VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

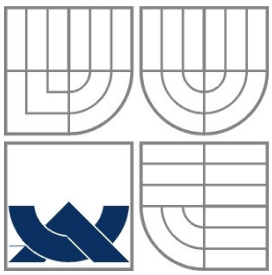
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM PRO DOMÁCNOST

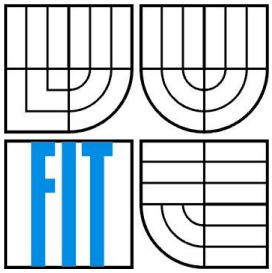
SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE ALEŠ KOLLNER
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM PRO DOMÁCNOST

HOUSEHOLD INFORMATION SYSTEM

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

ALEŠ KOLLNER

VEDOUCÍ PRÁCE
SUPERVISOR

ING. IGOR SZÖKE

BRNO 2007

Informační systém pro domácnost

Odevzdáno na Fakultě informačních technologií Vysokého učení technického v Brně, dne 4.1. 2008

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením Ing. Igora Szökeho. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Aleš Kollner
2.1.2008

Poděkování

Na tomto místě bych rád poděkoval Ing. Igoru Szökemu za odborné vedení, užitečné rady, připomínky a cenné konzultace po celou dobu vypracovávání semestrálního projektu.

Abstrakt

Semestrální projekt se zabývá prvotní analýzou a návrhem informačního systému pro domácnost. Systém je navržen pomocí univerzálního jazyka Unified Modeling Language (UML). Informační systém pro domácnost by měl umožňovat uživateli snadný přehled o financích a hospodaření domácnosti. Systém by měl být schopen předpovídat spotřebu energií v domácnosti. Implementován bude pomocí skriptovacího jazyka PHP a relačního databázového systému MySQL.

Klíčová slova

Unified Modeling Language , diagram případu užití, informační systém, PHP, MySQL, HTML, CSS, aktér, include, extends

Abstract

Term project is engaged in early analysis and proposal of household informational system. System has been designed with universal language Unified Modeling Language (UML). The household infromational system should make easy survey about a pocket book and economy of household possible to user. System should be able to make prediction of energy usage household. It will be implemetated with PHP language and relational database system MySQL.

Keywords

Unified Modeling Language , use case diagram, informational system, PHP, MySQL, HTML, CSS, actor, include, extends

Obsah

1 Úvod

Cílem mé diplomové práce je vytvoření informačního systému pro domácnost, který bude přístupný uživatelům prostřednictvím sítě Internet. V semestrálním projektu provedu analýzu požadavků a návrh informačního systému pro domácnost. Informační systém bude dostupný přes webové rozhraní, a proto se musím seznámit s požadavky kladenými na tvorbu webových informačních systémů a pro jeho tvorbu vybrat vhodné prostředky. Rozhodl jsem se použít jazyk HTML, CSS, skriptovací jazyk PHP, databázový systém MySQL a pro návrh použiji modelovací techniku UML.

Na základě získaných vědomostí provedu analýzu požadavků kladených na informační systém a navrhnu vhodný model systému pomocí metod UML. Podle navrhnuté koncepce v diplomové práci vytvořím informační systém pro domácnost. Při tvorbě systému se budu snažit, aby byl systém co nejvíce přehledný a snadno ovladatelný.

2 Vývoj informačního systému

Vývoj informačního systému můžeme rozdělit do pěti základních etap:

Analýza a specifikace požadavků – úvodní etapa vývoje softwaru. Představuje jeden z nejtěžších úkolů při řešení každého softwarového projektu. Obtížnost vyplívá z problémů souvisejících s komunikací mezi lidmi. Uživatel není často schopen zformulovat jasně svůj požadavek, může dávat přehnané požadavky na systém. Uživatel zadává požadavky, které si vzájemně odporují nebo jsou v rozporu s požadavky jiných uživatelů. Komunikace je založena na přirozeném jazyce, vývojáři proto často používají různé diagramy ke zpřesnění požadavků.

Architektonický a detailního návrh – je popis struktury softwarového systému, dat, rozhraní komponent, uživatelského rozhraní a případně i použitých algoritmů. V případě informačních systémů popis dat představuje návrh struktury databáze. Teoreticky by měl návrh začínat tam, kde končí analýza požadavků. Analýza znamená modelování bez omezení vyplívajících z úvah o implementaci, naopak návrh bere do úvahy platformu, na níž bude systém implementován. Hranice mezi analýzou a návrhem neexistuje, obě fáze se do určité míry překrývají. Proto existují dva hlavní důvody. Prvním je fakt, že se používá inkrementální a iterativní model životního cyklu. Etapy životního cyklu se v nich opakují v iteracích, kdy výsledkem každé iterace je přírůstek funkčnosti systému. Druhým důvodem je, že se pro modelování se v současnosti používá jazyk UML. Ten poskytuje prostředky, jak pro modelování požadavků tak návrhu. Přechod od analýzy k návrhu neznamená vytvoření nového modelu návrhu, který by byl striktně oddělen od modelu analýzy. Jde spíše o rozpracování modelu analýzy. Toto rozpracování je výsledkem detailního návrhu, který bývá uváděn jako jeden ze dvou aspektů návrhu. Tím druhým je architektonický návrh, který slouží k ujasnění koncepce systému a jeho dekompozici, která vyžaduje vymezení funkcionality jednotlivých podsystému a definování vztahů mezi nimi. Podrobný návrh se soustřeďuje na podrobnou specifikaci softwarových součástí, na stanovení logické a fyzické struktury dat a způsobu ošetřování chybových a neočekávaných stavů.

Implementace - znamená především programování. Neznamená pouhé kódování návrhu do příkazu programování jazyka. Uváděli jsme si, že návrh algoritmů není vždy součástí návrhu systému. Řadu algoritmů ve skutečnosti navrhuje až programátor v rámci psaní programu. V dnešní době se pro implementaci používají CASE (Computer – Aided/Assisted Software Engineering) systémy a nástroje a integrovaná vývojová prostředí, která jsou schopna generovat kostru programu na základě modelu návrhu. Do takové kostry potom programátor přidává úseky kódu, které tvoří podstatnou část programu. U mnoha projektů je implementace nejdelší fází a dominantní etapou vývoje systému.

Integrace a nasazení – znamená sestavení systému z již implementovaných a otestovaných komponent a následné nasazení u zadavatele k využívání. Informační systémy jsou typicky rozsáhlé

systémy sestávající z řady podsystémů, balíčků a komponent, z nichž některé mohou být implementovány souběžně, jiné postupně. V každém případě vzniká situace, kdy máme řadu implementovaných komponent, ze kterých potřebujeme sestavit celý systém nebo část. I když jsme v rámci implementace otestovali jednotlivé komponenty a odstranili nalezené chyby, není to zárukou, že po jejich spojení bude vše v pořádku. V případě klíčovou aktivitou integrace je testování, které se v tomto případě označuje jako integrační testování. Cílem integračního testování je ověření funkčnosti celého systému, resp. předávané části. Funkčnost systému je potřeba ověřovat postupně přidáváním dalších komponent. Takové postupné integrační testování lze provést shora dolů nebo zdola nahoru. Integrační testování vyžaduje určité programování navíc pro vytvoření náhražek a ovladačů. Nasazení není zpravidla jednorázovou aktivitou. Často bývá v souladu s inkrementálním vývojem nasazován informační systém postupně prostřednictvím verzí. Každá verze je tvořena řadou přírůstků vytvořených v několika iteracích iterativního vývoje. V tomto případě především systémové a přejímací testování. V prvním případě jde o testování prováděné vývojáři za podmínek blízkých provozním podmínkám. Ve druhém případě probíhají testy u zákazníka a jejich cílem je ověřit, že zákazník dostává produkt, jaký požadoval.

Provoz a údržba – značí období životního cyklu, kde je systém rutinně používán a rozvíjen. Zahájení provozu nového systému má za následek ukončení činnosti předchozího systému. Ukončení ale zpravidla nebývá okamžitě, nýbrž po určitou doby bývají systémy provozovány souběžně. Starý systém tak slouží jako určitá pojistka, kdyby nový systém selhal. Zahájením provozu nového systému současně znamená zahájení údržby tohoto systému. Údržbou software je chápáno více než je běžné chápání údržby technických zařízení. Údržba informačních systémů typicky plánována a náklady na ni odhadnuty již v počátečních fázích životního cyklu. Do údržby je zahrnuto nejen odstranění vzniklých problémů, ale i další rozvoj systému.

2.1 UML

Jazyk UML(Unifed Modeling Language) je univerzální jazyk pro vizuální modelování systémů, přestože je nejčastěji spojován s modelováním objektově orientovaných softwarových systémů, ale UML má mnohem širší využití vyplívajících z jeho zabudovaných systémů.

UML má již ve svém názvu slovo unifikovaný, protože jedním z jeho cílů je sjednocení používaných výrazových prostředků. Jazyk UML nabízí vizuální syntaxi pro modelování během celého vývojového cyklu projektu, požadavky na analýzu počínaje a implementací konče. UML je jazyk s bohatou sémantikou a syntaxí. Tvůrci jazyka UML věřili, že jazyk kompletně pokryje veškeré požadavky návrhářů, a proto přímo do jazyka zabudovali mechanismy rozšíření, které usnadňují deklaraci a definici nových elementů jazyka. UML umožňuje modelovat jednoduché i ty nejsložitější aplikace pomocí stejné formální syntaxe, a proto mohou výsledky své práce sdílet s ostatními návrháři.

Jazyk UML je složen ze třech hlavních stavebních bloků:

Předměty – do předmětů jsou řazeny samotné elementy modelu.

Vztahy(relace) – relace jsou pojítkem mezi předměty, které určují jak spolu dva nebo více předmětů spolu významově souvisejí. Typickým příkladem mohou být vztahy v rodině.

Diagramy – model UML se skládá z různých diagramů, jež představují pohledy na různé části sémantického základu navrhované aplikace. Sémantickým základem je souhrn specifikace aplikace, který vymezuje prostor v němž se může návrh pohybovat. Diagram ve vizuální podobě popisuje konkrétní aplikaci. Hlavním principem UML je, že každý model se může vyjádřit s rozlišnou přesností a na každý problém se můžeme dívat z několika pohledů.

UML chápe architekturu systému jako organizační strukturu systému včetně dekompozice na části, jejich propojení a mechanismů a principů návrhu systému. Jazyk UML rozeznává 4 + 1 základních pohledů na architekturu systému:

Logický pohled – zachycuje slovník domény řešeného problému jako množinu tříd a objektů. Důraz je kladen na to jak objekty a třídy implementují požadované chování

Procesní pohled – soustřeďuje se na chování systému, které musí splňovat požadavky a omezení z případu užití. Je to ve skutečnosti procesně orientovaná varianta logického pohledu

Implementační pohled – modeluje soubory a komponenty, které tvoří fyzický kód systému. Ukazuje rovněž závislosti komponent a souvisí se správou konfigurace při definování verzí systému.

Pohled nasazení – mapuje komponenty na množinu fyzických výpočetních uzlů v cílovém prostředí.

Pohled případu užití – zachycuje základní požadavky na systém v podobě množiny případu užití. Tyto případy použití jsou základem pro konstrukci dalších pohledů.

2.1.1 Diagram případu užití

Jsou hlavní technikou modelování chování systému na úrovni analýzy v UML. Lze vytvořit řadu diagramů, které reprezentují různé aspekty systému na úrovni podrobností.Diagram případu použití má v UML zvláštní postavení a to ze dvou důvodů. Jednak jeho síla nespočívá v samotném grafickém vyjádření, ale ve větší míře v textové specifikaci případů použití, které odráží požadavky uživatelů. Druhým důvodem je to , že právě tyto specifikace jsou významným zdrojem informací a návodem pro vývojáře ve všech fázích životního cyklu.

Aktér

Aktér je reprezentován jednoduchou ikonou lidské postavy. Jméno aktéra se uvádí přímo pod ikonu postavy. Aktéři jsou z pohledu systému externími entitami, přesto systém většinou udržuje o aktérovi určitou interní reprezentaci. Tři základní typy aktérů:

Fyzická osoba

Jiný systém, který se nachází mimo hranice aplikace

Čas se stává aktérem, když určitým počtem časových triggerů uvede v chod specifickou událost, např. automatická záloha systému

Případ užití

Nejlépe definujeme případy užití, pokud si projdeme seznam aktérů a zvážíme jakým způsobem budou systém využívat. Pomocí této strategie můžeme vytvořit seznam případu užití pro jednotlivé aktéry. Modelování případu užití je iterativní proces kdy postupně upřesňujeme případy užití.

Případ užití reprezentuje část funkcionality systému, který tvoří logický celek. Pomocí případu užití umožníme zákazníkovi pohled na celý systém a jeho funkce. Případy užití by měli kompletně popsat systém.

Případ užití: **Vložení zákazníka**
***Stručný popis*:** Systém nabídne obchodníkovi formulář pro zadání nového zákazníka.
***Aktéři*:** Obchodník
***Tok událostí*:** 1. Případ užití se spustí, když obchodník klikne na stránce „vytvoření nového zákazníka“ 2. Dokud nejsou zadané platné všechny povinné údaje: 2.1. Osobní údaje – jméno, příjmení 2.2. Kontaktní údaje – adresa, telefon, email 3. Systém ověří zadané údaje 3.1. Zadány všechny povinné údaje 3.2. Správný tvar zadaných údajů 4. Systém vytvoří nového zákazníka a vloží jej do DB
***Následné podmínky*:** Informace o zákazníkovi jsou uloženy do databáze

Obrázek 1. Případ užití

Vztahy mezi případy užití

Asociace – znázorňuje komunikaci mezi aktérem a případem užití. Asociace znázorňujeme šipkou mezi aktérem a případem užití

Zahrnutí (include) – chování určené jedním příkladem užití je zahrnuto v jiném případu užití. Vztah <<include>> znamená, že dodatkový případ použití je proveden vždy, když je proveden jeho bázový případ použití.

Rozšíření (extend) – způsob vytvoření nového případu použitím rozšíření z již existujícího případu užití. Vztah <<extend>> znamená, že rozšiřující případ použití je proveden pouze za jistých okolnosti, když je proveden jeho bázový případ užití.

Generalizace – používáme v případě, kdy účastníkům (aktér, případ užití) můžeme vyčlenit chování, které je společné dvěma a nebo více účastníkům do jednoho rodičovského účastníka. Rodičovský účastník je obecnější a jeho potomci jsou specializovanější než rodič. Potomci dědí všechny vlastnosti a funkce od svých předků a navíc mohou být doplněny o nové funkce a vlastnosti. Rodičovský účastník bývá obvykle abstraktní, a proto potomka můžeme dosadit všude tam, kde jsme mohli očekávat výskyt rodičovského účastníka.

3 Analýza informačního systému

Prvním krokem při vývoji softwaru je provedení analýzy a specifikace požadavků na informační systém pro domácnost. Cílem analýzy a specifikace požadavků je stanovení služeb, které požaduje zákazník od systému. Zákazník také musí vymezit podmínky pro vývoj a provoz systému. Smyslem této etapy vývoje softwaru je především:

Získání, analyzování a definovaní požadavků – použitím vhodných prostředků musíme zjistit o jaký softwarový produkt má zákazník skutečně zájem a musíme co nejpřesněji vymezit jeho funkcionalitu. Důraz je kladen na požadavky uživatele, nezajímáme se o to, jak požadavky budou realizovány. Cílem této etapy je získání představy o všech funkcích systému, které si zákazník přeje.

Transformace neformálních požadavků do strukturovaného popisu požadavků – pro komunikaci se zákazníkem používáme přirozený jazyk, který není jednoznačný. Proto může vzniknout problém významu konkrétních požadavků od zákazníka. Zákazník se nemusí přesně vyjádřit a nebo analyzátor nemusí pochopit představu zákazníka o konkrétním požadavku. Proto musíme provézt transformaci neformálních požadavků na jednoznačný seznam formálních požadavků.

3.1 Analýza požadavků

Ke správnému návrhu systému jsem musel určit potřebné požadavky, které na systém klade zákazník a potencionální uživatelé, kteří budou systém využívat. Od zákazníka jsem získal tyto neformální požadavky.

3.1.1 Neformální požadavky

Vytvoření informačního systému pro domácnost, který bude přístupný pomocí webového rozhraní. Systém bude umožňovat přihlášení různých uživatelů k systému. Cílem je navrhnout a implementovat informační systém pro zjednodušení správy a chodu domácnosti. Měl by obsahovat části, jako příjmy-výdaje, energie, „sklad“. Být modulární s možností přidání modulů „kuchařka v závislosti na tom, co je právě v lednici“ atd.

3.1.1.1 Nákupy

Nákupy – implementace „domácího skladu“, pokud uživatel přijde z nákupu, vloží věci které nakoupil do systému. Přiřazení nakoupených výrobků do kategorií, podpora průběžné tvorby kategorií. Kategorie výrobků by měli mít stromovou strukturu. Při vkládání nakoupených věcí do systému je možnost zadat kdo a pro koho jej koupil.

Plánování nákupu – do systému můžeme vkládat jednotlivé obchody a typ zboží které nabízejí. Záložka „Co je třeba koupit“, kde uživatel vkládá co je třeba koupit. Záložka „Jdu nakoupit“ kde si uživatel vybere, do kterého obchodu jde nakoupit a informační systém vygeneruje seznam věci, které by měl v konkrétním obchodu koupit.

Plánování velkých výdajů – možnost zadání do systému větší finanční výdaj na určitou dobu a předpokládanou cenu (např. dovolená)

3.1.1.2 Výdaje a příjmy

Přehled výdajů – z nákupů bude možné udělat přehled výdajů podle kategorií za období, podrobnější průchod výdaji. Přehled by měl být i pro „kdo nakoupil“ a „pro koho nakoupil“. Zadání pravidelných výdajů jako jsou zálohy za energie, nájemné, paušál za mobil. Systém by měl nabídnout i podporu půjček a vložení splátek. U příjmů i výdajů bude možno nastavit předpokládanou hodnotu v určitém období. Takto můžeme předpovídat hospodaření domácnosti na určitou dobu dopředu.

3.1.1.3 Energie

Systém bude mít přehled o výdajích za elektrickou energii, zemní plyn, vodné a stočné. U elektrické energie rozlišovat denní a noční proud, stejně tak i u vodného rozlišit teplou a studenou vodu. Uživatel bude moci měnit aktuální cenu za energii za jednotku. V systému budu mít možnost nastavení paušálu za energie. Ke každému měřidlu energií bude určen seznam zúčtovacích období a stavy měřidel ke konkrétnímu datu. Z těchto záznamů bude systém odhadovat kolik je skutečný účet za energie v daném období.

3.1.1.4 Předpověď spotřeby energií

Informační systém si bude z internetu stahovat průběhy venkovních teplot z předchozích let, z nichž si systém bude počítat průměr teplot pro určitý čas a bude schopen určovat přibližnou spotřebu energie v nadcházejícím období pro domácnost. Podobný princip využije IS pro solární systém, kde si bude stahovat údaje o slunečním svitu, intenzitu slunečních paprsků a z něj bude systém určovat ušetřenou energii.

3.1.2 Formální požadavky

Neformální požadavky jsem podrobil analýze, abych zjistil, které požadavky je schopen mnou navrhovaný systém splnit. Tyto požadavky nazveme funkčními požadavky. Požadavky, které navrhovaný systém nesplňuje, označujeme jako nefunkční a nadále s nimi při návrhu nebudeme pracovat. Po zjištění funkčních požadavků se pokusím o rozšíření požadavků, které je nutné realizovat pro lepší funkčnost systému. Seznam funkčních požadavků se pro mě stal základem pro další práci při návrhu systému.

3.1.2.1 Funkční požadavky

Seznam funkčních požadavků informačního systému:

- Registrace nového uživatele
- Přihlášení do systému
- Odhlášení ze systému
- Prohlížení a editace osobních údajů
- Změna přihlašovacího hesla
- Přidávání dalších členů rodiny
- Vkládání do skladu
- Odebírání ze skladu
- Přidávání kategorii výrobků
- Přidávání obchodů
- Editace obchodů
- Vkládání do nákupního seznamu
- Editace nákupního seznamu
- Nový nákup
- Přehled nákupů
- Přehled výdajů
- Přehled příjmů
- Nastavení záloh a paušálů
- Mimořádné výdaje
- Splátky, hypotéky
- Nastaveni aktualizací
- Nastavení cen energií
- Editace cen energii
- Nastavení domácnosti
- Zadávání stavu měřidel
- Prohlížení stavu měřidel
- Přehled průběhů spotřeby
- Předpověď spotřeby energie

3.1.2.2 Nefunkční požadavky

- Kuchařka

3.2 Uživatelé

Ze seznamu formálních požadavků vyplívá nutnost zavedení více typů uživatelů, kteří budou moci přistupovat k systému. Každý typ uživatelů bude mít jiná práva. Uživatele můžeme rozdělit na tři typy:

neregistrovaný uživatel

hlava rodiny

člen rodiny

3.2.1 Neregistrovaný uživatel

Typ uživatele systému mající minimální práva. Neregistrovaný uživatelem je náhodný návštěvník stránek, kterému je nabídnuto základní rozhraní hlavní stránky. Na hlavní stránce si může přečíst co systém umožňuje a jaké funkce nabízí po registraci. Dále má na výběr formulář pro přihlášení do systému a také možnost nové registrace. Po správném vyplnění registračních formulářů se z neregistrovaného uživatele může stát typ uživatele „hlava rodiny“.

3.2.2 Hlava rodiny

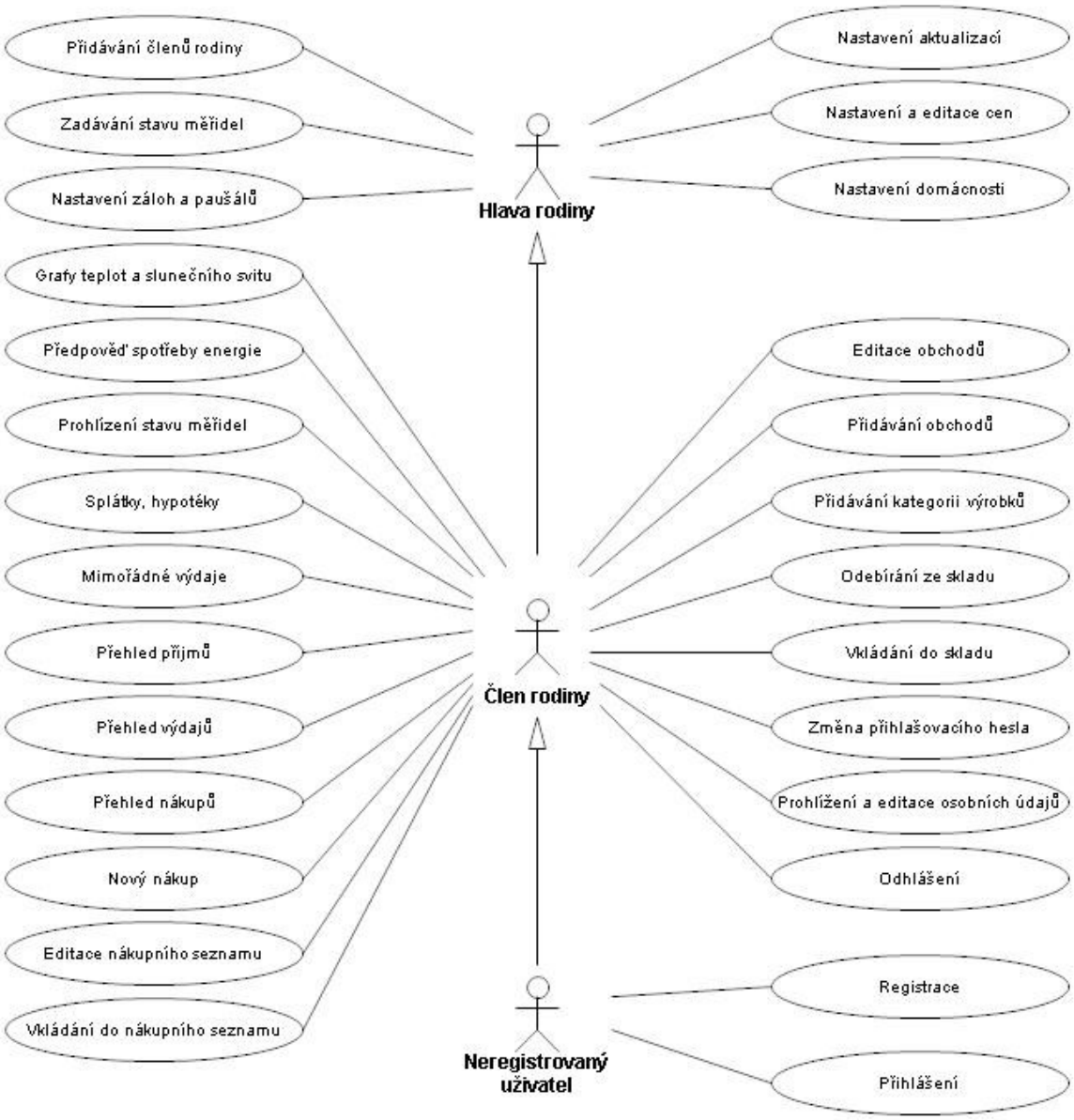
Oproti předešlému typu uživatele se liší tím, že už je v systému zaregistrován. K tomu aby mohla hlava rodiny využívat svých funkcí, které mu systém nabízí musí být přihlášen. Hlava rodiny může být v každé domácnosti právě a vždy jen jedna. Je to ten uživatel z konkrétní domácnosti, který se první zaregistruje a vstoupí do informačního systému. Hlava rodiny je správcem domácnosti. Hlava rodiny přidává do systému další členy rodiny, tím máme ošetřeno, že v domácnosti je pouze jeden uživatel typu hlava rodiny. Hlava rodiny má možnost zadání cen za jednotky elektrické energie, zemního plynu, studené vody, teplé vody a stočného. Dále má možnost nastavit vytápění domácnosti, zda je domácnost vytápěna plynovým či elektrickým kotlem. Zda je domácnost připojena k veřejnému rozvodu vody, zda odebírá z vodovodního řádu i teplou užitkovou vodu. Poslední možností nastavení je solární systém. Hlava rodiny má na výběr všechny funkce jako má člen rodiny.

3.2.3 Člen rodiny

Člen rodiny se do systému neregistruje sám, ale vkládá jej do systému správce domácnosti hlava rodiny. Při vkládání nového člena rodiny zadává přihlašovací jméno a heslo. Heslo si člen rodiny při přihlášení může kdykoliv změnit. Členů rodiny v informačním systému domácnosti může být libovolné množství. Člen rodiny se musí nejdříve přihlásit a pak může využívat funkce systému.

3.3 Konceptuální model

Pro snadnější pochopení, co má systém pro jednotlivé typy uživatelů umožňovat, jsem vytvořil konceptuální model. Ke tvorbě modelů jsem použil UML a konceptuální model jsem navrhoval pomocí diagramu případu užití (use case diagram). Diagramy případu užití znázorňují přehledně všechny činnosti, které bude moci daný uživatel v rámci systému vykonávat.



Obrázek 2. Diagram případu užití

3.4 Nákupy

Mezi hlavními částmi informačního systému patří podpora nákupů. Podpora nákupů se skládá ze tří hlavních částí, které si popíšeme v následujících odstavcích.

Plánování nákupu – uživatel systému si může v systému zapisovat zboží, které chce koupit při dalších nákupu. Takto si průběžně může zapisovat položky do seznamu na nákup, aby na žádnou nezapomněl. Systém řadí výrobky do kategorií, které mají stromovou strukturu. Uživatel do systému může ukládat obchody do kterých chodí nakupovat, a k těmto obchodům doplnit, které kategorie výrobku nabízí. Uživatel se chystá na nákup, a proto v systému vybere si obchod do kterého půjde nakupovat. Informační systém podle vybraného obchodu vybere v seznamu nákupu položky, jenž může uživatel v obchodě nakoupit. Uživatel si může vytisknout nákupní lístek. Systém podporuje výběr více obchodů pro nákup. Tento stav musí být na nákupním lístku ošetřen, a u každé položky bude vypsáno místo nákupu.

Domácí sklad – s nákupy souvisí domácí sklad. Sklad nám umožňuje sledovat co máme doma nakoupeno, zda nepotřebuje něco dokoupit. Pro každý uskutečněný nákup vložíme položky nákupu do domácího skladu. Při vkládání do domácího skladu, máme možnost vytvoření nové kategorie výrobků. Tyto kategorie mají stromovou strukturu. Také při vkládání můžeme zadat kdo zboží koupil a pro koho bylo nakoupeno. Těchto údajů můžeme pak využít pro přehled výdajů pro jednotlivé členy rodiny.

3.5 Trvalé výdaje a příjmy

V informačním systému mohou být nastaveny trvalé výdaje a příjmy pro jednotlivé měsíce roku. Trvalými příjmy bude převážně výplata jednotlivých členů domácnosti. Do trvalých výdajů můžeme zahrnout:

Záloha elektrické energie

Záloha zemního plynu

Záloha vodné, stočné

Koncesionářské poplatky

Daň z nemovitosti

Nájemné

Životní pojištění

Telefon

Hypotéka

Půjčky

Trvalé výdaje můžeme zadat do systému od kterého data a jak dlouho se budou platit. Trvalé výdaje i příjmy může přidávat, editovat a mazat pouze hlava rodiny.

4 Energie

Ve většině domácností je využívána elektrická energie a zemní plyn, které jsou dodávány do domácnosti z veřejných sítí, za které musíme platit. Proto se touto stránkou musí informační systém pro domácnost zaobírat. Tvoří velkou část výdajů na provoz domácnosti.

Elektrická energie má v každé domácnosti široké uplatnění. Každá domácnosti ji používá pro své elektrické spotřebiče (pračka, televize, lednička), tato část spotřebované energie bývá v průběhu roku relativně neměnná. V některých domácnostech je používána elektrická energie také k ohřevu TUV a k topení. Tato složka spotřeby elektřiny je v průběhu roku značně proměnlivá a závisí na venkovní teplotě a povětrnostních vlivech. U elektrické energie musíme rozlišovat denní a noční proud.

Zemní plyn je používán v domácnosti, avšak jeho použití není tak široké jako u elektřiny. Používá se převážně jako alternativa pro elektrickou energii při topení, ohřevu TUV a vaření.

4.1 Nastavení domácnosti

Hlava rodiny při své návštěvě informačního systému má možnost nastavení domácnosti. Může nastavit jak je domácnost připojena k inženýrským sítím, a jaké používá energie pro topení domácnosti. Možnosti nastavení domácnosti:

Připojení k vodovodnímu řádu – zda je domácnost připojena k vodovodnímu řádu či používá vlastní zdroj vody. Možnost že domácnost přijímá z vodovodního systému také teplou vodu.

Stočné – zda je domácnost připojena ke kanalizaci, či využívá vlastní jímky.

Připojení k plynovému rozvodu – zda domácnost je připojena k veřejnému rozvodu plynu. Nastavení zda se zemní plyn využívá k topení, ohřevu TUV či na vaření.

Elektrická energie – v dnešní době je již každá domácnost připojena k rozvodné síti, tady máme spíše na výběr zda je elektrická energie využívána na vaření, ohřev TUV nebo topení. Elektrická energie je zdrojem pro domácí spotřebiče, které tvoří velkou část spotřeby energie.

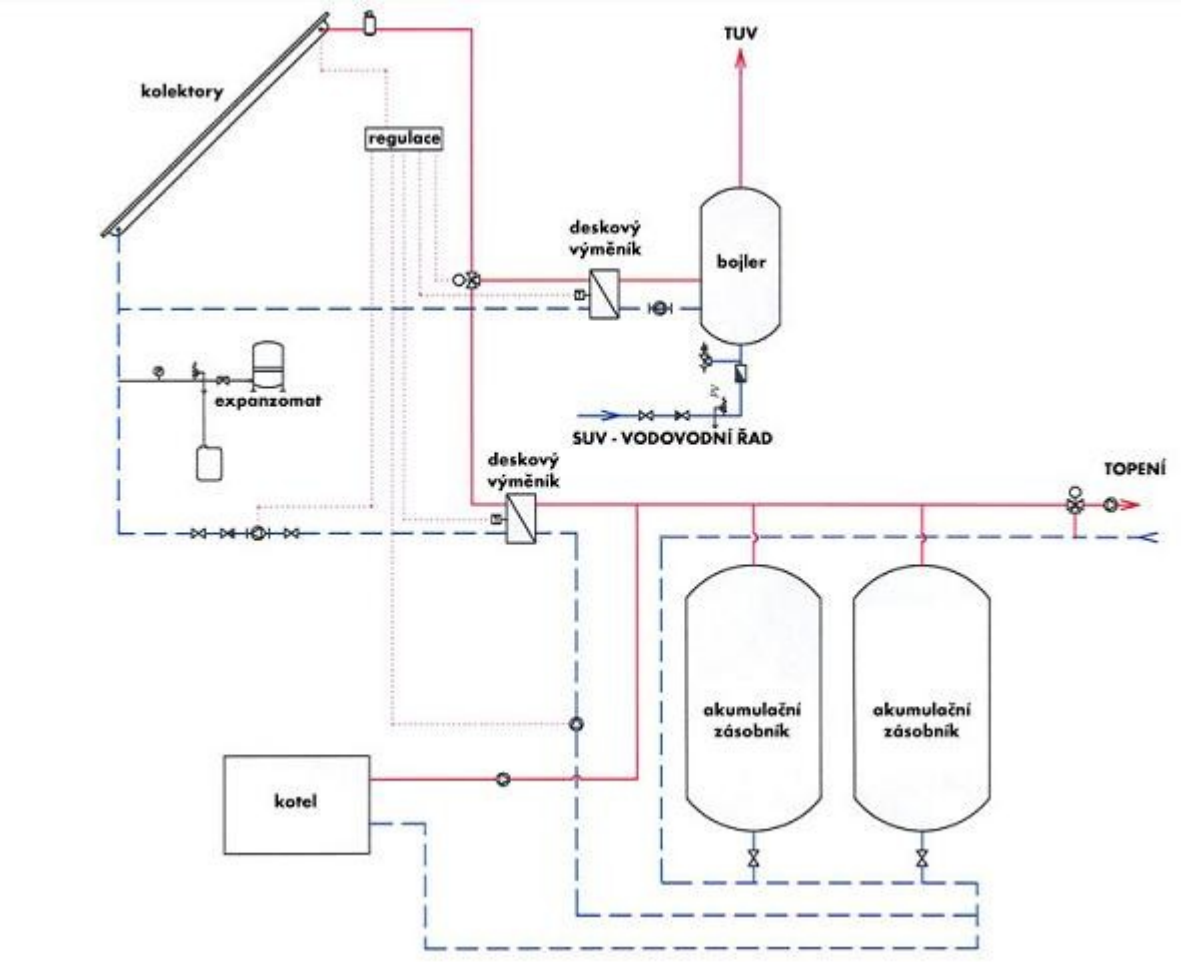
Solární kolektory – zda jsou v domácnosti a technické parametry solárních kolektorů.

V domácnosti musí také hlava rodiny nastavit ceny za jednotku elektrické energie(denní a noční proud), zemního plynu, teplé a studené vody, stočného.

4.2 Solární systém

Do spotřeby energie v domácnosti se také významnou měrou promítá použitý solární systém. V našich zeměpisných podmínkách máme dobré podmínky pro využití solárních systémů. Energii slunečního záření můžeme využít a ušetřit tím náklady na provoz domácnosti. Solární kolektory nám mohou ušetřit 80% nákladů na ohřev teplé užitkové vody a až 40% nákladů na vytápění.

Sluneční záření dopadá na plochu kolektoru, přibližně 90% záření proniká speciálním solárním sklem kolektoru a předává svou energii absorbéru kolektoru. Tepelná energie získaná absorberém se přenáší k trubkovému registru absorbéru ve kterém obíhá nemrznoucí kapalina, která je tímto teplem velmi rychle ohřívána. Elektronická regulace neustále vyhodnocuje prostřednictvím čidel teplotní rozdíl mezi teplotou vody v zásobníku TUV a teplotou kapaliny v kolektoru. Dle nastavené diference pak zabezpečuje spínání oběhového čerpadla. Ohřátá solární kapalina vystupuje z kolektorů protéká výměníkem solárního zásobníku a předává naakumulované teplo, ohřívá tak jeho obsah. Ochlazená kapalina pak putuje přes hnací náběhovou větev solárního okruhu zpět do kolektorů a znovu se ohřívá, celý cyklus se tak opakuje. Solární okruh je tedy uzavřen mezi absorbéry kolektorů a výměníkem solárního zásobníku TUV. Potrubní rozvod systému je obvykle vyhotoven z měděného potrubí.



Obrázek 6. Schéma zapojení slunečního kolektoru

Sluneční kolektory nemůžeme používat po celý rok jako jediný zdroj pro ohřev TUV a vytápění, proto musí v domácnosti jiný zdroj pro vytápění a ohřev TUV. Z energetické bilance vyplívá, že sluneční kolektory pracují i v zimním měsících, ale mají menší podíl na spotřebované energii na topení. Naopak v letních měsících energie vyrobená v solárních kolektorech pokryje veškeré náklady na ohřev TUV a také pro topení.

4.3 Předpověď spotřeby energie

Systém by měl umožňovat předpovídat spotřebu energie potřebnou pro vytápění domácnosti a ohřev TUV pro následující období. Systém má k dispozici údaje z předchozích let o průběhu teplot a intenzitu slunečního záření v průběhu celého roku. Takže je schopen vypočítat průměrnou spotřebu energie za jednotku času vzhledem k venkovní teplotě a slunečnímu záření.

Systém má přehled o aktuální spotřebě v daném roce, zda se spotřeba odchyluje od průměru a případně jakým způsobem. Z těchto údajů by systém mohl učinit predikci spotřeby energii pro potřebné období. Systém by mohl mít k dispozici více predikcí pro toto období:

Pesimistická predikce – systém by uvažoval pro předpovídané období, že teplota bude dlouhodobě pod průměrnou teplotou. Příkladem by mohla být predikce spotřeby pro zimní období, kdy bychom uvažovali výrazně chladnější zimu.

Optimistická predikce – systém pro předpovídané období očekává nadprůměrné teploty (mírná zima) oproti průměrné zimě.

Reálná predikce – systém dokáže spočítat, jaká je teplota v tomto roce oproti předcházejícími létům. Takže můžeme předpokládat, že i pro předpovídanou dobu bude hodnota podobná vůči průměru. Takto vypočítaná spotřeba by se měla nejvíce blížit reálné spotřebě v daném období.

4.4 Stahování dat z internetu

Abych mohl v informačním systému předpovídat spotřebu energie, musím získávat průběh teploty a intenzitu slunečního svitu. Nejlepším případem by bylo, kdyby v domácnosti byla čidla měřící konkrétní veličiny. To by znamenalo, že systém musí být 24 hodin připojen k čidlům a zpracovávat hodnoty z nich získané. Získané hodnoty by se ukládali do databáze. Výhodou by byla získaná data v malých časových intervalech. Toto řešení by bylo pro domácnost příliš drahé. Proto systém bude tyto informace stahovat z webového serveru, kde bude získávat jednotlivé hodnoty v daném intervalu. Data si systém bude stahovat sám v nastavených intervalech do textového souboru. Z textového souboru si systém extrahuje potřebná data, která bude ukládat ve správném tvaru do databáze. Data uložená v databázi bude systém využívat pro svoji činnost.

5 Implementace

Systém bude implementován jako webová aplikace pomocí technologie PHP a MySQL. Výhoda této implementace je v tom, že uživatel nepotřebuje na svém počítači mít nainstalován žádný speciální software a postačí mu pouze webový prohlížeč a připojení k internetu.

5.1 Implementační prostředí

5.1.1 HTML

Jazyk HTML vznikl v mezinárodním středisku pro jaderný výzkum CERN jako prostředek pro výměnu dokumentů mezi pracujícími vědci. Základ jazyku položil Tim Breners-Lee v roce 1991. Tato verze umožňovala vkládat do textu obrázky, hypertextové odkazy a umožňovala vytvářet několik logických úrovní a druhů zvýraznění. Byla označována jako HTML 0.9. Požadavky uživatelů se zvyšovaly a producenti začali vytvářet v HTML nové prvky. Tim Berners-Lee všechny používané prvky shrnul a popsal ve standardu HTML 2.0, který již plně vyhovuje SGML. V roce 1997 vzniká poslední vývojová verze jazyka HTML.

5.1.2 CSS

CSS neboli kaskádové styly vznikly jako souhrn metod pro úpravu vzhledu stránek, první návrh normy byl zveřejněn v roce 1994. Kaskádové styly se využívají k formátování obsahu HTML, XHTML a XML dokumentů. Kaskádové styly umožňují definovat způsob zobrazení každého prvku na stránce, který je označen konkrétním tagem. Styl však není přímo součástí textu stránky, proto může být zápis stránky přehlednější a dobře strukturovaný. Navíc umožňuje styly definovat jednotný vzhled určitého elementu v celém dokumentu jedním zápisem, proto jej nemusíme opakovat u každého elementu. Pokud styly vložíme do externího souboru, můžeme je využívat pro více stránek najednou. Definice designu stránek je pak uložena na jednom místě. Při požadavku na změnu vzhledu stránek stačí upravit styl a změny se promítnou do všech stránek.

Největší nevýhodou CSS je, že většina prohlížečů plně nepodporuje kaskádové styly, jediný prohlížeč, který se řídí všemi oficiálními standardy je Opera.

5.1.3 PHP

PHP je serverový skriptovací jazyk pro tvorbu dynamického webu. Počátky PHP spadají do roku 1994. Tehdy Rasmus Lerdort vytvořil v Perlu jednoduchý systém pro počítání přístupu ke svým stránkám, později jej přepsal do jazyka C. Sada těchto skriptů a jejich zdrojové kódy byly zveřejněny pod názvem Personal Home Page Tools (zkráceně PHP). Když Lerdorf systém rozšířil o možnost

začleňování SQL příkazů do stránek, práci s formuláři a zobrazování výsledků dotazu SQL dostal název PHP 2.0/FI. Tento systém ale pracoval pouze na operačním systému LINUX, proto bylo vyvinuto PHP 3.0, které již pracuje také na operačním systému Windows a Macintosh. Od této verze se opustil význam zkratky PHP a systém se označuje jako hypertextový preprocesor.

PHP je serverový skriptovací jazyk, který se provádí na straně serveru. PHP je na serveru závislé, protože na něm běží jeho interpret, který provádí PHP skripty. Výhodou PHP je, že ke zdrojovým kódům se nedostane nikdo jiný než autor. Samostatné PHP skripty se zapisují přímo do HTML stránky mezi značky, kterých máme několik druhů:

XML styl: <?php [PHP kód]>
Krátký styl: <? [PHP kód]>
Skript styl: <script language=’php’> [PHP kód] </php>

Interpret PHP zpracovává soubory s koncovkou php. Interpret HTML příkazy rovnou ukládá do výsledné HTML stránky, pokud ovšem narazí na PHP skript, tak jej nejprve provede a jeho výsledek se zapíše do HTML stránky. To je celý princip dynamického generování HTML stránek, což je základním posláním PHP.

Od počátku je PHP open source produkt, takže je volně šířitelný a má dostupné zdrojové kódy, které můžeme v případě potřeby upravovat a distribuovat.

Mezi hlavní přednosti PHP se řadí jeho vysoká efektivita a výkonnost. Jediný server dokáže denně zpracovat miliony požadavků. Další výhodou PHP je schopnost se připojovat bez jakýchkoliv prostředníků k mnoha databázovým systémům (např. PostgreSQL, mSQL, Oracle, Informix, Sybase). Další výhodou PHP je vysoká přenositelnost, bez jakýchkoliv modifikací bude fungovat na různých systémech. Od počátku bylo PHP navrhováno pro využití ve webových aplikacích, proto obsahuje velké množství funkcí, které můžeme využít při plnění úkolů spojených s webem.

5.1.4 MySQL

MySQL je velmi rychlý a robustní relační databázový systém. Databáze umožňuje efektivně ukládat, hledat, řadit a získávat data. Server MySQL je více-vláknový a více-uživatelský, umožňuje současné připojení více uživatelů k databázi a zajišťuje, aby to mohli být pouze oprávnění uživatelé. Jako dotazovací jazyk používáme SQL (Structured Query Language), což je celosvětově používaný standardní dotazovací jazyk pro databáze. MySQL databáze je jedním z nejvyužívanějších způsobů uchování dat. Počátky MySQL jsou v roce 1979, ale jako open source licence byla představena veřejnosti až v roce 1996.

Hlavní předností MySQL je vysoká rychlost, výkon a jednoduchost. Běží totiž jako samostatný server a pro práci s daty většinou využívá skriptování na straně serveru pomocí PHP. Výhodou je také open source licence, proto našlo MySQL velké využití zejména v oblasti internetu.

5.2 Zabezpečení IS

Na bezpečnost informačních systémů a dat v nich uložených je v dnešní době kladen velký důraz. Proto by měl mít každý informační systém určité zabezpečení.

Hrozbou pro bezpečnost informačních systémů je komunikace mezi počítačem uživatele a serverem, na němž jsou uloženy webové stránky a databáze MySQL, která probíhá nezašifrovaně. Potencionální útočník může tuto komunikaci sledovat a zjistit přihlašovací jména, hesla a další citlivé údaje. V této situaci je řešením šifrovaný přenos dat. Typický příklad je využití protokolu SSL, které se stará o to, aby nedošlo k neoprávněnému zachycení či pozměnění přenášených dat.

Pro zvýšení bezpečnosti informačního systému jsem si vybral zabezpečení pomocí přihlašovacího jména, hesla a session.

5.2.1 Přihlašovací jméno a heslo

Zabezpečení pomocí přihlašovacího jména a hesla patří mezi základní způsoby zabezpečení informačního systému. Přihlašovací jméno a heslo je uloženo v databázi. Pro vyšší bezpečnost bude stanovena nejnižší délka hesla. Uživatel by si při výběru hesla neměl vybírat slova související s jeho osobou, který by mohlo být snadněji překonáno. Správně zvolené heslo by se mělo skládat z různých znaků, jako jsou číslovky, malá, velká písmena a speciální znaky.

Heslo by nemělo být ukládáno do databáze přímo, proto bychom měli využívat speciální hashovací funkce, které nám na zadané heslo při registraci aplikuje svůj jednostranný šifrovací algoritmus. Šifrovací algoritmus vrací hash pevné délky, který uložíme do databáze. Při přihlášení tedy zadáváme heslo, na které je aplikována hashovací funkce a výsledný hash porovnáváme s hashem uloženým v databázi. Pro systém vyberu hashovací funkci MD5, protože ji podporuje PHP.

5.2.2 Session

Na začátek php skriptu je vložen dotaz na session, zda je nastaven a jaké hodnoty nabývá. Tento dotaz je vkládán do všech php skriptů, ke kterým mohou přistupovat pouze přihlášení uživatelé. Pokud uživatelovo session nevyhovuje podmínce pro přístup k požadovanému skriptu je mu přistup odmítnut. Tímto způsobem ošetříme případy připojení, kdy uživatel ve svém prohlížeči přímo napíše cestu k tomuto skriptu. Přestože je uživatelem zadaná cesta správná, uživatel není autorizovaný a přístup je mu odmítnut.

6 Závěr

Cílem mého semestrálního projektu je vytvoření informačního systému pro domácnost, který bude mít přehled o finančním hospodaření a spotřebě energie v domácnosti. V semestrálním části jsem musel analyzovat obecné požadavky na informační systém. Dalším krokem bylo vytvoření požadavků na konkrétní zadání systém, kdy jsem z neformáních požadavků od zadavatele systému vytvořil seznam formálních požadavků. Tím jsem si stanovil všechny potřebné funkce, které bude muset systém umožňovat a nabízet uživatelům.

V diplomové části budu ze seznamu formálních požadavků vytvářet návrh architektury a podrobný návrh systému. Posledním krokem bude implementace systému. Ve výsledném informačním systém pro domácnost bude klást důraz na funkčnost a intuitivní ovládání.

Literatura

[1] Castro E.: HTLM 4 PRO WORL WIDE WEB, Softpress s.r.o., 2004.

[2] Staníček P.: CSS Kaskádové styly kompletní průvodce, Computer Press, Brno, 2005.

[3] Welling L., Thomsonová L.: PHP a MySQL rozvoj webových aplikací, druhé vydání, Softpress s.r.o., 2004.

[4] Arlow J., Neustadt I.: UML a unifikovaný proces vývoje aplikací, Grada Published spol. s.r.o., 1999.

[5] WWW stránky. http://www.envi.cz.

[6] WWW stránky. http://www.solarobchod.cz.

[7] WWW stránky. http://www.jiranek.cz.

[8] WWW stránky. http://www.cez.cz/presentation/static/solarni/k32.htm.

[9] WWW stránky. http://www.enoleka.cz.

[10] Zendulka J., Bártík V., Květoňová Š.: Analýza a návrh informačních systému AIS – studijní opora 2006.