ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra řídicí techniky

BAKALÁŘSKÁ PRÁCE

Vlastnosti TABS (Thermally Activated Building System)

květen 2013

Vypracoval: Petr Štěpán
Vedoucí práce: Doc.Ing. Lukáš Ferkl, Ph.D.

Anotace

Tepelně aktivní stavební systém je systém založený na chlazení či ohřívání betonových desek pomocí vody, která koluje v potrubí zabudovaných přímo do betonových desek. Systém TABS má mnoho výhod. Tento systém maximálně využívá energii určenou pro vytápění či chlazení a tudíž zajišťuje patřičný komfort a šetření energie. Systém je zatížen pomalou odezvou, která způsobuje řídicí problémy. Tyto problémy jsou řešeny pomocí prediktivního řízení, které je pokročilou technikou určenou pro regulaci pomalých systémů.
Tato práce je tedy zaměřena jak na teoretický popis systému TABS, tak na návrh konstrukce a řízení pomocí PLC (Programmable Logic Controler).

Annotation

Thermally Activated Building System (TABS) are water based heating or cooling system which consist of pipes embedded in the concrete slab. System TABS has a lot of advantages. This system maximally uses energy intended for heating or cooling and therefore ensures appropriate komfort and saving energy. System is loaded by slow response which causes control problematic. This problems are solved using preditive control which is advanced control method intended for slow systems.

This work is focused for teoretical description TABS systems and proposal of construct and control by PLC (Programmable Logic Controler)

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Petr Štěpán**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Vlastnosti TABS (Thermally Activated Building System)**

Pokyny pro vypracování:

1. Seznamte se s hardwarovou PLC platformou Domat a s vývojovým prostředím RcWare SoftPLC.
2. Navrhněte experimentální zapojení, které bude testovat tepelné vlastnosti prvku TABS (Thermally Activated Building System), tj. jak hydraulický obvod, tak související řídicí systém.
3. Hydraulický obvod realizujte a naprogramujte ovládání a řízení systému.
4. Hotový vzorek připravte pro napojení do prefabrikátu TABS a připravte k převozu do testovací laboratoře (vzorek a prefabrikát budou zvlášť převezeny k testování v laboratoři DTU, Lyngby, Dánsko).

Seznam odborné literatury:

[1] Prívara, Široký, Ferkl, Cigler: Model predictive control of a building heating system: The first experience. Energy and Buildings, vol. 43, pp. 564-572. 2012
[2] Technická dokumentace systému Domat
[3] Technická dokumentace prostředí RcWare

Vedoucí: Doc.Ing. Lukáš Ferkl, Ph.D.

Platnost zadání: do konce zimního semestru 2013/2014

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 26. 3. 2013

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Kladně dne 22.5.2013 podpis Petr Štěpán

Poděkování

Děkuji vedoucímu práce Doc.Ing. Lukáši Ferklovi, Ph.D za trpělivost a rady, které mi pomohli rozšířit znalosti v dané problematice.
Dále děkuji Ing. Janu Širokému a Ing. Filipu Petrovičovi z ENERGOCENTRA Plus s.r.o za ochotu, rady a pomoc s danou technologií, která mi umožnila hlubší pochopení a zpracování tématu této práce.

Obsah

1 Úvod

Spotřeba energie komerčních budov ve vyspělých zemích se pohybuje kolem 20 až 40% celkové spotřeby energie. Tato hodnota se průměrně ročně zvyšuje o 0,5 - 4%. Z těchto čísel vystupují obavy o spotřebu fosilních paliv a o ochranu životního prostředí. Z obavy vyčerpání fosilních paliv roste zájem o nízkoenergetické budovy.

Koncept vyvinutý v Nizozemsku v roce 1996 zavedl 3 pravidla, která pomáhají snížit spotřebu energie a závislost budov na spalování fosilních paliv. Tato pravidla jsou [1]:

- Lepší využití energie a snížení poptávky po energii.
- Maximální využití obnovitelných zdrojů energie.
- Efektivní využití energie z fosilních paliv.

Z prvního požadavku se začaly vyvíjet budovy, které měly zvýšenou tepelnou rezistanci stěn, oken a budovy byly orientovány na stranu, kde byl největší sluneční svit.
Postupně se nároky na úsporu energie začaly zvyšovat a to vedlo k zaměření na úsporu ve vytápění budov. Tyto důvody vedly k rozvoji TABS systémů, které pomocí pokročilé regulace dokáží ušetřit 15 až 30% energie ročně.
Právě tyto hodnoty vedly k zahájení experimentu ohledně chování TABS systému, neboť se tyto údaje neshodovaly s hodnotami naměřenými na modelu a bylo potřeba daný model upravit tak, aby odpovídal realitě. A k tomu nám dopomůžou odměřené výsledky na skutečném TABS panelu.

2 Systémy TABS

Jedná se o systém vytápění či chlazení budov pomocí betonových desek. Do desek jsou přímo instalována vodní potrubí, ve kterých proudí voda ohřátá na požadovanou teplotu. Díky proudění kapaliny je deska ohřáta a dochází k tepelnému vyzařováni, pomocí něhož lze v místnosti nastavit požadovanou teplotu.
Tyto systémy můžeme také nazývat jako zářiče, jelikož 50 a více procent energie je vyzářeno do okolí. Schopnost uchovávání energie v desce v podobě tepla je také ovlivněna tloušťkou vrstvy betonu. Mohutné betonové desky dokáží udržet teplo/chlad po dlouhou dobu, ale nejsou schopny rychle reagovat na potřebné změny při regulaci, naopak tenčí desky reagují na změny teplot rychleji, ale je potřeba častěji dodávat energii. Protože jsou tyto systémy velmi masivní a s obrovskou tepelnou kapacitou, tak mají pomalou dynamickou odezvu.

2.1 Typy TABS

TABS může být členěn podle typu panelu, který má být aktivním prvkem.

Podlaha jako aktivní prvek

Trubky jsou umístěny vedle sebe v podlaze. Hloubka umístění trubek je v rozmezí od 40 do 100 mm. Tato hodnota se pohybuje v závislosti na šířce betonu. Vzdálenost mezi trubkami, vzhledem k jejich středu, je v rozmezí 100 až 300 mm. Rozmístění trubek je důležité, neboť může docházet k nerovnoměrnému ohřevu povrchu betonu. Dále je pod deskou použita vrstva tepelné izolace k zajištění co nejmenších tepelných ztrát přes zadní část desky.

Obrázek 1: Podlaha jako aktivní prvek - rozmístění [1]

Strop jako aktivní prvek

Trubky jsou podobně jako v podlaze umístěny do betonu v požadovaných vzdálenostech. Tepelná izolace je zde použita mezi střechou a betonem ze stejného důvodu jako u podlahy, tedy kvůli minimalizaci tepelných ztrát. Hloubka položení trubek se zde většinou pohybuje od 100 do 200 mm.

Obrázek 2: Strop jako aktivní prvek - rozmístění [1]

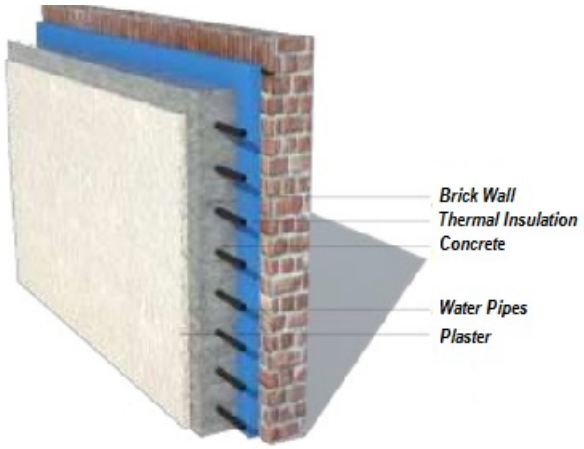
Mezi-podlažní aktivace

Mezi patry se používají dva typy vytápění. První typ je založen na vyhřívání pouze stropu nebo podlahy, druhý typ je smíšený, tedy že se zároveň vytápí strop spodního patra a podlaha horního patra a mezi jednotlivými kusy betonu je položena akustická ochrana. Jedná se v podstatě o složení stropní a podlažní aktivace dohromady, oddělené akustickou či jinou tepelnou izolací.

Obrázek 3: Mezi-podlaží jako aktivní prvek - rozmístění [1]

Stěna jako aktivní prvek

Tato část pracuje na stejném principu jako předchozí části.



Obrázek 4: Zeď jako aktivní prvek - rozmístění [1]

2.2 Provozní hodnoty

Provozní teploty povrchu betonových desek záleží na typu provozu. Při chlazení se teplota může pohybovat například v rozmezí 5 °C - 16 °C, naopak při ohřívání se teplota může vyšplhat až na hodnotu 90 °C.

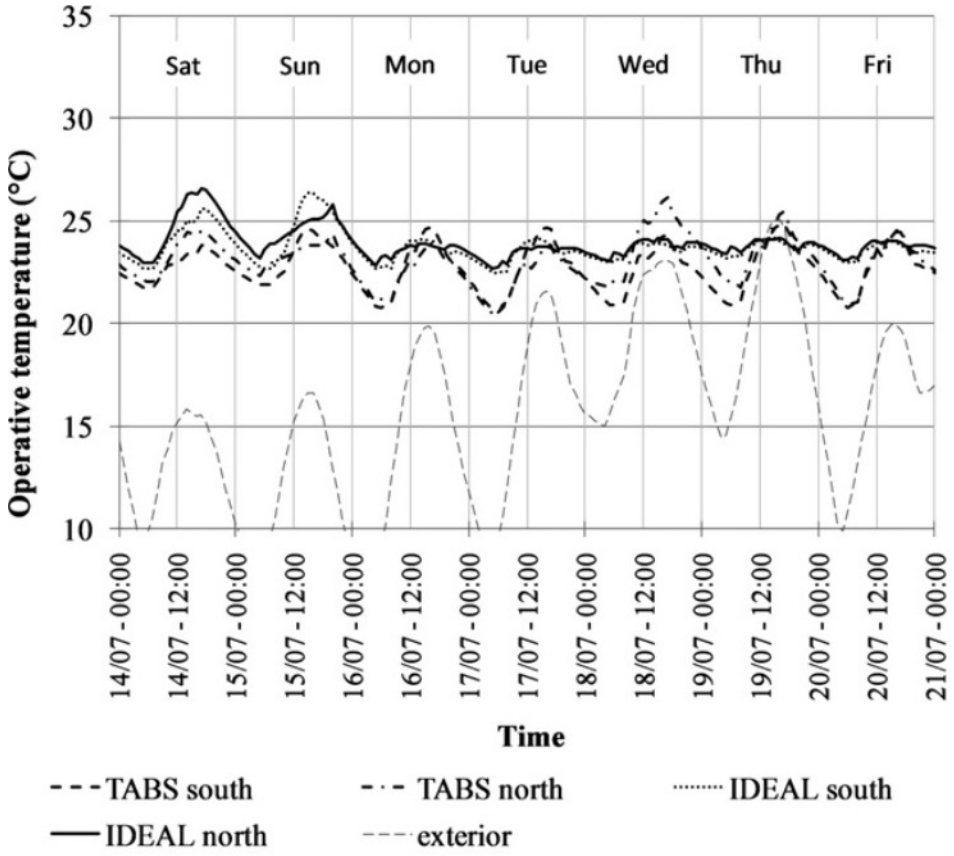
Nastavení povrchových teplot je velmi důležité z hlediska komfortu. Při oteplování či chlazení místnosti je možné, že budou mít jednotlivé povrchy betonových desek (strop, stěna, podlaha) velkou rozdílnou teplotu vůči okolí. Měření prokázala, že pokud je teplota podlahy o 10 °C nižší než teplota místnosti, tak je vertikální teplotní rozdíl mezi hlavou a chodily větší než komfortní limit 3 °C [9].

Tento limit je samozřejmě upraven pro lidi nosící obuv. Běžná povrchová teplota podlahy se pohybuje v rozmezí 19 °C - 29 °C. Komfortní teplota v místnosti se odhaduje na 28 °C [9], ale záleží samozřejmě na mnoha okolnostech, podle kterých se teplota upravuje. Obvyklé provozní teploty pro jednotlivé části jsou uvedeny v tabulce 1.

Regulace požadované teploty v místnosti nezáleží pouze na teplotě proudící vody. Do počtu musí být také započítána ventilace, účinnost výměníku, světová strana na kterou je budova orientována nebo čas spuštění oběhu vody.
Příkladem může být dynamická simulace kancelářských budov programu TRNSYS (Transient System Simulation Program, SEL. Madison), provedená v USA roku 2006 [2].

Jednalo se o ochlazování kanceláří v budově. Požadováná výsledná teplota byla 17 °C. Z hygienických důvodů byla použita ventilace vzduchu s koeficientem $50m^3/h$, doprovázená tepelným výměníkem se 70% účinností. Během pracovních hodin (7:00 - 20:00) byla spuštěna ventilace stejně tak jako chlazení či ohřev. Chlazení bylo také spuštěno každou noc před pracovním dnem v hodinách od 21:00 do 7:00. Pro ilustraci je na obrázku 5 uvedena provozní teplota vody proudící v trubkách.

Chlazení je spuštěno hlavně v noci z důvodu velké setrvačnosti systému.



Obrázek 5: Provozní teplota vody v TABS v programu TRNSYS [2]

	Maximum	**Minimum**
Podlaha	29 °C	19 °C
Strop	28 °C	17 °C
Zdi	40 °C	17 °C

Tabulka 1: Provozní teploty [1]

2.3 Řízení TABS

Celkový tepelný energetický tok lze vypočítat jako součin hmotnostního průtoku vody a rozdílem teplot mezi přívodem a zpátečkou:

$$W_{TABS} = \dot{m} c_p \Delta T \tag{1}$$

Jednotlivé veličiny vyjadřují:

W_{TABS} -Tepelný zářivý tok dodaný nebo přijatý do/z vody. $[W]$
$\dot{m}$ - Hmotnostní průtok vody. $\left[\frac{kg}{s}\right]$
c_p - Tepelná kapacita vody (4.18×10^3). $\left[\frac{J}{kg \cdot K}\right]$
ΔT - teplotní rozdíl mezi přívodem a zpátečkou. $[K]$

Z rovnice (1) vyplývá, že pro řízení velikosti tepelného vyzařování musíme řídit proměnné $\dot{m}$ a ΔT. Člen ΔT obsahuje teplotu přívodu a zpátečky:

$$\Delta T = (T_p - T_z)\,[K]$$

Kde:
T_p - Teplota přívodu $[K]$
T_z - Teplota zpátečky$[K]$

Z předchozího vztahu se bere jako kontrolní proměnná pouze teplota T_p. Teplota přívodu T_p a hmotnostní průtok vody $\dot{m}$ jsou dvě kontrolní proměnné. TABS lze tedy řídit pomocí těchto dvou proměnných za předpokladu nízko úrovňového řízení.
Na druhou stranu, teplota vratné vody je ovlivněna účinností tepelné výměny mezi vodou a deskou a to záleží na teplotě desky a hloubce trubek. Z pohledu betonové desky je teplota závislá na teplotě vody a teplotě místnosti. Příkladem může být vytápění: Při zvýšení teploty v místnosti se zvýší i teplota desky a zpátečky a to má za následek snížení potenciálu W_{TABS}. Tudíž regulace teploty zpátečky T_z by mohla být také účinná.
Nízko-úrovňové řízení lze dosáhnout za pomoci směšovacích ventilů či regulace otáček tepelných zdrojů nebo pulsně šířkové module (PWM) z oběhových čerpadel.

Při vyšší úrovni řízení se používají regulátory. Lze použít následující typy řízení:

- **On-Off regulace** - Jedná se o jednoduchou zpětnovazební regulaci, kde nevíme nic o dynamice systému. Vytápěcí systém je zapnut či vypnut v závislosti na tzv. teplotní chybě. Chyba je dána vztahem: $e = (t_{pozadovana} - t_{mistnosti})$. Tato regulace je většinou implementována pomocí hystereze. Regulace má dva stavy $S = (On, Off)$. Jednotlivé stavy jsou aktivní v závislosti na hysterezi a na chybě : $S = f(e)$. Výhoda spočívá v jednoduchosti.
- **Přímovazební regulace** - Typ řízení, který opět nenese žádnou informaci ohledně dynamiky systému. Teplota vody je nastavena v závislosti na venkovní teplotě. Teplota vody je tedy funkcí venkovní teploty $t_{water} = f(t_{out})$. Výhoda toho řízení je robustnost a jednoduché ladění.
- **Zpětnovazební řízení** - Pro tento typ řízení se používají jednotlivé typy regulátorů (PID, PI, P). Toto řízení nese informaci ohledně dynamiky systému a nějakým způsobem vyhodnocuje teplotu v místnosti v závislosti na teplotní chybě e a historii, $t_{water} = f_{PI}(e, history)$. Jedná se opět o robustní řízení, ale nedokáže započítat vliv venkovní teploty. Často se tento typ řízení kombinuje s přímovazebním řízením a jsou tak dosaženy optimální výsledky.

Studie řízení ukazují, že nejlepší energetický výkon a komfort je zajištěn řízením teploty vody jako funkcí venkovní teploty. Tedy přímovazebním řízením. Ještě lepších výsledků je dosaženo použitím (pokud je to možné) prediktivního řízení, krátce popsané v následujících částech.

2.4 Prediktivní řízení TABS

Zpětná vazba je efektivní metoda řízení, ale nedokáže kompenzovat zpoždění, které je charakteristické pro systémy TABS. TABS jsou také náchylné na nepravidelné poruchy. Např.: počasí, aplikace budov a obyvatelé produkují teplo, CO_2 a nastavují tak podmínky pro teplotu. Tyto skutečnosti představují nárůst řídicí problematiky. Cílem je využít předpověď počasí nezbytnou pro vhodné využití tepelné kapacity budov. Model prediktivního řízení (MPC - Model Predictive Control) slibuje řešení této problematiky s využitím všech charakteristik z výše uvedených metod. MPC přináší zpětnou vazbu, která přinese informace ohledně tepelné chyby e, venkovní teploty t_{out} a předpovědí venkovního počasí, samozřejmě vše za předpokladu znalosti dynamiky systému [3].

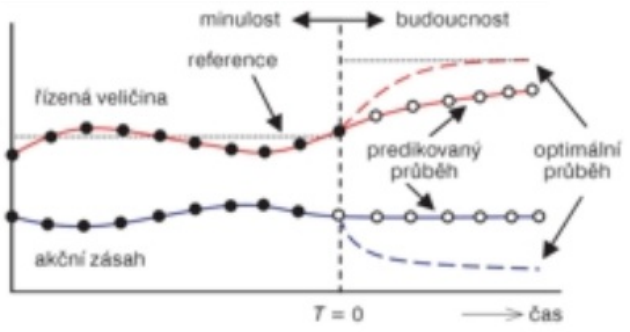
MPC tedy využívá model k predikci budoucích odezev systému. Díky této predikci je systém schopný vhodně spočítat hodnoty výstupů v každé vzorkovací periodě systému. MPC lze tedy přiřadit tyto rysy [5]:

- Trajektorie žádané veličiny je předem známa. U TABS je tedy potřeba znát přesnou trajektorii teploty, která bude v budově.
- Matematický model systému je použit pro predikci budoucího akčního zásahu.
- Výpočet budoucích akčních zásahů obsahuje i minimalizaci účelové funkce.

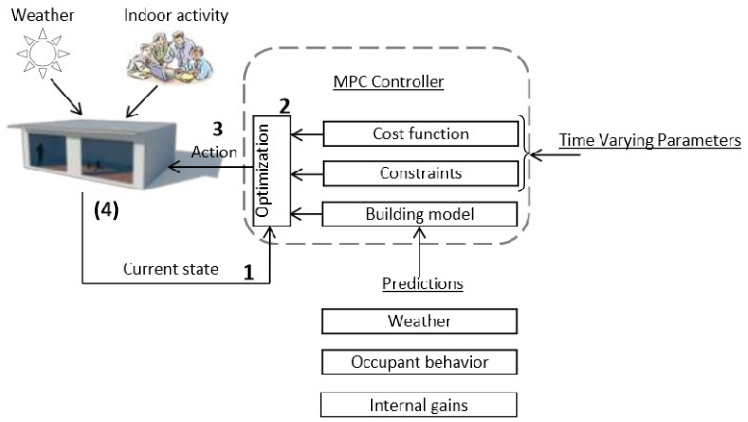
Kvalita řízení u MPC je vyšší něž kvalita řízení u "obyčejného"PID regulátoru. MPC řízení se vypořádá se systémy s velkým dopravním zpožděním, s nestabilním systémem či systémem nelineárním. Dále je toto řízení mnohostranné a robustní. Velkou výhodou je možnost řízení nebo sledování systému "online", což je také velká zásluha rozšíření těchto systémů.

2.4.1 Princip prediktivního řízení

Prediktivní řízení, jak už naznačuje předchozí kapitola, je pokročilejší typ řízení, který nehledá pouze vhodný akční zásah pro následující periodu vzorkování, ale hledá optimální posloupnost hodnot pro budoucí horizont, vše za pomocí modelu procesu. Konkrétně pro TABS tedy hledá optimální podmínky pro aktuální a budoucí operace. Sleduje tedy počasí, vnitřní teplotu, venkovní teplotu či přítomnost zaměstnanců.
Model je použit na predikci budoucích akčních zásahů, je třeba mít na paměti, že při řízení je třeba mít přesný model řízeného procesu. Pomocí tohoto modelu dostaneme budoucí možné trajektorie regulované veličiny, které se následně minimalizační funkcí upraví a výsledkem je optimální trajektorie budoucího zásahu akční veličiny. Minimalizace je samozřejmě provedena s ohledem na zachování všech vlastností, které mají jednotlivé veličiny procesu.
Nalezená posloupnost akčních zásahů by mohla být postupně použita v daném časovém horizontu. Po zavedení všech zásahů, by mohla být vypočítána další posloupnost a následně použita. Toto je ovšem chybné řešení, protože se jedná pouze o přímovazební řízení, které nedokáže eliminovat jednotlivé náhodné poruchy působící na systém. Tento problém lze vyřešit zavedením zpětné vazby v tzv. klouzavém horizontu. Klouzavý horizont je založen na principu neustálého počítaní nových posloupností akčních zásahů. Je-li vypočítána posloupnost akčních zásahů, tak je použita pouze první hodnota a následně je vypočítána nová posloupnost. Tímto je zavedena zpětná vazba [5].



Obrázek 6: Průběh posloupnosti akčních zásahů v horizontu [5]



Obrázek 7: Princip prediktivní řízení [1]

V první fázi je zaznamenán stav budovy, počasí a vnitřní aktivity. Ve druhé fázi jsou zaznamenané hodnoty poslány do MPC kontroléru, kde je vytvořen model systému, který popisuje dynamiku systému. Dále se vezmou v úvahu jednotlivé poruchy a vytvoří se posloupnost akčních zásahů pro daný horizont. Třetím krokem je aplikace prvního akčního zásahu z vytvořené posloupnosti. Konečně čtvrtý krok je zaznamenání výsledku a změření nových hodnot pro vypočtení nové posloupnosti [1].

Již bylo zmíněno, že minimalizace hledá optimální posloupnosti akčních zásahů pro daný horizont. Proto je nutné definovat kritérium minimalizace tak, aby definovala cíle pro daný prediktivní regulátor [5]. Kritéria jsou různá pro různé typy úlohy.
Protože se však tato práce nezabývá návrhem prediktivního regulátoru, nebudeme se tedy dále touto problematikou zabývat. Pro čtenáře, který by chtěl prediktivnímu řízení více porozumět, je uveden popis prediktivní regulace v příloze.

2.5 Výhody a nevýhody

Hlavní výhodou a důvodem, proč se začaly rozvíjet systémy TABS je úspora energie. Tepelné ztráty potrubí zalitého v betonových deskách nejsou příliš velké a kvalita ovzduší je dána pouze ventilačními podmínkami. Díky vytápění či chlazení pomocí betonových desek není potřeba žádných mechanických zařízení, jako jsou ústřední topení nebo jiné tepelné prostředky, a proto je v místnostech daleko více využitelného prostoru. To je především užitečné ve školách, nemocnicích a kancelářích, které mají velké požadavky na volný prostor [1].
Další výhodou je malý rozdíl teplot mezi povrchem a vzduchem uvnitř místnosti. Jelikož je při aktivaci aktivní celá plocha (strop, zem) je tepelné záření rovnoměrně vyzařováno do místnosti.
Dále lze za výhody považovat bezpečnost, úsporu pracovních sil a lepší pohodlí. Nakonec záleží k jakému účelu bude budova postavena. Výhody jsou pro každého uživatele jiné.
Hlavní nevýhodou je obrovské dopravní zpoždění, které se odvíjí od mohutnosti betonové desky. Je nutné mít předem vypočítané časy, kdy je potřeba aktivaci zapnout, tak aby přes den budova měla požadovanou teplotu.
Dá se říci, že masivní budovy jsou vhodné tam, kde je stabilní zátěž a můžeme minimalizovat solární účinky. V budově kde je třeba náhle měnit teplotu je tento systém nevyhovující.

Problematiku je třeba řešit i s použitým materiálem ohledně použitých senzorů a vodních trubek. Je třeba volit kvalitní materiály s velmi dlouhou životností. Je velmi obtížné měnit vodní trubku, která v průběhu času praskla uvnitř zalitého betonu. S tímto je spojeno i nebezpečí neúmyslného přerušení trubky. Příkladem může být vrtání potřebných otvorů do zdi a provrtání vodních trubek. Použité senzory musí být vyrobeny z nerezových materiálů, jinak dochází ke korozi a následně k destrukci senzoru.

TABS neřeší kvalitu ovzduší uvnitř místností, a proto je třeba s každým takovým systémem spojit i ventilační zřízení, které bude efektivně udržovat kvalitní prostředí. Spojením těchto dvou mechanismů lze vytvořit velmi kvalitní a pohodlné prostředí, které je vysoce ekonomické.

3 Rozbor experimentu

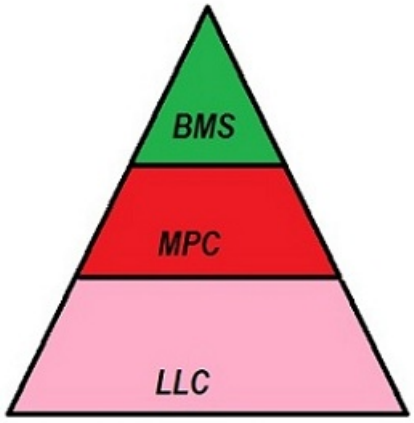
Úspora energie TABS objektu řízeného prediktivním řízením se pohybuje okolo 15 až 30% ročně. Ovšem navržený algoritmus prediktivního řízení běžící na matematickém modelu měl úsporu 5%. Je v celku jasné, že úspora 5% energie ročně je chybný údaj, ale na druhou stranu nebyla ani nalezena chyba v testovaném modelu, která by tuto nesrovnalost objasňovala. Z důvodu takto odlišných výsledků ohledně úspory energie byl navržen experiment, který ověří daný algoritmus v praxi a bude tak relativně přesně odměřena úspora energie pro budoucí aplikace takto navržených TABS objektů.

Při zkoumaní a měření vlastností TABS systémů je důležité navrhnout komponenty, ze kterých se řízení bude skládat a prodiskutovat varianty, které by pro daný experiment byly nejlepší.

Základní a velmi důležitou otázkou je, jaké vývojové prostředí použijeme pro realizaci programu jednak pro řídící systém, tak pro komunikaci mezi řízením a deskou.

Druhou otázkou je jak vytvořit koncept řízení, zahrnující komunikaci mezi řídicím systémem a senzory a definicí jednotlivých kroků implementace.

Implementace je navržena následující hierarchií tvořenou ze tří částí.

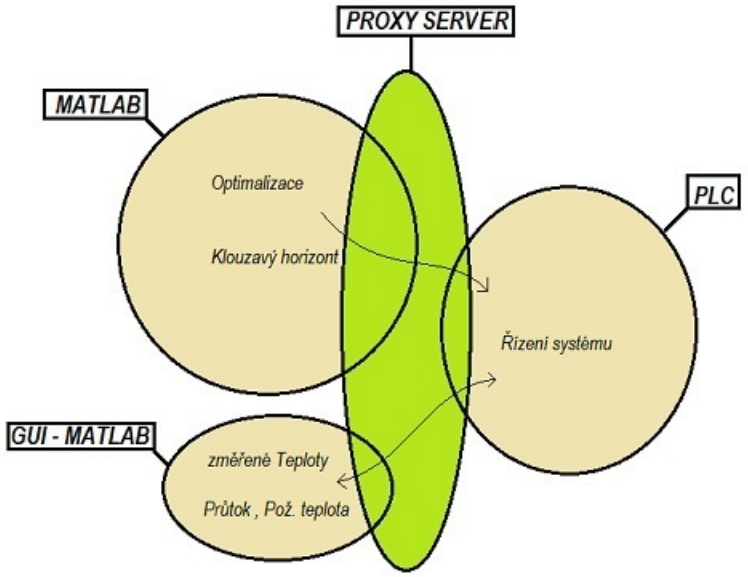


Obrázek 8: Koncept řízení pro TABS systémy

Informace jsou předávány od vrcholu směrem dolů.

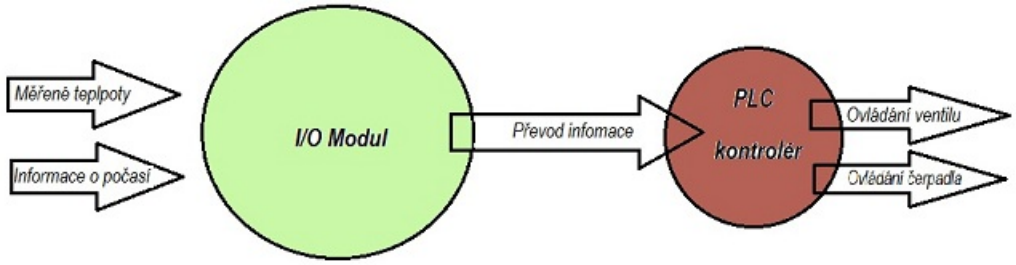
Uživatelský koncept **BMS** (Building Managment System) nastavuje požadavky kladené na celkové chování systému. V našem případě se jedná o zadání požadované teploty v místnosti. Tato informace je předána dále do softwarové části.

Softwarový koncept **MPC** (Model Predictive Control) je složen ze dvou skriptů a rozhraní, přes které bude komunikovat. První skript bude obsahovat minimalizační algoritmy, nastavení klouzavého horizontu a omezující podmínky. Druhý skript bude obsahovat grafické uživatelské rozhraní, ve kterém budou zaznamenány změřené hodnoty ze senzorů a bude umožňovat nastavení vybraných softwarových vstupů. Tyto skripty budou přes rozhraní vzdáleně komunikovat s řídícím systémem (např. PLC). Vzdálená komunikace je nutná, aby byla možnost běhu skriptů na různých počítačích a různých místech.



Obrázek 9: Příklad softwarového konceptu

Hardwarový koncept **LLC**(Low Logic Control) se skládá ze dvou modulů. První modul bude obsluhovat komunikaci mezi jednotlivými vstupy a výstupy ať už digitálními nebo analogovými a tyto informace bude posílat do druhého zařízení (modulu), který bude tvořit programovatelný počítač PLC. PLC bude obsahovat program pro zpracování dat ze senzorů a dat z příslušného softwaru. Program vyhodnotí tyto informace a bude generovat příslušné řídicí signály pro jednotlivé výstupy. Konkrétní vizualizace tohoto konceptu je na obrázku 10.



Obrázek 10: Příklad hardwarového konceptu

Zakomponováním všech těchto konceptů do sebe by mohlo vzniknout kvalitní řízení pro TABS experiment. Experiment samozřejmě obsahuje další problémy, spojené s použitou technologií a instalací, které budou diskutovány níže.

3.1 PLC kontrolér

Jak již bylo zmíněno, pro řízení se použije kontrolér PLC. Existuje mnoho druhů a je na každém projektantovi, který typ zařízení si vybere. Aby však zařízení neplnilo pouze funkci "obyčejných"regulátorů jako jsou například (PID,PI,PD), tak musí pracovat v reálném čase a mít možnost vysílat i přijímat informace "on-line", aby mohl včas přizpůsobit řízení například kvůli náhlé změně počasí.

3.1.1 Struktura PLC

PLC nebo-li průmyslový počítač je zařízení, které cyklicky vykonává program a řídí tak sled výstupů do aktivních úrovní. Používá se převážně pro automatizaci procesů v reálném čase. PLC jsou odlišné od běžných počítačů nejen cyklickém vykonáváním programu, ale i tím, že jsou jednotlivé vstupy/výstupy přímo připraveny na připojení k technologickému procesu. Cyklické vykonávání programu znamená, že PLC opravdu **nepřetržitě** čte jednotlivé instrukce dokola. Nelze tedy ani na krátkou dobu proces "uspat" a po čase opět proces spustit. Kdyby to bylo možné, pak by řízení nebylo bezpečné a kdyby zrovna nastala chyba, při které by měl být proces zastaven, tak by na tuto chybu PLC reagovalo až při opětovném spuštění. Nicméně čekací smyčky lze také v PLC programovat a řeší se pomocí logického zásobníku.

3.1.2 Dělení PLC

Obvykle kontrolér pracuje ve třech režimech:

STOP - PLC nevykonává program a čeká dokud nebude nahrán do paměti. Vstupy a výstupy nejsou aktivní.

ERROR - V tomto režimu došlo v PLC k chybě ať už při čtení programu nebo špatném nastavení. Program není vykonáván a čeká na opravu. Obvykle se tento stav řeší připojením na uživatelský počítač a vyhledáním chyby.

RUN - Program je cyklicky vykonáván a jsou aktivní vstupy a výstupy.

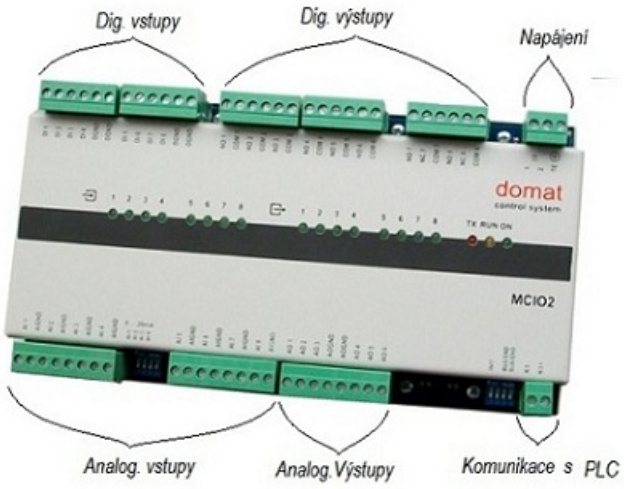
Existuje více druhů PLC a můžeme je rozdělit opět do tří skupin.

Minisystémy - Jedná se o nejednodušší verzi, která má nízký počet vstupů a výstupů, obvykle není vybavena klávesnicí ani obrazovkou. Nepracuje v reálném čase, má malou interní paměť a je vhodný pro řízení jednoduchých systémů. Výhodou je cena.

Kompakty - Složitější zařízení pracující v reálném čase. Má větší počet vstupů a výstupů. Používá se k řízení větších automatizačních procesů.

Modulové systémy Nejlepší a zároveň nejdražší verze. PLC se skládá z jednotlivých modulů a hlavní výpočetní jednotky. Jednotlivé moduly obsahují například pouze vstupy/výstupy. Používají se k řízení velmi složitých technologických systémů.

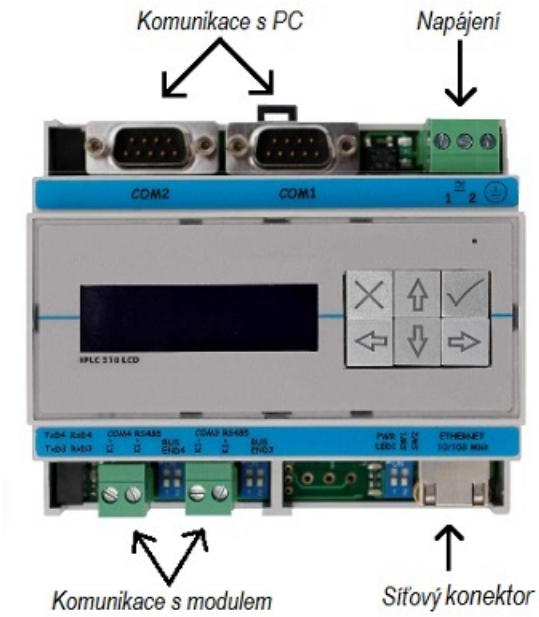
Do projektu bylo vybráno PLC od firmy DOMAT s jedním přípojným modulem. Modul obsahuje 8 analogových a digitálních vstupů, 6 analogových výstupů a 8 digitálních výstupů. Tento modul nazývaný MCIO2 komunikuje s PLC typu IPLC510B. Jedná se o PLC pracující v reálném čase s možností online komunikace po připojení na internet.



Obrázek 11: Použitý I/O modul

Základní parametry modulu:
Napájení: 12 - 24 V ss.
Komunikace: RS485, 1200 - 19200 Bit/s.
Maximální délka sběrnice: 1200m.
Rozměry: $217 \times 115 \times 40mm$.

Použité PLC:



Obrázek 12: Použité PLC

Základní parametry:
Napájení:$10 \div 35$ V ss.
Prac. teplota: $0 \div 60°C$, použité součástky s odolností $-20 \div 80°C$
Procesor: MPC5200, 400MHz, 760 MIPS.
Paměť: 64MB RAM, 32MB Flash, 128Kb NVRAM FRAM
Rozměry: $105 \times 90 \times 58mm$.
Komunikace pomocí RS485.

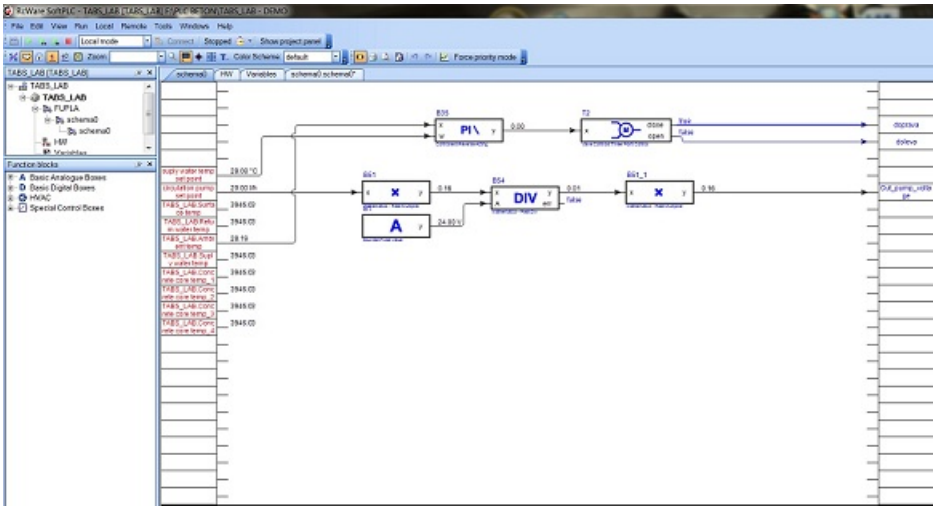
Obě zařízení budou převezena s betonovou deskou do klimakomory v Dánsku, kde bude proveden příslušný experiment.

3.2 Vývojové prostředí

Programování PLC bylo tvořeno ve vývojovém prostředí RcWare SoftPLC. Toto prostředí obsahuje sadu programů pro regulaci a řízení technologických procesů. Typické použití je především v systémech větrání, vytápění a klimatizace. Pro navrhnutí řízení TABS si vystačíme s programem SoftPLC IDE z balíčku programů SoftPLC.

3.2.1 SoftPLC IDE

IDE neboli integrované vývojové prostředí slouží k editaci aplikací, jako jsou fyzické vstupy, výstupy, moduly nebo cizí systémy. Pomocí funkčních bloků se určí, jak se má aplikace chovat.



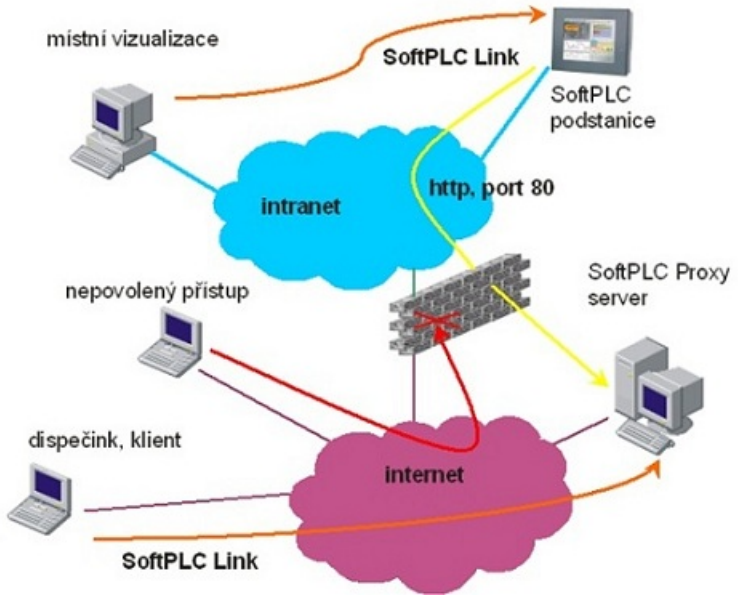
Obrázek 13: Vývojové prostředí SoftPLC IDE

V editoru je kontextová nápověda, která podrobně popisuje jednotlivé funkční bloky, jejich chování a příklad.

Knihovny obsahují velké množství funkčních bloků. Jsou k dispozici analogové bloky, digitální bloky, matematické funkce, PID regulátory, časové bloky, alarmové bloky, ekvitermní křivky atd.
Prostředí obsahuje i funkci testování komunikace, ve schématech jsou vidět "on-line" aktuální procesní hodnoty a lze je zobrazovat do grafu, což velmi usnadňuje ladění regulačních smyček.

3.2.2 SoftPLC Proxy server

Software RcWare poskytuje i možnost připojení na proxy server. Přes toto rozhraní bude PLC komunikovat s jednotlivými skripty běžících na personálních počítačích. Funkce spočívá ve spojení podstanic bez nutnosti mít veřejnou IP adresu. Spojení mezi proxy serverem a podstanicemi zajišťuje SoftPLC Proxy. Podstanice (PLC) po startu naváže odchozí spojení proto-



Obrázek 14: Principiální funkce proxy serveru [7]

kolem http na proxy server (žlutá šipka). Příchozí připojení do sítě nejsou povolena (červená šipka). Případná místní centrála komunikuje s podstanicí přímo v síti zákazníka, externí dispečink pak sdílí data přes proxy server (oranžová šipka) [7].

3.2.3 Uživatelské prostředí

Prostředí, ve kterém budou naprogramovány jednotlivé skripty může být libovolné. Protože však budeme převážně pracovat se signály, tak jsme zvolili prostředí MATLAB. MATLAB je velmi univerzální vyšší jazyk, který podporuje práci se signály. Je to výborný nástroj pro návrh regulace a obsahuje možnost tvorby grafického uživatelského prostředí.

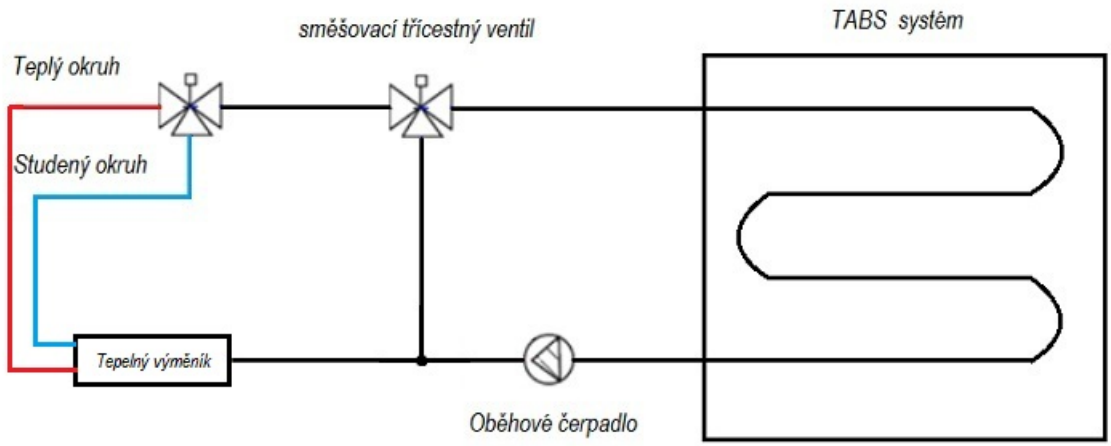
4 Návrh TABS

Experiment je založen na zkoumání vlastností TABS systémů, abychom mohli však úspěšně odhadnout chování celé aktivované budovy, tak je třeba vyzkoušet chování samotné části aktivované betonové desky. S návrhem je spojeno plno návrhových problémů, které je nutné před realizací vyřešit. Jedná se především o použité senzory, čerpadla, rozměry trubek, rozteče trubek, hloubku uložení trubek a o rozmyšlení, které části budou aktivovány (strop, podlaha, stěny). Tato práce se zabývá pouze návrhem samotné betonové desky a nízkoúrovňového řízení LLC viz obrázek 9 (poslední část). Prediktivní regulaci si navrhne energocentrum individuálně.

Navržený TABS panel musí vystihovat typické chování TABS a MPC řízení. Musí mít co nejrychlejší dynamiku (drahý provoz) a musí být vhodně osazený senzory. Toto jsou podmínky, které musí být splněny, aby odměřené výsledky byly použitelné.

4.1 Návrh Hydraulické části

Funkcí betonové desky by mělo být jak vytápění, tak chlazení. Proto je třeba navrhnout systém, který bude obsahovat dva okruhy a nejméně jeden směšovací ventil, který bude zajišťovat nastavení teplot v intervalu $< t_{min}, t_{max} >$. První okruh bude zajišťovat teplotu t_{min} a naopak druhý okruh teplotu t_{max}. Hydraulický obvod musí také obsahovat oběhové čerpadlo, které bude zajišťovat zvolený objemový průtok.



Obrázek 15: Navržený hydraulický obvod

Trojcestný ventil jsme zvolili od firmy DOMAT, jedná se o ventil typu VD131 20-6,3. Volba ventilu je libovolná, se změnou ventilu je třeba upravit řízení, protože každý ventil představuje jiný systém s jinou směšovací charakteristikou. Jak již typ ventilu naznačuje, má vnitřní průměr $20mm$ a maximální průtok ventilem je $6.3m^3/hod$.
Volili jsme čerpadlo s proměnným průtokem od firmy GRUNDFOS, typ Grundfos MAGNA 25-40. Maximální výtlačná výška jsou 4 metry, vnitřní průměr otvorů je $25mm$. Tento typ jsem zvolil z důvodu kompatibility s modulem GENI, též od firmy Grundfos, který umožňuje spojitou regulaci otáček pomocí napětí $0 - 10V$ a další ovládání čerpadla externími signály. Maximální průtok se pohybuje okolo $6m^3/hod$ a pracovat s ním lze ve třech významných režimech (viz dokumentace k čerpadlu).

4.2 Rozmístění senzorů

Nezbytnou část pro regulaci tvoří jednotlivé teplotní senzory. Protože PLC modul obsahuje 8 analogových vstupů, tak je možné použít pouze 8 teplotních senzorů nebo přikoupit další I/O modul. Pro náš experiment si vystačíme pouze s jedním modulem. Zde vznikají tři základní problémy: jaké teploty budeme měřit, jak senzory do desky vložíme a jak je rozmístíme. Volili jsme mezi třemi typy teplotních senzorů.

- Termočlánky
- NTC termistory
- odporové prvky

Vysoká odolnost, dlouhá životnost, přesnost a linearita jsou kritéria, podle kterých jsme volili senzory do betonové desky.

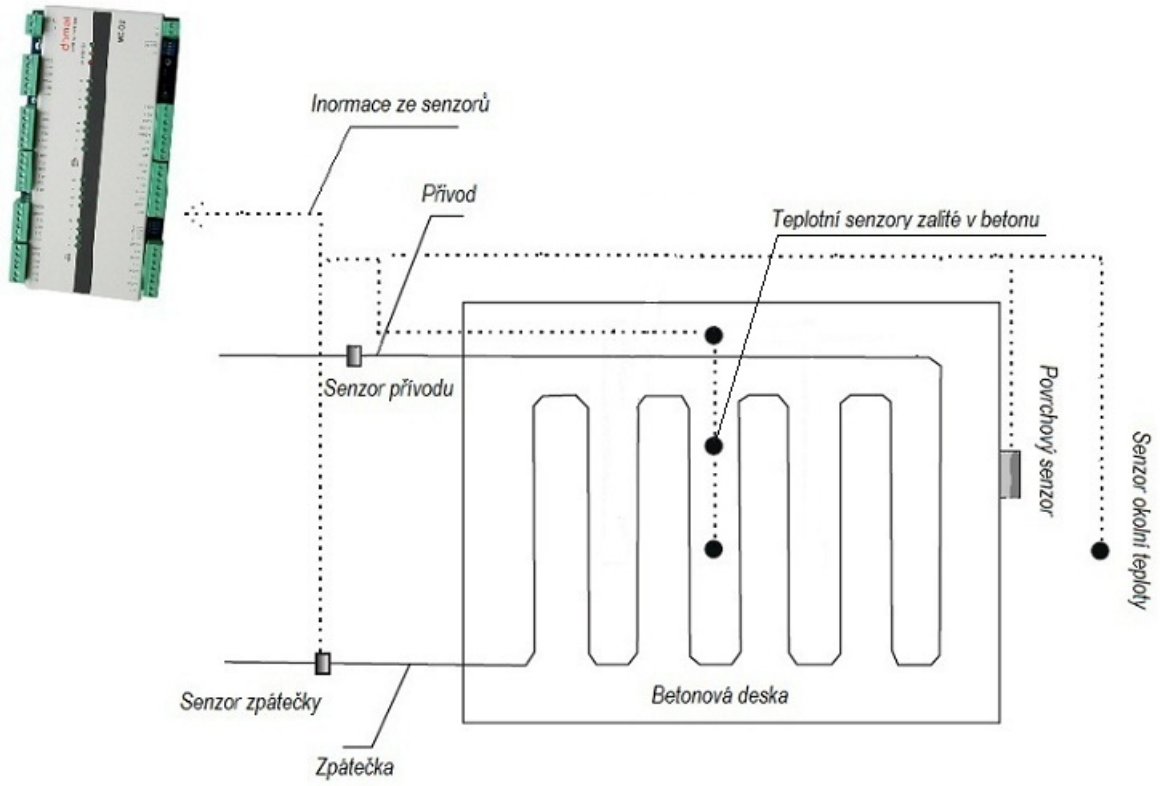
Termočlánky jsou nevýhodné pro jejich relativně krátkou životnost a přesnost.

NTC termistory jsou nevhodné kvůli teplotní charakteristice a částečně také kvůli životnosti.

Odporové senzory kalibrace-A jsou díky velmi dlouhé životnosti, linearitě a přesnosti optimální senzory.

Další otázkou je, jak vybrané senzory vložit do desky. Nabízí se možnost přímého zalití senzoru betonem nebo zalít do betonu trubičku, do které by byl senzor zastrčen. Přímé zalití je nevýhodné ohledně výměny senzoru. Při poruše nebo při přetržení kabelu je senzor nevyměnitelný. Druhé provedení výměnu senzoru dovoluje, ale jsou zde problémy s materiálem trubičky. Je třeba volit nerezový materiál. Dále je zde nepřímý dotek s betonem, tedy horší měření teploty. Vyskytuje se zde i problém s kondenzací vody. Příklad z praxe, z 16 senzorů měřící teploty jich bylo funkčních pouze 8 z důvodu kondenzace vody v trubičkách a následné koroze.

Pro náš experiment jsme tedy zvolili přímé zalití senzoru do betonu. Senzor je chráněn nerezovou ocelí.



Obrázek 16: Zvolené rozmístění senzorů

Na obrázku výše je rozvržení senzorů. Volili jsme jeden senzor okolní teploty, jeden povrchový senzor, jeden senzor teploty přívodu a zpátečky a 3 senzory měřící teplotu uvnitř betonu. Je zde otázka, proč nepoužít více povrchových senzorů, protože teplota povrchu nám dává rozhodující informace pro řízení. Toto je dobrá myšlenka a při budování celého systému (TABS budovy) by to takto bylo nejspíše řešeno. Informace ohledně teploty uvnitř betonového jádra je také velmi zajímavá. Můžeme sledovat vývoj teploty v závislosti na vzdáleností od jádra aktivované desky směrem k povrchu. Jelikož experiment je založen na zkoumání vlastností TABS, tak je důležité změřit rozprostření energie uvnitř betonu a proto použijeme většinu senzorů pro měření vnitřní teploty.

Důležité je mít jednu stranu desky maximálně izolovanou. Pokud bude deska ideálně tepelně izolována, pak bude energie proudit pouze jedním směrem a to směrem od izolace přes jednotlivé senzory k povrchu a výsledky měření tak nebudou zkreslené.

Všechny zařízení (senzory) použité v experimentu jsou pro přehlednost zaneseny v tabulce 2.

Název	***Zařízení***	***Typ***
Čidlo povrchové teploty (Surface temp)	Příložný senzor	Domat OFTF - Pt1000
Senzor betonového jádra_1 (Concrete core temp_1)	Odporový senzor	Domat HTF- Pt1000
Senzor betonového jádra_2 (Concrete core temp_2)	Odporový senzor	Domat HTF- Pt1000
Senzor betonového jádra_3 (Concrete core temp_3)	Odporový senzor	Domat HTF- Pt1000
Senzor teploty zpátečky (Return water temp)	Ponorný senzor(jímkový)	Domat ETF1 - Pt1000
Senzor teploty přívodu (Ambient Temp)	Ponorný senzor(jímkový)	Domat ETF1 - Pt1000
Senzor okolní teploty (Outdoor sensor temp)	Venkovní senzor	Domat ATF1 - Pt1000
Průtokoměr (Flow rate)	kalorimetr	Megatron 2
Čerpadlo	Oběhové čerpadlo	Grundfos Magna 25-40
Ventil	3-cestný směšovací ventil	Domat VD131 20-6.3

Tabulka 2: Přehled použitých zařízení pro experiment

4.3 Rozložení vodních trubek v betonu

Vodní trubky budou v desce rozloženy do jednoduchého meandru. Vnitřní průměr, rozteč a hloubka položení trubek hrají významnou roli ohledně chlazení/vytápění, účinnosti a spotřebě energie. Rozložení "hadů"záleží na zvolené aktivní části. Otázkou bylo, zda zvolit aktivní strop nebo podlahu.

Aktivní strop - Jedná se o masivní konstrukci, která zároveň plní funkci nosné desky. Proto se celková šířka betonu pohybuje okolo $30cm$. Hloubka položení trubek se nejčastěji volí $15cm$.

Aktivní podlaha - Zde se jedná naopak o poměrně tenkou desku o maximální šířce $10cm$. Vodní trubky se pokládají těsně pod povrch okolo $5cm$, protože se zde počítá s povrchovou izolací (koberec, linoleum, apod.), která snižuje efekt vyzařovaní.

Experiment bude probíhat v Dánsku, a proto je nutné počítat s váhou desky ohledně převozu. Budeme-li uvažovat aktivaci stropní desky s rozměry $100 \times 100 \times 30cm$, tak se hmotnost bude pohybovat okolo jedné tuny. Váha podlahy je ve stejných rozměrech znatelně lehčí. Dále je nutné počítat s tím, že v klimakomoře, ve které bude deska v Dánsku testována, musí být schopna udržet stropní panel.
I přes všechny "nevýhody", které volba stropní aktivace obsahuje jsme se rozhodli právě pro tuto variantu. Stropní aktivace je podle naší úvahy ohledně budoucnosti mnohem významnější. Různé experimenty ukazují, že velké masivní budovy téměř nepotřebují vytápění, ale spíše chlazení. Do budoucna je tedy možné, že TABS budovy budou převážně fungovat pro ochlazování prostředí a ochlazování přes podlažní desku jsme uznali za vcelku nevhodné.

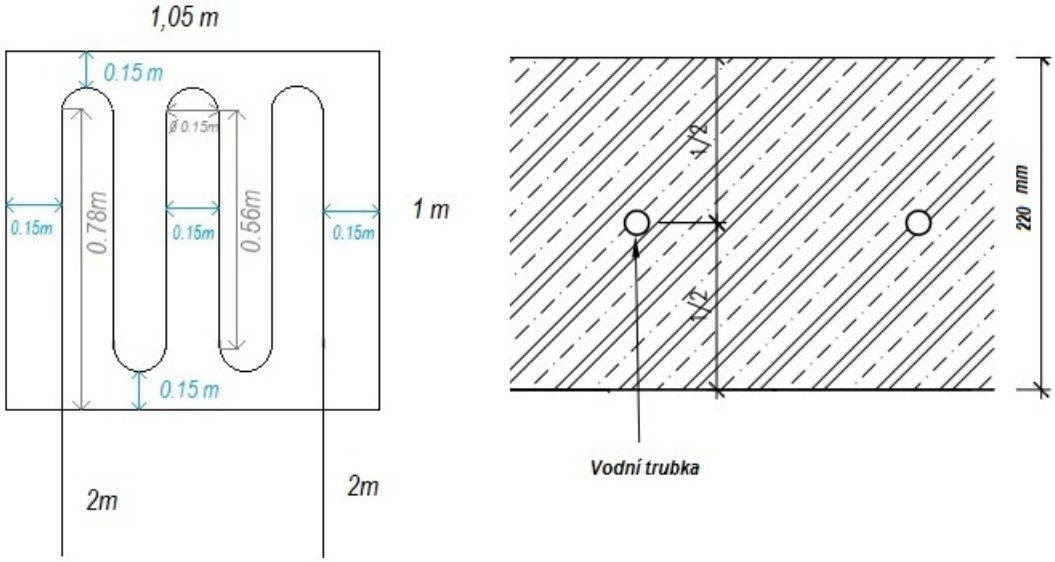
Zvolili jsme tedy aktivaci stropní betonové desky o rozměrech $105 \times 100 \times 22cm$. Hloubka položení vodních trubek je přímo uprostřed desky, tedy $11cm$. Rozteč mezi "hady"jsme voli také $15cm$, protože výzkumné studie firmy REHAU dokázaly, že pokládka s roztečí $10cm$ je nehospodárná. Hustota pokládky $10cm$ způsobuje při vytápění cca o 10% zvýšení výkonu

a spotřeba materiálu je v tom to případě větší zhruba o 33%. Tento rozměr má tedy nevýhodný poměr mezi užitkovostí a náklady [10]. Rozteč $20cm$ je naopak vhodná jen v některých aplikacích [10].

roteč	$10cm$	$15cm$	$20cm$
Délka trubky	$10,0m/m^2$	$6,6m/m^2$	$5,0m/m^2$
Hodnota výkonu	110%	100%	90%

Tabulka 3: srovnání délky trubky a hodnoty výkonu při použití rozdílných roztečí udávané firmou REHAU [10]

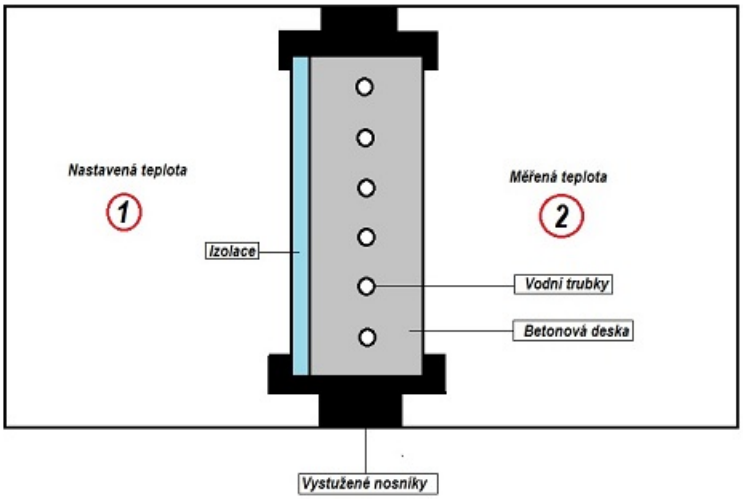
Do betonové konstrukce bude vložena trubka REHAU Rautherm S ze sítěného polyetylénu PE-Xa s rozměry $20 \times 2,0$. Celkovou délku "hada" jsme odhadli na $10m$.



Obrázek 17: Rozměry desky včetně rozprostření trubky

4.4 Klimakomora

Jak již bylo zmíněno, celý experiment bude odsimulován v klimakomoře v Dánsku.

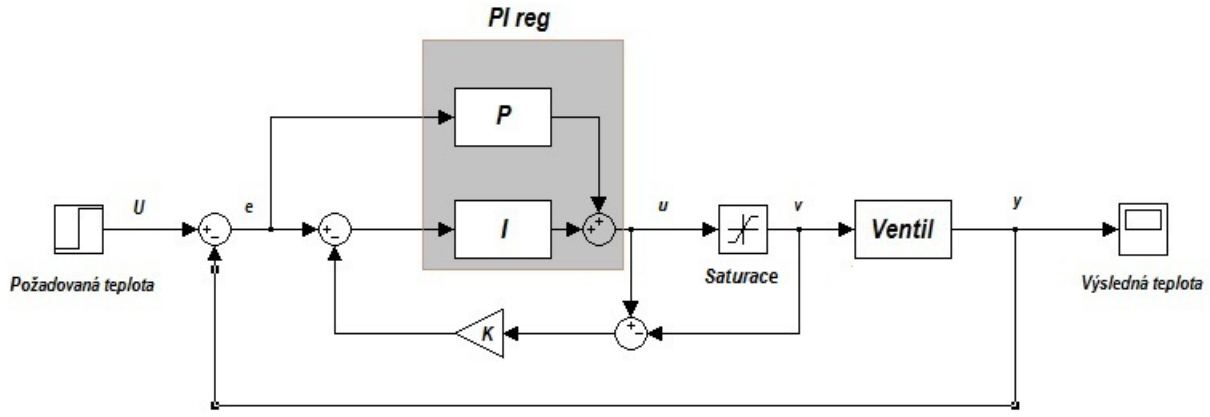


Obrázek 18: Vizualizace klimakomory

Jedná se o izolovaný komplex, který bude rozdělen na dvě části (místnosti) pomocí betonové desky. Nosníky desky musí být speciálně vyztužené, protože deska bude velmi hmotná (počítáme-li uvedené rozměry $105 \times 100 \times 22cm$, tak bude deska vážit okolo $700kg$) a musí být zajištěna potřebná bezpečnost. Uchycená deska bude představovat stropní aktivaci popsanou výše. Izolace (v praxi protizvuková bariéra nebo přímo střešní izolace) je při experimentu velmi důležitá a deska musí být natočena tak, aby izolace směřovala do komory č.1. Díky této izolaci totiž bude tok energie proudit právě jedním směrem a to do místnosti č.2.
Komora č.1 představuje okolní prostředí, respektive venkovní teplotu. Tato teplota je nastavována dle potřeb, ovšem je třeba brát v potaz i spotřebu energie. Jedná-li se o prostor $2 \times 3m$, tak udržování teploty v místnosti např. $-5°$C vyžaduje obrovský příkon.
V komoře č.2 je dále pak zaznamenáván průběh teploty, kterou zajišťuje uchycený TABS panel řízený již popsanou strategií. Na základě změřených dat bude pak možné ověřit správnost chování regulace na modelu.

5 Návrh Řízení

Návrh nízkoúrovňového řízení (LLC) je nezbytnou součástí celého algoritmu. Zajišťuje totiž požadovanou teplotu vody proudící ve vodních trubkách v desce a zajišťuje požadovaný průtok. Tuto regulaci bude vykonávat již zmíněný PLC kontrolér.
Pro řízení využijeme funkční bloky v předdefinované knihovně v softwaru. Jedná se o pokročilý software a práce je v něm mnohem efektivnější než práce s jazykem instrukcí, který se dá přirovnat k assembleru.
Hodnotu teploty vody zajišťuje trojcestný směšovací ventil viz hydraulické schéma. Ventil tedy budeme ovládat v jednoduché zpětnovazební uzavřené smyčce.



Obrázek 19: Návrh uzavřené smyčky

Řízení bude obsluhovat jednoduchý PI regulátor s vhodně nastaveným anti-windupem. Volba PID regulátoru by nebyla příliš vhodná, protože se jedná o poměrně jednoduchou regulaci a D složka by do regulace nepřinášela téměř žádný užitek. Samotnou regulaci by byl schopen zařídit i samotný P regulátor, ale trvalou regulační odchylku, kterou tento regulátor vytváří, si nemůžeme při kvalitním řízení TABS panelu dovolit.

5.1 Anti-Windup

Anti-Windup ideálně zajišťuje při regulaci náběh řízené veličiny bez překmitu. Anti-Windup je zjednodušeně hysterezní smyčka s vhodně nastavenými hranicemi. Pokud není chyba regulace e rovna 0, pak integrační složka stále "dorovnává" chybu v podobě akčního zásahu. Ovšem ventil nemůže být více otevřený, než je jeho maximální poloha nebo naopak nemůže být více než úplně zavřený. Integrační složka tuto skutečnost nezaznamenává a stále zvyšuje/snižuje svou hodnotu dokud se $e = 0$ a ve většině případů tak způsobí vysoký překmit regulované veličiny.
Funkce anti-windupu je omezit integrační složku, jestliže je dosaženo maximální úrovně (ventil je zcela otevřený/zavřený), s tím souvisí vhodně nastavené meze anti-windupu (horní mez = zcela otevřený ventil, dolní mez = zcela zavřený ventil). Při dosažení maximální úrovně je hodnota I složky ovlivněna zpětnou vazbou $z = K(u - v)$, kde K je konstanta, u je akční zásah a v je omezený akční zásah vstupující do soustavy(ventilu). Pokud není dosaženo maximální úrovně, tak je zpětná vazba z rozpojena ($u = v \Rightarrow K(u - v) = 0$) a integrační složka není ničím omezena.
Dá se říci, že při saturaci je hlavní smyčka v podstatě otevřená a integrační člen se v otevřené smyčce stává nestabilním prvkem a musí být extra stabilizován. Pokud není dosaženo saturace, pak hlavní smyčka není rozpojená a není třeba pomocné zpětné vazby.

5.2 Obecná regulace

akční zásah do soustavy je řízen PI regulátorem podle následující rovnice:

$$v = \begin{cases} max & u > max \\ k_p e(t) + k_I \int\limits_{t_0}^{t} [e(\tau) - K(u - v)] d\tau & u \in < min, max > \\ min & u < min \end{cases} \tag{2}$$

kde

v je akční signál
$e(t) = U - y$ je regulační chyba
k_p je proporcionální konstanta
k_I je integrační konstanta

Celá soustava bude mít jistě dopravní zpoždění na výstupu. Toto zpoždění bude způsobeno jednak přejezdem ventilu, tak přenosem informace ze senzoru umístěného v jímce zabudované ve vodní trubce.
Celkové zpoždění můžeme předpokládat jako:

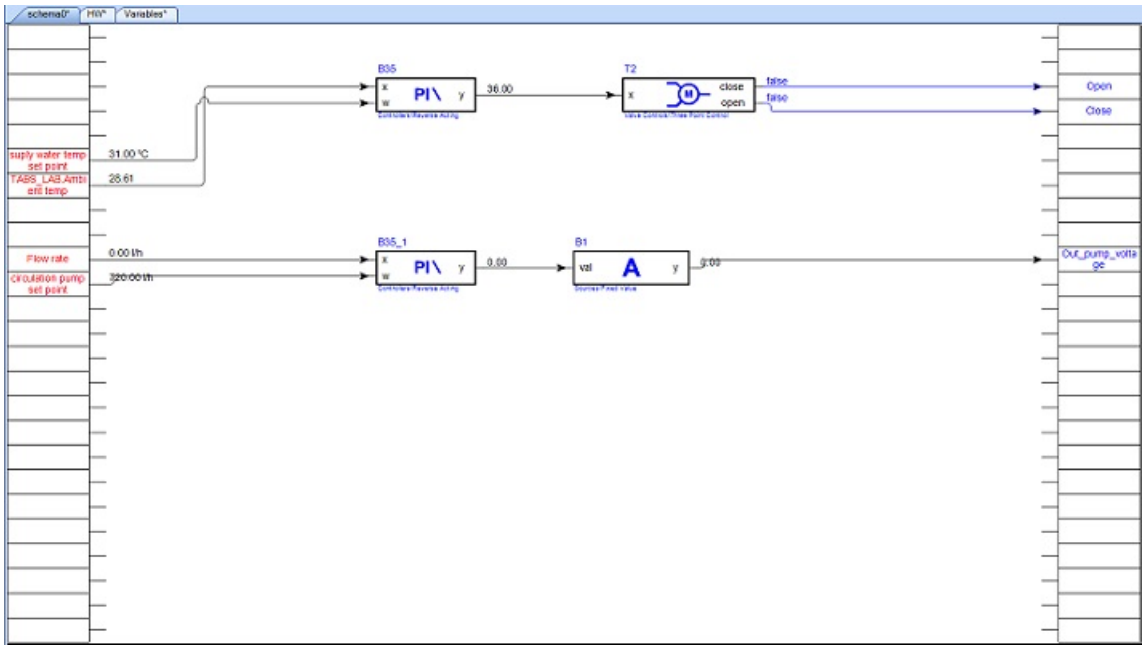
$$t_{delay} = t_{valve} + t_{sensor}$$

Zpoždění bude kompenzováno přímo v regulátoru. Obecně by se to dalo řešit také zvětšením periody vzorkování, která by byla větší(rovna), něž celková doba zpoždění a tím by se zpoždění "odstranilo". Použitý software má již tak propracovaný, že je zpoždění řešeno již ve funkčních blocích.

Hodnota min a max v rovnici (2) značí procentuálně maximální zavření/otevření ventilu, tedy 0 a 100%.

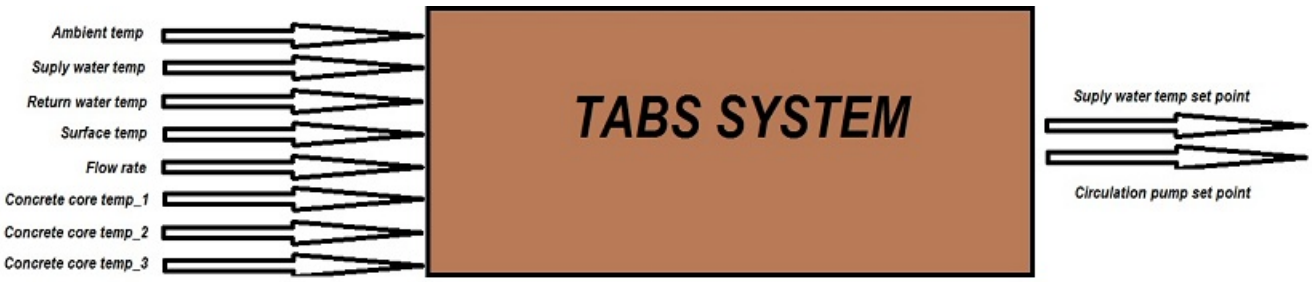
Protože většina čerpadel (naše není výjimkou) mívá za požadovanou hodnotu tlak a ne průtok, tak nelze čerpadlo připojit přímo na systém, který by nastavoval na čerpadle hodnotu průtoku. Čerpadlo vyžaduje regulaci podobně jako ventil. Čerpadlo umístíme do uzavřené smyčky viz obr.19, ale na místo ventilu vložíme čerpadlo a na místo senzoru teploty přijde průtokoměr. Tím půjde v softwaru nastavit požadovaný průtok a PI regulátor už zařídí příslušné otáčky na čerpadle.
Jako průtokoměr zvolíme kalorimetr, který měří a po sběrnici posílá okamžitý průtok a teplo odevzdané do desky prostřednictvím ohřáté vody.

Regulace čerpadla ventilu lze zakomponovat do jednoho schématu v softwaru RcWare (obr.20).



Obrázek 20: Regulační smyčky v PLC softwaru

Anti-Windup a všelijaká další omezení se dají nastavit po otevření přímo v blocích. Kolonky vlevo značí jednotlivé vstupy do systému, kolonky na opačné straně jsou výstupy ze systému (zaznamenávané již popsanými senzory). Celý systém představuje MIMO(Multiple In Multiple Out) systém s 8 analogovými vstupy a 2 analogovými výstupy.



Obrázek 21: Blokové schéma systému

Celkový počet osmi vstupů je zavádějící, neboť do soustavy vstupují ještě informace z minimalizační funkce v prediktivní části řízení a jsou zde i dva softwarové vstupy nastavující požadovaný průtok a povrchovou teplotu.

5.3 Volba konstant

Přenos PI regulátoru po aplikaci Laplaceovy transformace je:

$$C(s) = K_p + \frac{K_I}{s} = \frac{K_p s + K_I}{s} \tag{3}$$

Přenos soustavy budeme předpokládat ve tvaru :

$$H(s) = \frac{K}{1 + Ts} = \frac{\frac{K}{T}}{\frac{1}{T} + s} \tag{4}$$

Po aplikaci zpětné vazby dostanu charakteristický polynom uzavřené smyčky.

$$\begin{aligned} F(s) &= C(s)H(s) = \frac{q(s)}{p(s)} \cdot \frac{b(s)}{a(s)} \\ G(s) &= \frac{F(s)}{1 + F(s)} \end{aligned}$$

$$\boxed{c(s) = a(s)p(s) + b(s)q(s) = s(\frac{1}{T} + s) + \frac{K}{T}(K_p s + K_I)} \tag{5}$$

charakteristický polynom soustavy $c(s)$ porovnám s naším požadovaným polynomem, respektive umístím póly soustavy do požadovaných poloh a dostanu tím potřebné konstanty na nastavení PI regulátoru.
Pomocí navrženého polynomu chci umístit póly soustavy do poloh $-\alpha$ a $-\beta$ nejlépe tak, abychom dostali nekmitavou soustavu. Póly nemohu umisťovat libovolně a platí, že se nesmí příliš vzdálit od původních poloh, jinak je většinou akční zásah do soustavy nerealizovatelný.

Námi vytvořený polynom tedy bude ve tvaru:

$$l(s) = (s+\alpha)(s+\beta) = s^2 + (\alpha+\beta)s + \alpha\beta \tag{6}$$

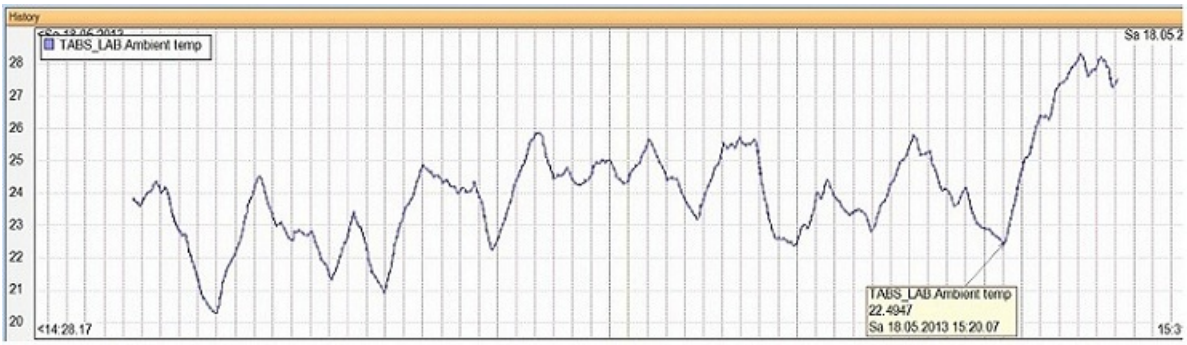
oba polynomy porovnáme a dostaneme požadované konstanty.

$$\begin{aligned}
l(s) &= c(s) \\
s^2 + (\alpha+\beta)s + \alpha\beta &= s^2 + (\frac{1}{T} + \frac{K}{T}K_p)s + \frac{K}{T}K_I \\
\alpha+\beta &= \frac{1}{T} + \frac{K}{T}K_p \Longrightarrow K_p = \frac{T(\alpha+\beta)-1}{K} \\
\alpha\beta &= \frac{K}{T}K_I \Longrightarrow K_I = \frac{T\alpha\beta}{K}
\end{aligned}$$

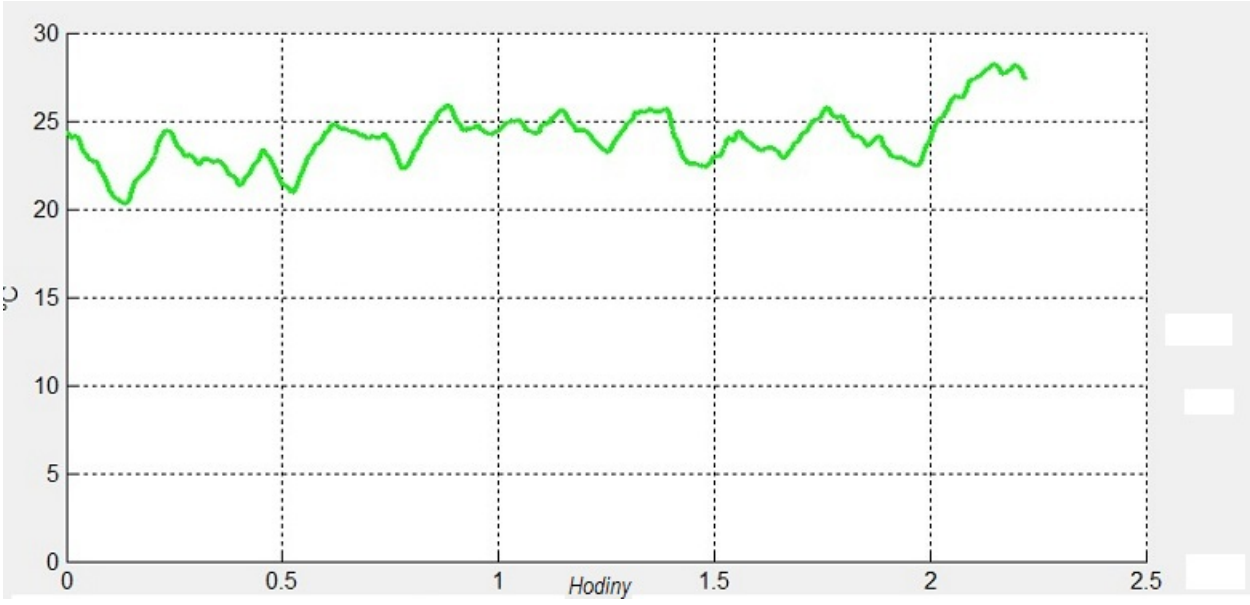
Druhá možnost nastavení konstant je čistě experimentálně. Metoda experimentálního nastavení konstant bude nejspíše použita u regulace čerpadla, protože identifikace systému čerpadla by byla náročnější.
Pro náš případ zkusím navrhnout póly do poloh $\alpha = -1$, $\beta = -2$, což dává $l(s) = s^2 + 3s + 2$ a tvoří tak teoreticky "pomalejší"a nekmitavý systém. Saturaci nastavím na polohy $W_{max} = 100$,$W_{min} = 0$.
Ohledně čerpadla bude saturace nastavena na $W_{max} = 10$,$W_{min} = 0$, protože signál regulující otáčky čerpadla se pohybuje v intervalu $< 0, 10 > [V]$

5.4 Výsledky měření

Funkčnost skriptu komunikující přes proxy server s PLC byla ověřena na měření venkovní teploty. Ověřila se tak i funkčnost měření teploty přes I/O modul, který bude použit. Graf ze softwaru SoftPLC IDE je na obrázku 22 a graf ze skriptu je na obrázku 23.



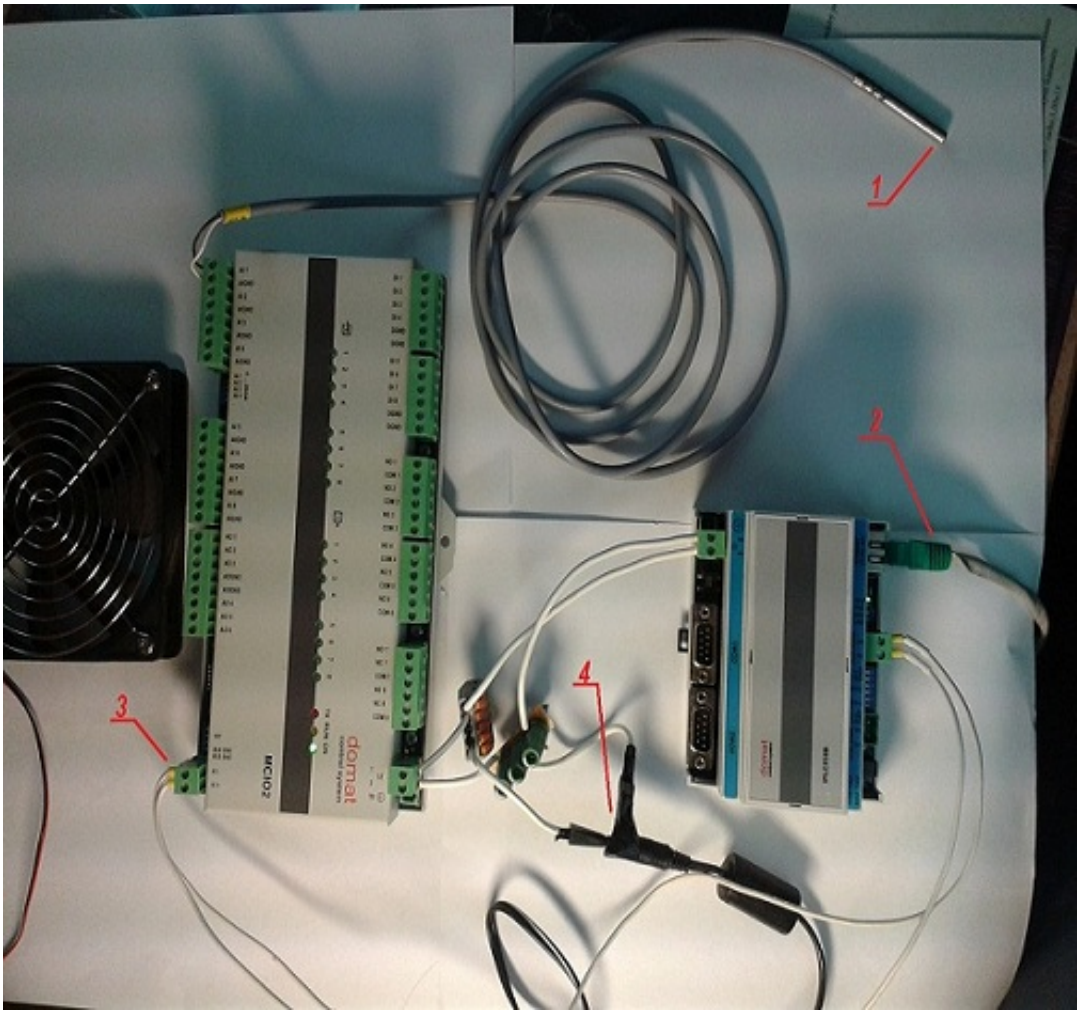
Obrázek 22: Graf teploty v programu SoftPLC IDE



Obrázek 23: Graf teploty v Matlab sriptu

Při porovnání obou grafů zjistíme, že jsou odměřená data stejná, a proto takto navržený skript může být použit pro komunikaci s PLC v průběhu experimentu.

Venkovní teplota byla změřena podle zapojení na obrázku 24. Jedná se o výsledné zapojení, kde bude všech 8 senzorů zapojeno stejně tak, jako zkušební senzor Ni1000 označený na obrázku.



Obrázek 24: Propojení PLC, modulu a senzoru

Označené komponenty jsou :

- **1** - Odporový senzor Ni1000, připojený na analogový vstup modulu.
- **2** - Síťový adaptér
- **3** - Datová komunikace s PLC
- **4** - Napájení modulu a PLC

6 Závěr

Práce je zaměřena na návrh experimentu, který bude svými vlastnostmi modelovat TABS systém a který umožní ověřit účinnost a spotřebu jeho energie. Rozměry desky, použité senzory, zařízení a všechny ostatní výše diskutované prvky byly pečlivě promyšleny a projednány s osobami, které s danou problematikou měli již mnoho zkušeností. Deska je tedy navržena tak, aby se neopakovaly již zaznamenané chyby a tak, aby měla co největší účinnost a nejmenší spotřebu energie.
Ve většině TABS aplikacích je použit konstantní průtok, což způsobuje omezené provozní podmínky. Možnost změny velikosti průtoku souvisí s velikostí vyzářené energie prostřednictvím vody (podle rovnice (1)), a proto jednotlivé výsledky pro různé průtoky budou pro tento experiment také klíčovou informací.
Deska bude zhotovena až v červnu roku 2013, proto jsem nemohl ověřit funkci regulace. Nicméně řešení regulace je uvedeno ve schématu na obr.19, respektive na obr.20. Takto navržená regulace by měla vystihovat všechny podmínky kladené na LLC.
Konstanty v PI regulátoru se navrhnou podle výše popsaného postupu, ale nejdříve se však musí odměřit přechodová charakteristika ventilu.
Naprogramoval jsem script v Matlabu, který komunikuje s PLC kontrolérem přes vzdálený proxy server. Script zaznamenává informace ze všech senzorů připojených na I/O modul. Lze v něm nastavit požadované hodnoty výstupů (celkový tok, požadovaná teplota).
Jednotlivé vstupy a výstupy I/O modulu byly v softwaru nastaveny na přesný typ zařízení, které do nich bude připojeno kvůli správnému přepočtu teploty. Přepočtenou teplotu ze senzorů modul odešle do PLC, který podle určité logiky nastaví výstupy na I/O modulu na určitou hodnotu.
Zkušební senzor Ni1000 po připojení na modul komunikoval bez problémů. Komunikace mezi PLC a Matlab skriptem zaznamenávajícím hodnoty ze senzorů byla ověřena a funguje bez problémů. Nutnou podmínkou této komunikace je možnost připojení PLC k síti a konfigurace PLC k zasílání hodnot na server.

7 Literatura

[1] Prof.Dr.ir. Jan Hensen, Model Predictive Control Model for Thermally Activated Building Systems, Master Thesis Project. Vystaveno roku 2011. Přístupné na: <http://www.bwk.tue.nl/bps/hensen/team/past/master/Sakellariou_2011.pdf>

[2] Dirk Saelens, Wout Parys, Ruben Baetens, Energy and comfort performance of thermally activated building systems including occupant behavior. Building and Environment 46(2010) 835-839. Přístupné na: <ftp://iristor.vub.ac.be/patio/ARCH/pub/fdescamp/bruface/literature/tabs/Saelens.pdf>

[3] Frauke Oldewurtel, Alessandra Parisio, Colin N. Jones, Dimitrios Gyalistras, Markus Gwerder, Vanessa Stauch, Beat Lehmann, Manfred Morari, Use of model predictive control and weather forecasts for energy efficient building climate control. Energy and Buildings 45 (2012) 15-18. Přístupné na: <http://infoscience.epfl.ch/record/176005/files/frauke_eab_2012.pdf>

[4] *Bc. Pavel Velecký*, Predikce v prediktivním řízení. Vystaveno roku 2010. Přístupné na: <http://dspace.k.utb.cz/bitstream/handle/10563/13990/veleck\%C3\%BD_2010_dp.pdf?sequence=1>

[5] Prediktivní řízení průmyslových procesů. Automa 2 (2007). Přístupné na: <http://www.odbornecasopisy.cz/index.php?id_document=34170>

[6] Programovatelný logický automat. Wikipedia. Vystaveno roku 2013. přístupné na: <http://cs.wikipedia.org/wiki/Programovateln\%C3\%BD_logick\%C3\%BD_automat>

[7] Popis softwaru RcWare. Přístupné na: <http://rcware.eu/public/rcware/softplc>

[8] Privara, S.,at al., 2010, Model predictive control of a building system: The first experience. Energy and Buildings, 2-3: 560 -572

[9] Olesen, B.W., 1997 Possibilities and limitations of radiant floor cooling. ASHRAE Transaction, 103:1

[10] Systémy REHAU pro vytápění / chlazení nosných betonových konstrukcí (BKT) - tipy pro projektanty TZB. REHAU, s.r.o. (2008). přístupné na: <http://www.tzb-info.cz/4579-systemy-rehau-pro-vytapeni-chlazeni-nosnych-betonovych-konstrukci-bkt-tipy-pro-projektanty-tzb>

8 Přílohy

8.1 Prediktivní řízení

8.1.1 Základní model MPC

Obecný MPC rámec řeší následující optimalizační problém [3]:

$$
\begin{aligned}
E(x_0) &= min \sum_{i=0}^{N-1} l_i(x_k, u_k) - \textit{nákladová funkce} && (7)\\
(x_i, u_i) &\in X_i \times U_i - \textit{omezení} && (8)\\
x_0 &= x - \textit{počáteční stav} && (9)\\
x_{i+1} &= f(x_i, u_i) - \textit{dynamika} && (10)
\end{aligned}
$$

Kde N je predikovaný horizont, X_i a U_i jsou omezené stavy ze stavu x_i a vstupu u_i v čase t. Základní části MPC tvoří nákladová funkce (1) a omezené stavy (2). Počáteční stav (3) slouží k počáteční inicializaci a Dynamika systému (4) musí být přesně modelována, abychom dostali uspokojivé výsledky.

Nákladová funkce Tato funkce popisuje požadované chování systému. Obecně slouží ke dvěma účelům :

- **Stabilita** Tato funkce se volí tak, aby byla zachována stabilita uzavřené smyčky. Tento požadavek se obecně uvádí pro systémy s pomalou dynamikou, jako jsou budovy.
- **Výkonové přizpůsobení** Většinou je požadavek ohledně výkonu jasně specifikován, ale může se stát že nebude vždy jednoznačný. Různé výkonové rozdíly můžou být použity pro specifikaci jednoznačné zpětné vazby. Například výkonové přizpůsobení můžeme nastavit z hlediska spotřeby energie nebo z hlediska komfortu. Obě cesty vedou na jinou sestavu zpětné vazby.

Základní typy běžně používaných nákladových funkcí jsou uvedeny v následující tabulce.

Nákladová funkce	**Matematický popis**
Kvadratická forma	$l_i(x_i, u_i) = x_i^T Q x_i + u_i^T R u_i$
Lineární forma	$l_i(x_i, u_i) = C^T U_i$
Pravděpodobnostní funkce	$l_i(x_i, u_i) = \xi\left[g_k(x_k, u_k)\right]$

Tabulka 4: Základní typy nákladových funkcí [3]

Kvadratická forma Určuje relativní vyváženost mezi stavy a vstupy. Volbou matic Q a R je poskytnut kompromis mezi kvalitou řízení a vstupní energií. Jestliže systém nemá žádné omezení nebo nejsou-li aktivní, pak bude kvadratická forma rovna lineárnímu kvadratickému regulátoru, což je klasická optimální regulace.

Lineární forma Pokud je nežádoucí regulační odchylka, pak je lepší volbou lineární forma. Tato funkce může být také klasická pro minimalizace spotřeby energie v budovách.

Pravděpodobnostní funkce Je-li systém ovlivněn náhodnými poruchami, pak může být požadovaná hodnota spočítána pomocí funkce $g_k(x_k, u_k)$ kde $g_k(x_k, u_k)$ může být například kvadratická či lineární forma diskutovaná výše.

8.1.2 Typy omezení

V praxi je použito mnoho typu omezení, některé běžně používané jsou uvedeny v tabulce 3. Síla prediktivního řízení spočívá ve schopnosti aproximovat tyto omezení a zařadit je do svých podmínek. Mimo omezení uvedená v tabulce, může být omezením také konstanta, protože některé systémy nemohou přesáhnout určitou hodnotu [3]. Například ventil nemůže být vychýlen více, jak do maximální polohy otevření tedy 100 %.

Typ omezení	Matematický popis
Lineární	$Ax_i < b$
Konvexní kvadratické	$(x_k - x)^T Q(x_k - x) \leq 1, Q \geq 0$
pravděpodobnostní	$P\left[Ax_k \leq b\right] \geq 1 - \alpha, \alpha \in (0, 0.5]$
Nelineární	$h(x_k, u_k) \leq 0$

Tabulka 5: Základní typy omezení [3]

Lineární omezení Toto omezení je nejpoužívanější a slouží k nastavení horní a dolní meze proměnné. Je nejlepší metoda k řešení optimalizační úlohy.

Konvexní kvadratická omezení Toto omezení je použito pro vázanou proměnou uvnitř elipsoidu. V řízení ovzduší v budovách by tento typ mohl představovat součet vstupní energie rozdělený mezi několik akčních členů.

Pravděpodobnostní Tento typ formuluje podmínky, které budou splněny s určitou pravděpodobností. Optimalizační problém může být v tomto případě řešen pouze pokud jsou všechny proměnné deterministické.

nelineární Toto omezení je kompromisem, jestliže nelze použít žádné z výše uvedených omezení. Funkce $h(x_k, u_k)$ je nějaká nelineární funkce. Je velmi obtížné manipulovat s tímto omezením při řešení optimalizačních úloh.

8.1.3 Počáteční stav

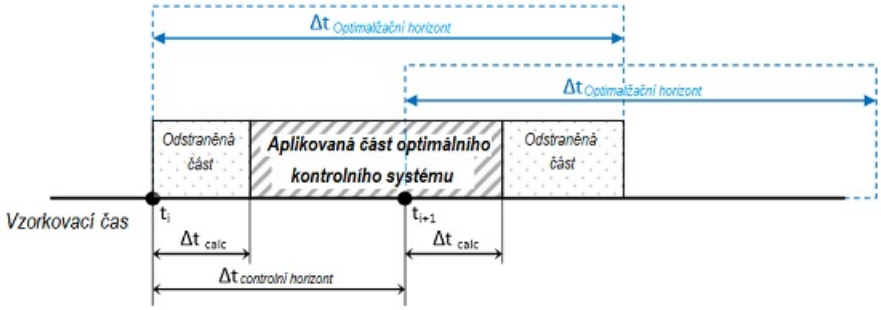
Tento stav nese informace o veškerých počátečních podmínkách. Definuje odkud se daný model bude dále vyvíjet. Pokud je nějaký stav neměřitelný, ale je pozorovatelný, pak můžeme použít co nejpřesnější odhad.

8.1.4 Dynamika

Model MPC je kritickou částí kontroleru. Dynamika systému určuje chování celého systému a díky známe dynamice víme, jak sestavit model na kterém simulujeme akční zásahy.

8.2 Klouzavý horizont

Jak již bylo řečeno, MPC kontrolér funguje na principu predikce klouzavého horizontu. Nechť budou 2 vzorkovací časy t_i a t_{i+1} vzdáleny o $\Delta t_{\textit{řídící horizont}}$. Na začátků každé vzorkovací doby je potřeba určitý čas pro řešení problému a následného doručení řídícího signálu. Tento čas může být označen jako Δt_{calk}. Aplikace řídícího signálu včetně doby výpočtu trvá přes celý optimalizační horizont $\Delta t_{\textit{optimalizační horizont}}$. Po odebrání druhého vzorku v čase t_{i+1} začne nový výpočet řídícího signálů pro nově změřené hodnoty. Můžeme tedy říci, že část řídícího signálu, který leží v intervalu $[t_i + \Delta t_{calk}, t_{i+1}\Delta t_{calk}]$ je aplikován do řízení budovy, zatímco zbytek signálu je zahozen [1].



Obrázek 25: Principiální schema klouzavého horizontu [1]

Takto zavedený klouzavý horizont přináší do systému zpětnou vazbu, jelikož každý nový řídící signál je počítaný pro stav, který sebou nese nejnovější informace ohledně poruch vstupujících do systému.

8.3 Typy modelů prediktivního řízení

Zde budou uvedeny 3 základní typy řízení. Jedná se o základní řízení RBC (Rule-Basic Control), deterministické řízení (DMPC) a stochastické řízení (SMPC). RBC je v praxi základním používaným řízením a je proto použit jako měřítko. DMPC se dá také považovat za základní prvek řízení, který se neustále rozvíjí. SMPC je nejnovější typ řízení vyvinutý pro TABS, který dokáže operovat(eliminovat) poruchy, které plynou z předpovědí počasí. Jednotlivé typy jsou podrobněji popsány níže [3].

8.3.1 RBC řízení

Základní typ řízení, který je založen na podmínce "Jestliže je splněna podmínka proveď akci". RBC tedy zajišťuje řízení všech vstupů v závislosti na různých podmínkách. Podmínky jsou většinou určeny prahovými podmínky, které musí být nutně určeny. Vlastnosti RBC řízení jsou kritický ovlivněny výběrem podmínek.

8.3.2 DMPC řízení

Pokročilejší typ řízení, který je použit ve všech komerčních MPC aplikacích. Jedná se o typ, který využívá nejisté předpovědí počasí a podle různých kritérií vybere nejvhodnější variantu řízení, kterou považuje za správnou. Obvyklá nastavení jsou : 24 hodinový horizont a hodinová vzorkovací perioda.

8.3.3 SMPC řízení

Typ, který má velmi dobré předpoklady k řízení klimatu uvnitř budov. Strategie spočívá ve dvou krocích. Určit nejistou předpověď počasí a formulovat šance pro náhodné poruchy.

9 Scripty v Matlabu

9.1 Třída SoftPLCProxy

```
classdef SoftPLCProxy

    properties (GetAccess = private, SetAccess = private)
        count_of_variables_in_one_request = 100;
        count_of_variables_in_one_write_request = 10;
        client = [];
    end

    properties (GetAccess = public, SetAccess = private)
        proxyurl;
        password;
        username;
    end



    methods
        function this = SoftPLCProxy(proxyurl, username, password)
            % Creates SoftPLCProxy class
            % Example of ussage
            % s = SoftPLCProxy('https://softplcproxy.rcware.eu', '
               K7qGcqA5gV8O', 'mPoaVFZgFg==')

            this.proxyurl = proxyurl;
            this.username = username;
            this.password = password;

            % add jsonlab into path
            pth=mfilename('fullpath');
            scriptName=mfilename;

            dp1 = [pth(1:end-length(scriptName)) 'dll' filesep '
               SoftPLCProxyMatlabClient.dll'];
            if ~SoftPLCProxy.IsAssemblyAdded( dp1 )
                NET.addAssembly(dp1);
            end
            dp2 = [pth(1:end-length(scriptName)) 'dll' filesep 'ESG.
               SoftPLCProxy.Shared.dll'];
            if ~SoftPLCProxy.IsAssemblyAdded( dp2 )
                NET.addAssembly(dp2);
            end
```

```
            import ('SoftPLCProxyMatlabClient.SoftPLCProxyMatlabClientImplementation');
            this.client = SoftPLCProxyMatlabClientImplementation(this.proxyurl, this.username, this.password);
        end


        function varStruct = GetVariables(this,varargin)
            % Returns array of structs containing all accesible variables
            % Args:
            % - none i.e. GetVariables():returns all variables in the system OR
            % - cell of guids i.e. GetVariables(guids) : return variables with the specified guids.
            vars = this.client.GetVariables();
            varStruct = struct([]);
            guids={};
            for i=1:vars.Length
                var = vars(i);
                varStruct(i).Guid = var(2).char;
                guids{i}=varStruct(i).Guid;
                varStruct(i).Name = var(1).char;
                varStruct(i).NamePrefix = var(3).char;
                varStruct(i).Type = var(4).char;
                varStruct(i).Unit = var(5).char;
              end
           % filter out the required guids
            if numel(varargin)>0
                [~,ind]=ismember(varargin{1},{varStruct.Guid});
                varStruct=varStruct(ind);
            end
        end

        function valStruct = GetValuesByVariables(this, vars)
            % Returns array of structs containing the most recent values
            %    vars - array of structs returned by GetVariables() function

            % at first, initialize System.String array of an appropriate dimension
            nvars = numel(vars);
            arrGuid = NET.createArray('System.String',nvars);
```

```
            % fill the array with guids
            for i=1:nvars
                arrGuid(i)=vars(i).Guid;
            end

            % download data
            results = this.client.GetValues(arrGuid);
            if double(results.Length)~= nvars
                error('SoftPLCProxy:GetValuesByVariables:NotEnoughDataInDb','NotEnoughDataInDb - server returned different number of variables that was desired');
            end
            % process data
            valStruct = vars;
            for i=1:nvars
                result = results(i);

                valStruct(i).Quality = result(2).char;
                valStruct(i).UpdateTime = datenum(result(3).char,'dd.mm.yyyy HH:MM:SS'); % in UTC

                switch vars(i).Type
                    case {'VT_Int64', 'VT_Double'}
                        valStruct(i).Value = str2double(result(1).char);
                    case 'VT_DateTime'
                        warning('SoftPLC:GetValues:NotImplemented','Not yet implemented-unrecognized format of date');
                        valStruct(i).Value = result(1).char;

                    case {'VT_Bool'}
                        if strcmpi(result(1).char,'True')
                            valStruct(i).Value = true;
                        else
                            valStruct(i).Value = false;
                        end
                    case {'VT_String'}
                        valStruct(i).Value = result(1).char;
                    otherwise
                        %valStruct(i).Value = result(1).char;
                        error('SoftPLCProxy:unrecognizedType','Unrecognized file type');
```

```
            end
        end
    end

    function vals = GetValuesByGuids(this, guids)
        % Returns array of structs containing the most recent values
        %    guids - cell of guid strings
        vars=this.GetVariables(guids);
        vals=this.GetValuesByVariables(vars);
    end

    function vals = GetValuesByName(this,Name_1)
        vars = this.GetVariables();
        vars1 = this.GetValuesByVariables(vars);
        delka = length(vars1);
        try
        names = {vars.Name};
        catch
            vals = 0;
        end
       for i=1:delka
          if (strcmp(names(i),Name_1))
              vals = vars1(i).Value;
              return
          else
              vals = 0;
          end
       end
    end

    function vals = GetValuesVectorByVariables(this, vars)
        % Returns vector containing the most recent values
        %    guids - cell of guid strings
        valStruct = this.GetValuesByVariables(vars);
        try
            vals = [valStruct.Value];
        catch %not the same data type
            vals = {valStruct.Value};
        end
    end

    function vals = GetValuesVectorByGuids(this, guids)
```

```
157             % Returns vector containing the most recent values
158             %    guids - cell of guid strings
159             valStruct = GetValuesByGuids(this, guids);
160             try
161                 vals = [valStruct.Value];
162             catch %not the same data type
163                 vals = {valStruct.Value};
164             end
165         end
166
167         function SetValues(this, varStruct)
168             %set values
169             % input args:
170             % varStruct - array of vars that contain also field Value
171             nvars = numel(varStruct);
172             arrGuid = NET.createArray('System.String',nvars);
173             arrValue = NET.createArray('System.String',nvars);
174
175             % fill the array with guids
176             for i=1:nvars
177                 arrGuid(i)=varStruct(i).Guid;
178                 switch varStruct(i).Type
179                     case {'VT_Int64', 'VT_Double'}
180                         arrValue(i)=num2str(varStruct(i).Value);
181                     case 'VT_DateTime'
182                         warning('SoftPLC:SetValues:NotImplemented','
                                Not yet implemented-unrecognized format of
                                date');
183                         arrValue(i)=varStruct(i).Value;
184                     case {'VT_Bool'}
185                         if varStruct(i).Value
186                             arrValue(i)='True';
187                         else
188                             arrValue(i)='False';
189                         end
190                     case {'VT_String'}
191                         arrValue(i) = varStruct(i).Value;
192                     otherwise
193                         error('SoftPLCProxy:unrecognizedType','SoftPLC
                                - Unrecognized file type');
194
195
196                 end
197             end
198
```

```
            this.client.SetValues(arrGuid,arrValue);
        end
    end

    methods (Static)
        function flag = IsAssemblyAdded( MyName )
            % obtained from http://stackoverflow.com/questions
               /5368974/how-to-check-if-net-assembly-was-already-added
               -in-matlab
            domain = System.AppDomain.CurrentDomain;
            assemblies = domain.GetAssemblies;
            flag = false;
            for i= 1:assemblies.Length
                asm = assemblies.Get(i-1);
                %disp(char(asm.FullName));
                if strcmpi(asm.Location.char, MyName)
                    flag = true;
                end
            end
        end
    end
end
```

9.2 GUI MATLAB

```
function varargout = Beton(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Beton_OpeningFcn, ...
                   'gui_OutputFcn',  @Beton_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end



function Beton_OpeningFcn(hObject, eventdata, handles, varargin)

handles.client  = SoftPLCProxy('https://softplcproxy.rcware.eu', '
    lSgibAeTkb2d', 'A0KrMKwtYg==');
handles.output = hObject;
handles.isLoaded = 0;
handles.x = 0;
handles.x_min = 0;
handles.y = 0;
handles.y_min = handles.client.GetValuesByName('TABS_LAB.Ambient temp'
    );
handles.value_y = 0;
handles.value_y_min = 0;
handles.graf = line(0,0);
handles.pocatek = true;
handles.first = true;

assignin('base','client',handles.client);
ylabel('°C');

guidata(hObject, handles);

```

```
function varargout = Beton_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;



function button_load_Callback(hObject, eventdata, handles)

input_def_names = {'TABS_LAB.Ambient temp' 'TABS_LAB.Return water temp' 'TABS_LAB.Suply water temp' ...
    'TABS_LAB.Surface temp' 'TABS_LAB.Concrete core temp_1' 'TABS_LAB.Concrete core temp_2' ...
    'TABS_LAB.Concrete core temp_3' 'TABS_LAB.Concrete core temp_4'};
 output_def_names = {'circulation pump set point' 'suply water temp set point'};
 output_jednotky = {'l/h',' °C'};

set(handles.pop,'Value',2);
vars = handles.client.GetVariables();
variables = handles.client.GetValuesByVariables(vars);

try
names = {vars.Name};
catch
   for x=1:length(vars)
       names(x)='';
   end
end
write = false;
delka_jmen = length(input_def_names);
indexOfValue  = 0;

% setting variables and values from proxy
for i = 1:length(input_def_names) % delka zobrazovacich polí
   str = num2str(i);
   teploty = strcat('temp',str);
   hodnoty = strcat('hodnota',str);
    for j =1:length(vars)

                if (strncmpi(names(j),input_def_names(i),20))
                    write = true;
                    indexOfValue = j;
                end
```

```
    end
    if write
       set(handles.(teploty),'string',input_def_names(i));
       set(handles.(hodnoty),'ForegroundColor','white');
       set(handles.(hodnoty),'string',strcat(num2str(variables(
          indexOfValue).Value),' °C'));
    else
      set(handles.(teploty),'string','Not Avalaible');
      set(handles.(hodnoty),'ForegroundColor','red');
      set(handles.(hodnoty),'string','NULL');
    end

    write = false;
end

write = false;

for k = 1:length(output_def_names) % delka zobrazovacich polí
   str = num2str(k);
   teploty = strcat('tempo',str);
   hodnoty = strcat('Req_out',str);
    for o =1:length(vars)

                if (strncmpi(names(o),output_def_names(k),20))
                    write = true;
                    hodnota = handles.client.GetValuesByName(
                       output_def_names(k));
                    if(k == 2)
                        handles.value_y_min = handles.value_y;
                        handles.value_y = hodnota;
                    end
                end
    end
    if write
       set(handles.(teploty),'string',output_def_names(k));
       set(handles.(hodnoty),'string',strcat(num2str(hodnota),
          output_jednotky(k)));

    else
      set(handles.(teploty),'string','Not Avalaible');
    end

    write = false;
end

```

```
handles.isLoaded = 1;

if (handles.first)
handles.graf = line([0 10],[handles.value_y handles.value_y]);
set(handles.graf,'Color','red','LineWidth',2);
grid on;
handles.first = false;
end
     if(handles.value_y_min ~= handles.value_y)
         delete(handles.graf);
         handles.graf = line([0 10],[handles.value_y handles.value_y])
            ;
         set(handles.graf,'Color','red','LineWidth',2);
     end

   guidata(hObject, handles);




function button_set_Callback(hObject, eventdata, handles)

if (handles.isLoaded)
vars    = handles.client.GetVariables();
variables = handles.client.GetValuesByVariables(vars);

 output_def_names = {'circulation pump set point' 'suply water temp
    set point'};
 names = {vars.Name};
index_value = 0;

for i = 1:length(output_def_names) % delka zobrazovacich polí
    for j =1:length(vars)

                if (strncmpi(names(j),output_def_names(i),20))
                    write = true;
                    index_value = j;
                end
    end
    if (write && i==1)
       variables(index_value).Value = str2num(get(handles.out1,'string
          '));
       handles.client.SetValues(variables(index_value));
```

```
        set(handles.out1,'string','');
    end
    if ( write && i==2)
     variables(index_value).Value = str2num(get(handles.out2,'string')
        );
     handles.client.SetValues(variables(index_value));
     set(handles.out2,'string','');
    end

    index_value = 0;
    write = false;
end
end



function out1_Callback(hObject, eventdata, handles)
    num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String','');
    errordlg('Input must be a number', 'Error');

elseif (~handles.isLoaded)
    set(hObject,'String','');
    errordlg('Data must be loaded', 'Error');
end


function out1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
   defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function pop_Callback(hObject, eventdata, handles)
 while ((get(handles.pop,'Value')== 2) && (handles.isLoaded))


     handles.x = (handles.x)+0.003;
     handles.y = handles.client.GetValuesByName('TABS_LAB.Ambient temp
        ');
```

```
  x_sour = [handles.x_min handles.x];
  y_sour = [handles.y_min handles.y];
 % set(handles.graf,'NextPlot','add');

   l = line(x_sour,y_sour);
   set(l,'Color','green','LineWidth',2);
   grid on;

   handles.x_min = handles.x;
   handles.y_min  = handles.y;

   button_load_Callback(hObject, eventdata, handles);
       pause(5);
     end

function pop_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function out2_Callback(hObject, eventdata, handles)
 num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String','');
    errordlg('Input must be a number', 'Error');
    elseif (~handles.isLoaded)
    set(hObject,'String','');
    errordlg('Data must be loaded', 'Error');
end

function out2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```


Beton
Input Variables
TABS_LAB.Ambient temp
TABS_LAB.Return water temp
TABS_LAB.Suply water temp
TABS_LAB.Surface temp
TABS_LAB.Concrete core temp_1
TABS_LAB.Concrete core temp_2
TABS_LAB.Concrete core temp_3
TABS_LAB.Concrete core temp_4
Values
28.6994 °C
3946.0263 °C
3946.0263 °C
3946.0263 °C
3946.0263 °C
3946.0263 °C
3946.0263 °C
3946.0263 °C
Output Variables
circulation pump set point
suply water temp set point
New Outputs
Actual Outputs
300l/h
28 °C
°C
35
30
25
20
15
10
5
0
0
1
2
3
4
5
6
7
8
9
10
Load Variables
On-line
Set Output