátorů proudu jednotlivých fázi generátoru
6. Měřící svorky napětí generátoru
7. Měřící svorky napětí fází sítě
8. Binární vstupy ŘJ (hodnoty stavů stykačů, tlak oleje gen-setu, okamžité zastavení gen-setu atd.)
9. Binární výstupy ŘJ (množství paliva, zapalování, stav ventilátoru, chlazení atd.)
10. Komunikační rozhraní ŘJ pro doplňkové moduly/nadřazené systémy
11. Napájení ŘJ
12. Zapalovač motoru

Chování řídicí jednotky *InteliSys*^{NTC} závisí na její provozním režimu. Dostupné režimy (8 stránky 11-12):

1) OFF

a) výstupy *STARTER, GCB CLOSE/OPEN* and *FUEL SOLENOID* nejsou napájeny.
b) Není možné nastartovat gen-set
c) Není možné sepnout/rozepnout stykače MCB a GCB, avšak lze nastavit automatické chování MCB stykače při přepnutí do tohoto režimu
d) Pokud gen-set běží, do tohoto modu nelze se přepnout. Nejdřív gen-set se musí zastavit.

2) MAN

a) Při aktivaci binárního vstupu *START* kontrolér nastartuje gen-set
b) Při aktivaci binárního vstupu *STOP*
   i) Ostrovní režim: otevře stykač GCB, začne chladit gen-set, pak ho vypne
   ii) Paralelní režim: přehodí veškerou zátěž na síť, otevře GCB, začne chladit gen-set, pak ho vypne
   iii) Pokud gen-set není zatížen: začne ho chladit a pak vypne
   iv) Pokud gen-set se již chladí: okamžitě ho zastaví
c) Při aktivaci/deaktivaci binárního vstupu *GCB close/open*
d) Při překročení dovolených mezí napětí generátoru ŘJ nepovolí operací sepnutí/rozepnutí GCB

e) V případě synchronizace se síti kontrolér ověří, jestli síť je zdravá (výstup *MainsOK*) a stykač MCB je sepnut. Pak nastartuje gen-set a po splnění podmínek synchronizace připojí ho k silovému rozvodu. Další chování systému po synchronizaci lze definovat v nastavení setpointů karty **Process Control** *InteliSys*^{NTC}

f) Při požadavku na otevření stykače GCB v ostrovním režimu rozepne ho okamžitě. V paralelním režimu nejdřív odlehčí gen-set a potom rozepne stykač.

3) SEM

a) Při aktivaci binárního vstupu *START* se nastartuje gen-set, synchronizuje se sítí a poběží v paralelním režimu.

b) Při aktivaci binárního vstupu *STOP* řídicí jednotka měkce odlehčí gen-set, otevře GCB stykač, zapne chlazení motoru a potom vypne.

4) TEST

a) V tomto režimu gen-set se nastartuje automaticky

b) Pokud setpoint řídicí jednotky *Return To Mains=ENABLED*, pak vstupy *GCB ON/OFF , STOP, START* se ignorují

5) **AUT**

a) Tento režim provádí veškeré operace řízení toků elektřiny automaticky na základě přednastavení setpointů karty **Process Control** *InteliSys*^{NTC}

b) Pokud se stal výpadek:

i) Rozepne se MCB a nastartuje gen-set. Pokud síť se vrátí za náběhu gen-setu, řídicí jednotka sepne MCB a vypne gen-set

ii) Počká se, až generátor se roztočí na nominální otáčky. Pokud hodnota napětí generátoru se nachází v povolených mezích, řídicí jednotka zavře stykač GCB. Pokud ne, vypne gen-set a nahlásí chybu

c) Pokud síť se vrátila:

i) Provede se zpětná synchronizace s gen-setem, stykač MCB se zavře

ii) Řídicí jednotka odlehčí gen-set a rozepne GCB stykač

iii) Gen-set se vychladí a následně se zastaví

2.2.3.2 Komunikační rozhraní

InteliSys^{NTC} má několik možností realizace komunikačního rozhraní. Mezi ní řadíme:

1.) MODBUS – byl vyvinuty společností Modicon v roce 1979. Je založen na architektuře „master-slave“. Nadřízené a podřízená zařízení vysílají zprávy, které vypadají takto:

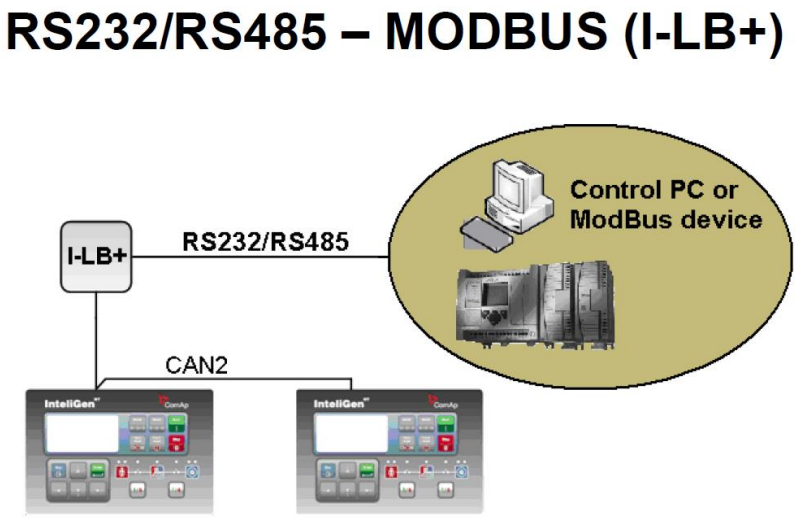
Adresa	*Kód funkce*	*Data*	*Kontrolní součet*

Adresa **–** unikátní číslo jednotky, pro niž je zpráva určena.

Kód funkce **–** popisuje příslušnou funkci (čtení, zápis, chyba atd.)

Data **–** tady je uložena informace o proměnných (jméno a hodnota) nebo příkazech vysílaných zařízením typu „master“

Kontrolní součet (CheckSum) **–** obsahuje důležité informace nutné pro kontrolu integrity zpráv a možných chyb komunikace.

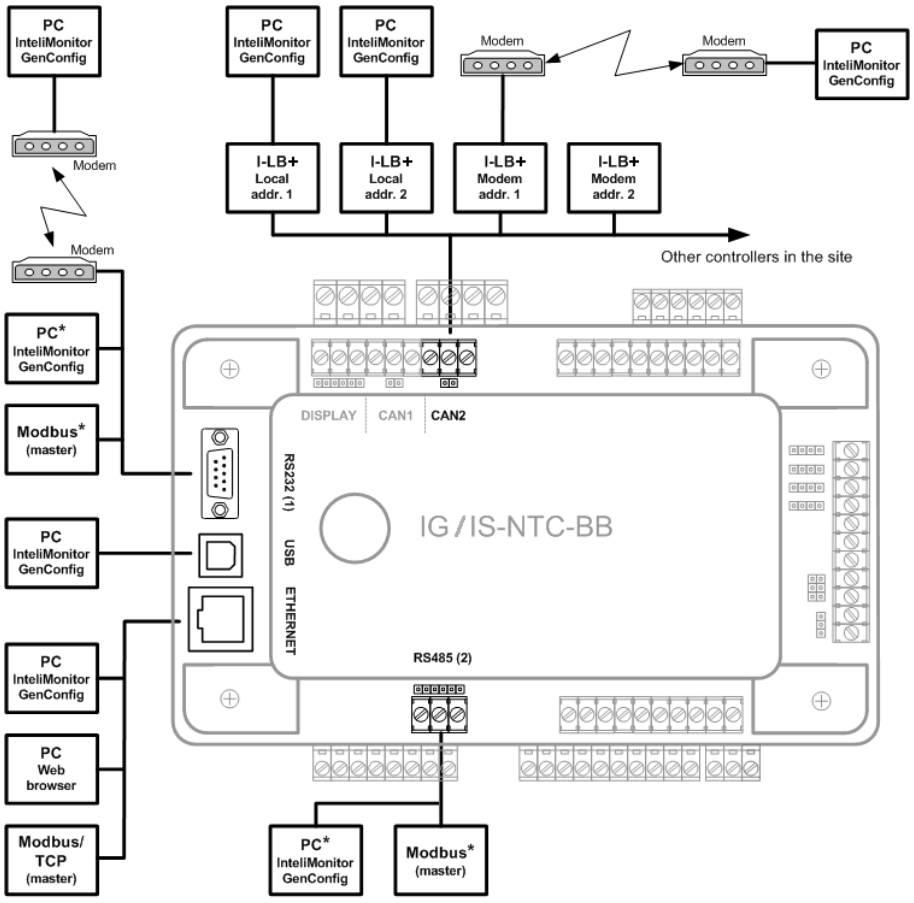


Obr. 11 Komunikace MODBUS s využitím I-LB+ modulu. Zdroj: (6 str. 29)

Pokud potřebujeme se připojit k několika kontrolérům zároveň, ale máme jednou Modbus sběrnici přes RS232/RS485, můžeme použit speciální komunikační modul *I-LB+*. Ten je přes CAN2 sběrnici propojen s kontroléry. Pro komunikaci s počítačem, například monitorovací PC nástroj, využijeme RS232. Jsou-li „slave“ kontroléry propojeny přes CAN sběrnici, modul převádí zprávy do RS232/RS485 a posílá „masteru“.

2.) CAN – *Controller Area Network*, sériová datová sběrnice, vyvinuta firmou Robert Bosch. Slouží především pro sjednocení do jediné sítě akčních členů a senzoru, tzn. patří k nízké úrovni (*fieldbus*). Data se posílají tzv. rámci. Užitečná informace v rámci se skládá z identifikátoru o délce 11 bit a 8-bitového datového pole. Na základě čísla identifikátoru se posuzuje priorita vysílání rámce po sběrnici.

InteliSys^{NTC} má dva CAN porty pro připojení periferií a rozšiřujících modulů, například pro další binární nebo analogové I/O (IS-AIN8 (*Analog Input Module*), IS-BIN8/16 (*Binary Input/Output Module)* a ostatní (viz stránky ComAp)). Sběrnice CAN dle standartu J1939 se využívá i pro komunikaci s řídicí jednotkou motorů ECU (*Electronic Control Unit*).



* Only one device can be connected to the respective communication port, device type is selectable by setpoint

Obr. 12 Komunikační rozhraní InteliSys^{NTC}. Zdroj: (6 str. 7)

3.) SNMP – *Simple Network Management Protocol*, internetové rozhraní, které umožňuje sběr dat pro správy a vyhodnocování. SNMP se hodně využívá v šítových zařízeních (servery, routery). Momentálně existují tři verze protokolu. I když první verze je poměrně zastaralá, mnozí výrobci ji podporují jako primární (9 str. 2). U poslední třetí verze se podporuje šifrování. Uvedeme základní pojmy:

Agent – monitorované zařízení (kontrolér). Představuje server.

Manager – typicky je to program na počítači, kterým spravujeme kontroléry. Představuje klienta. Manažéři často odkazují na tzv. *Network Management Stations (NMS)*. NMS je zodpovědná za příjem hlášení od agentů a odesílání příkazů od manažérů.

Management Information Base (MIB) – speciální tabulka, která charakterizuje datové objekty konkrétního agenta. Každý výrobce má svůj speciální klíč, pomocí kterého může vytvářet svoje vlastní MIB tabulky a připojovat se k celosvětovému MIB stromu. U *InteliSys*^{NTC} soubor s MIB tabulkou lze vygenerovat přes ovládací nástroj *InteliMonitor*, který je krátce popsán v podkapitole 2.2.3.4.

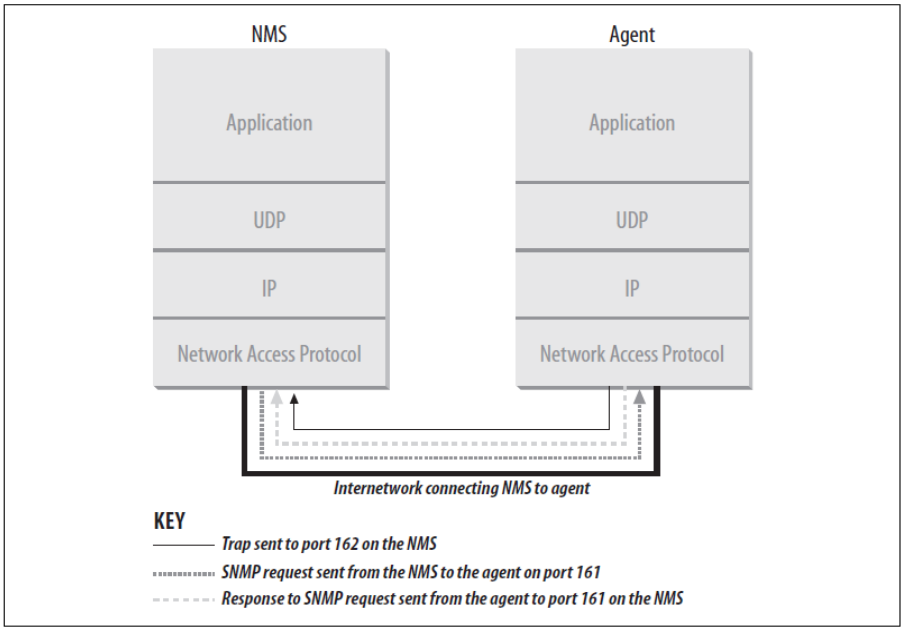
Object Identifier (OID) – identifikátor datového objektu.

Pro operace s daty se nejvíce využívá příkazů:

- GET – čtení datového objektu s určitým OID. Tento příkaz posílá manažér agentovi a agent následně odpovídá.
- SET – zápis datového objektu s určitým OID. Tento příkaz posílá manažér agentovi a agent následně odpovídá.
- TRAP – asynchronní notifikace manažerovi, která obsahuje systémový čas, OID objektu a případně vybrané datové objekty (*bindings*). Tento příkaz se odešle agentem, pokud se stane určitá událost (porucha) (10). Například, pokud router zjistí přerušení připojení k Internetu, pošle *trap* nadřazenému počítači.

SNMP používá UDP port číslo 161 pro příkazy GET a SET, zatímco TRAP funguje na portu 162. Každé zařízení, které implementuje SNMP musí mít tyto porty nastavené pro komunikaci jako výchozí. Avšak u některých výrobců zařízení-agentů je povoleno měnit výchozí konfiguraci. Použití UDP místo TCP přináší své výhody a nevýhody. Nevýhodou je nespolehlivost v případě, když agent při poruše posílá trap do NMS. Pokud ten trap se ztratí, NMS se nikdy nedozví, že tato porucha nastala. Naopak, výhodou v porovnání s TCP je menší zatížení komunikační sítě.

Na obrázku 13 je zobrazen způsob SNMP komunikace. Pokud buď NMS nebo agent chtějí vykonat SNMP příkaz, v protokolovém stacku probíhají následující události:



Obr. 13 Komunikační souprava v SNMP. Zdroj: (9 str. 20)

Aplikační vrstva (Application)

Zaprvé, aktuální SNMP objekt (NMS nebo agent) rozhoduje o tom, jaký bude postup při vyplnění operace (GET, SET, TRAP). Aplikační pak vrstva poskytuje informace, o stavu cílového portu v zařízení-manažéru nebo agentu.

UDP

Další vrstva, UDP, dovoluje dvojím zařízením „mluvit“ mezi sebou. Hlavička UDP zprávy obsahuje informace o konečném portu zařízení, kam se posílá příkaz. Číslo portu může být buď 161 (GET, SET) nebo 162 (TRAP).

IP

IP vrstva zkouší odeslat SNMP paket do cíle, specifikovaného jeho IP-adresou.

Media Access Control (MAC)

Konečná událost, která by měla proběhnout, aby SNMP paket byl doručen správně přes fyzickou síť. Tato vrstva se skládá z hardwaru a ovládačů zařízení, které obstarají přenos dat (síťové karta). MAC je rovněž zodpovědná za příjem paketů z fyzické sítě a jejích posílání nahoru do aplikační vrstvy SNMP.

Protokol SNMP jsem vzhledem k jeho jednoduchosti vybral pro komunikaci mezi aplikaci na počítači a kontrolérem. Použitá verze je SNMP v1.0.

2.2.3.3 Vybrané funkce

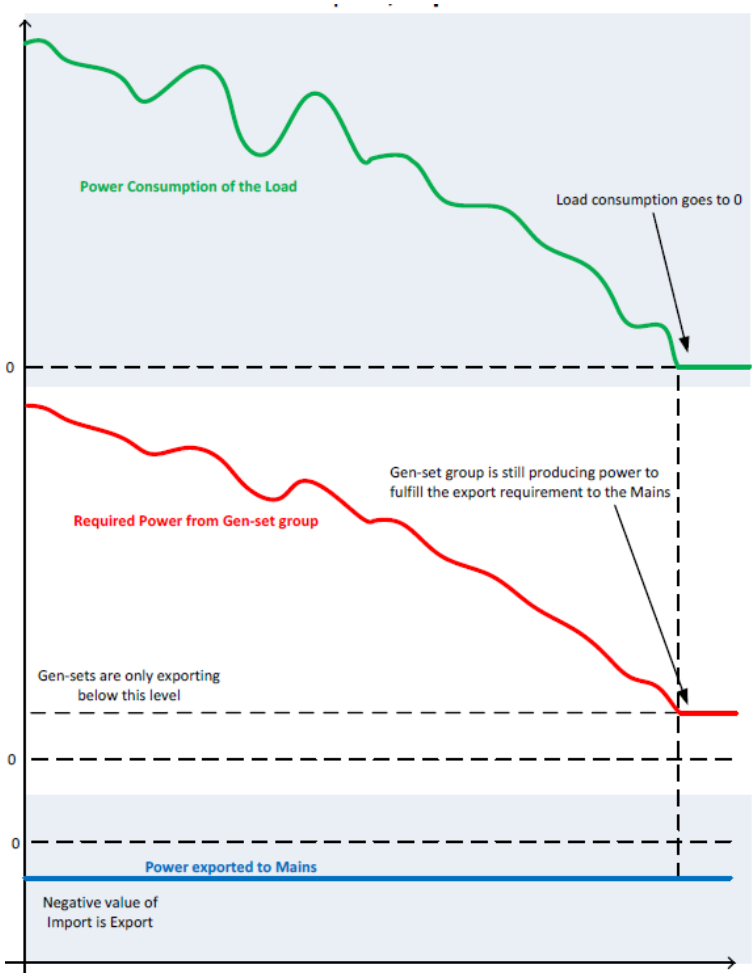
2.2.3.3.1 AMF - Auto Mains Failure

Funkce AMF neboli záskok je uplatněna u instalací, kde je nutné trvalé napájení. Při výpadku tato funkce zajistí nastartování gen-setu a napájení zátěže gen-setem. Jestliže si klient přeje bezvýpadkový provoz, systém doplní o UPS zdroj, který bude nepájet zátěž než se gen-set nebo gen-sety nastartují. Po obnovení sítě se gen-set nebo gen-sety se synchronizují se sítí. Zátěž je následně postupně převzata sítí a gen-sety jsou tím odlehčeny. Přechod napájení z gen-setu na sítí je tedy bez výpadku. Funkce AMF je realizována buď jedním gen-setem a kontrolérem *InteliGen*^{NTC}/*InteliSys*^{NTC} s aplikací „1 gen-set, 1 síť“. Kontrolér monitoruje parametry sítě a v případě výpadku vyšle příkaz k nastartování gen-setu.

2.2.3.3.2 Load Control PtM (System BaseLoad, Import/Export, T-BY-PWR)

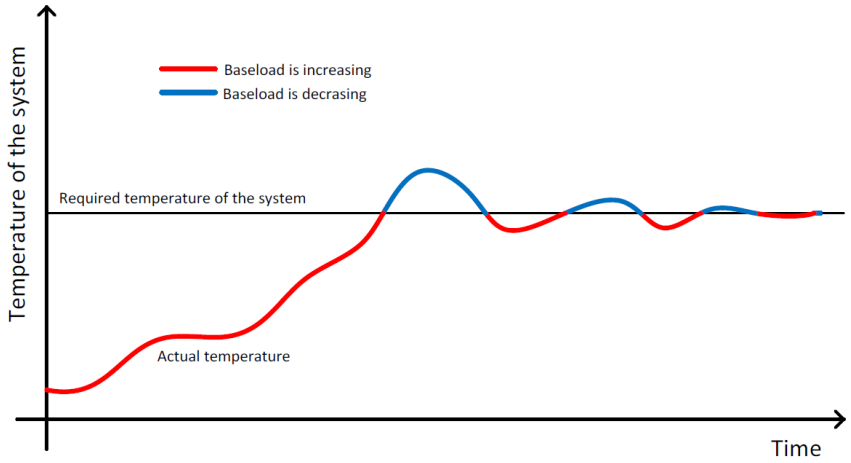
Funkce *Load Control PtM* (parallel to mains) se uplatňuje, když gen-sety pracují v paralelním režimu se sítí. Pomocí této funkce určujeme, na jaký výkon budou kontroléry regulovat. Následující volby jsou možné:

- *System Baseload*. Je-li tato volba vybrána, gen-sety dodávají výkon, který uživatel nastaví parametrem. Rozdíl vůči zátěži je importován nebo exportován z nebo do sítě.
- *Import/Export*. Je-li tato volba vybrána, gen-sety udržují konstantní importovaný nebo exportovaný výkon z nebo do sítě i při změně zátěže. Na obrázku 14 jsou uvedeny křivky znázorňující zátěž, výkon gen-setů a exportovaný výkon do sítě.



Obr. 14 Křivky znázorňující zátěž (zelená), výkon gen-setů (červená) a exportovány výkon do sítě. Zdroj: (11)

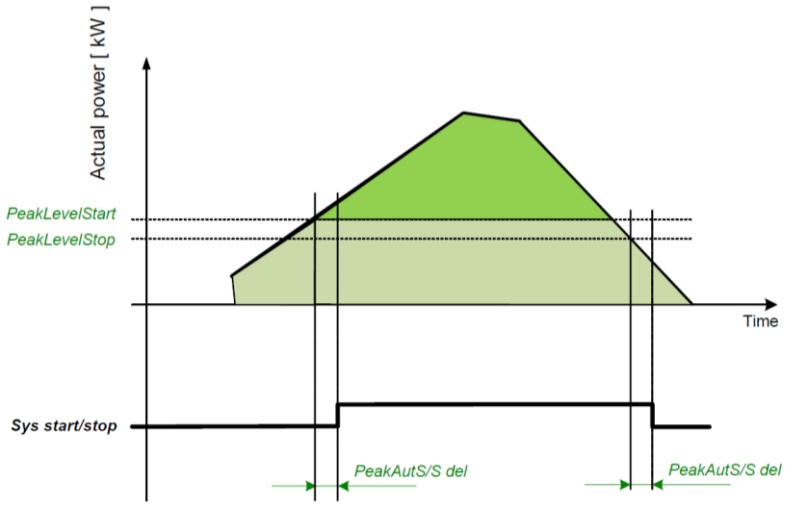
- *T-BY-PWR*. Tato volba je používána tehdy, chceme primárně ohřívat vodu a druhotným produktem je elektřina. Na obrázku 15 je uveden příklad takové regulace.



Obr. 15 Regulace T-BY-PWR. Zdroj: (11)

2.2.3.3.3 Peak Shaving

Funkce *Peak Shaving* se uplatňuje v například případech, kdy je síť dimenzována na určitý odebíraný výkon. V případě jeho překročení kontrolér nastartuje gen-sety a špičku vykryjí. Na obrázku 16 je funkce znázorněna. Kritický výkon je nastaven parametrem *PeakLevelStart*. Vypnutí gen-setů nastane při poklesu odebíraného výkonu pod hodnotu nastavenou parametrem *PeakLevelStop*. Hodnota tohoto parametru je menší než *PeakLevelStart*. Touto hysterezí zabráníme častým startům gen-setů při malé a časté změně zátěže.



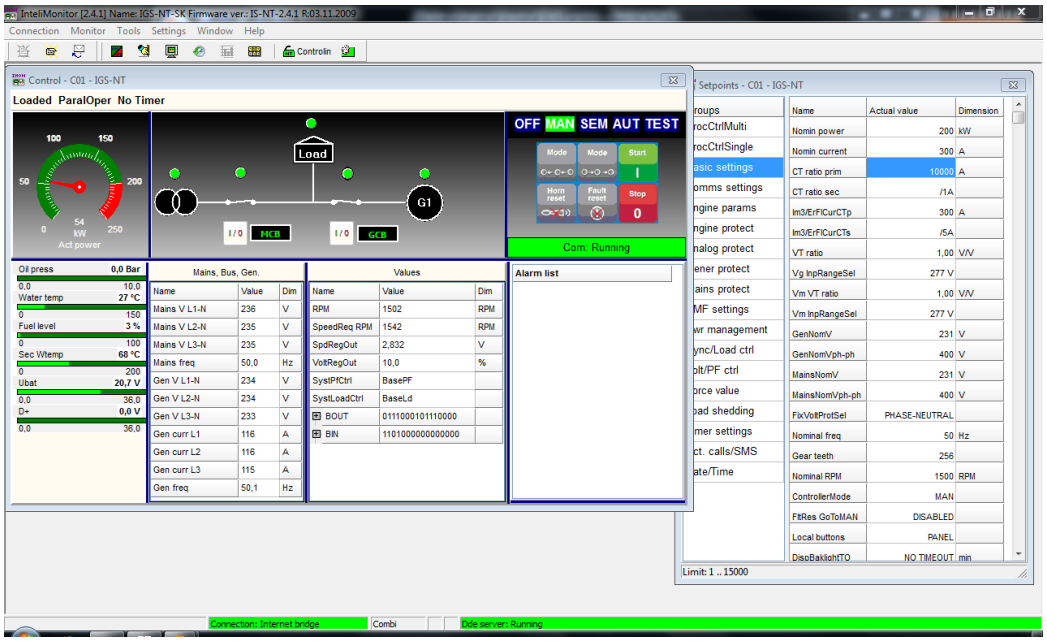
Obr. 16 Funkce Peak shaving. Zdroj: (11)

2.2.3.3.4 Power Management

Funkce *Power Management* zajišťuje, aby byl vždy k dispozici dostatečný výkon. Tato funkce hlídá výkonovou rezervu, která je označena *Load reserve*. Tato rezerva je rozdílem mezi skutečným odebíraným výkonem a součtem nominálních výkonů běžících gen-setů, které mají aktivní funkci *Power Management*. Jakmile tato rezerva poklesne pod hodnotu *LoadResStrt*, nastartuje další gen-set, který se synchronizuje a připojí se na silovou sběrnici. Naopak pro jeho vypnutí je potřeba, aby rezerva stoupla nad hodnotu danou parametrem *LoadResStop*, která je vyšší než *LoadResStrt*. Touto hysterezí je opět omezeno časté startování gen-setů.

2.2.3.4 Softwarový nástroj InteliMonitor

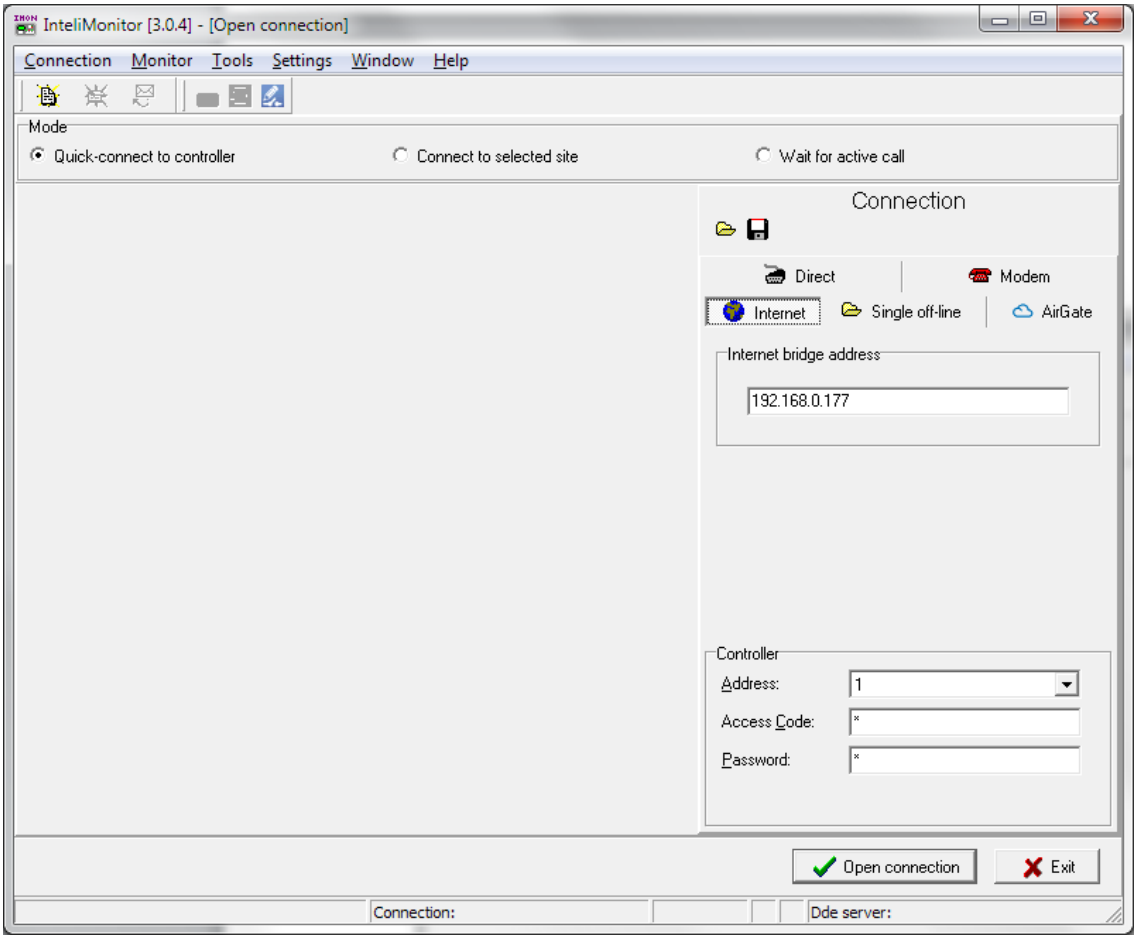
V předchozích podkapitolách jsme rozebrali co to je *StarterKit* a kontrolér sady *InteliSys*^{NTC}, který je jeho součásti. V podkapitole „Vybrané funkce“ bylo popsáno, jak dokáže kontrolér na základě svých vnitřních hodnot (tzv. „setpointů“) měnit chování energosystému. Jedním z možných způsobů je nastavení přes obrazovku *InteliVision 8* přímo v simulátoru. Avšak máme k dispozici i druhou variantu. Změny vnitřních stavů, pracovního režimu, monitorování analogových diskrétních veličin a řadu dalších úkolu lze provést přes ovládací nástroj *InteliMonitor*, který lze stáhnout po registraci zde.



Obr. 17 Nástroj InteliMonitor. Zdroj: vlastní

InteliMonitor je kompatibilní s Windows 2000/XP/Vista/Win7/Win8 programový prostředek, který poskytuje možnosti:

- Online sledování stavu kontroléru nebo jejích sady (*site*)
- Vytváření SCADA diagramů
- Funkce PLC monitor (návrh žebříčkových PLC diagramů v softwaru GenConfig)
- Přehled všech měřených a vypočtených hodnot
- Přehled historie změn kontroléru
- Změna parametrů řídicí jednotky (*setpoints*)
- Příjem hlášek v pasivním režimu

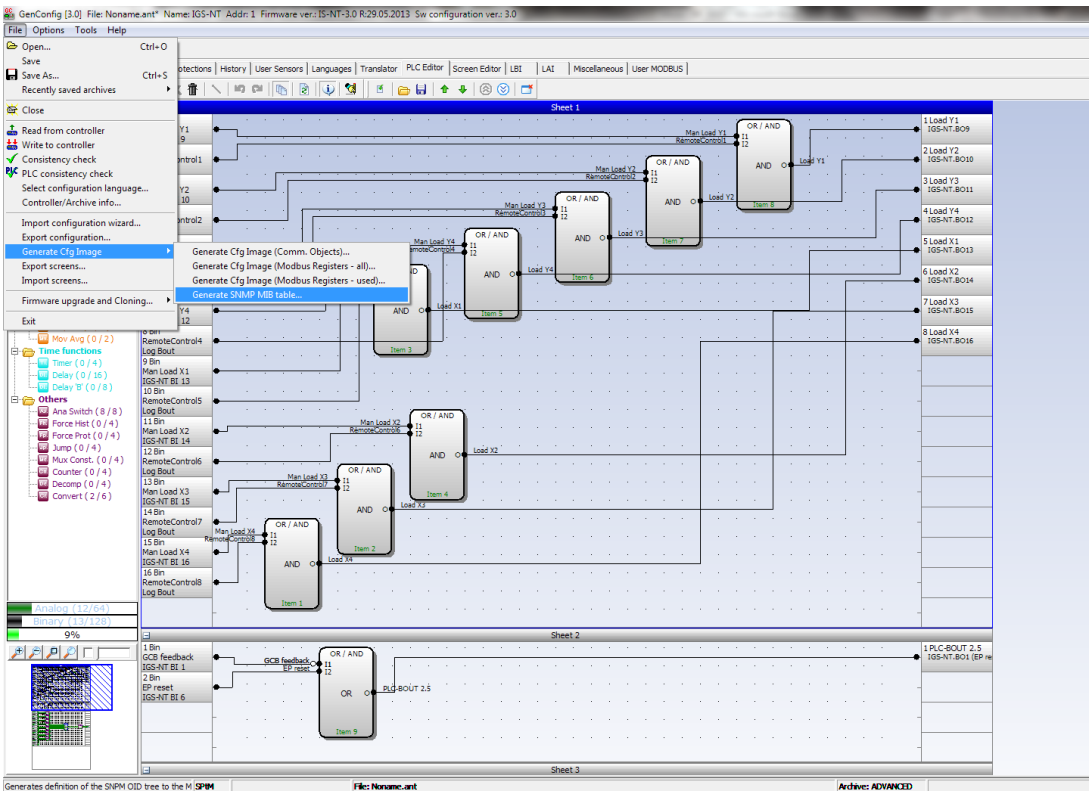


Obr. 18 Varianty připojení k zařízení v InteliMonitor. Zdroj: vlastní

InteliMonitor podporuje připojení ke kontroléru nebo sadě kontrolérů přes:

- Kabel (USB-A, USB-B). Na klientském počítači se vytvoří virtuální COM port.
- Vzdáleně pomocí modemu, Internetu nebo služby AirGate (cloudové řešení, které sleduje stav kontroléru a umožňuje ovládání přes prohlížeč)
- Off-line na základě předem uložených dat. To znamená, že pokud máme uloženy konfigurační archiv, tak ho můžeme změnit lokálně a novou verzi nahrát do kontroléru (12)

Po připojení pro změnu setpointů se nejdřív musíme přihlásit. Pak v okně „Setpoints“ dokážeme měnit jejích hodnoty. Pro pokročilejší monitoring spustíme nástroj *GenConfig* (Tools→GenConfig). Nástroj umožňuje konfigurace PLC logiky kontroléru, výhledu obrazovky *InteliVision 8,* volbu řídící jednotky dieselagregátu, který pohání generátor atd.



Obr. 19 Nástroj GenConfig s otevřenou kartou „PLC Editor“. Zdroj: vlastní

K navázání spojení mezi počítačem s aplikací, popsané v kapitole 4, a simulátorem přes SNMP rozhraní potřebujeme mít k dispozici soubor s MIB tabulkou (viz podkapitola 2.2.3.2). Ten vygenerujeme přes program GenConfig (File→ Generate Cfg Image→ Generate SNMP MIB table) jako na obrázku 19. Aplikace pak použije tuto tabulku k určení hodnot pro čtení/zápis.

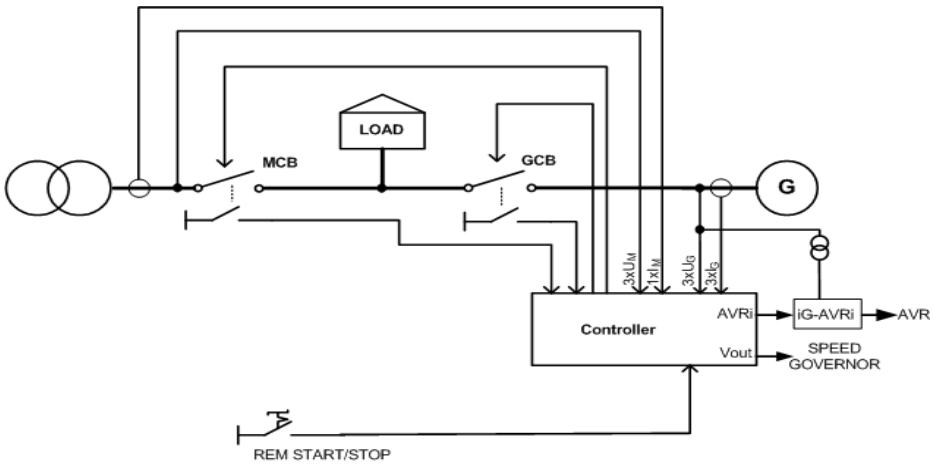
2.3 Simulační kufr Starter Kit

Jak bylo zmíněno v podkapitole Úvod, systém řízení budov se skládá ze softwarové a hardwarové časti. V daném projektu aplikace představuje programovou část. Co se tyká hardwaru, tak vzhledem k nemožnosti testování ve skutečné budově ho nahradil simulátor. Původně se předpokládalo, že bude použita pokročilejší verze simulátoru inteligentní budovy *Multi Kit* od společnosti ComAp. Bohužel během konzultací s odborníkem projekčního oddělení firmy se zjistilo, že kvůli poměrně dlouhému procesu projektování a výroby simulátoru, a také vysoké ceně, tohle řešení se pro daný projekt nehodí. Proto byla vybrána „zjednodušená verze“ pod názvem *Starter Kit*.

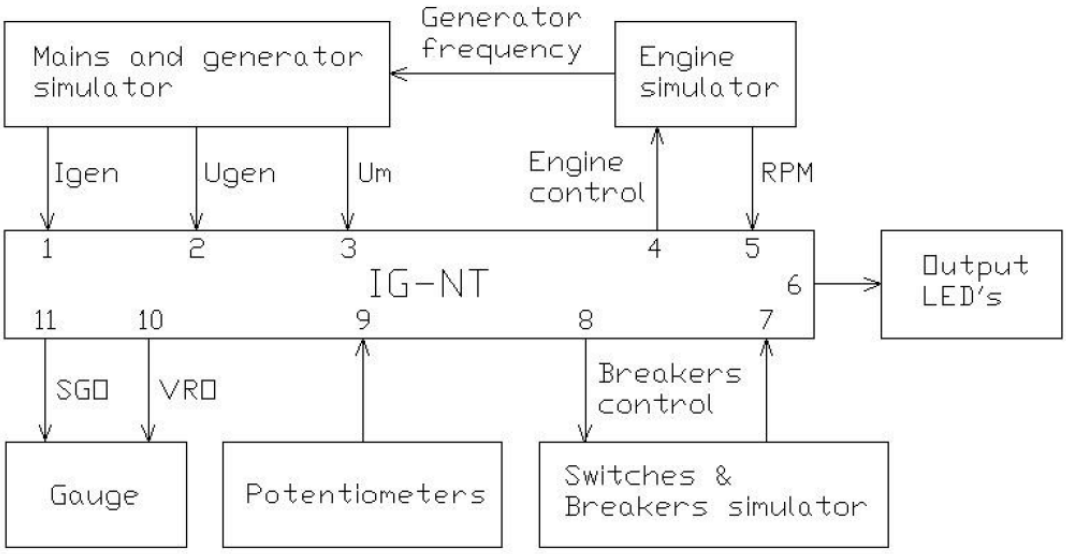
Starter Kit je simulátor operací gen-setu a sítě druhé generace, sloužící k testování funkcí kontroléru nebo k prezentačním účelům. V této práci představuje systém napájení inteligentní budovy tvořený tvrdou sítí a záložním zdrojem. Na simulátoru můžeme vyzkoušet:

- Výpadek sítě a obnovení provozu (funkce *Auto Mains Failure*)
- Zapínání/vypínání generátoru vzdáleně přes komunikační sběrnici
- Synchronizaci generátoru a sítě
- Připojení a odpojení zátěže
- Další funkce zmíněné v podkapitole 2.2.3.3

Simulátor má několik důležitých omezení. Vestavený *InteliSys*^{NTC} kontrolér neřídí otáčky generátoru, musejí se nastavit ručně pomocí potenciometru RPM (viz obrázek 20). Nominální hodnota je 1500 ot/min. Ve Starter Kitu můžeme nasimulovat pouze činný výkon, regulace účiníku není tedy možná. Napětí a proud generátoru rovněž nastavíme potenciometry U_{gen} a I_{gen}. V závislosti na připojené zátěži je ovládán elektronický potenciometr, který ovládá zdroj proudu. Ten je připojen k měřicím svorkám proudu generátoru. Síť není také řízená automaticky, proto pro řízení napětí použijeme potenciometr U_m. Měření síťového proudu není podporováno. Pro simulaci výpadku (*blackout*) slouží přepínač U_m.



Obr. 20 Schéma SPtM aplikace. Zdroj: (8 str. 10)

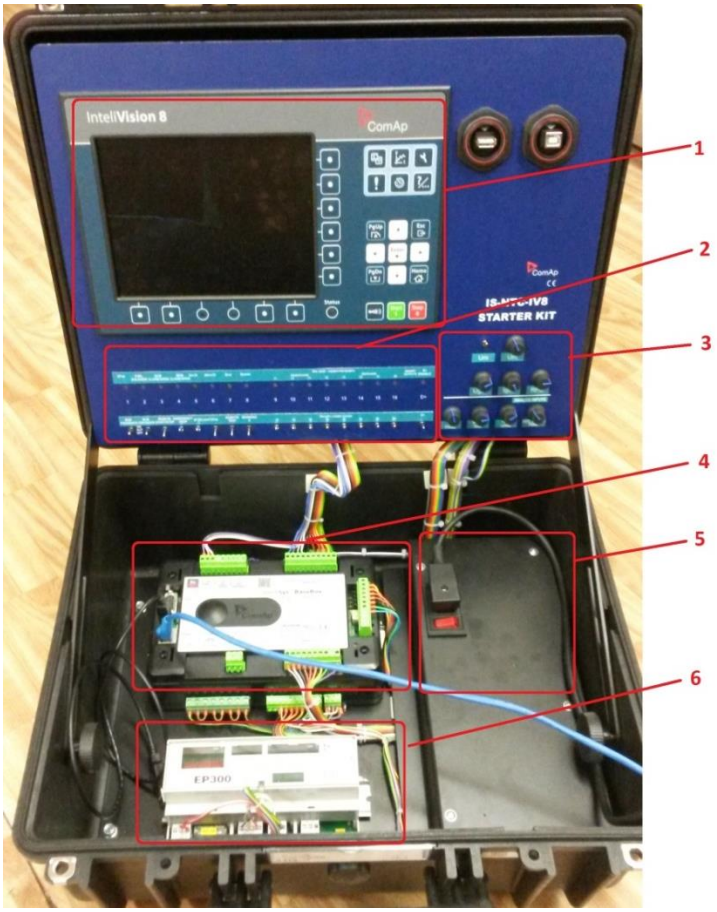


Obr. 21. Funkční schéma Starter Kitu. Zdroj: (13 str. 6)

1. *Proudové svorky generátoru*
2. *Napěťové svorky generátoru*
3. *Napěťové svorky sítě*
4. *Binární výstupy 1 (Starter) a 2 (Fuel Solenoid)*
5. *Binární vstup otáček gen-setu*
6. *Výstupní svorky*
7. *Vstupní svorky*
8. *Binární vstupy stavů GCB a MCB*
9. *Svorky pro analogové vstupy*
10. *Udržování rychlosti otáčení gen-setu (nepoužívá se)*
11. *Automatické řízení napětí generátoru (nepoužívá se)*

Jak již bylo zmíněno, simulátor představuje napájení budovy typu SPtM (*Single Parallel to Mains*). SPtM obsahuje následující vlastnosti:

- Široce nastavitelný rozsah ochran gen-setu
- **Funkce provozu generátoru spolu se sítí**
- **Ostrovní režim (zátěž napájí jenom generátor, síť je vypnuta)**
- Dvojice stykačů – GCB (*Generator Circuit Breaker*) a MCB (*Mains Circuit Breaker*) s možností jak automatické, tak i ruční synchronizace
- Měkký náběh generátoru a zátěží

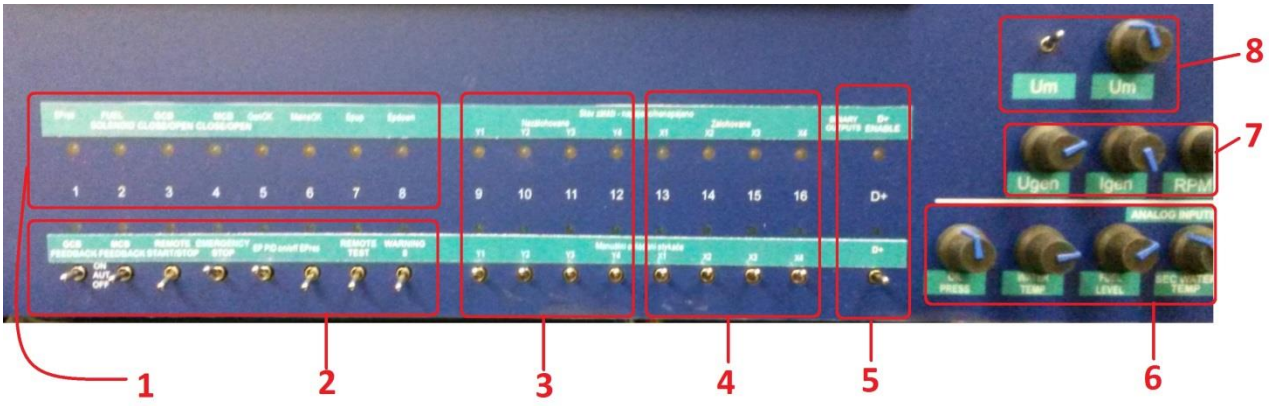


Obr. 22 Starter Kit. Zdroj: vlastní

Na obrázku Obr. 22 jsou označené prvky tvořící simulátor:

1. Obrazovka *InteliVision 8*. Pomocí tlačítek na panelu lze prohlížet historii změn a hlášky, nastartovat/zastavit gen-set, měnit setpointy kontroléru.
2. Přístrojový panel. Přes ně ovládáme stykače jednotlivých zátěží, sítě a generátoru, povolujeme vzdálenou kontrolu generátoru a řídíme elektronický potenciometr 6. Podrobnější popis je na obrázku 23.

3. Panel potenciometrů pro ovládání elektrických veličin generátoru a sítě, regulaci neelektrických veličin (otáčky, teplotu, množství paliva a oleje) generátoru. Podrobnější popis je na obrázku 23.
4. Kontrolér *InteliSys*^{NTC}.
5. Blok napájení.
6. Elektronický potenciometr, který obsahuje PID regulátor. Slouží k postupnému zatížení gen-setu v ostrovním nebo paralelním režimu při připojení nebo odpojení zátěže.



Obr. 23 Panel přepínačů a potenciometrů Starter Kit. Zdroj: vlastní

Obrázek Obr. 23 znázorňuje panel, přes který se provádí ruční regulaci simulátoru. LED diody nahoře patří k binárním výstupům, zatímco ty zelené dole jsou binárními vstupy kontroléru. Na obrázku jsou zobrazeny:

1) Nakonfigurované binární výstupy:
   (a) EP_{res} – resetování elektronického potenciometru
   (b) *Fuel Solenoid* – ukazuje stav palivové nádrže.
   (c) *GCB close/open* – pokud v log. 1, tak stykač generátoru je zavřen
   (d) *MCB close/open* – pokud v log. 1, tak stykač sítě je zavřen
   (e) *GenOk* – ukazuje, jestli generátor běží a jeho parametry jsou v pořádku.
   (f) *MainsOk* – ukazuje, jestli síť funguje a její parametry jsou v pořádku.
   (g) EP_{up} – zvýšení zatížení generátoru elektronickým potenciometrem.
   (h) EP_{down} – snížení zatížení generátoru elektronickým potenciometrem.

2) Nakonfigurované binární vstupy:
   (a) *GCB Feedback* – ovládá stykač generátoru. Může se nacházet v polohách ON, OFF, AUTO. V režimu AUTO stykač reguluje kontrolér. Výchozí stav je AUTO.
   (b) *MCB Feedback* – ovládá stykač sítě. Může se nacházet v polohách ON, OFF, AUTO. V režimu AUTO stykač reguluje kontrolér. Výchozí stav je AUTO.
   (c) *Remote Start/Stop* – nastavuje možnost dálkového ovládání generátoru. Výchozí stav je OFF.
   (d) *Emergency Stop* – okamžitě zastaví gen-set a vyvolá hlášku. Výchozí stav je ON.
   (e) E_{pid} *on/off* – vypíná nebo zapíná PID regulaci elektronický potenciometr se zachováním jeho původní hodnoty. Výchozí stav je ON.
   (f) EP_{res} – resetuje potenciometru a vynuluje jeho hodnotu. Výchozí stav je OFF.
   (g) *Remote Test* – povoluje vzdálené provedení periodických testů funkčnosti gen-setu. Výchozí stav je OFF.
   (h) *Warning* – aktivace binárního vstupu vede k vyhlášení alarmu typu „Warning“.
3) Nejištěné zátěže. Dolní přepínače jednotlivých zátěží tvoří jeden ze dvou stykačů. Druhý je virtuální a zapíná se programově. Pokud jsou sepnuty oba stykače, zátěž je připojena (svítí žlutá LED nahoře). Výchozí stav je ON.
4) Jištěné zátěže. Dolní přepínače jednotlivých zátěží tvoří jeden ze dvou stykačů. Druhý je virtuální a zapíná se programově. Pokud jsou sepnuty oba stykače, zátěž je připojena (svítí žlutá LED nahoře). Výchozí stav je ON.
5) D+ – přídavná detekce běhu gen-setu. V této úloze nebudeme tuto přídavnou funkci využívat. Výchozí stav je tedy OFF.
6) Ovládání neelektrických veličin gen-setu.
7) Ovládání elektrických veličin gen-setu.
8) Ovládání elektrických veličin sítě.

Simulátor *Starter Kit* bude propojen přes Ethernet kabel s počítačem, na kterém poběží aplikace, popsaná v kapitole 2.3. Vazba mezi simulátorem a programem je oboustranná, tzn. všechny děje provedené se *Starter Kit* se zobrazí v aplikace i naopak, uživatel přes aplikační uživatelské rozhraní dokáže měnit chování simulátoru.

3. Praktická část

3.1 Aplikace

3.1.1 Jazyk a technologie

Monitorovací nástroj, který je výsledkem této práce, byl mnou napsán v jazyce C#. Je to objektově orientovaný jazyk programování, vyvíjený společností Microsoft od roku 1998. Je založen na jazycích Java a C++. C# lze využít k tvorbě databázových aplikací, webových aplikací a stránek, webových služeb, aplikací ve Windows, softwaru pro mobilní zařízení. Přičemž C# může fungovat nejenom s operačním systémem Windows. Existuje platforma Mono, která běží na počítačích s Linux, Mac OS X, FreeBSD a Solaris.

Některé vlastnosti jazyka (14):

- V C# neexistuje vícenásobná dědičnost - to znamená, že každá třída může být potomkem pouze jedné třídy. Toto rozhodnutí bylo přijato, aby se předešlo komplikacím a přílišné složitosti, která je spojena s vícenásobnou dědičností. Třída může implementovat libovolný počet rozhraní.
- Neexistují žádné globální proměnné a metody. Všechny funkce a metody musí být deklarovány uvnitř tříd. Náhradou za ně jsou statické metody a proměnné veřejných tříd.
- V Objektově orientovaném programování se z důvodu dodržení principu zapouzdření často používá vzor, kdy k datovým atributům třídy lze zvenčí přistupovat pouze nepřímo a to pomocí dvou metod get (accessor) a set (mutator). V C# lze místo toho definovat tzv. Property, která zvenčí stále funguje jako datový atribut, ale uvnitř Property si můžeme definovat get a set metody. Výhodou je jednodušší práce s datovým atributem při zachování principu zapouzdření.
- C# je typově bezpečnější než C++. Jediné výchozí implicitní konverze jsou takové, které jsou považovány za bezpečné jako rozšiřování Integerů (např. z 32 bitového na 64 bitový) nebo konverze z odvozeného typu na typ rodičovský. Neexistuje implicitní konverze z typu Integer na Boolean, ani mezi výčtovým typem enum a typem Integer.
- C# neobsahuje a ani nepotřebuje dopřednou deklaraci - není důležité pořadí deklarace metod.

- Jazyk C# je „case sensitive“ - to znamená, že rozlišuje mezi velkými a malými písmeny. Identifikátory "hodnota" a "Hodnota" tedy nejsou na rozdíl od VB.NET ekvivalentní.

C# je tvrdě spojen s platformou .NET. Je to soubor technologií pro tvorbu softwarových produktů.

Součástí .NET jsou:

- WCF – implementuje webové a komunikační služby.
- ASP.NET – technologie pro vývoj webových stránek.
- WPF a WinForms – technologie pro vytváření grafického uživatelského rozhraní pro desktopové aplikace.
- LINQ – technologie, poskytující přístup k datům z databází nebo XML souborů a umožňující objektovou reprezentaci těchto dat.

Existuje několik vývojových prostředí pro C#: *SharpDeveloper*, *Turbo C# Explorer* nebo Microsoft *Visual Studio*. Použil jsem to poslední, protože s ním jíž mám dobrou zkušenost. Navíc je pro studenty Visual Studio je zadarmo. Lze ho stáhnout na webových stránkách *DreamsPark*. Jako technologii pro tvorbu aplikace jsem si zvolil WPF (*Windows Presentation Foundation*).

3.1.2 Nároky na architekturu aplikace

Jako každá složitá struktura, počítačové nebo mobilní aplikace musejí mít pod sebou pevný základ. Špatná definice klíčových scénářů, špatné projektování součástí systému nebo neschopnost prognózování dlouhodobých následků návrhových řešení mohou vést ke komplikaci podpory aplikace v budoucnu anebo dokonce ji ohrozit.

Cílem architektury je v podstatě sjednocení dvou typu požadavků: technického a klientského. Velmi často mezi nimi se musí najít kompromis. Například, zákazník chce co nejvyšší výkon aplikace, ale kvůli použitým řešením může se snížit škálovatelnost nebo kompatibilita. Dobře navržená struktura programu je flexibilní, aby odpovídala jak budoucím uživatelským požadavkům, tak i z pohledu narůstajícího tempa technického pokroku.

Co musí architektura umět:

- Odhalovat strukturu systému, ale skrývat detaily realizace
- Pokud je možné, vyhovovat požadavkům všech stran
- Realizovat všechny varianty použití (funkcionalita)

U té moje aplikace jsem se snažil vycházet ze základních principů projektování. Přinášejí úsporu času (méně náročná správa kódu), pohodlnost ve využití a škálovatelnost. Tyhle principy jsou:

- Správné rozdělení funkcí a co nemenší spojení a co nejvyšší soudružnost
- Princip „jediného zodpovědného“. To znamená, že každá komponenta odpovídá pouze za svou vlastnost nebo funkci.
- Neopakovatelnost. Komponenty nesmějí se dublovat.
- Princip minimální znalosti. Jednotlivá struktura by neměla vědět vnitřní detaily jiných struktur.

Existuje několik šablon projektování v závislosti na typu aplikace (webová služba, databázová aplikace, mobilní aplikace atd.). Každá má své výhody a nevýhody. Asi nejvíce známá je architektura „klient-server“, avšak s rostoucí složitostí a požadavky se často uplatňuje tři- a vícevrstvá architektura. V této práci jsem se snažil vycházet z třívrstvé objektově-orientované architektury.

Objektově-orientována architektura je paradigma projektování, založena na rozdělení aplikace na samostatné a vhodné pro jiná použití objekty. Pokládáme celý systém ne jako sadu

podprogramů a procedur, ale jako seznam vzájemně působících objektů. Tyto objekty jsou na sebe nezávislé.

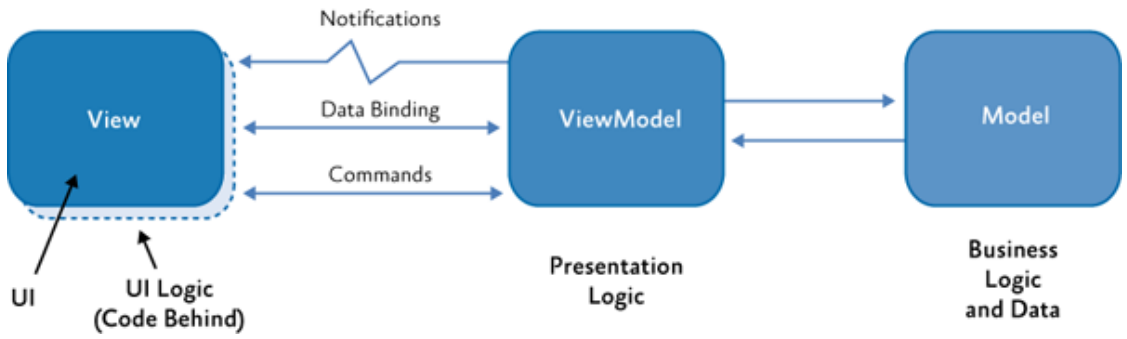
Objektově-orientována architektura má následující rysy:

- Abstrakce. Například, potřebujeme programově popsat autobazar. Prodávají se tam však nejenom osobní auta, ale třeba i nákladové. Mohli bychom založit dva objekty, „Škodovku“ a „Tatru“, a definovat u každého všechny jeho vlastnosti (objem motoru, barvu, poloměr kol atd.). Takovým postupem bychom ale značně zvětšili objem kódu. Co kdybychom chtěli přidat vlastnost „objem kufru“ ke všem autům? Měla by se tedy tato vlastnost zapsat do obojích souborů. A co když těch druhů aut máme stovky? Proto nejlepší by bylo definovat abstraktní třídu nebo rozhraní „Auto“ a odvodit od ní naše objekty.
- Kompozice. Objekty můžou mít v sobě jiné objekty případně je skrývat od ostatních. Třeba auto má klimatizační jednotku uvnitř, kterou lze vnímat jako objekt se svými vlastnostmi a metody.
- Polymorfizmus. Dovoluje předefinovat chování bázového typu přes vytvoření nové metody. Například, obyčejné auto natankujeme na pumpě, zatímco elektromobil musí se zapojit do zásuvky. To znamená, že by se přepsala metoda „Natankuj” v objektu „Elektromobil“. (15)

3.1.3 Návrhový vzor MVVM

V dnešní době samotný proces vývoje softwaru prošel výraznými změnami. Jak bylo řečeno v předchozí podkapitole, to je spojeno s narůstající složitostí aplikací, nutnosti testování, analýzy a designu. Proto ve světě IT vznikla potřeba rozdělit vývoj na jednotlivé moduly a při práci používat unifikována řešení neboli šablony. Šablony značně usnadňují komunikaci mezi vývojáři, můžou odkazovat na již známé konstrukce a také snižují počet chyb v programování a návrhu.

Existuje mnoho návrhových vzorů pro různé technologie a jazyky. U Microsoft WPF se nejvíc uplatnila šablona MVVM (***Model-View-ViewModel***). Řekneme o ni pár slov.



Obr. 24. Návrhový vzor MVVM. Zdroj: (16)

Pod pojmem ***Model*** se rozumí část, která obsahuje „business logiku“ aplikace a ví pouze sama o sobě. Například, našim modelem je auto. V představení budeme mít vlastností auta (objem motoru, barvu karoserie, typ převodovky), nějaké metody (nastartovat, zapnout/vypnout klimatizaci, otevřít kufr). Avšak to auto nepotřebuje vědět, kdo ho bude řídit.

View obsahuje prvky uživatelského rozhraní a může mít kód, který realizuje interakci uživatele a aplikace. ***View*** definuje grafickou úpravu, umístění a strukturu elementů, které se zobrazují uživateli. Příkladem jsou tlačítka, tabulky, textová pole. U technologie WPF většinou tuto část definujeme přes XML-podobnou syntaxi, která se nazývá *XAML*.

Část ***Model-View*** slouží především k oddělení dvou výše zmíněných součástí. Je to abstrakce modelu, která ho upravuje a zároveň představuje data ve vhodném formátu pro ***View***. ***Model-View*** spouští události a příkazy a také hrají roli zdroje dat pro jakýkoliv prvek v jejím grafickém zobrazení. Například pokud zmáčkneme tlačítko v programu, ***Model-View*** zpracuje událost a vybarví to tlačítko modrou barvou.

Jedněm z nejdůležitějších aspektů WPF, který dělá MVVM velice flexibilním a komfortním je tzv. „vázání dat“ (*Data Binding*). Je to přiřazení vlastností modelu grafickým

elementům. *Data Binding* podporuje několik režimů. Jednosměrný režim vázání prvků UI předpokládá zobrazení dat při vizualizaci bez zásahu uživatele. Obousměrné vázání obnoví údaj v modelu, když ho uživatel v rozhraní změnil, a následně zobrazí. Jednosměrný zdrojový režim mění data v modelu při zásahu uživatele, ale neukazuje změny rozhraní při změně modelu.

Abychom mohli měnit hodnoty modelu přes UI a naopak, třída modelu musí implementovat speciální rozhraní *INotifyPropertyChanged*. Toto rozhraní obsahuje jediný prvek – událost *PropertyChanged*. Pokud bychom chtěli změnit vlastnost modelu, například pro auto zvětšit objem motoru z 2,5 litrů na 3,5, pak přiřadíme hodnotu „3,5“ vlastnosti „Engine capacity“ a vyvoláme událost *PropertyChanged*, kde jako vstupní parametr uvedeme název obnovené vlastnosti.

Šablona MVVM je zahrnuta do aplikace a používá se hodně při definici modelu budovy a připojení ke kontroléru *InteliSys*^{NTC} v simulátoru.

3.1.4 Knihovna SnmpSharpLib

Protože SNMP je světově známým protokolem, dá se najít na Internetu hodně knihoven (nejen v C# a Java) pro realizaci interakce mezi klientským programem a počítači nebo jinými zařízení. Našel jsem dvě vhodné Open Source varianty: *snmpsharpnet* a *SnmpSharpLib*, a zastavil jsem se u té druhé.

SnmpSharpLib se vyvíjí od roku 2008 čínským programátorem Lex Li a je určena především pro vývojáře, kteří dělají v .NET nebo Xamarin Mono platformách. Zdrojový kód knihovny lze najít na *GitHub*. Konzultace a technická podpora se poskytují za peníze.

Knihovna pracuje s verzemi v1, v2 a v3 SNMP protokolu. Má v sobě parser MIB tabulek, který poskytuje seznam všech objektů definovaných v tabulce. Také je k dispozici třída *Assembler*, umožňující sestavit z vyparsovaných objektů dřevo (MIB tree).

Níže jsou příklady některých příkazů SNMP protokolu.

a.) **GET** – tento kód ukazuje, jak poslat příkaz SNMP agentu na adrese 192.168.1.2 a dotaz pro pole s OID „1.3.6.1.2.1.1.1.0“.

```
var result = Messenger.Get(VersionCode.V1,
                    new IPEndPoint(IPAddress.Parse("192.168.1.2"), 161),
                      new OctetString("public"),
                      new List<Variable> { new Variable(new
                ObjectIdentifier("1.3.6.1.2.1.1.1.0")) },
                  60000);
```

b.) **SET** – ten to kód ukazuje, jak poslat příkaz SNMP agentu na adrese 192.168.1.2 , aby změnil hodnotu pole s OID „1.3.6.1.2.1.1.1.0“ na „Test“.

```
var result = Messenger.Set(VersionCode.V1,
                        new IPEndPoint(IPAddress.Parse("192.168.1.2"), 161),
                        new OctetString("public"),
                       new List<Variable> { new Variable(new
                ObjectIdentifier("1.3.6.1.2.1.1.6.0"), new OctetString("Test")) },
                     60000);
```

Této operace vyhodí chybu, pokud se nedočkají odpovědi po 60000 mil vteřinách (jedna minuta). Pokud dostanou odpověď, bude ve výhledu seznamu proměnných, na které se dotazovalo. (17)

3.1.5 Struktura aplikace

Řešení (*Solution*) vytvořené ve Visual Studio se skládá ze sedmi projektů: *MainApplication, UserControls, InterfaceLibrary, BMS, CircularGauge, Logger a SNMP. MainApplication* je tzv. počátečním projektem (*startup project*). V něm se nadefinují hlavní pracovní okna aplikace, popíše se jejích výhled a chování elementů uživatelského rozhraní: karet, tlačítek, tabulek s daty a ostatních. Projekt *InterfaceLibrary* obsahuje obecnou definici všech důležitých tříd aplikace, hrubě řečeno, „model modelů“. Projekt *Logger* slouží k přehledu procesů celého řešení, v něm se nadefinuje logovací uživatelský element, který zobrazuje události případně chyby aplikace. Projekt *SNMP* realizuje strukturu datového modelu pro připojení ke kontroléru. V projektu s názvem *BMS* se reprezentuje model inteligentní budovy včetně napájecího a zátěžového podsystému a jednotlivé prvky (zařízení) těchto podsystémů. Projekty *SNMP* a *BMS* představují ***Model*** v návrhovém vzoru Návrhový vzor MVVM (viz podkapitola 3.1.3). Projekt *CircularGauge* (18) jeví knihovnu s prvkem měřidlo k přehlednějšímu zobrazení množství elektřiny dodané/exportované budovou. Rozebírat se v práci nebude, protože je jenom pomocným elementem a nemá vliv na chování aplikace. Nakonec projekt *UserControls* realizuje jednoduchý operátorský panel, který se skládá ze seznamu přístrojů používaných v budovách a tzv. pracovní plochy. Na tuto plochu bude možné přetáhnout myší zařízení z výše zmíněného seznamu a tím pádem měnit model budovy, realizovány v projektu *BMS*. Také je v projektu definován výhled karty detailu podsystému budovy.

3.1.5.1 Projekt InterfaceLibrary

Projekt představuje kostru celé aplikace. V něm pomocí rozhraní (*interface*) je nadefinováno, jak má vypadat jakékoliv zařízení (*Device*) nebo připojení (*Connection*). Rozebereme tyto pojmy podrobněji.

Zařízení je objekt, který vykonává nějakou práci a působí s elektřinou. Jako zařízení můžeme brát ventilátor, transformátor nebo výtah. Mezi vlastnosti zařízení patří například:

- Identifikátor (*ID*)
- Vyráběný nebo spotřebovaný výkon (*Power*)
- Stav spínače (*Switch Enabled/Disabled*)
- Pracovní stav (*Enabled*/Disabled)
- Typ podsystému budovy, k němuž zařízení patří (*Building systém type*)

V aplikaci všechna zařízení jsou virtuální, čili existují jenom v programu. Všechna zařízení mají implementovat rozhraní *IDevice*.

Připojení, buď to SNMP nebo třeba ModBus připojení, je způsobem komunikace s reálním zařízením. Všechna připojení implementují rozhraní *IConnection*. Každé připojení obsahuje vlastnosti, popsané rozhraním *IConnectionProperty.* Každá z vlastností připojení má svoje jméno, adresu (identifikátor) a hodnotu. Například, připojení ke kontroléru přes SNMP má vlastnost s názvem „*vBOUT*“ a identifikátorem „*1.3.6.1.4.1.28634.5.3088647955.2.8239*“. Její hodnotou je řetězec z 16 symbolů (nuly nebo jedničky), který říká, kolik binárních výstupů je vypnuto nebo zapnuto, přičemž pořadí bitů je obráceno v porovnáním s číslováním na panelu simulátoru, čili první znak řetězce odpovídá poslednímu 16-mu výstupu (zátěž X4). Více informací o binárních výstupech, vstupech a konfiguraci poskytuje podkapitola 2.3.

Velice důležitou součásti aplikace, která se nachází v projektu *InterfaceLibrary*, jsou tzv. **scénáře**. Scénáře obecně představují logickou konstrukci:

IF (*sada podmínek***) THEN (***sada akcí***) ELSE (***sada alternativních akcí***),** kde:

- *Sada podmínek* – seznam podmínek spojených přes operátory *AND* nebo *OR*. V jednotlivých podmínkách určíme operand. Operandem může být virtuální zařízení, připojení anebo celý model budovy. Pak uvedeme sledovanou vlastnost operandu, operátor porovnání a nakonec hodnotu porovnání, přičemž u virtuálních zařízení a připojení sledované vlastnosti se můžou lišit. Například, pokud máme k dispozici zařízení „Ventilátor“, můžeme vytvořit podmínku, kde

operandem bude samotné zařízení; sledovanou vlastností bude pracovní stav (funguje/nefunguje); jako operátor porovnání zvolíme „rovná se“; hodnotou bude „true“ nebo „false“, protože sledovaná vlastnost je typu „Bool“. Sada podmínek se vyhodnotí, pokud se změní aspoň jedna hodnota sledované vlastnosti daného zařízení.

- *Sada akcí* – seznam dějů, které se vykonají, pokud bude splněna sada podmínek. Když v podmínkách jsme sledovali nějaké vlastnosti, tak u akcí naopak té vlastnosti měníme. Součástmi akce jsou cíl, typ příkazu (*Action Command*) závislý na cílu, název měnící se vlastnosti a nová hodnota, která ji bude přiřazena. Cílem může být virtuální zařízení nebo připojení. Typu příkazů zařízení a připojení můžou se lišit mezi sebou. Například, u virtuálního zařízení „Ventilátor“ při zvoleném typu příkazu „Set value“ můžeme změnit hodnotu jeho výkonu ze 100 W na 150 W.
- *Sada alternativních akcí* – seznam dějů, které se vykonají, pokud *sada podmínek* nebude splněna. Platí pro ni stejné principu jako pro *sadu akcí*.

Scénáře jsou užitečné z toho pohledu, že nemusíme přepisovat kód aplikace, pokud se změní nároky na ni. Dovolují provázat virtuální zařízení a připojení mezi sebou a tím dosáhnout flexibilního chování celého systému.

Další podrobnosti lze najít ve třídním diagramu (viz 6.1).

3.1.5.2 Projekt BMS

Tento projekt realizuje vlastně řečeno model inteligentní budovy. Při navrhování struktury jsem vycházel z několika předpokladů:

- Budovu lze vnímat jako soustavu z několika podsystémů. Vydělil jsem podsystém napájení, HVAC, požární systém a také skupinu účelových spotřebičů (dále v textu **zátěžový systém**), která se může lišit u různých typů budov (poliklinika, vojenská základna, objekt státní správy). V aplikaci však lze použit jenom zátěžový a napájecí systémy.
- Ke každému z podsystémů patří několik zařízení (viz podkapitola 3.1.5.1). Například, požární pumpy nebo evakuační výtahy jsou součástí požárního systému. Dieselagregát a elektrickou síť řadíme k napájecímu systému.
- Jednotlivé systémy se charakterizují jejích okamžitým výkonem a celkovým výkonem patřících zařízení.
- Zařízení se dělí na zdroje elektřiny a její spotřebiče.
- Každý spotřebič má přidělenou prioritu napájení.U spotřebiče ona uvádí, jestli má být zásobován elektrickou energií, pokud napájecí systém budovy najednou ztratí část výkonu. Priorita napájení se rozděluje na tři úrovně: vysoká, střední a nízká.

Jak již bylo zmíněno, projekt *BMS* slouží k reprezentaci modelu inteligentní budovy. Avšak je zřejmé, že tento model není statický. Proto důležité je, aby všechny změny se zobrazovali v uživatelském rozhraní. K tomu se používá vázání dat. Aby toto vázání fungovalo, musejí všechny závislé části třídy modelu budovy realizovat rozhraní *INotifyPropertyChanged* (viz podkapitola 3.1.3)

Realizace tohoto rozhraní je udělána ve třídě *PropertyChangedClass* pomocí metody *SetField()*. Tato třída je předkem následujících tříd:

- *BuildingSystem* – abstraktní zobrazení podsystému budovy.
- *Device* – abstraktní představení jednotlivého zařízení v budově.
- *LoadSystem* – zátěžový systém.
- *ElectricSystem* – realizace napájecího systému
- *PowerConsumer* – abstraktní představení spotřebiče elektrické energie.

Model budovy obsahuje seznam připojení k reálným zařízením jako třeba řídicí jednotka *InteliSys*^{NTC}. Přes připojení lze monitorovat a měnit stav jednotky.

Podle logiky systému, pokud zdroje elektřiny jsou odpojené nebo nefunkční, žádný spotřebič nemůže pracovat. Nebude fungovat i v případě když elektřina je, ale spínač zařízení je otevřen, anebo když mu to zakáže nějaký scénář. Připojení spotřebiče k energosystému omezuji vlastnosti toku elektrické energie modelu. Vyhodnotit směr a velikost těchto toků můžeme pomocí pěti parametrů:

- *Total Load* – vyhodnocuje celkový výkon *aktivních* spotřebičů. Například, pokud máme tři ventilátory a z nich zapnut a funguje pouze jeden, pak celkový výkon zátěže bude se rovnat výkonu tohoto ventilátoru. Tuto vlastnost lze uvést jako tag pro hodnotu akce/alternativní akce scénáře.
- *Exported (produced) Power* – popisuje celkový výkon *interních* zdrojů budovy, čili nebere v úvahu výkon elektrické sítě (sítí). Například, pokud máme továrnu s dvěma běžícími gen-sety, které pokrývají část zátěže továrny v paralelním režimu se síti, pak tento parametr se rovná celkovému výkonu obojích gen-setů. Tuto vlastnost lze uvést jako tag pro hodnotu akce/alternativní akce scénáře.
- *Load Reserve* – říká, jak je velká výkonová rezerva. Počítá se jako rozdíl mezi celkovou vyráběnou energii jak interních, tak i externích zdrojů, a celkovou spotřebovanou energii (*Total Load*). Tuto vlastnost lze uvést jako tag pro hodnotu akce/alternativní akce scénáře.
- *Required Load Reserve* – uvádí omezení na minimální hodnotu výkonové rezervy (*Load Reserve*). To znamená, že když máme rezervu 50 kW a parametr *Required Load Reserve* se rovná 30 kW, pak nedokážeme připojit zátěž o výkonu 40 kW (*50-30<40*).
- *Power Saldo* – vyhodnocuje směr „čistého“ toku elektřiny. Například, pokud připojená k síti budova produkuje 30 kW pomocí vnitřních zdrojů a zároveň jen 20 kW spotřebovává, pak tento parametr je záporný (rozdíl v 10 kW jde do sítě). Tuto vlastnost lze uvést jako tag pro hodnotu akce/alternativní akce scénáře.

V projektu se nachází jedna z realizací typu příkazu pro scénáře, zmíněných v předchozí podkapitole. Nazývá se „SetValueCommand“ a slouží ke změně parametrů virtuálních zařízení (výkon, stav stykače atd.). Ostatní typu příkazů jsou popsané v projektu *SNMP* (viz 3.1.5.4).

Další podrobnosti lze najít ve třídním diagramu projektu (viz 6.2).

3.1.5.3 Projekt Logger

Součástí každé složitější aplikace je logování všech významných událostí. Přináší to výhodu jak z pohledu ověření funkčnosti, tak i z pohledu testování při vývoji. Logovat se můžou vnitřní a vnější události. V dané aplikaci pozor při logování je věnován spíše vnějšímu prostředí, čili komunikací s *InteliSys*[NTC], kde se sledují výsledky čtení/zápisu hodnot kontroléru. Z vnitřních dějů aplikace ukládají se jenom záznamy o výjimkách. Zápis do logu se koná pouze za běhu programu a neukládá se na disk. Avšak je možné nakopírovat záznamy v aplikaci a uložit je do textového souboru samostatně.

Výsledkem projektu *Logger* je textové pole, ve kterém se zobrazují informace o jednotlivých událostech aplikace, jako například připojení ke kontroléru, úspěšné nebo chybné ukončení příkazu apod. Projekt odkazuje na logovací knihovnu *log4net* pro platformu .NET, kterou lze stáhnout zde. Pokud v kódu je nutno něco zaznamenat do logu, použije se příkaz *Log.Write.* Jako příklad důležité události pro logování vezmeme serializaci, která je určená pro ukládání modelu, dat a konfigurací do binárních souborů. Serializace se hodí, když chceme výsledky práce v aplikaci někam uložit nebo nahradit existující soubor novou verzi. Kód metody serializace, která ukládá aktuální seznam připojení, je uveden dole:

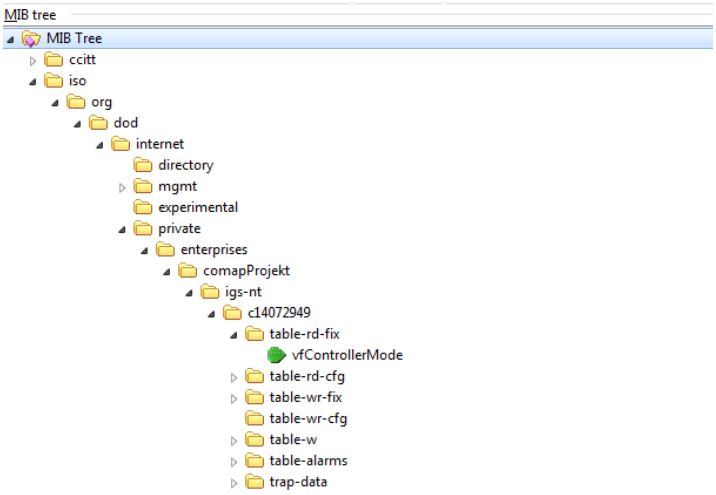
```
private void Serialize(object serializedObject, string filename)
 {
            Stream stream; //Datový proud, kterým zapisujeme hodnoty do souboru
            BinaryFormatter formatter = new BinaryFormatter();
            string projectPath =
Path.GetDirectoryName(Path.GetDirectoryName(Directory.GetCurrentDirectory()));//Cesta k projektu

            string filePath = System.IO.Path.Combine(projectPath,
"Saves\\SNMPConnections",filename); //Úplná cesta k souboru

            stream = new FileStream(filePath, FileMode.OpenOrCreate);

            try
            {
                formatter.Serialize(stream, serializedObject);//Zápis serializovaného objektu do
souboru
            }
            catch (SerializationException ex)
            {
                MessageBox.Show("Failed to save connections");
                Log.Write(LogLevel.Error, "Failed to save connections. Exception details:" +
ex.Message); //Pokud se zachytí výjimka, do logu se zapíše zpráva
            }
            finally
            {
                stream.Close();//Nakonec zavřeme datový proud
            }
}
```

Použité v metodě *Log.Write* vyjmenování *LogLevel* má v sobě definované stavy pro popis typu zpráv. Existuje pět typů: *Debug, Error, Fatal, Info, Warning.* Stav spolu s časovou známkou a popisem události se uvádějí při zobrazení v textovém poli. Log lze vymazat zmačknutím tlačítka „Clear log“ nad textovým polem.

Další podrobnosti lze najít ve třídním diagramu projektu (viz 6.3).

3.1.5.4 Projekt SNMP

Projekt *SNMP* se stará o vhodnou objektovou realizaci spojení s kontrolérem. Jak již víme, ovládání se uskutečňuje přes vygenerovanou pomocí *InteliMonitor* MIB tabulku kontroléru, která v podstatě vypadá jako strom.

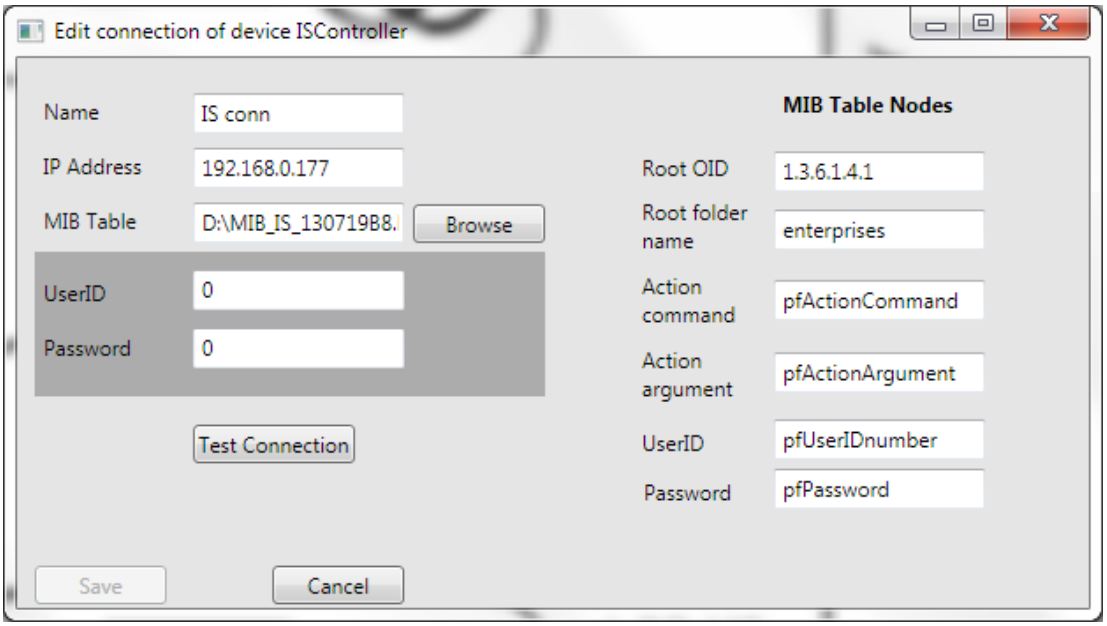


Obr. 25 MIB strom. Zdroj: vlastní

Konečným prvkem toho stromu je vlastnost kontroléru, například, napětí na generátoru, provozní stav případně výskyt poplachu. Některé vlastnosti můžeme jen číst (typ „read-only“), některé můžeme jen zapisovat (typ „write-only“) a některé dovolují provedení obou operací. Ke „write-only“ patří třeba jméno uživatele a heslo. Kromě toho, můžeme zadávat příkazy (zastavit generátor, rozepnout stykač) zápisem určitých hodnot do vlastností *pfActionCommand* a *pfActionArgument*.

V projektu *SNMP* se nachází několik tříd, každá ze kterých je potomkem třídy *PropertyChangedClass* (viz kapitola 3.1.5.1**Error! Reference source not found.**). Třída *SNMPConnection* je klíčová, představuje připojení se všemi parametry jako objekt. Právě na ni se budou vázat UI elementy karty „Connection Explorer“ z projektu *MainApplication*. Ve třídě *SNMPConnection* všechny prvky, vyparsované z MIB tabulky, ukládají se do číselníku *SNMPProperties*, který obsahuje prvky typu *SNMPProperty*. Třída *SNMPProperty* je objektově představení konečných prvků neboli listů MIB stromu. Objektům této třídy lze přiřadit jméno, OID (*Object Identifier*), hodnotu atd. Objekt této třídy může vykonávat příkazy *SetValue()* a *RefreshValue()* vázané na knihovnu *SnmpSharpLib* (viz podkapitola 3.1.4), což je přiřazení nebo obnovení hodnoty celku MIB stromu.

Další třídou je *SNMPCommand*, což je realizace příkazu, vysílaných do zařízení. *SNMPCommand* má v sobě dvě proměnné typu *SNMPProperty.* První odpovídá MIB prvku *pfActionCommand* a druhá - *pfActionArgument*. Abychom zadali příkaz, musíme nejdřív přiřadit hodnotu elementu *pfActionArgument* a potom *pfActionCommand*, což uděláme metodou *SetValue()*. Všechny vlastnosti této třídy jako *CommandName*, *HEXCommandArgument*, *DECCommand* atd. se načítají ze souboru **Commands.csv**, který se umísťuje ve složce **Config** projektu.



26. Výhled okna editace SNMP připojení v aplikaci. Zdroj: vlastní

Kontroléry ComAp mají uložené několik registrů, na jejíchž hodnotách můžou záviset externí přístroje, které ten kontrolér ovládá, například, relé. Nejsou to setpointy, kterými nastavujeme konfiguraci samotného kontroléru, ale pouze hodnoty v paměti. Jmenují se vnější hodnoty (*External values*). V projektu SNMP oni jsou reprezentované třídou se stejným názvem (*ExternalValue*). Konfigurace vnějších hodnot se nacházejí rovněž ve složce **Config** projektu.

Vnější hodnoty jsou čtyři v rozsahu od -32000 do 32000. Jsou typu „write-only“, čili se nemůžou načítat, pouze zapisovat. Jejích pomocí budeme sdělovat kontroléru velikost zátěže a okamžitý výkon napájecí soustavy modelu budovy**. *External value 1* bude odpovídat hodnotě výkonu zátěže, kterou pokryje síť, zatímco *External value 2* – hodnotě výkonu zátěže pokryté z gen-setu.**

Jak bylo řečeno v podkapitole 2.3, konkrétně v popisu obrázku panelu přepínačů a potenciometrů, jištěná a nejištěná zátěž je představená binárními výstupy 9 až 16. Binární

výstupy jsou vázané na hodnoty osmi odpovídajících binárních vstupů 9 až 16 a také osmi virtuálních přepínačů kontroléru, které mají název *Remote Switch*. To znamená, že pokud třeba chceme zapnout zátěž Y1, čili nastavit na binárním výstupu B_{out9} logickou 1, potřebujeme mít sepnutý B_{in9} (což uděláme přepínačem dole) a virtuální přepínač s indexem 0 (což uděláme programově v aplikaci).

V projektu se nachází několik realizací typu příkazu pro scénáře, zmíněné v podkapitole 3.1.5.1. Lze je vybrat v sadě akcí/alternativních akcí pouze při volbě reálného zařízení s připojením typu *SNMPConnection*. Jsou to příkazy:

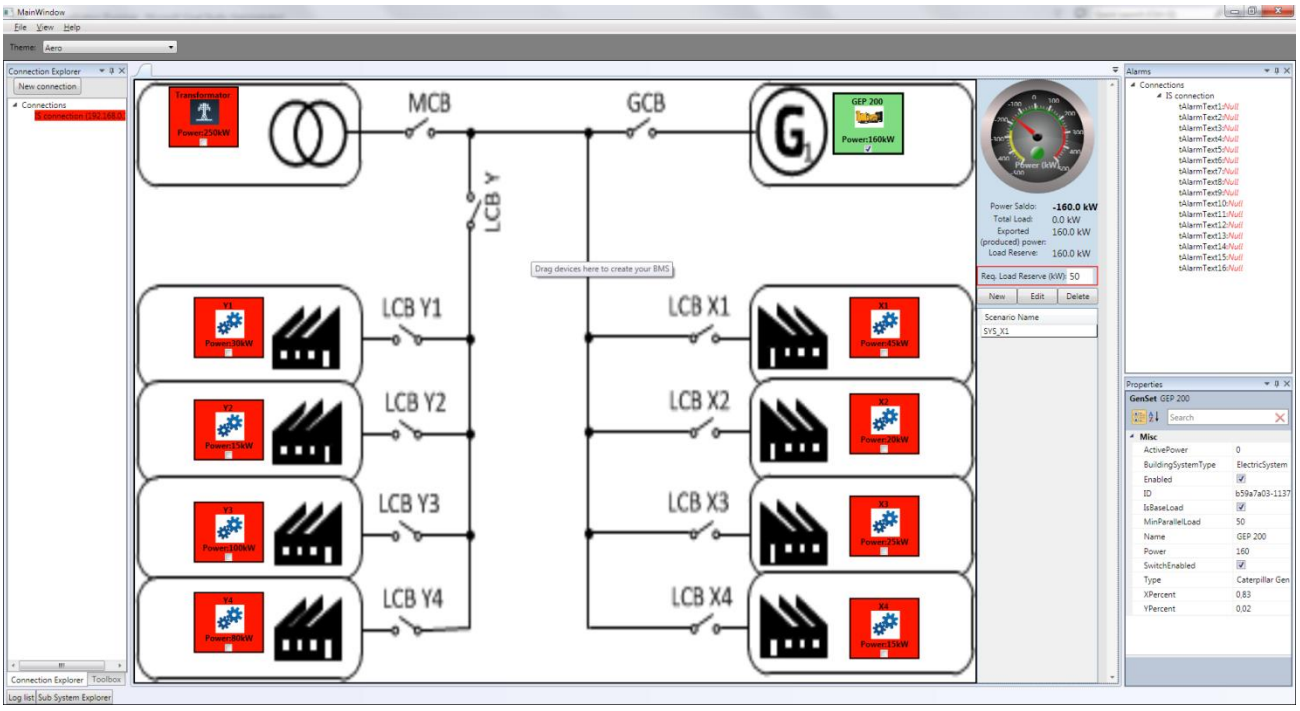
- *SetExternalValueCommand* – při volbě tohoto příkazu uživatel také zvolí číslo externí hodnoty kontroléru (tj. jedna až čtyři) a novou hodnotu, která se odešle přes SNMP do *InteliSys*^{NTC} po vyhodnocení sady podmínek scénáře.
- *SetModeCommand* – při volbě tohoto příkazu uživatel má uvést navíc režim práce kontroléru (tj. AUTO, MANUAL, TEST, SEL, OFF). Pak po vyhodnocení podmínek scénáře do kontroléru se odešle povel ke změně režimu.
- *RunCommand* – při volbě tohoto příkazu uživatel si vybere jako parametr příkazu jeden z dostupných objektu typu *SNMPCommand* dané instance *SNMPConnection*. Můžou to být třeba příkazy k otevření síťového nebo generátorového stykače, zastavení generátoru a podobně. Po vyhodnocení podmínek scénáře příkaz může být odeslán do kontroléru.
- *SetRemoteSwitchCommand* – při volbě tohoto příkazu uživatel zvolí číslo jednoho z osmi virtuálních přepínačů kontroléru a novou hodnotu, která se odešle do *InteliSys*^{NTC} po vyhodnocení sady podmínek scénáře. Poněvadž virtuální přepínač má jen dvě polohy, zapnuto nebo vypnuto, nová hodnotu může být jen „true“ nebo „false“.

Další podrobnosti lze najít ve třídním diagramu (viz 6.5).

3.1.5.5 Projekt UserControls

Projekt obsahuje elementy, nezbytné pro panel operátora (dispečera) pro zobrazení procesů, probíhajících v budově. Panel (obrázek 27) se skládá z:

- Okna se seznamem zařízení, používaných v budovách. Seznam zařízení pro okno se načítá ze souboru *Devices.xml*, který se umísťuje ve složce **Config** projektu. Taková konfigurace dovoluje měnit seznam bez zásahu do kódu aplikace, což zvyšuje flexibilitu. Nachází se v levé části panelu.
- Pracovní plochy, na kterou lze myší přetáhnout přístroje ze seznamu zařízení. Po přetahování přístroj se stává částí příslušného systému budovy. Přístroje můžou se přetahovat po ploše nebo můžou být smazané kliknutím pravým čudlíkem myší a volbou „Delete“. Pracovní plocha se nachází uprostřed panelu.
- Prvků uživatelského rozhraní, odpovídajících jednotlivým zařízením. Jsou to čtverce na pracovní ploše s obrázkem, popisem a červeným nebo zeleným pozadím v závislosti na pracovním stavu.
- Panelu konfigurace na pravé straně. V něm se umísťují ukazatele toků elektřiny (viz podkapitola 3.1.5.2) a tabulka scénářů pro danou budovu (viz podkapitola 3.1.5.1). Scénáře lze vytvářet, mazat nebo editovat.



Obr. 27 Výhled operátorského ovládacího panelu. Zdroj:vlastní

V projektu se nachází definice uživatelského prvku *SystemPanelView*. Tento prvek jako vstupní parametr vyžaduje objekt jednoho z podsystémů budovy (napájecího, zátěžového atd.). Výstupem pro uživatele bude plocha s tabulkou všech zařízení daného podsystému a graf změn aktivního a celkového výkonu podsystému. Graf se obnovuje s periodou 3 vteřiny. V aplikaci se daný prvek použije pro napájecí a zátěžový systém v kartě „Systém Explorer“.

Další informace lze najít ve třídním diagramu (viz 6.6).

3.1.5.6 Projekt MainApplication

Projekt identifikuje elementy počátečního okna v uživatelském rozhraní aplikace a obsahuje odkazy na všechny ostatní projekty. Jako model podle vzoru MVVM se používá objekt třídy *BuildingManagementSystem* (viz podkapitola 3.1.5.2). V tomto projektu se také nacházejí:

- Okna vytvoření, editace a řízení připojení
- Seznam všech aktuálních připojení
- Seznam hlášek všech aktivních připojení
- Tabulka editace vlastnosti virtuálních zařízení
- Menu se záložkami pro vytváření, ukládání souboru modelu budovy.

Při programování projektu jsem narazil na několik komplikaci. Jedna z nich spočívala v tom, že při aktivaci připojení aplikace posílala povely k přihlášení, obnovení hodnot binárních výstupu a vstupu kontroléru a také nastavení jeho virtuálních přepínačů a vnějších hodnot v závislosti na pracovním stavu zařízení v modelu budovy. Zároveň se spouštěl čítač, který jednou za vteřinu se snažil obnovit binární vstupy a výstupy. Obnovení nebo nastavení hodnot jsou ale procesy velice pomalé (řadově vteřiny), které mi navíc zastavovali běh uživatelského rozhraní, čili program neodpovídal na děje uživatele do doby, než by nedostal od kontroléru odezvu. Důvodem těchto zadržení bylo to, že všechny instrukce se vykonávaly pouze v jednom vláknu. K tomu, aby uživatelské rozhraní fungovalo stabilně a bez zpomalení, rozdělil jsem procesy UI a obnovení hodnot z *InteliSys*^{NTC} na více vláken.

Zavedení několika vláken přináší svoje výhody a nevýhody. Výhodou je značné zrychlení běhu celého programu, protože instrukce ve vláknech se zpracovávají paralelně. Nevýhodou (a to bylo dalším problémem, na který jsem narazil) je složitost programování a zejména testování aplikace. Jako příklad se podíváme na kód dole:

```
private void MyMethod()
 {
     bool IsObjectPrepared = false;

     PrepareObject();

     IsObjectPrepared = true;

     if (IsObjectPrepared)
     {
         StartProcess();
     }

  }
```

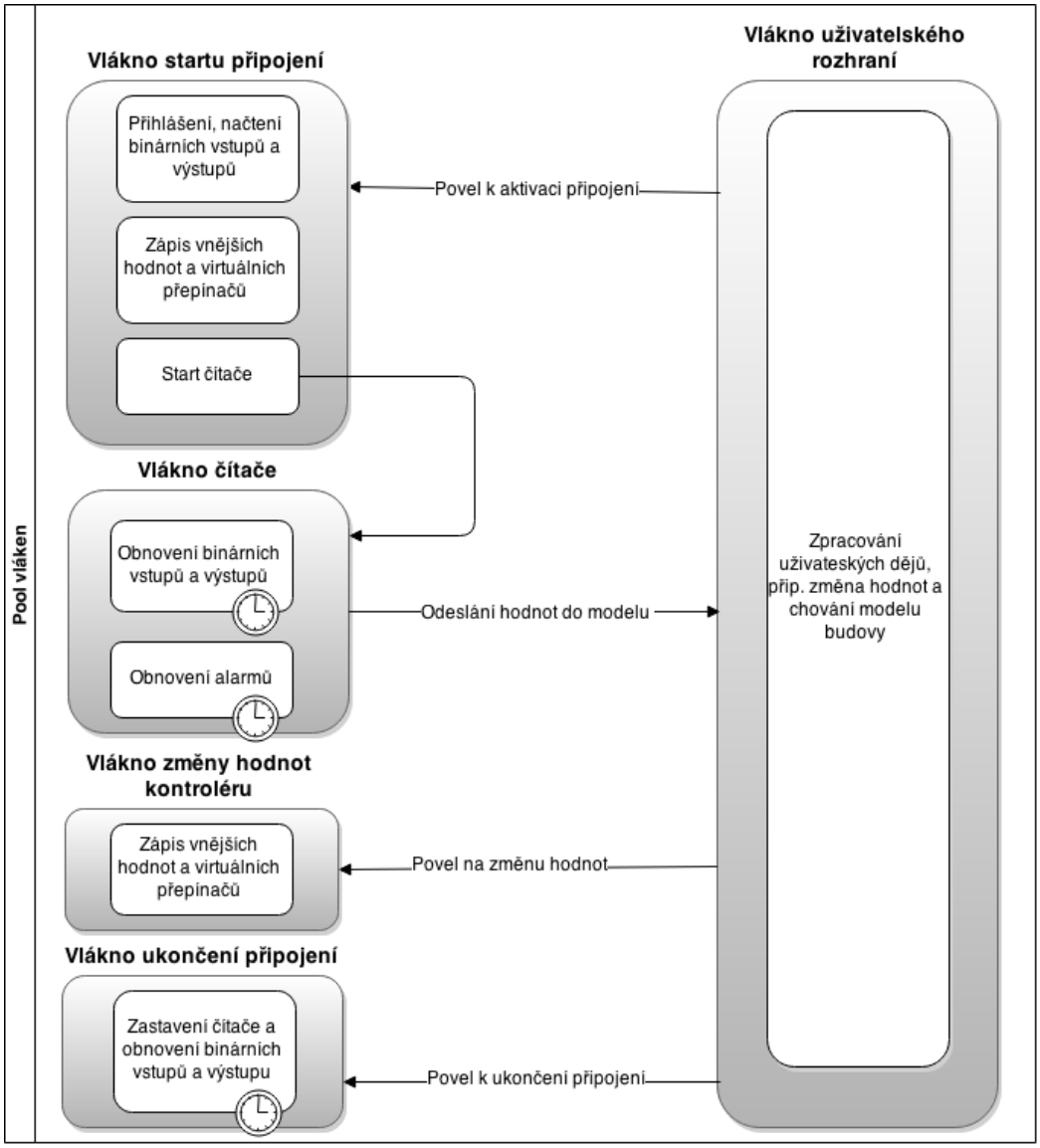
Metoda *MyMethod* () předpokládá, že její tělo se bude zpracovávat synchronně, tj. krok za krokem. Zavede proměnnou stavu objektu, zavolá metodu *PrepareObject()*, potom změní stav objektu a na základě tohoto stavu provede další metodu *StartProcess()*. Avšak při několika běžících vláknech, které budou volat metodu *MyMethod(),* daná metoda nebude fungovat správně. Důvodem je možnost změny proměnné *IsObjectPrepared* jiným vláknem. Čili pokud jedno vlákno provedlo přípravu objektu a ověřuje v konstrukci „if“ jeho stav, druhé vlákno může změnit proměnnou *IsObjectPrepared* na „false“. Tím pádem první vlákno při vyhodnocení stavu objektu nezavolá metodu *StartProcess()* i když podle logiky by mělo.

Správná realizace této metody u multivláknové aplikace by vypadalo pravděpodobně takto (záleží ale na tom, co dělá metoda *StartProcess()* a v jakém kontextu je volaná metoda *MyMethod()*):

```
private static readonly Object _objLock = new Object();
private static bool _globalIsObjectPrepared = false;
//Multivláknová realizace metody
private void MyMethod()
{
    bool IsObjectPrepared = _globalIsObjectPrepared;
    lock (_objLock)
    {
        PrepareObject();
        _globalIsObjectPrepared = true;
        IsObjectPrepared = _globalIsObjectPrepared;
    }
    if (IsObjectPrepared)
    {
        StartProcess();
        _globalIsObjectPrepared = false;
    }

  }
```

Na začátku založíme dvě globální proměnné: *_objLock* a *_globalIsObjectPrepared*. První slouží k vytvoření zámku (*lock*) v metodě *MyMethod()*. Druhá je určená pro uložení hodnoty stavu objektu globálně, aby několik vláken nemohli jeho ovlivnit jako v realizaci metody *MyMethod()* pro synchronní běh. Zámek na objektu říká ostatním vláknům, že musejí počkat, až ten zámek neuvolní jiné vlákno, které získalo přístup k tomuto objektu jako první. Pak v těle zámku, které se vykonává synchronně, provedeme přípravu objektu, nastavíme hodnotu proměnné *IsObjectPrepared* a *_globalIsObjectPrepared* na „true“. Protože proměnné *IsObjectPrepared* přiřazujeme pouze hodnoty globální proměnné, tím eliminujeme její vláknovou závislost. Pak po volání metody *StartProcess()* změníme hodnotu

_globalIsObjectPrepared, aby v kontextu jiných vláken objekt zůstával nepřipravený. Taková realizace zajistí, že metoda *StartProcess()* se bude vykonávat jen ve správném kontextu.



Obr. 28 Schéma interakce mezi vlákny v aplikaci. Zdroj: vlastní

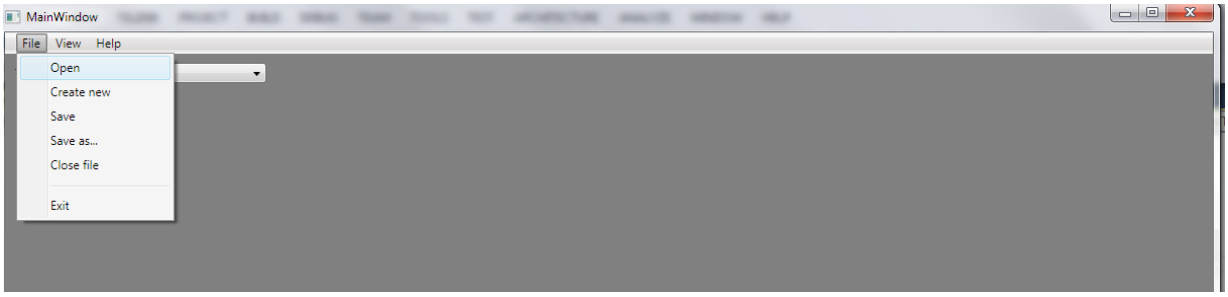
Podle požadavků, kladených na rychlost čtení a zápisu hodnot do kontroléru při nezadržené funkčnosti uživatelského rozhraní, navrhl jsem schéma interakce mezi vlákny. Je představeno na obrázku 28.

Další informace lze najít ve třídním diagramu projektu (viz 6.4**Error! Reference source not found.**).

3.1.6 Popis Aplikace

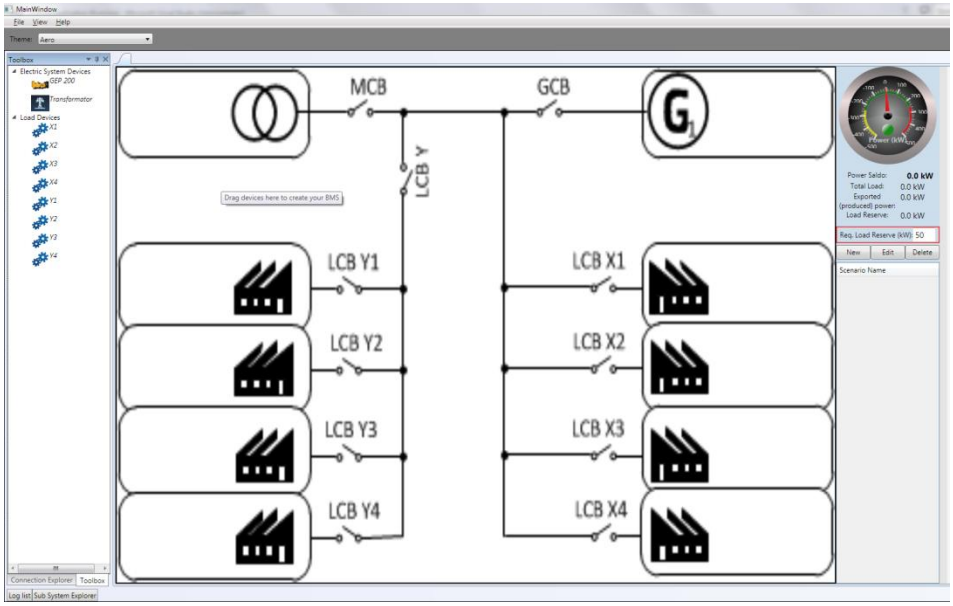
3.1.6.1 Začátek práce

Při spuštění *.exe* souboru aplikace se objeví výchozí okno. Po kliknutí na záložku „File“ menu nahoře máme možnost vytvořit nový model budovy nebo načíst již existující. Zvolíme první variantu.



Obr. 29 Výchozí okna aplikace. Zdroj: vlastní

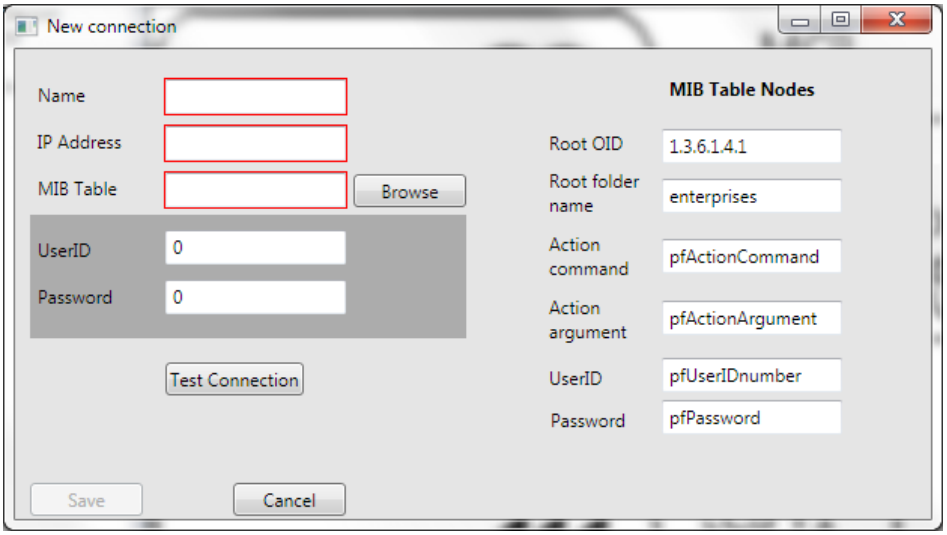
Dalším krokem je volba cesty k souboru na disku. Přípona souboru je *.bin*, čili všechny informace o modelu budovy ukládají se v binárním formátu. Po volbě cesty otevře se operátorsky panel budovy. Schéma elektrické sítě na obrázku 30 je pouze pozadí pro přehlednost a nekoná žádnou funkčnost.



Obr. 30 Operátorský panel. Zdroj: vlastní

Abychom přidali nějaké zařízení do budovy, stačí zmáčknout jedno ze seznamu (karta *Toolbox*) na levé straně a přetáhnout ho na prostřední plochu, nejlépe tak, aby sedělo s pozadím.

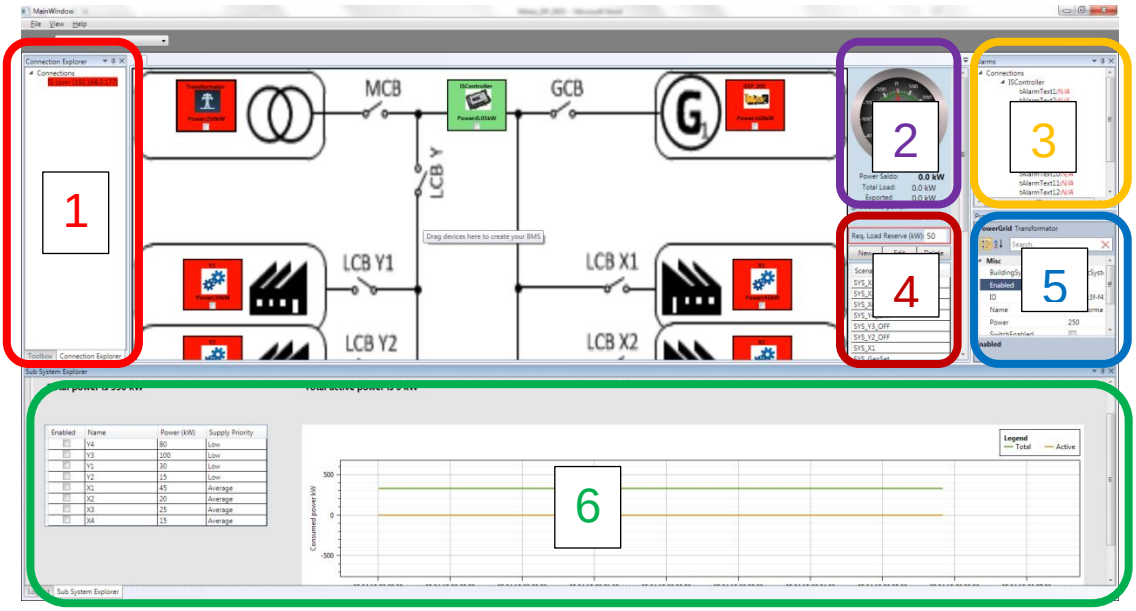
Dále vytvoříme připojení. Přepneme se na kartu *Connection Explorer* a zmáčkneme tlačítko „New Connection“. Objeví se okno jako na obrázku 31.



Obr. 31 Okno založení připojení. Zdroj: vlastní

Na obrázku 31 máme vyplnit pole s názvem připojení, IP-adresou kontroléru a cestu k souboru s MIB tabulkou. Ostatní údaje jsou defaultně nastavené. Po vyplnění všech položek se má zmáčknout tlačítko „Test connection“ a po ověření připojení zařízení lze uložit na pracovní plochu.

Po přidání potřebných zařízení do budovy a navázání spojení s řídicí jednotkou *InteliSys*^{NTC} pustíme se k popisu způsobu jejích ovládání a monitoringu. Na obrázku 32 jsou zobrazená důležitá pracovní okna aplikace.



Obr. 32 Pracovní okna aplikace. Zdroj: vlastní

Číslo okna	Název	Popis
1.	Connection Explorer	Poskytuje seznam všech připojení. Připojení označené červeně nejsou aktivní.
2.	-	Okno hodnot toků elektřiny v budově. V tomto okně lze určit minimální výkonovou rezervu budovy
3.	Alarms	Seznam hlášek aktivních připojení
4.	Scenarios	Seznam scénářů modelu budovy s možností přidávání, editace nebo mazání
5.	Property window	Při kliknutí myší na nějaké zařízení na pracovní ploše zobrazí jeho vlastnosti
6.	Sub systém explorer	Poskytuje informace o stavu jednotlivých podsystémů v budově. Jeho součástí je tabulka všech zařízení daného podsystému a graf celkového a aktivního výkonu

Tabulka 1. Popis pracovních oken aplikace. Zdroj:vlastní

Abychom propojili virtuální zařízení modelu s binárními výstupy, vstupy a virtuálními přepínači kontroléru, máme definovat určité scénáře pro každé zařízení. V další podkapitole uvedeme několik příkladů.

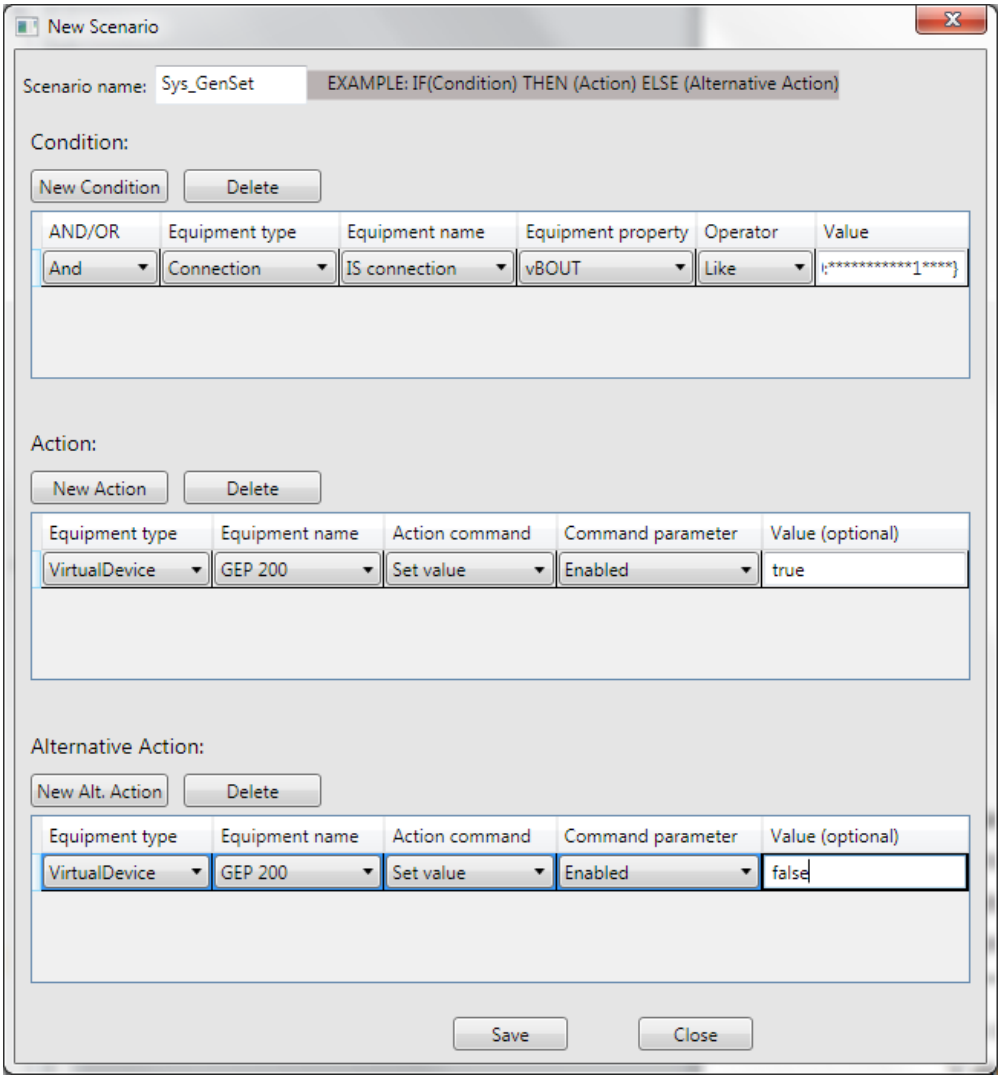
3.1.6.2 Příklady scénářů

Příklad 1. *Jednosměrné propojení virtuálního zařízení „GEP 200" typu „Gen-set" a výstupů simulátoru.*

Jak již víme z podkapitoly 3.1.5.1, gen-set jako každé zařízení má vlastnost stavu spínače a pracovního stavu. V simulátoru těmto vlastnostem odpovídají binární vstup 1 a binární výstup 5 (viz obrázekObr. 33).

Obr. 33 Část konfiguračního panelu simulátoru Starter Kit. Zdroj: vlastní

Stanovením závislosti mezi simulátorem a virtuálním gen-setem v aplikaci vznikají dvě logické konstrukce „IF-ELSE": pro spínač a pro pracovní stav. To znamená, že měli bychom založit dva scénáře. Kliknutím tlačítka „New" v okně „Scenarios" (okno č. 4 na obrázku 32) se objeví okno nového scénáře. Vyplníme ho stejně jako na obrázku 34.



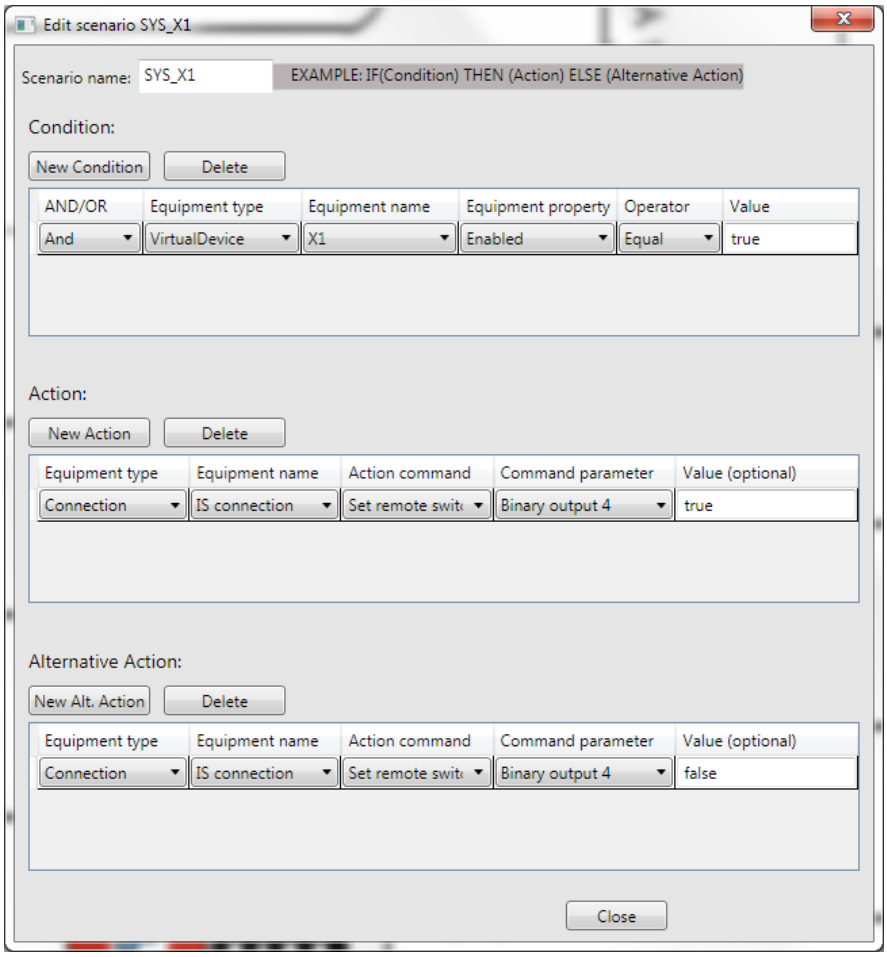
Obr. 34 Okno založení scénáře pro gen-set. Zdroj: vlastní

V daném scénáři sledujeme stav vlastnosti „*vBOUT*“ (binární výstupy) kontroléru (viz 3.1.5.1), kterou jsme získali parsováním MIB tabulky při založení připojení. Operátor porovnání „Like“ je vybrán, protože budeme pracovat s bitovou maskou. Poněvadž zajímá nás jen výstup „GenOK“ pod číslem 5, maska bude vypadat takto **{0:**********1****}**, kde **{0:}** – identifikátor masky, a ************1****** – obsah řetězce. Hvězdička znamená jakýkoliv symbol. Jednička má pátý index od konce, protože vysílané kontrolérem pořadí bitů je obráceno.

V akcích přiměřeně stanovíme, jak se změní pracovní stav virtuálního gen-setu při splnění či nesplnění podmínky.

Stejným způsobem založíme scénář pro propojení binárního vstupu 1 simulátoru a stavu spínače zařízení v aplikaci.

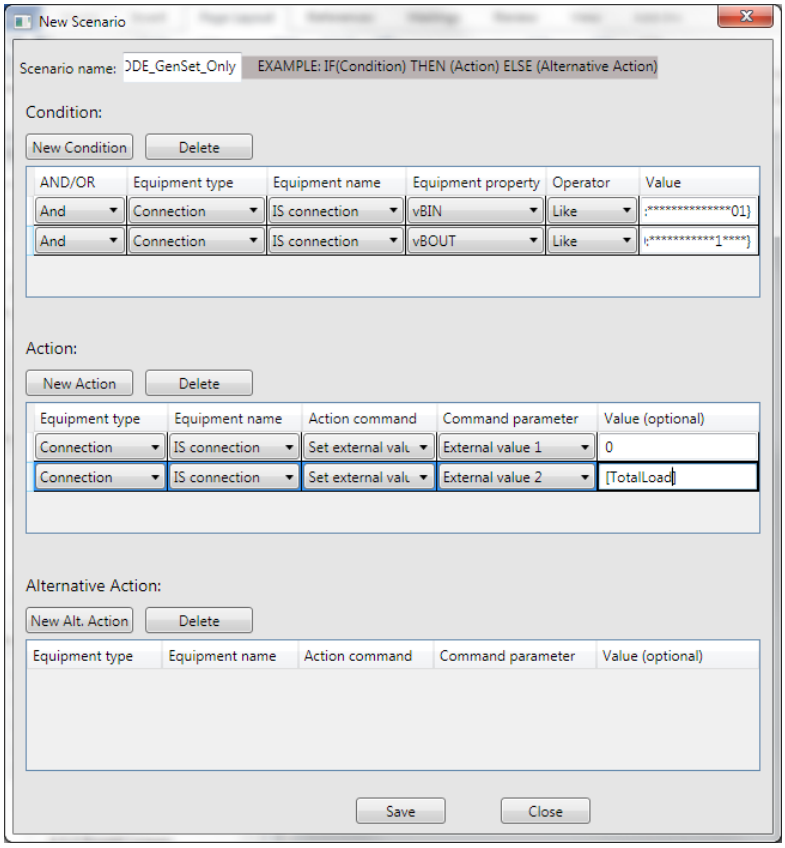
Příklad 2. *Jednosměrné propojení virtuální zátěže „X1" a virtuálního přepínače simulátoru.*



Obr. 35 Okno editace scénáře pro zátěž Zdroj: vlastní

V daném příkladu propojíme jištěnou zátěž X1 a virtuální přepínač simulátoru. Číslování virtuálních přepínačů začíná nulou a přepínač s indexem **0** odpovídá binárnímu výstupu **9** *Starter Kitu*, což je zátěž Y1. Čili aby jméno zátěže X1 s odpovídajícím binárním výstupem **13** bylo stejné jak pro simulátor, tak i pro aplikaci, zvolíme v scénáři virtuální přepínač **4**. V seznamu akcí stanovíme hodnotu tohoto přepínače. Pokud uživatel v aplikaci změní pracovní stav virtuální zátěže na „Zapnuto", do simulátoru se odešle povel k zapínání virtuálního přepínače **4**. Poněvadž binární výstup **13** simulátoru je závislý na binárním vstupu **13** a výše zmíněným přepínači, jeho stav se změní na logickou jedničku a LED dioda **13** simulátoru se rozsvítí.

Příklad 3. *Změna vnějších hodnot (External value) simulátoru v ostrovním režimu.*



Obr. 36 Okno scénáře pro ostrovní režim. Zdroj: vlastní

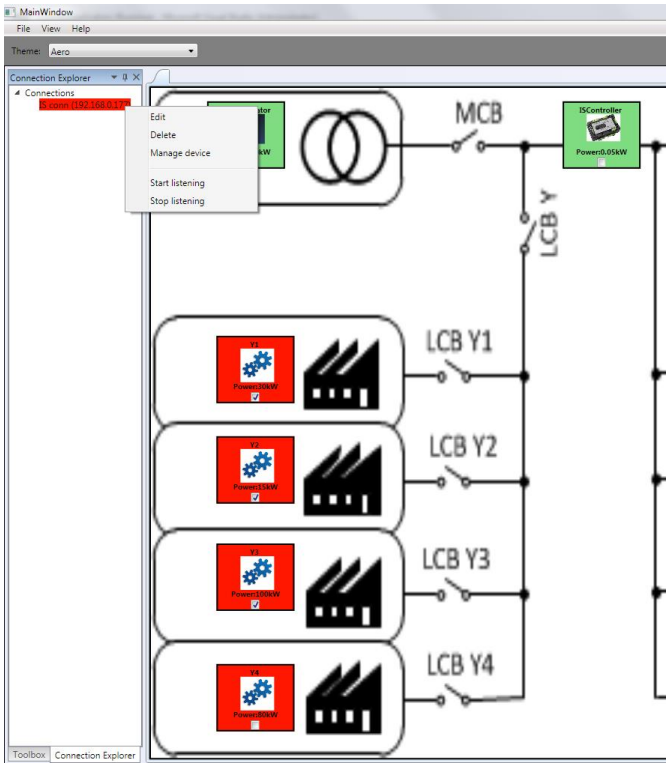
Ostrovní režim znamená, že v aplikaci budova je napájená pouze z gen-setu, zatímco síť je odpojena nebo nefunguje. Jak bylo řečeno v podkapitole 3.1.5.4, první vnější hodnota kontroléru (*External value 1*) popisuje zátěž pokrytou ze sítě a druhá vnější hodnota (*External value 2)* – zátěž pokrytou z gen-setu. Pokud jsme v ostrovním režimu, síť nepokryje nic a veškerou zátěž převezme gen-set.

Proto v podmínkách scénáře určíme okamžik, kdy ten režim nastane. Budeme mít zapnuty binární výstup 5 „GenOK“ v simulátoru a binární vstup „GCB close/open“. Vstup číslo 2 „MCB close/open“ bude v nule. Výstup 6 „MainsOK“ nepotřebujeme, protože podle logiky kontroléru pokud síť spadne, síťový stykač se rozepne. Výhled masky pro výstupy bude **{0:**********1****}** a pro vstupy - **{0:**********01}**.

V akcích určíme, co se stane při splnění podmínek. Pošleme povel k nastavení hodnoty *External value 1* na nulu. Pro hodnotu *External value 2* uvedeme tag „TotalLoad“ v hranatých závorkách, čili pošleme do simulátoru hodnotu aktivního výkonu zátěží, vypočítaného modelem.

3.1.6.3 Řízení a naslouchání připojení

Pokud v levé části hlavního okna aplikace rozklikneme kartu „*Connection explorer*“, objeví se seznam všech dostupných připojení. Pokud chceme posílat povely řídicí jednotce v manuálním režimu i když připojení není aktivní, klikněme pravým čudlíkem na potřebné připojení a zvolíme „*Manage device*“. Objeví se okno konfigurace, v němž lze zadávat příkazy, měnit hodnoty virtuálních přepínačů nebo režimu kontroléru a také monitorovat hodnoty jednotlivých položek MIB stromu.



Obr. 37 Seznam manipulací s připojením. Zdroj: vlastní

Pokud bychom potřebovali provázat model budovy a simulátor *Starter Kit* mezi sebou a zároveň sledovat alarmy kontroléru, musíme připojení aktivovat a stále naslouchat. K tomu v seznamu manipulací s připojením zvolíme příkaz „*Start listening*“ a připojení změní barvu pozadí z červené (neaktivní) na zelenou. Při aktivaci se vykonají všechny existující scénáře modelu budovy. Schéma naslouchávání z programového hlediska popisuje obrázek Obr. 28 Schéma interakce mezi vlákny v aplikaci. Pak po aktivaci a běžícím procesu naslouchání v okně „*Log list*“ v levém dolním rohu aplikace lze monitorovat zápisy o provedených povelech a obnovených hodnotách *InteliSys*^{NTC} nezbytných pro vyhodnocení podmínek jednotlivých scénářů.

3.2 Laboratorní úloha

Součástí této práce bylo navrhnout na základě simulačního kufru *Starter Kit* a naprogramované aplikace laboratorní úlohu pro studenty. Cílem úlohy je dostupně a přehledně vysvětlit, jak funguje systém napájení inteligentní budovy se soustrojí motor-generator (gen-set) jako záložním zdrojem. Součástí inteligentní budovy jsou jištěné a nejištěné zátěže. Budova jako celek je představena *Starter Kitem.* Úloha se skládá ze zadání; teoretického podkladu s popisem typu napájení v budově, provozních režimu řídicí jednotky a vlastně popisem jednotlivých částí simulátoru *Starter Kit*; praktické části s popisem jednotlivých kroků. Uživatel (student) během splnění zadání úlohy se může seznámit s jednotlivými funkcemi *Energy Managementu* v oblasti spotřeby a výroby elektřiny. Postup vykonání zadání je rozdělen podle provozních režimu řídicí jednotky *InteliSys*^{NTC}: ***TEST, AUT*** a ***MAN***. Režim ***TEST*** dovoluje ověřit funkčnost gen-setu v budově. Uplatňujeme ho v pravidelních kontrolních intervalech pro ověření spolehlivosti systému. V režimu ***AUT*** uživatel dokáže pomocí aplikaci na počítači zapínat nebo vypínat zátěže, přepínačem simulátoru vyvolat výpadek sítě a její obnovení. Při konání těchto dějů jak simulátor, tak i aplikace poskytnou informace o krocích, které řídicí jednotka provede automaticky. Režim ***MAN*** nabízí možnost ručního ovládání a regulace pracovního stavu sítě a gen-setu a také stavu jejích stykačů.

Po splnění úlohy uživatel (student) získá obecné dovednosti o tom, jak funguje gen-set a jaké parametry jsou důležité pro jeho správný provoz; pochopí smysl synchronizace; dostane přehled o způsobech zajištěného napájení kritických spotřebičů v budově; nakonec, seznámí se s dostupnými výrobky na trhu v oblasti řízení zdrojů elektřiny.

Úplnou verzi laboratorní úlohy poskytuje příloha 6.7.

4. Shrnutí a závěr

Cílem diplomové práce bylo naprogramovat aplikaci, která by představovala počítačový model inteligentní budovy. Přes SNMP rozhraní byl tento model propojen s řídicí jednotkou *InteliSys*^{NTC} od firmy ComAp. Jednotka je vestavená do kufru s přepínači, relé a elektrickým potenciometrem (tzv. *Starter Kit*), který simuluje reálnou budovu se zdroji elektřiny a spotřebiči energie. Aplikace byla napsaná v jazyce C# ve vývojovém prostředí *Visual Studio* od Microsoft.

Celá práce je rozdělená na dvě velké části: teoretickou a praktickou. Teoretická část je věnovaná systémům řízení. V první polovině této části je rozebrán systém řízení budovy (*BMS*). V podkapitolách 2.1.1 a 2.1.2 definujeme tento pojem a seznamujeme se strukturou BMS a se základními cílí všech operací a procesů, prováděných v něm. V podkapitole 2.1.3 je čtenář stručně seznámen s problematikou výroby a distribuce elektrické energie (*Energy Management*). Nakonec v podkapitole 2.1.4 se hovoří o realizaci BMS v rámci Národní Technické Knihovny.

Druhá polovina teoretické části se zabývá otázkou řízení soustrojí motor-generátor (gen-set). Je vysvětleno z čeho se skládá gen-set, k čemu slouží a jaké důležité parametry by se měli u něho sledovat. Uvedl jsem podmínky napojení generátoru na síť a obecně popsal, jak funguje jeho řídicí jednotka při změně napájecích parametrů.

Značná část podkapitoly řízení gen-setu pojednává o řídicích jednotkách pro zdroje elektřiny firmy ComAp. Nejdřív je krátce popsán profil společnosti a seznam její výrobků. Další podkapitola 2.2.3 poskytuje informace o kontroléru pro řízení dieselagregátů *InteliSys*^{NTC}, s nímž se pracuje v praktické části. V ní jsem rozebral možnosti měření a vyhodnocení elektrických veličin, fyzické vstupy a výstupy jednotky a podporované komunikační protokoly včetně protokolu SNMP, použitému v praktické části. Velká část této podkapitoly je věnovaná vybraným funkcím kontroléru, týkajících se *Energy Managementu* a sloužících pro ošetření případných problémů spolupráce gen-setu (nebo skupiny gen-setů) a elektrických sítí. V podkapitole 2.2.3.4 jsem stručně popsal programový nástroj firmy ComAp *InteliMonitor*, sloužící k ovládání a monitoringu jejích řídicích jednotek. Poslední podkapitola 2.3 teoretické části uvádí simulátor budovy *Starter Kit*, do něhož je řídicí jednotka *InteliSys*^{NTC} vestavena.

Praktická část diplomové práce je rovněž rozdělená na dvě poloviny. První polovina (podkapitola 3.1) popisuje navrženou aplikaci. Nejdřív je probrán programovací jazyk aplikace C# v kontextu platformy. NET od společnosti Microsoft a typy aplikací, které tato platforma

umožňuje projektovat. Podkapitoly 3.1.2 a 3.1.3 se věnují obecné problematice designu programů a zvláště návrhovému vzoru MVVM, použitém v mé aplikaci. V podkapitole 3.1.4 je udělán krátký rozbor Open-Source knihovny *SharpSnmpLib*, která je použita k parsování MIB tabulky generované kontrolérem. Tabulka pak se používá k navázaní spojení přes SNMP mezi aplikací a simulátorem. Podkapitola 3.1.5 je nejdůležitější z hlediska provedené práce, protože popisuje vlastně strukturu celé aplikace a zodpovědnost jejích bloků (projektů). Nakonec v podkapitole 3.1.6 je představen krátký uživatelský manuál pro práci s aplikací.

Druhá polovina praktické části popisuje laboratorní úlohu a její možný přínos pro studenty. Plná verze této úlohy se nachází v příloze 6.7. Také ji lze najít v připojených souborech, stejně jako zdrojový kód aplikace a uživatelský manuál firmy ComAp pro řídicí jednotky *InteliGen*^{NTC}, *InteliSys*^{NTC} a simulátor *Starter Kit.*

Při vypracování praktické části diplomové práce, programování aplikace, jsem narazil na řadu problémů. Některé z nich vyplývaly z volby protokolu SNMP. Tento protokol není stoprocentně spolehlivý, protože vychází z UDP rozhraní. Proto například v aplikaci při okamžitém vypínání/zapínání více zátěží najednou docházelo občas ke ztrátě komunikačních paketů a několik binárních výstupů simulátoru zůstávalo nezměněnými. Zlepšit situaci by mohlo použití protokolu ModBus, který se y toho pohledu jeví jako spolehlivější. Další komplikaci tvoří neideálně optimalizovány paralelní běh vláken aplikace. Největší potíže mi dělalo současné obnovení hodnot binárních vstupů a výstupů, posílání povelů do řídicí jednotky a načítání alarmů. Příčinou nedokonalosti běhu bylo hlavně obtížné testování. Při případném použití výsledků této práce je nezbytné tyto problémy brát v úvahu.

Během práce nad diplomkou jsem si vyzkoušel spolupráci na reálnem projektu, který by měl najít další uplatnění. Jak už bylo zmíněno, vytvořena souprava *Starter Kit* + aplikace by se měla stát laboratorní úlohou, která by byla k dispozici studentům Katedrou měření Fakulty elektrotechnické ČVUT v Praze. Kromě toho, firma ComAp by mohla tuto naprogramovanou aplikaci použít na odborném veletrhu. Pro mě bylo přínosem mít příležitost seznámit se s hodně zajímavými lidmi ze společnosti ComAp, kteří mi poskytli spoustu nových informací, za což bych jim a mému vedoucímu p. Jiřičkovi chtěl poděkovat. Zároveň jsem se dozvěděl, jak probíhají vnitrofiremní procesy. Doufám, že na vylepšení aplikace a implementaci nových možnosti budu pokračovat i nadále.

Během psaní kódu aplikace jsem se naučil nejpoužívanějším technikám programování (design patterns). Také jsem si získal zkušenost s návrhem architektury větších aplikací, což v budoucnu nepochybně využiji.

5. Seznam literatury

1. **Herman M., Hansemann T., Hubner Ch.** *Automatizováné systémy budov.* Praha : Grada.

2. **Siemens.** Technická specifikace BMS DESIGO. [Online] 2012. www.buildingtechnologies.siemens.com.

3. *www.novinky.cz.* [Online]

4. **Phoenix-Zeppelin Slovensko.** Technická specifikace rotační UPS 250i. [Online] 2012. http://www.zeppelin.sk/public/data/cat_data/newmachine_types/2532/RUPS250i.pdf.

5. **Caterpillar.** 3412C Generator set. *Caterpillar.* [Online] www.cat.com.

6. **ComAp.** *IGS-NT Communication Guide.* 2013.

7. —. *IGS-NT Installation Guige.* 2014.

8. —. *IGS-NT SPtM Reference Guide.* Prague : ComAp, 2013.

9. **Douhlas R. Mauro, Kevin J. Schmidt.** *Essential SNMP.* místo neznámé : O'Reilly, 2005. ISBN: 0-596-00840-6.

10. **ComAp.** *Confluence-Komunikacni protokol SNMP.* 2014.

11. —. *Specifikace InteliGen, InteliSys, InteliMains.* 2014.

12. Intelimonitor. *Oficiální web firmy ComAp.* [Online] http://www.comap.cz/products/detail/intelimonitor/.

13. **ComAp.** *IGS-NT Starter Kit. Getting started.* 2008.

14. C sharp. *Wikipedia.* [Online] https://cs.wikipedia.org/wiki/C_Sharp.

15. **Microsoft.** *Application architecture design guide.* 2009.

16. —. Implementing the MVVM Pattern. *Microsoft Developer Network.* [Online] www.msdn.microsoft.com.

17. Introduction to sharpSNMP library. *CodeProject.* [Online] http://www.codeproject.com/Articles/468892/An-introduction-to-sharpSNMP-an-Open-Source-SNMP.

18. **EvelynT(nick).** Circular gauge custom control for Silverlight 3 and WPF. *www.codeproject.com.* [Online] 2009. http://www.codeproject.com/Articles/38361/Circular-gauge-custom-control-for-Silverlight-an.

19. **ComAp.** Intelimonitor. *Oficiální web firmy ComAp.* [Online] http://www.comap.cz/products/detail/intelimonitor/.

20. **Mauro D., Schmidt J.** *Essential SNMP.* Beijing : O'Reilly.

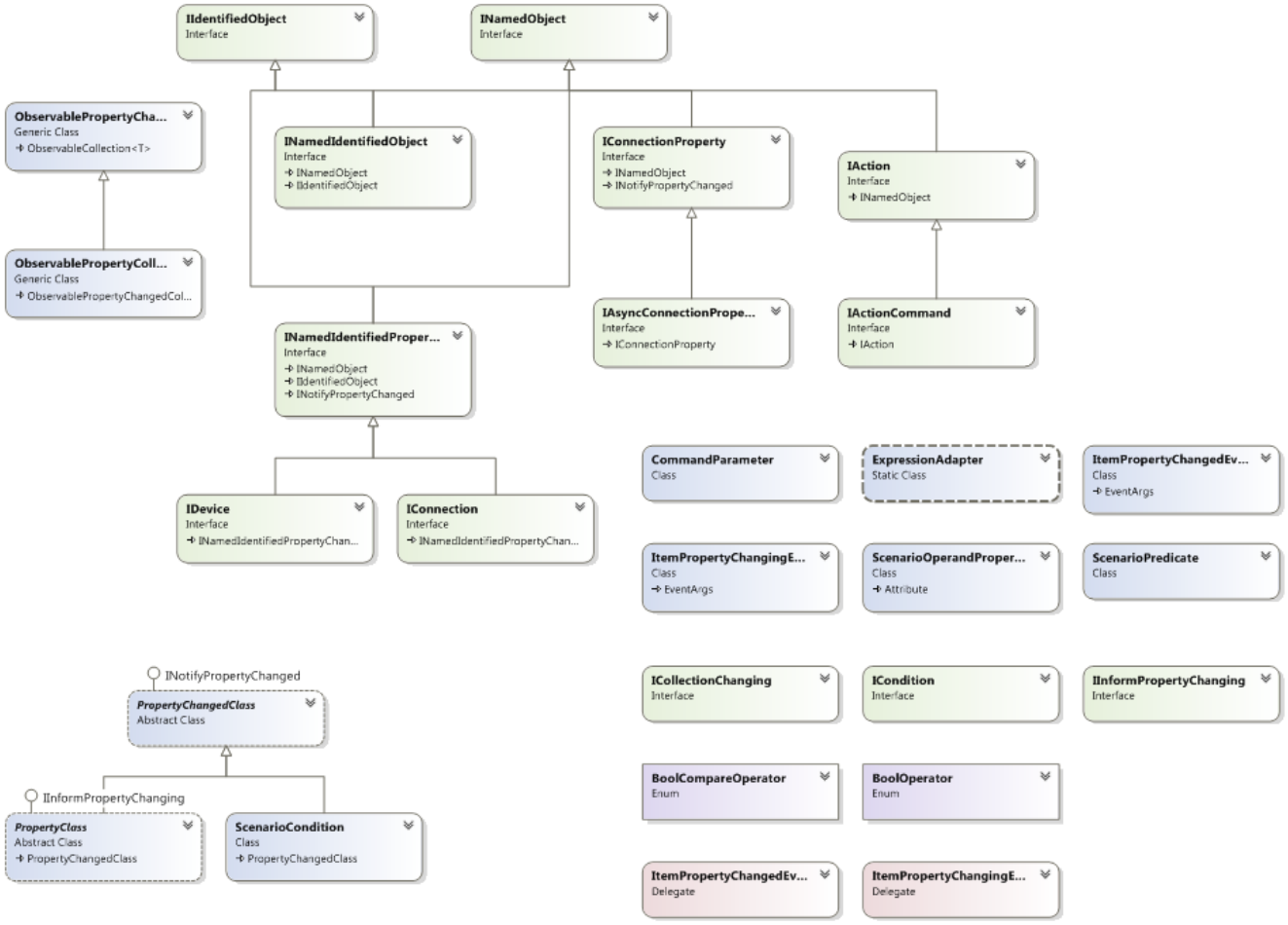
21. **ComAp.** *Specifikace InteliGen, InteliSys, InteliMains.* 2014.

22. **Siemens.** Technická specifikace BMS DESIGO. [Online] 2012. http://www.automatedbuildings.ru.

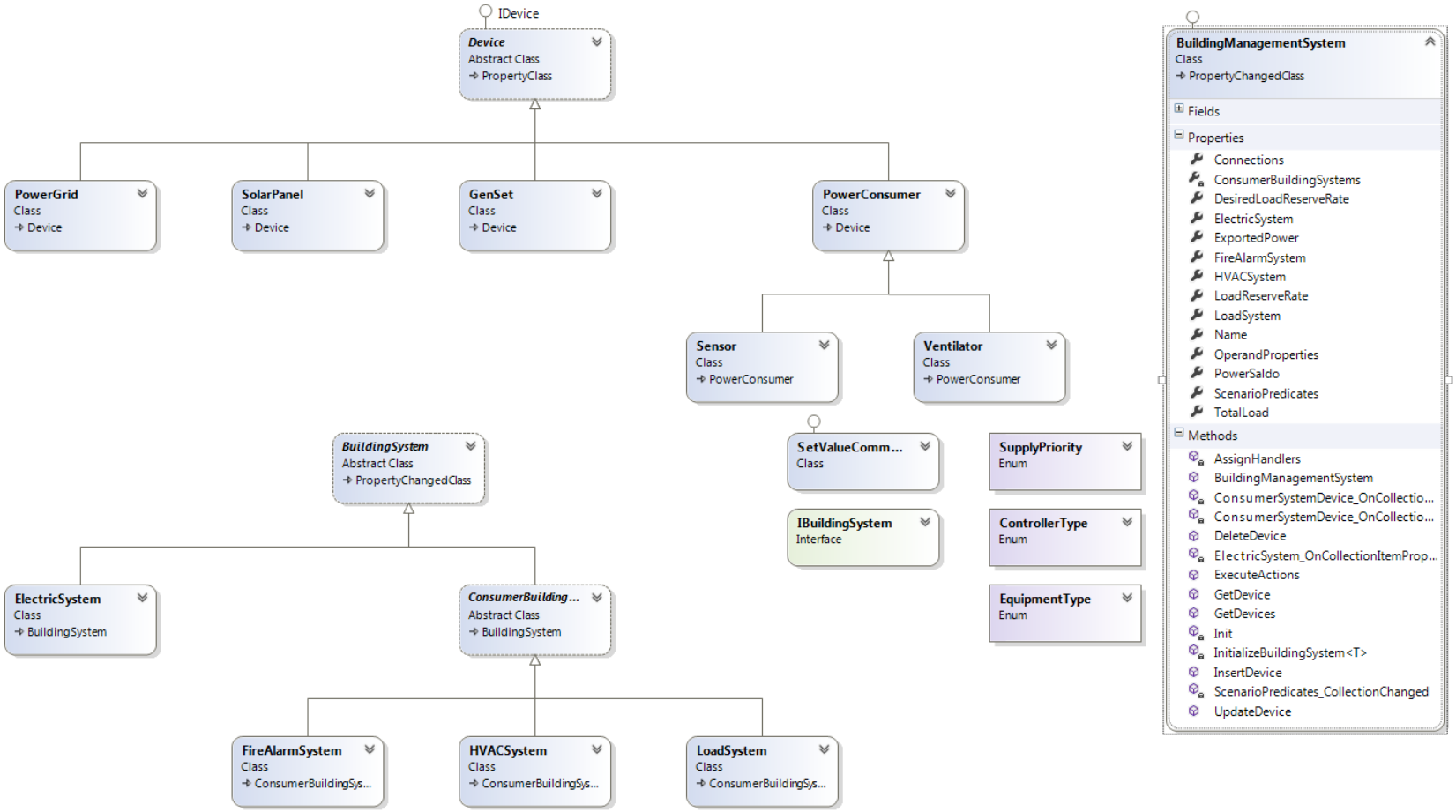
23. **Herrmann Merz, Thomas Hansemann, Christof Hubner.** *Automatizované systémy budov.* Havlíčkův brod : Grada Publishing, 2008. ISBN 978-80-247-2367-9.

6. Seznam příloh

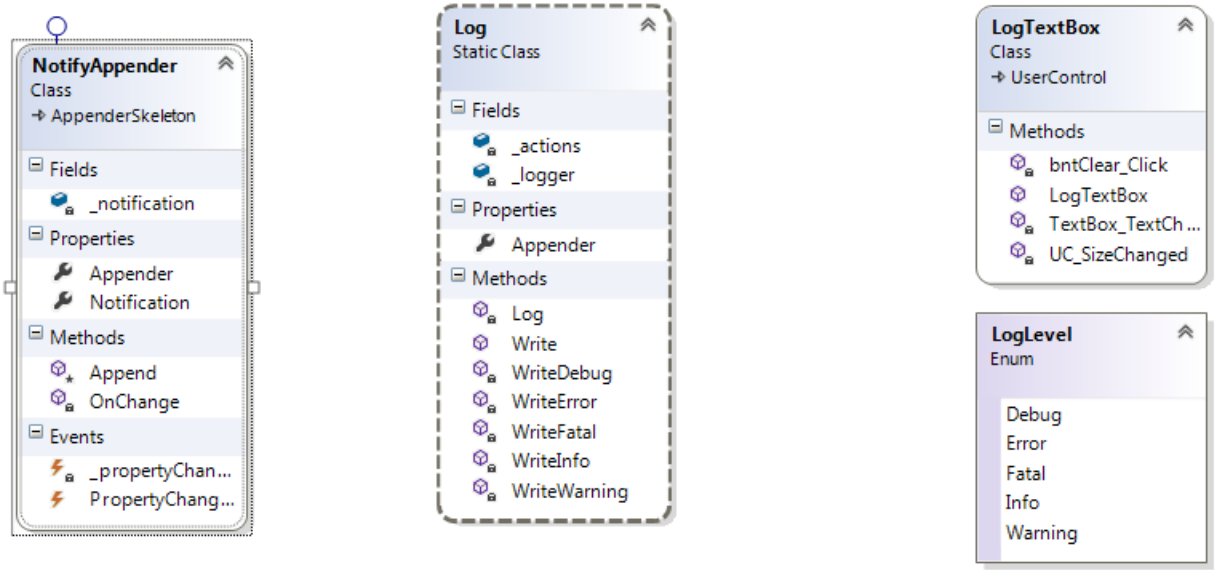
6.1 Příloha A. Třídový diagram projektu InterfaceLibrary



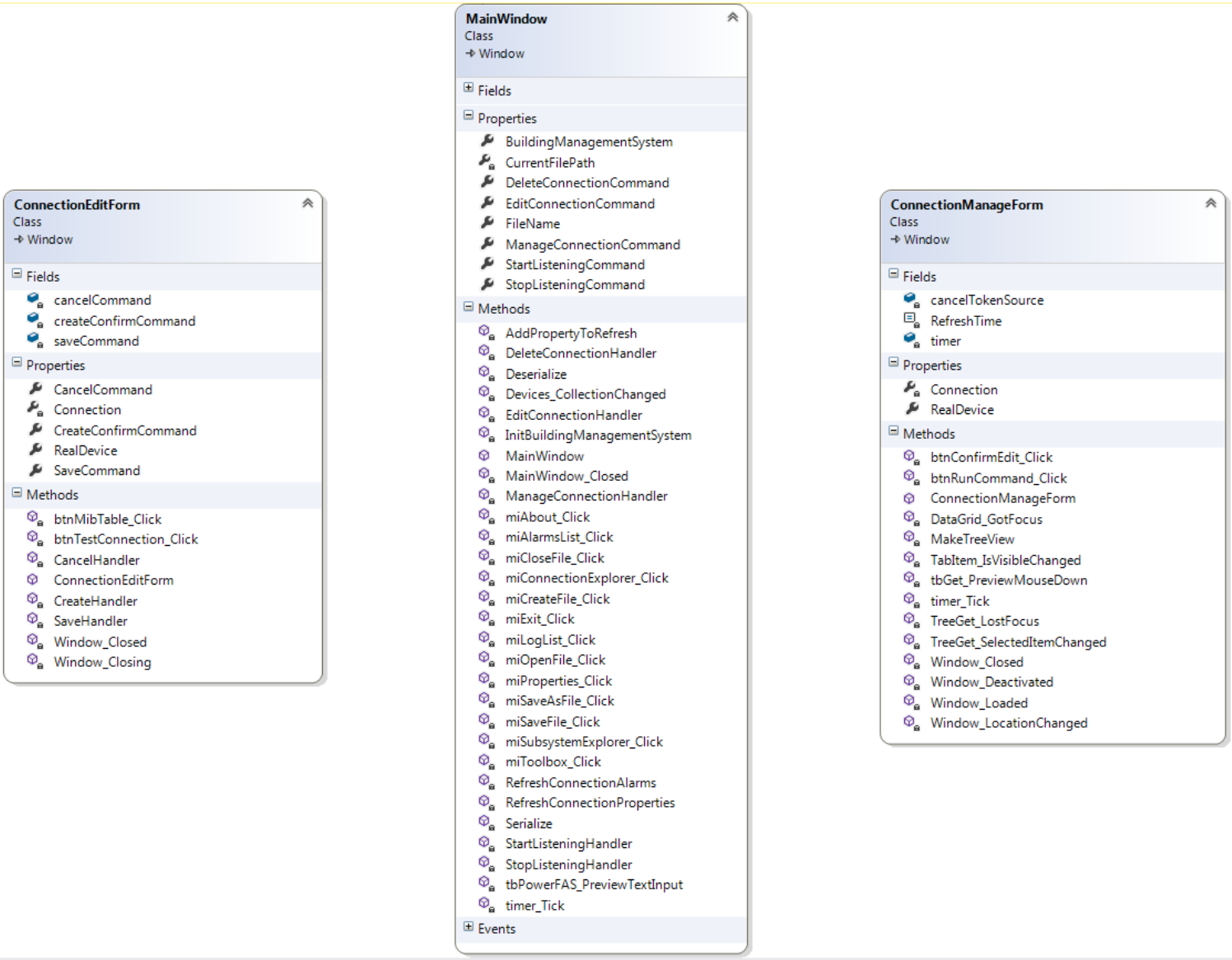
6.2 Příloha B. Třídový diagram projektu BMS



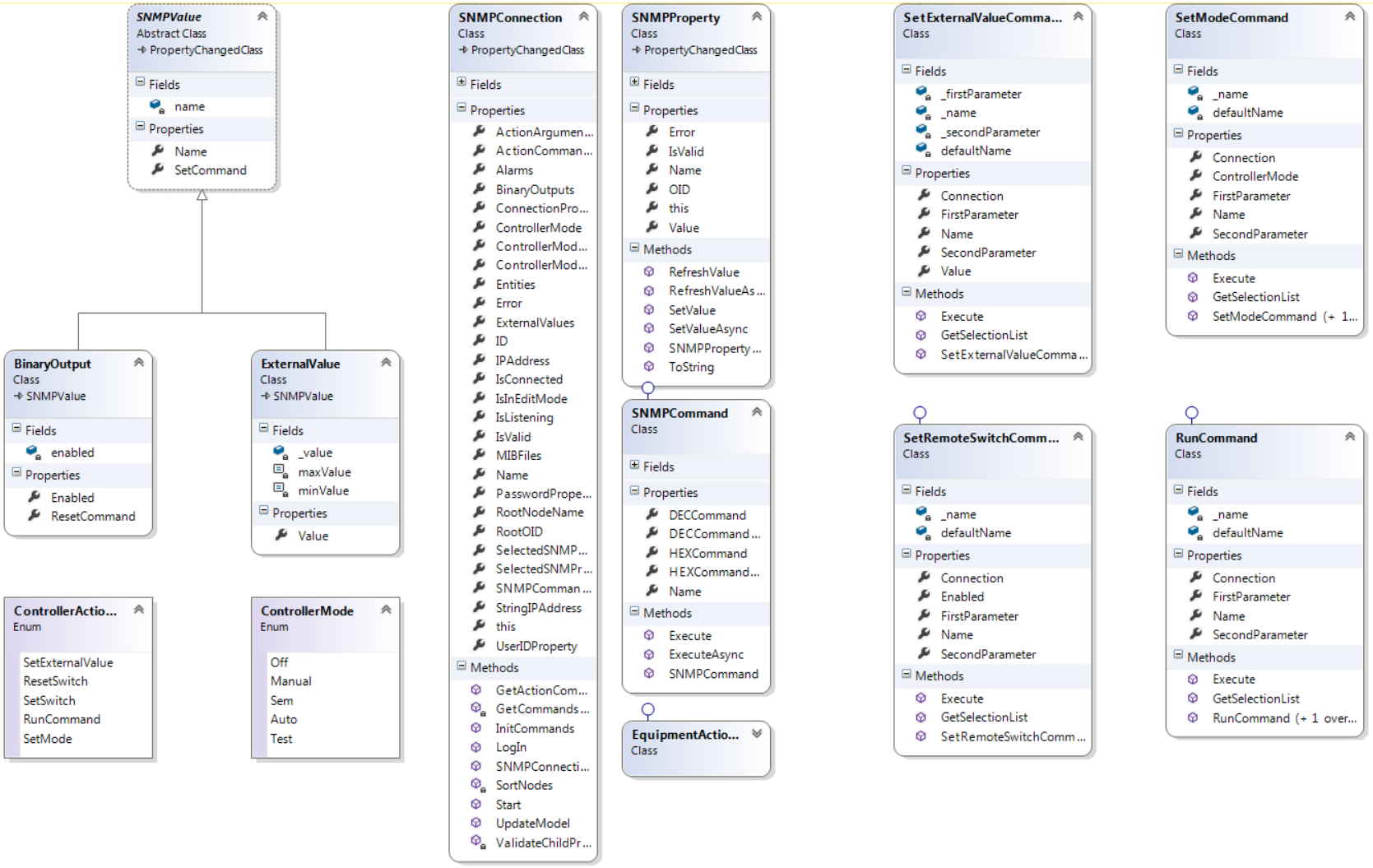
6.3 Příloha C. Třídový diagram projektu Logger



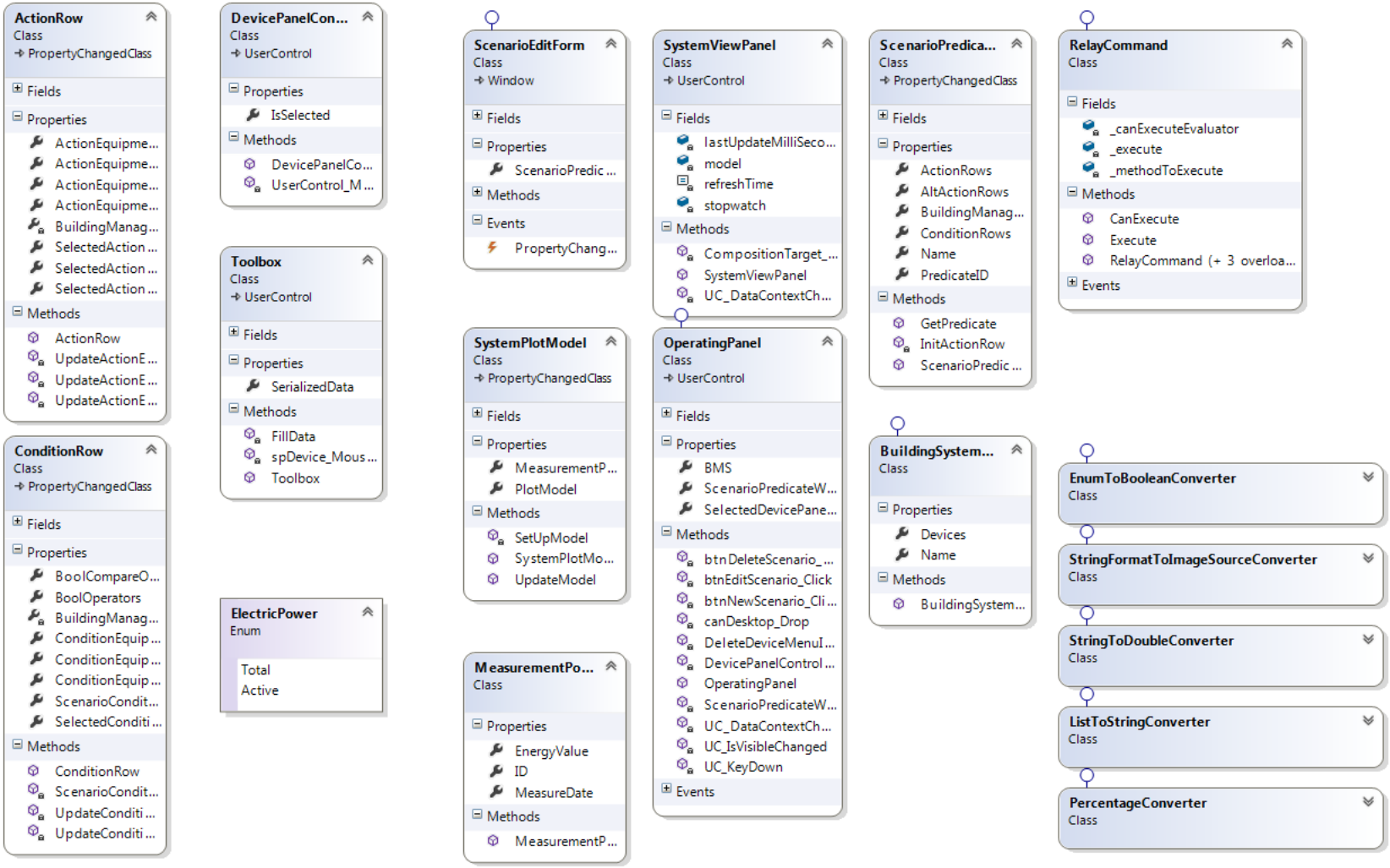
6.4 Příloha D. Třídový diagram projektu MainApplication



6.5 Příloha E. Třídový diagram projektu SNMP



6.6 Příloha F. Třídový diagram projektu UserControls



6.7 Příloha G. Laboratorní úloha

Demonstrace funkcí Energy Management na simulátoru budovy

Zadání úlohy:

1. Seznamit se se základními funkcemi simulátoru inteligentní budovy *Starter Kit* s vestaveným kontrolérem *InteliSys*^{NTC}
2. V aplikaci na počítači navázat komunikaci se simulátorem přes SNMP rozhraní
3. Otestovat funkce Energy Management při různých provozních režimech kontroléru *InteliSys*^{NTC}

Teoretický rozbor:

Úloha je navržena k tomu, aby student se mohl seznámit s tím, jak funguje napájecí systém v inteligentních účelových budovách. K tomu se využije speciální aplikace na počítači, který je spojen přes Ethernet kabel s řídicí jednotkou ***InteliSys***^{NTC} vestavenou do demonstračního kufru ***Starter Kit*** od české firmy ComAp a.s.

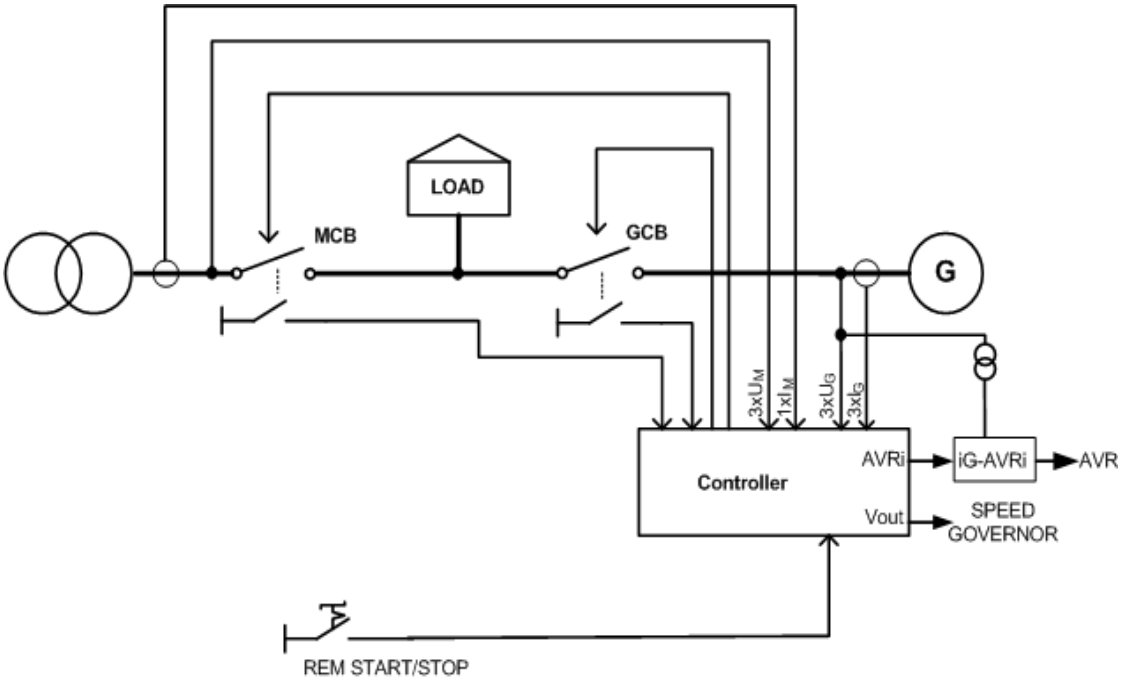
InteliSys^{NTC} představuje speciální kontrolér k monitorování a ovládání zdrojů elektrické energie a kogeneračních jednotek. Většinou takovým zdrojem je soustrojí spalovacího motoru a generátoru, který při pohánění vyrábí elektrickou energii. Toto soustrojí se nazývá ***gen-set***. Gen-sety se používají jako záložní zdroje v budovách, kde je důležité nepřetržité zásobování elektrickou energií. Mezi takové budovy lze zařadit letiště, nemocnice nebo vojenské objekty.

Starter Kit je simulátor operací gen-setu a sítě druhé generace, sloužící k testování funkcí kontroléru nebo k prezentačním účelům. V této úloze představuje systém napájení inteligentní budovy tvořený tvrdou sítí a záložním zdrojem (gen-set). Takovému typu aplikace se říká **SPtM** (*Single Parallel to Mains*). *Starter Kit* rovněž simuluje i elektro spotřebiče budovy (jištěné nebo nejištěné zátěže). Na simulátoru můžeme vyzkoušet následující funkce Energy Management:

- Synchronizaci generátoru a sítě
- Připojení a odpojení zátěže
- Výpadek sítě a obnovení provozu (funkce *Auto Mains Failure*)
- Řízení importovaného/exportovaného výkonu při paralelním běhu sítě a gen-setu (funkce *Load Control PtM (Parallel to Mains)*)

- Zapínání/vypínání gen-setu při nedostatku/přebytku výkonu z tvrdé sítě (funkce *Peak Shaving*)
- Hlídaní dostatečné výkonové rezervy (funkce *Power Management*)

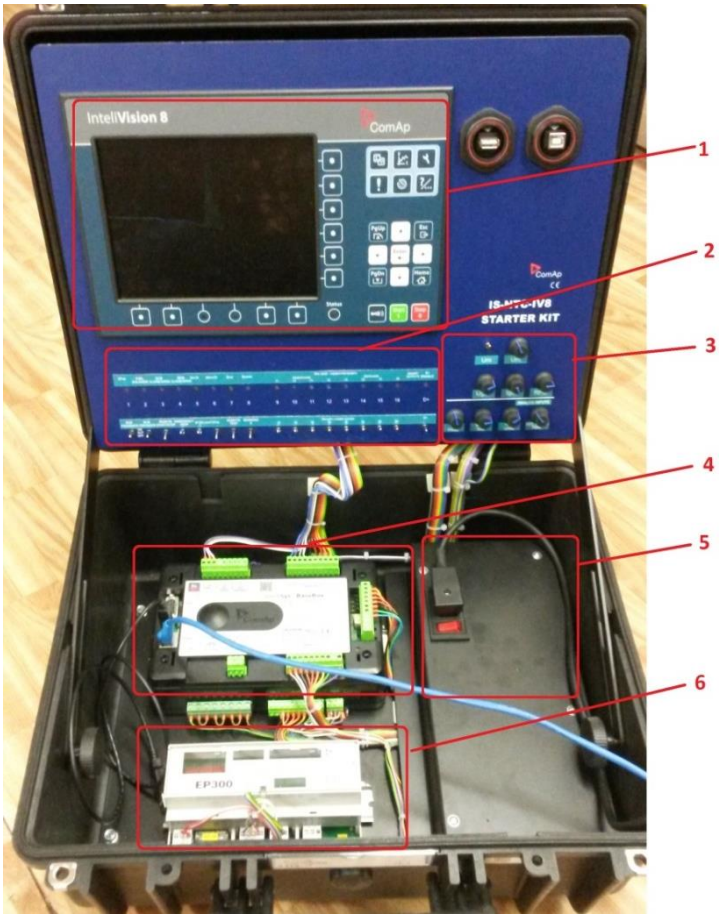
Simulátor má několik důležitých omezení. Zaprvé **pracujeme pouze s činnou zátěží**. Vestavený *InteliSys*^{NTC} kontrolér neřídí otáčky generátoru, proto se musejí nastavit ručně pomocí potenciometru RPM (viz Obrázek 3). Nominální hodnota otáček je **1500 ot/min**. Napětí a proud generátoru rovněž nastavíme potenciometry U_{gen} a I_{gen}. Síť není také řízená automaticky, proto pro řízení napětí použijeme potenciometr U_m. Hodnota napětí sítě a generátoru – **230 V**. Měření síťového proudu není podporováno. Pro simulaci výpadku (*blackout*) slouží přepínač U_m.



Obrázek 1. Schéma SPtM aplikace

Jak jíž bylo zmíněno, simulátor představuje napájení budovy typu SPtM (*Single Parallel to Mains*). SPtM obsahuje následující vlastnosti:

- Široce nastavitelný rozsah ochran gen-setu
- **Funkce provozu generátoru spolu se sítí**
- **Ostrovní režim (zátěž napájí jenom generátor, síť je vypnuta)**
- Dvojice stykačů – GCB (*Generator Circuit Breaker*) a MCB (*Mains Circuit Breaker*) s možností jak automatické, tak i ruční synchronizace
- Měkký náběh generátoru a zátěží



Obrázek 2. Starter Kit

Na obrázku Obrázek 2 jsou označené prvky tvořící simulátor:

1. Obrazovka *InteliVision 8*. Pomocí tlačítek na panelu lze prohlížet historii změn a hlášky, nastartovat/zastavit gen-set, měnit setpointy kontroléru.
2. Přístrojový panel. Přes ně ovládáme stykače jednotlivých zátěží, sítě a generátoru, povolujeme vzdálenou kontrolu generátoru a řídíme elektronický potenciometr 6. Podrobnější popis je na obrázku Obrázek 3.
3. Panel potenciometrů pro ovládání elektrických veličin generátoru a sítě, regulaci neelektrických veličin (otáčky, teplotu, množství paliva a oleje) generátoru. Podrobnější popis je na obrázku Obrázek 3.
4. Kontrolér *InteliSys*^{NTC}.
5. Blok napájení.
6. Elektronický potenciometr, který obsahuje PID regulátor. Slouží k postupnému zatížení gen-setu v ostrovním nebo paralelním režimu při připojení nebo odpojení zátěže.

Chování řídicí jednotky *InteliSys*^{NTC} závisí na její provozním režimu. Dostupné režimy práce:

1) OFF

d) výstupy *STARTER, GCB CLOSE/OPEN* and *FUEL SOLENOID* nejsou napájeny.

e) Není možné nastartovat gen-set

f) Není možné sepnout/rozepnout stykače MCB a GCB, avšak lze nastavit automatické chování MCB stykače při přepnutí do tohoto režimu

g) Pokud gen-set běží, do tohoto modu nelze se přepnout. Nejdřív gen-set se musí zastavit.

2) MAN

a) Při aktivaci binárního vstupu *START* kontrolér nastartuje gen-set

b) Při aktivaci binárního vstupu *STOP*

i) Ostrovní režim: otevře stykač GCB, začne chladit gen-set, pak ho vypne

ii) Paralelní režim: přehodí veškerou zátěž na síť, otevře GCB, začne chladit gen-set, pak ho vypne

iii) Pokud gen-set není zatížen: začne ho chladit a pak vypne

iv) Pokud gen-set se již chladí: okamžitě ho zastaví

c) Při aktivaci/deaktivaci binárního vstupu *GCB close/open*

d) Při překročení dovolených mezí napětí generátoru ŘJ nepovolí operací sepnutí/rozepnutí GCB

e) V případě synchronizace se síti kontrolér ověří, jestli síť je zdravá (výstup *MainsOK*) a stykač MCB je sepnut. Pak nastartuje gen-set a po splnění podmínek synchronizace připojí ho k silovému rozvodu. Další chování systému po synchronizaci lze definovat v nastavení setpointů karty **Process Control** *InteliSys*^{NTC}

f) Při požadavku na otevření stykače GCB v ostrovním režimu rozepne ho okamžitě. V paralelním režimu nejdřív odlehčí gen-set a potom rozepne stykač.

3) SEM

a) Při aktivaci binárního vstupu *START* se nastartuje gen-set, synchronizuje se sítí a poběží v paralelním režimu.

b) Při aktivaci binárního vstupu *STOP* řídicí jednotka měkce odlehčí gen-set, otevře GCB stykač, zapne chlazení motoru a potom vypne.

4) TEST

a) V tomto režimu gen-set se nastartuje automaticky

b) Pokud setpoint řídicí jednotky *Return To Mains=ENABLED*, pak vstupy *GCB ON/OFF , STOP, START* se ignorují

5) **AUT**

a) Tento režim provádí veškeré operace řízení toků elektřiny automaticky na základě přednastavení setpointů karty **Process Control** *InteliSys*^{NTC}

b) Pokud se stal výpadek:

   i) Rozepne se MCB a nastartuje gen-set. Pokud síť se vrátí za náběhu gen-setu, řídicí jednotka sepne MCB a vypne gen-set

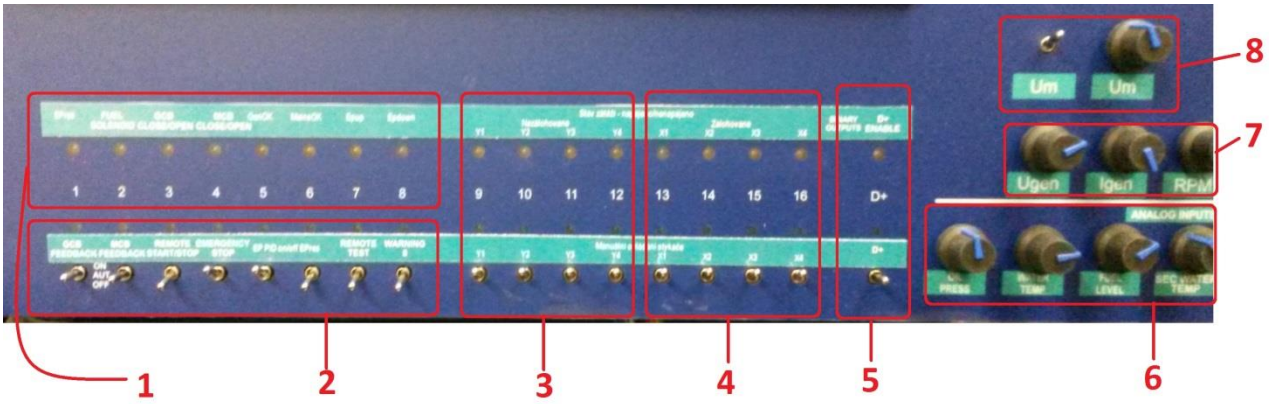
   ii) Počká se, až generátor se roztočí na nominální otáčky. Pokud hodnota napětí generátoru se nachází v povolených mezích, řídicí jednotka zavře stykač GCB. Pokud ne, vypne gen-set a nahlásí chybu

c) Pokud síť se vrátila:

   i) Provede se zpětná synchronizace s gen-setem, stykač MCB se zavře

   ii) Řídicí jednotka odlehčí gen-set a rozepne GCB stykač

   iii) Gen-set se vychladí a následně se zastaví



Obrázek 3. Panel přepínačů a potenciometrů Starter Kit

Obrázek 3 znázorňuje panel, přes který se provádí ruční regulaci simulátoru. LED diody nahoře patří k binárním výstupům, zatímco ty zelené dole jsou binárními vstupy kontroléru. Na obrázku jsou zobrazeny:

1) **Nakonfigurované binární výstupy:**
   (a) EP_{res} – resetování elektronického potenciometru
   (b) *Fuel Solenoid* – ukazuje stav palivové nádrže.
   (c) *GCB close/open* – pokud v log. 1, tak stykač generátoru je zavřen
   (d) *MCB close/open* – pokud v log. 1, tak stykač sítě je zavřen
   (e) *GenOk* – ukazuje, jestli generátor běží a jeho parametry jsou v pořádku.
   (f) *MainsOk* – ukazuje, jestli síť funguje a její parametry jsou v pořádku.
   (g) EP_{up} – zvýšení zatížení generátoru elektronickým potenciometrem.
   (h) EP_{down} – snížení zatížení generátoru elektronickým potenciometrem.
2) **Nakonfigurované binární vstupy:**
   (a) *GCB Feedback* – ovládá stykač generátoru. Může se nacházet v polohách ON, OFF, AUTO. V režimu AUTO stykač reguluje kontrolér. Výchozí stav je AUTO.
   (b) *MCB Feedback* – ovládá stykač sítě. Může se nacházet v polohách ON, OFF, AUTO. V režimu AUTO stykač reguluje kontrolér. Výchozí stav je AUTO.
   (c) *Remote Start/Stop* – nastavuje možnost dálkového ovládání generátoru. Výchozí stav je OFF.
   (d) *Emergency Stop* – okamžitě zastaví gen-set a vyvolá hlášku. Výchozí stav je ON.
   (e) E_{pid} *on/off* – vypíná nebo zapíná PID regulaci elektronický potenciometr se zachováním jeho původní hodnoty. Výchozí stav je ON.
   (f) EP_{res} – resetuje potenciometru a vynuluje jeho hodnotu. Výchozí stav je OFF.
   (g) *Remote Test* – povoluje vzdálené provedení periodických testů funkčnosti gen-setu. Výchozí stav je OFF.

(h) *Warning* – aktivace binárního vstupu vede k vyhlášení alarmu typu „Warning“. Výchozí stav je OFF.

3) **Nejištěné zátěže**. Dolní přepínače jednotlivých zátěží tvoří jeden ze dvou stykačů. Druhý je virtuální a zapíná se programově. Pokud jsou sepnuty oba stykače, zátěž je připojena (svítí žlutá LED nahoře). Výchozí stav je ON.
4) **Jištěné zátěže.** Dolní přepínače jednotlivých zátěží tvoří jeden ze dvou stykačů. Druhý je virtuální a zapíná se programově. Pokud jsou sepnuty oba stykače, zátěž je připojena (svítí žlutá LED nahoře). Výchozí stav je ON.
5) **D+** – přídavná detekce běhu gen-setu. V této úloze nebudeme tuto přídavnou funkci využívat. Výchozí stav je OFF.
6) **Ovládání neelektrických veličin gen-setu.**
7) **Ovládání elektrických veličin gen-setu.**
8) **Ovládání elektrických veličin sítě.**

Na obrazovce *InteliVision 8* pomocí tlačítek nahoru ↑ a dolu ↓ se dá přepínat mezi deseti měřícími displeji. Pracovat se ale bude pouze s dvěma, druhou a pátou. Display č. 2 zobrazuje stavy stykačů gen-setu a sítě, jejích okamžitý výkon dodaný do budovy a také hodnotu výkonu všech zátěží budovy. Display č. 5 představuje synchroskop, který se použije při ručním fázování generátoru na síť.

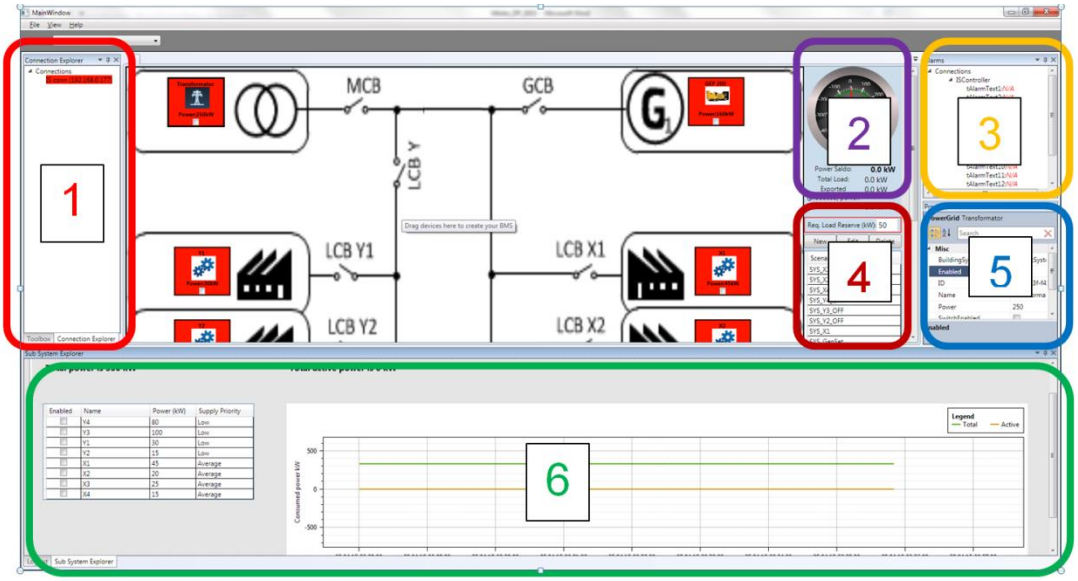
Display č.2

Display č.5

Simulátor *Starter Kit* bude propojen přes Ethernet kabel s počítačem, na kterém poběží speciální aplikace. Vazba mezi simulátorem a aplikací je oboustranná, tzn. všechny děje provedené se *Starter Kit* se zobrazí v aplikace i naopak, uživatel přes aplikační uživatelské rozhraní dokáže měnit chování simulátoru.

Postup:

1. Spustit „BMSComAp“
2. File → Open, vybrat soubor „**Lab_uloha_BMSComAp.bin**“
3. Objeví se panel (Obrázek 4)
4. Uprostřed panelu se nachází pracovní plocha s elementy budovy. Jsou to Trasformator, Gen-set, čtyři nezálohované zátěže (Y1 až Y4) a čtyři zálohované (X1 až X4). Pokud pozadí elementu je červené – element nefunguje. V dolní části každého elementu je umístěn checkbox, který je stykačem elementu se sběrnicí. **Uživatel by měl připínat/odepínat pouze zátěže, protože Transformátor a Gen-set jsou řízeny výstupy z kontroléru**.

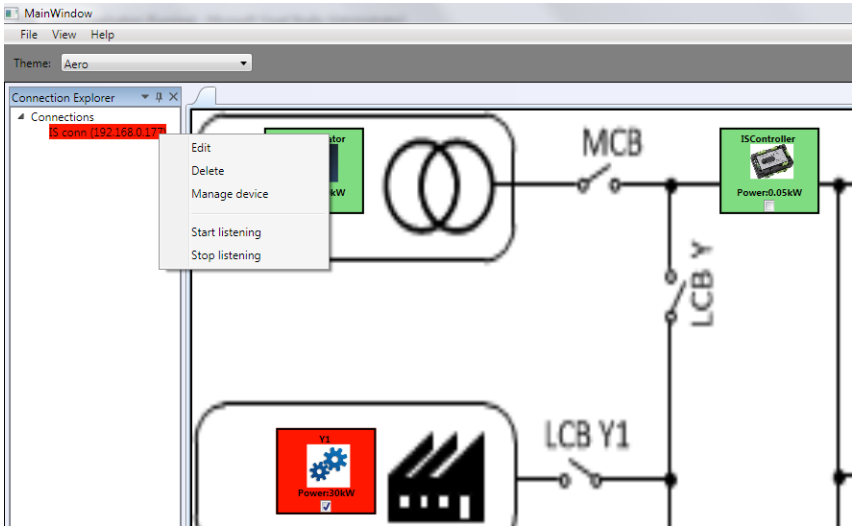


Obrázek 4. Operátorský panel

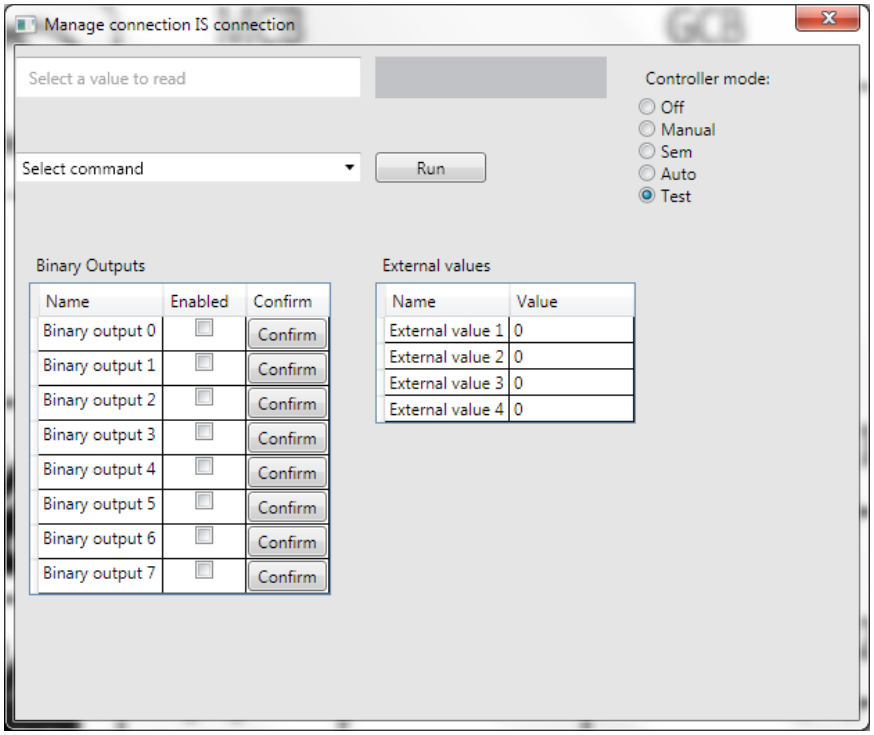
Číslo okna	Název	Popis
1.	Connection Explorer	Poskytuje seznam všech připojení. Připojení označené červeně nejsou aktivní.
2.	-	Okno hodnot toků elektřiny v budově. Tady lze určit minimální výkonovou rezervu budovy
3.	Alarms	Seznam hlášek aktivních připojení
4.	Scenarios	Seznam scénářů modelu budovy s možností přidávání, editace nebo mazání
5.	Property window	Při kliknutí myší na nějaké zařízení na pracovní ploše zobrazí jeho vlastnosti
6.	Sub systém explorer	Poskytuje informace o stavu jednotlivých podsystémů v budově. Jeho součástí je tabulka všech zařízení daného podsystému a graf celkového a aktivního výkonu

Tabulka 2. Popis pracovních oken aplikace

5. V okne č.1 (Connection Explorer) klikněme pravým na ***IS Connection***, zvolit “Manage Device” (Obrázek 5). Objeví se okno konfigurace kontroléru (Obrázek 6). Přes tohle okno lze měnit pracovní režimy řídicí jednotky *InteliSys*^{NTC}, avšak jen když připojení není aktivováno (červené pozadí). Preferovaným způsobem je měnit režim rovnou v simulátoru tlačítkem „Controller Mode“ na obrazovce *InteliVision 8.*



Obrázek 5



Obrázek 6

TESTOVACÍ REŽIM (TEST)

A. V simulátoru přepněme režim kontroléru na TEST (tlačítko „Controller mode“ na displeji *InteliVision 8*). Automaticky by se měl nastartovat gen-set.

B. Na obrazovce InteliVision 8 zvolíme měřící displej č. 5 se synchroskopem. Ověříme, jestli jsou splněny podmínky pro synchronizaci gen-setu a sítě (napětí, frekvence).

AUTOMATICKÝ REŽIM (AUT)

A. V okně „Manage connection“ přepnout do automatického režimu. Ověřit jestli se změnil režim na obrazovce simulátoru.

B. Kliknout pravým na připojení a zvolit „Start Listening“ (Obrázek 5)

C. Počkat až se na pracovní ploše operátorského panelu prvek „Transformator“ rozsvítí zeleným a jeho stykač se sepne. Postupně (ne hned za sebou) zapneme stykače zátěží X1,X2,X3,X4 a Y2. Ověříme, jestli příslušné výstupy simulátoru svítí, a na měřícím displeji č.2 jsou odpovídající hodnoty výkonu zátěže a transformátoru (Mains) .

D. Vyvoláme výpadek sítě (přepínač ***Um*** dolu, viz Obrázek 3). Po několika vteřinách se automatický nastartuje gen-set, který napájí pouze zálohované zátěže. K přetížení gen-setu nedojde, protože jeho výkon (160 kW) je větší než celkový výkon zálohované zátěže (105 kW). Před zahájením dalšího kroku odstraníme alarmy tlačítkem „Fault reset“ na obrazovce *InteliVision 8.*

E. Obnovíme napájení ze sítě (přepínač ***Um*** nahoru, viz Obrázek 3). V simulátoru se přepneme tlačítkem ↑ nebo ↓ na displej č. 5 se synchroskopem. Zkontrolujeme, jestli hodnoty frekvence a napětí gen-setu a sítě jsou shodné. Cca po minutě proběhne fázování na síť tzv. „***reverse synchronization***“ přes MCB stykač. Na synchroskopu můžeme si všimnout, že tento děj se může stát pouze když synchronizační úhel mezi gen-setem a sítí je v dovolených mezích (šipka v zelené zóně).

F. Po nafázování gen-set se odpojí, protože hodnota zátěže (105 kW) je menší než hodnota při které je nutno gen-set zapnout (*Peak Shaving = 160 kW*)

G. Připneme zátěže Y1a Y2. Suma zátěží bude větší než hodnota *Peak Shaving* (165 kW > 160 kW). Funkce se aktivuje a nastartuje gen-set a následně připne ho k síti. Tomuto procesu se říká „***forward synchronization***“ přes GCB stykač. Postupně připojíme další zátěže Y3 a Y4.

H. Vyvoláme poruchu gen-setu (přepínač ***Emergency stop*** dolu, viz Obrázek 3). Celkový výkon všech zátěží by měl klesnout pod hodnotu rozdílu výkonu sítě a požadované výkonové rezervy (*Required Load Rate*). Vrátíme přepínač do výchozí polohy. Před zahájením dalšího kroku odstraníme alarmy tlačítkem „Fault reset“ na obrazovce *InteliVision 8.*

I. Postupně odpojíme zátěže Y1, Y3 a Y4. Poněvadž výkon zátěže je menší než hodnota *Peak Shaving* (130 kW<160 kW). Funkce se deaktivuje, gen-set se odlehčí, odepne a pak se zastaví.

J. Vyzkoušíme dva způsoby zásobování budovy elektřinou v paralelním režimu (gen-set a síť běží): **BaseLoad** a **Import/Export**.

   a. Klikněme na prvek gen-set (GEP 200), aby se zvýraznil. Pak v okně „Properties“ (okno č.5 na Obrázek 4) zaškrtněme checkbox *IsBaseLoad.* Postupně připojíme všechny zátěže a potom postupně je také budeme odpojovat než kontrolér na základě funkce *Peak Shaving* neodepne gen-set. Na měřícím displeji 2 v simulátoru uvidíme, že při jakémkoliv zatížení výkon dodávaný gen-setem se nemění. Pokud zátěž je menší než výkon gen-setu, část výkonu gen-setu se exportuje do sítě. Pokud je zátěž větší, tak výkon, který nepokryje gen-set, se importuje ze sítě.

   b. Pokud v okně „Properties“ gen-setu odškrtněme checkbox *IsBaseLoad*, tak potom síť bude dodávat do budovy konstantní výkon (100 kW), zatímco výkon gen-setu se bude měnit podle velikosti zátěže. Tomuto režimu říkáme **Import/Export**. Změny výkonu zase pozorujeme na displeji 2. Stejně jako v bodě *a.* změníme velikost zátěže. Udržování konstantního importovaného výkonu ze sítě má z pohledu celkového okamžitého výkonu zátěže omezení „zdola“ a „ze vzhoru“. Omezení „zdola“ spočívá v tom, že při paralelním běhu gen-set musí být minimálně zatížen (výchozí hodnota – 50 kW). Minimální zatížení určuje mez, pod kterou výkon gen-setu by neměl klesnout kvůli možnému poškození spalovacího motoru. Pokud žátěž poklesne pod 150 kW (100 kW import + 50 kW min. zatížení), importovat se bude méně, ale výkon gen-setu bude stále konstantní. Omezení „ze vzhoru“ znamená, že pokud zátěž bude větší než 260 kW (100 kW import + 160 kW výkon gen-setu), tak potom budeme importovat více výkonu ze sítě, protože gen-set bude plně zatížen.

MANUÁLNÍ REŽIM (MAN)

A. V ručním režimu vyvoláme výpadek sítě (přepínač ***Um*** dolu, viz Obrázek 3). Nastartujeme gen-set (zelené tlačítko „Start“). Počkat až se stabilizují otáčky a napětí, poté ručně otevřeme MCB stykač (tlačítko „Open MCB“ na displeji) zapneme GCB stykač (tlačítko „Close GCB“ na displeji).

B. Vrátíme síť zpátky (přepínač ***Um*** nahoru, viz Obrázek 3). Protože jsme v manuálním režimu, stykač sítě se nesepne jako v automatickém režimu. Gen-set zbytečně pálí naftu. Přepneme se na display č. 5 se synchroskopem a manuálně zavřeme síťový stykač (tlačítko „Close MCB“ na displeji). Na synchroskopu uvidíme, kdy proběhla synchronizace (šipka v zelené zóně). Poté pošleme požadavek na otevření GCB stykače (tlačítko „Open GCB“ na displeji). Nakonec zastavíme gen-set červeným tlačítkem „Stop“.