UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Tine Mlakar

Razvoj informacijskega sistema za multimedijski center

Mentor: doc. dr. Janez Demšar

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00518/2010
Datum: 05.04.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **TINE MLAKAR**

Naslov: **RAZVOJ INFORMACIJSKEGA SISTEMA ZA MULTIMEDIJSKI CENTER**

DEVELOPMENT OF INFORMATION SYSTEM FOR MULTIMEDIA CENTER

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Manjša podjetja in druge organizacije, ki se odločajo za vpeljavo ali prenovo informacijskih sistemov, k reševanju problema pogosto pristopajo brez jasnih ciljev, kar vodi v slabo načrtovane sisteme. V diplomskem delu na primeru manjše organizacije prikažite, kakšen je ustrezno voden - to je, ne pretirano formalističen, a vseeno primerno sistematičen - pristop k racionalni celoviti prenovi informacijskega sistema. V delu opišite tudi uporabljene pristope in orodja ter implementirajte tolikšen del sistema, da bo razvidna uspešnost uporabljenega pristopa.

Mentor:

doc. dr. Janez Demšar

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00518/2010
Datum: 05.04.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **TINE MLAKAR**

Naslov: **RAZVOJ INFORMACIJSKEGA SISTEMA ZA MULTIMEDIJSKI CENTER**

DEVELOPMENT OF INFORMATION SYSTEM FOR MULTIMEDIA CENTER

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Manjša podjetja in druge organizacije, ki se odločajo za vpeljavo ali prenovo informacijskih sistemov, k reševanju problema pogosto pristopajo brez jasnih ciljev, kar vodi v slabo načrtovane sisteme. V diplomskem delu na primeru manjše organizacije prikažite, kakšen je ustrezno voden - to je, ne pretirano formalističen, a vseeno primerno sistematičen - pristop k racionalni celoviti prenovi informacijskega sistema. V delu opišite tudi uporabljene pristope in orodja ter implementirajte tolikšen del sistema, da bo razvidna uspešnost uporabljenega pristopa.

Mentor:

doc. dr. Janez Demšar

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a **Tine Mlakar**,

z vpisno številko **63040105**,

sem avtor/-ica diplomskega dela z naslovom:

Razvoj informacijskega sistema za multimedijski center

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek)

  doc. dr. Janeza Demšarja

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 15.10.2010 Podpis avtorja/-ice: _____

Zahvala

Najprej bi se zahvalil mentorju, prof. Janezu Demšarju, za pomoč pri izdelavi te diplomske naloge.

Še posebej pa bi se rad zahvalil svojim staršem za pomoč in podporo skozi vsa leta šolanja in študija.

Zahvalil bi se tudi Kristini, ki me je spodbujala, ko sem to najbolj potreboval.

KAZALO

Seznam uporabljenih kratic in simbolov

API (angl. application program interface) − aplikacijski programski vmesnik

ERP (angl. Enterprise resource planning) − integriran poslovni informacijski sistem

IDE (angl. Integrated development environment) − integrirano razvojno okolje

IS − informacijski sistem

M3C − mreža multimedijskih centrov

MDI (angl. Multi Document Interface) − večokenski vmesnik

MIS (angl. Management Information System) − upraviteljski informacijski sistem

MMC − multimedijski center

SDI (angl. Single Document Interface) − enookenski vmesnik

SLDC (angl. System Developement Life Cycle) − življenjski model razvoja sistema

SUPB − sistem za upravljanje s podatkovnimi bazami

SQL (angl. Structured Query Language) − poizvedovalni jezik za delo s podatkovnimi bazami

TIS − transakcijski informacijski sistem

VPN (angl. Virtual Private Network) − navidezno privatno omrežje

.NET − programska knjižnica ".NET Framework"

Povzetek

Diplomsko delo opisuje razvoj informacijskega sistema za podporo delovanja multimedijskega centra. Sistem je funkcionalno razdeljen na več delov − modulov. Vsak modul s hranjenjem, urejanjem in obdelovanjem podatkov informacijsko podpira določeno dejavnost znotraj multimedijskega centra. V začetku je opisana organizacija multimedijskega centra Pulsar, za katerega je informacijski sistem narejen. Predstavljeno je delo v organizaciji, problematika in cilji, ki jih želimo doseči z uvedbo računalniško podprtega informacijskega sistema. V nadaljevanju je opisana vrsta in življenjski model razvoja sistema. Sledi opis načrtovanja podatkovnega modela in razvoj pomembnejših modulov ter podpornih razredov. V zadnjem delu pa so predstavljene idejne rešitve za module in dejavnosti, ki še niso razvite. Predstavljeni so tudi problemi pri razvoju in načrti za nadaljnji razvoj.

Ključne besede:

informacijski sistem, Windows forms, inkrementalni razvoj

Abstract

The thesis deals with the development of an information system for operational support of a multimedia center. The system is functionally divided into several parts − modules. Each module supports a particular activity within the multimedia center by saving, editing and processing data. At the beginning, the multimedia center Pulsar is described, for which the system is designed. It presents work, issues and objectives to be achieved by implementing computer supported information system in the multimedia center. Further, the description of the type and the development life cycle of the system is provided. It continues with the description of designing the data model and the development of major modules and support classes. The last part provides conceptual solutions for the modules and activities that are not yet developed. Above all, it also presents problems of the development and plans for further progress.

Keywords:

information system, Windows forms, incremental development

1 Uvod

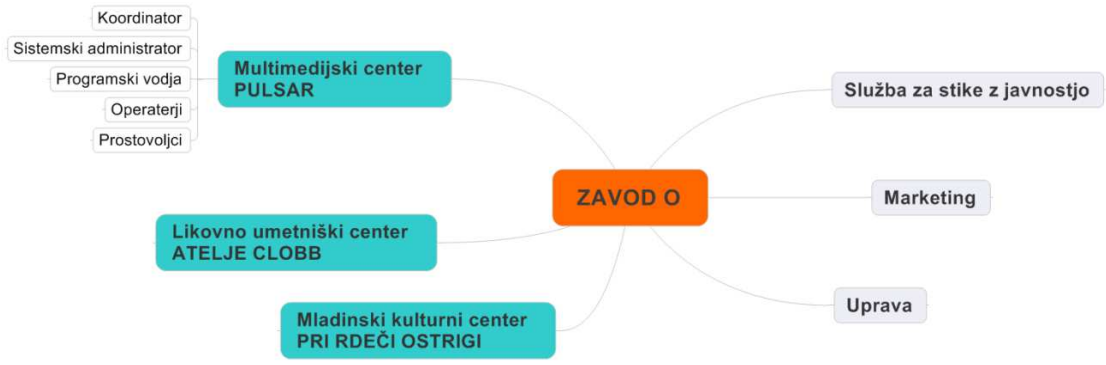
1.1 Predstavitev multimedijskega centra Pulsar

Da bi lahko dobro predstavili reševanje informacijske problematike multimedijskega centra, je ključno, da bralca seznanimo z organizacijo in načinom njenega delovanja.

1.1.1 Organizacija

Multimedijski center Pulsar je organizacija, ki se nahaja v starem mestnem jedru Škofje Loke. Je edini tovrstni center na Gorenjskem, ki je bil ustanovljen s pomočjo evropskih strukturnih skladov pod okriljem Zavoda O − zavoda škofjeloške mladine. Nudi številne brezplačne storitve, kot so svetovanje, dostop do interneta, uporaba računalnikov za pisarniške namene, grafično oblikovanje in drugo multimedijsko produkcijo. Poleg tega izvaja brezplačne tečaje in cenovno ugodne delavnice predvsem na področju multimedijev in informacijsko-komunikacijskih tehnologij. Kot organizatorji, soorganizatorji ali kot tehnična podpora se pojavlja tudi pri različnih kulturnoumetniških prireditvah na lokalni ravni in znotraj slovenske mreže multimedijskih centrov M3C.

1.1.2 Osebje in delovanje



Slika 1: Shema strukture Zavoda O in delovnih mest v MMC Pulsar

MMC Pulsar nima redno zaposlenih delavcev. Za delovanje skrbijo študentje, ki delajo preko študentskega dela v svojem prostem času. Večina opravljenih ur je prostovoljnih; plačane so le ure operaterja, opravljene med uradnimi urami. Mesečni honorar dobijo tudi koordinator in programski vodja ter sistemski administrator. Zunanji sodelavci, kot so na primer predavatelji, so plačani po dogovoru.

Ker je MMC Pulsar del Zavoda O, večji del administrativnega dela opravljajo službe zavoda. Za ostalo pa so zadolženi koordinator, programski vodja, operaterji in sistemski administrator. Koordinator je zadolžen za administrativna dela, poročanje upravi, koordinacijo operaterjev, nadzor nad delovanjem in podobno. Programski vodja ima nalogo, da priskrbi program in finančni, tehnični ter kadrovski načrt za njegovo izvedbo. Operaterjeva naloga je izvedba programa, nudenje tehnične pomoči in

asistenca obiskovalcem MMC-ja. Poleg tega je operater zadolžen tudi za asistenco programskemu vodji in koordinatorju, za čistočo in red v prostorih, delovanje opreme itd. Sistemski administrator skrbi za delovanje in posodabljanje podatkovnih, spletnih in drugih strežnikov ter žičnega in brezžičnega računalniškega omrežja.

Zavod O nudi tudi službo za stike z javnostjo in oglaševanje (oz. marketing). V praksi veliko tega dela opravijo kar operaterji z objavljanjem novic na različnih spletnih straneh in s pošiljanjem elektronske pošte na naslove zainteresirane javnosti, pa tudi z izdelavo tiskovin, kot so plakati in letaki.

1.2 Problematika

1.2.1 Predstavitev problema

Pred časom so v MMC-ju uporabljali program za izračun plač in beleženje stanja v blagajni, vendar je imel veliko napak in se je čez čas pokazal kot dokaj zastarel in pomanjkljiv. Ko so ga nehali uporabljati, se je porodila ideja o novem informacijskem sistemu. Od takrat za razna delovna poročila, računanje plač in stanje v blagajni uporabljajo elektronske preglednice (Microsoft Excell, Google Docs), spletne forume in papirnate obrazce. Zaradi tega je zelo oteženo iskanje določenih zapisov. Poleg tega tudi ni kontrol, ki bi preprečile neizpolnjene ali napačno izpolnjene obrazce. Seznam opravil se nahaja kar na listu papirja na steni ob operaterjevi mizi. To seveda ne omogoča nikakršnega sistema za opominjanje, zato mnoge stvari niso pravočasno opravljene. Komunikacija v glavnem poteka preko elektronskega poštarja, ki omogoča hkratno pošiljanje elektronske pošte na več naslovov. Skratka, trenutno stanje informatike je za multimedijski center skoraj sramotno.

Problem, ki ga želimo rešiti, je slabo stanje informatike v MMC-ju. Ker nekateri poslovni procesi niso dobro informacijsko podprti s programsko opremo, je delo večkrat oteženo, neracionalno in včasih neučinkovito. Prav tako je velik problem hranjenje, iskanje in dostop do podatkov, na primer osebnih podatkov strank in zaposlenih, podatkov o opremi, inventarju itd.

1.2.2 Predstavitev ciljev

Izboljšanje informatike v organizaciji je mogoče le, če so podatki pravilno urejeni in informacije operaterjem vedno dostopne. Z izdelavo celovitega informacijskega sistema, ki bi bil popolnoma usklajen s poslovnim sistemom in dobro podprt s strani programske opreme, bi operaterjem in administraciji omogočili hiter in zanesljiv dostop do podatkov. Tako bi hitreje prišli do informacij, ki so potrebne za reševanje vsakodnevnih vprašanj, izvajanje upravljavskih ukrepov in sprejemanje strateških odločitev.

Cilj projekta je izdelati računalniško podprt informacijski sistem, ki bo hranil podatke o zaposlenih, strankah, storitvah, opremi, inventarju, financah itd. Te podatke bo obdeloval in s tem omogočil, da bodo informacije hitro dostopne in posredovane v taki obliki, da bo operater lahko svoje delo opravil hitro in učinkovito. V tej diplomski nalogi bomo opisali načrtovanje in razvoj informacijskega sistema, imenovanega Pulsmaster.

1.3 Informacijski sistem

1.3.1 Teorija o informacijskih sistemih

Ko govorimo o informatiki, je seveda potrebno, da najprej razjasnimo nekatere ključne pojme. Informatika je namreč interdisciplinarna znanstvena disciplina, ki se v prvi vrsti ukvarja z analizo, zbiranjem, razvrščanjem, upravljanjem, shranjevanjem in iskanjem informacij. [5] Ko danes govorimo o informatiki, predvsem mislimo na avtomatično in sistematično obdelavo podatkov z računalniki.

Podatek je le predstavitev informacije na način, ki je primeren za obdelavo, prenos ali hranjenje. Podatki so nestrukturirana dejstva o dogodkih, objektih, ali ljudeh. [2] Sam podatek nam še ne zagotavlja informacije, dokler ga ne znamo pravilno interpretirati. Informacija pa je novo znanje, ki ga dobimo z interpretacijo določenih podatkov in se nanaša na določene objekte. Informacija ima pomen in uporabo za naslovnika v določenem kontekstu in se jo lahko uporabi pri sprejemanju odločitev.

Informacija nastane z zbiranjem, obdelavo in predstavitvijo podatkov na naslovniku uporaben način (tekst, grafi, preglednice itd.). [2] Prenos in uporaba informacij je v današnji informacijski družbi za ekonomsko, politično in kulturno dejavnost vedno večjega pomena.

Informacijski sistem (IS) je kombinacija informacijske tehnologije in ljudi, ki to tehnologijo uporabljajo za podporo njihove dejavnosti, upravljanja in odločanja. [6] Med informacijsko tehnologijo štejemo strojno, komunikacijsko in programsko opremo. Informacijski sistem zbira, obdeluje, hrani in porazdeljuje podatke ter s tem nudi informacije, ki jih uporabljajo člani in stranke organizacije. Informacije lahko zadevajo stranke, dobavitelje, produkte, opremo, postopke itd. [2]

Poznamo več vrst formalnih informacijskih sistemov. Lahko jih razdelimo na

- sisteme za podporo operativnim nalogam:
  - transakcijski IS,
  - IS za nadzorovanje procesov,
  - IS za poslovno sodelovanje,
- IS za podporo odločanju:
  - upraviteljski (poslovodni) IS,
  - odločitveni IS,
  - direktorski IS,
- ostalo (druge delitve):
  - ekspertni IS,
  - IS za upravljanje znanja,
  - strateški IS,
  - funkcionalni informacijski podsistemi.

Pulsmaster je predvsem transakcijski informacijski sistem, v katerega bomo vključili tudi nekaj prvin upraviteljskih informacijskih sistemov.

Transakcijski informacijski sistem (TIS) hrani in obdeluje podatke o poslovnih dogodkih oz. transakcijah. Transakcija je standardni poslovni dogodek, ki generira ali spremeni podatke v podatkovni bazi informacijskega sistema. Tipični primeri transakcij, s katerimi se bo ukvarjal naš informacijski sistem, so prodaja storitev, računanje plače operaterja, izposoja opreme in drugo. TIS običajno hranijo veliko količino podatkov in je namenjen operativnemu nivoju organizacije. Zato je število uporabnikov relativno veliko, obdelava podatkov je enostavna, pomembna pa sta tudi hitrost in zanesljivost sistema.

Upraviteljski IS (Management Information System − MIS) na osnovi podatkov, dobljenih od transakcijskega IS, in na osnovi delovanja organizacije zagotavlja informacije in poročila, ki jih uprava organizacije lahko uporabi za pomoč pri sprejemanju odločitev. Glede na transakcijske IS imajo MIS manjše število uporabnikov, obdelava podatkov je zahtevnejša, odzivni časi pa so nekoliko manj pomembni.

S stališča poslovne informatike lahko IS Pulsmaster uvrstimo med integrirane poslovne informacijske sisteme (Enterprise resource planning − ERP). ERP je računalniško podprt sistem, ki v eni centralni podatkovni bazi združuje in upravlja podatke notranjih in zunanjih virov, vključno s stvarnim premoženjem, finančnimi sredstvi, materiali in človeškimi viri. Njegov namen je izboljšati pretok informacij med poslovnimi funkcijami znotraj organizacije in upravljanje povezav z zunanjimi interesnimi skupinami. [4]

2 Razvoj informacijskega sistema Pulsmaster

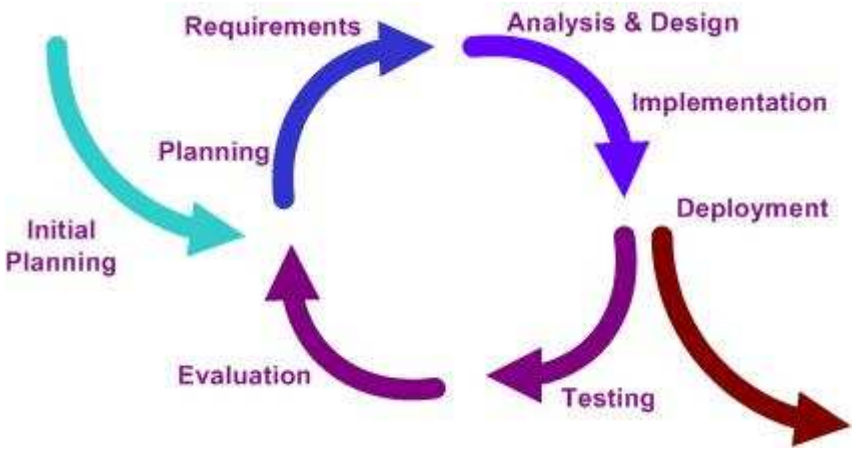
2.1 Izbira življenjskega modela razvoja

Pri razvoju informacijskih sistemov opravila navadno razdelimo na naslednje faze:

- zajem in specifikacijo uporabniških zahtev,
- analizo,
- načrtovanje,
- izvedbo,
- testiranje,
- namestitev in uvedbo,
- vzdrževanje.

Z izborom življenjskega modela razvoja IS (angl. System Developement Life Cycle − SLDC) določimo, na kakšen način si sledijo posamezne faze razvoja.

Za razvoj našega IS smo uporabili kombinacijo iterativnega in inkrementalnega razvoja. Osnovna ideja je, da razvijamo sistem skozi ponavljajoče cikle (iteracije) in po manjših delih (inkrementih), kar nam omogoča, da uporabimo, kar smo se naučili med razvojem prejšnjih delov ali verzij sistema. [7]



Slika 2: Shema iterativnega modela razvoja [7]

Iterativni razvoj praktično pomeni odlašanje izvedbe za določeno časovno obdobje, da bi hitreje napredovali in se kasneje vračali k istemu problemu, kjer bi dosežene rezultate izboljšali oziroma popravili (slika 2). S stališča projektnega vodje je tak pristop teže

obvladljiv, saj je pri planiranju težko oceniti koliko iteracij bo potrebnih za dosego zastavljenega cilja z želeno kakovostjo izdelka. [10] Omogoča pa, da se sistem razvija hitreje, razvoj je bolj pester in prispeva k motivaciji razvijalcev.

Inkrementalni razvoj je strategija napredovanja v malih korakih, da bi prišli do ustreznih rezultatov. [10] Problematiko moramo najprej razdeliti na podprobleme, ki jih nato rešujemo vzporedno in/ali izmenično. Ko posamezen podproblem rešimo, ga testiramo in vključimo v obstoječi sistem.

IS Pulsmaster bomo razdelili na več delov oz. modulov, katere bomo razvijali posamično iterativno. Vsak modul bo postal svoj inkrement, ki ga bomo vključili v aplikacijo. Tako bo mogoče, da damo sistem v uporabo, še preden bodo razvite vse njegove funkcionalnosti.

2.2 Zajem uporabniških zahtev

Po pogovoru z naročnikom − osebjem multimedijskega centra smo izluščili njihove uporabniške zahteve. V nadaljevanju so te zahteve na kratko predstavljene. Ker so te zahteve zelo obširne, smo za potrebo diplomske nalogo izdelali le del želenega sistema. Po dogovoru z naročnikom, naj bi se sistem postopoma izdeloval in uvajal ter nadgrajeval. Ker gre za brezplačno in prostovoljno delo, končni rok za izdelavo ni bil postavljen.

2.2.1 Uporabniške zahteve

Prijava v sistem naj bo večnivojska. Administrator lahko v programu ustvari nove uporabniške profile in jim določi pooblastila. Glede na pooblastila lahko uporabnik dodaja, ureja in pregleduje šifrante.

Sistem naj ima možnost generiranja poročil in sicer:

- obračun delovnih ur operaterja,
- poročila projektov − prikaz stroškov in delovnih ur,
- prikaz prihodkov in odhodkov storitev (delavnic, tečajev).

Program operaterju prikazuje zadolžitve oz. naloge, ki jih je potrebno opraviti, razvrščene po prioriteti in bližini roka za zaključek naloge. Ko operater zaključi nalogo, jo v sistemu označi kot opravljeno in vpiše podatke o opravljenem delu v dnevnik dela. Obvezen bo vnos vrste dela, opis dela in število ur. Glede na te podatke se operaterju obračunava plača.
Sistem mora imeti možnost pošiljanja sporočil med operaterji. Omogočati mora pošiljanje več naslovnikom hkrati.
Sistem mora imeti možnost hranjenja podatkov o inventarju in njegovi lokaciji. Prav tako mora biti podprta izposoja opreme. Pri izposoji je potrebno vedeti, kdo si kaj izposoja in kdaj bo oprema vrnjena. Prav tako mora imeti možnost izstavitve računa za izposojnino.

Sistem mora omogočati prijavo strank na izobraževalne dogodke, vodenje evidence prostih mest pri dogodkih in izstavitev računov za prijavnine. Potrebno je tudi hraniti podatke strank za potrebe marketinga. Sistem mora imeti možnost dodajanja storitev in izstavljanja računov za uporabnike teh storitev.
Sistem ima lahko tudi spletni vmesnik. Spletni vmesnik bo na voljo le za uporabnike sistema − operaterje. Na spletnem vmesniku bo operater lahko označil termine, ko ima čas za delo v MMC-ju. Poleg tega ima možnost pregledovati tudi sporočila iz različnih virov preko RSS tehnologije. Prav tako bo možen vpogled v sistemski dnevnik. Dodatno bi lahko prikazovali tudi posnetke varnostnih kamer iz MMC-ja in s tem omogočili nadzor na daljavo.
Dobro bi bilo, da bi v sistem vključili tudi modul za vodenje projektov. Tu bo vidno trenutno stanje projektov, kdo sodeluje pri projektu in kakšni so trenutni in predvideni stroški projekta. Po končanem projektu se mora generirati podrobno poročilo.
V kolikor bi bilo možno, naj bi imel operater preko sistema možnost, da na enem mestu ureja več spletnih portalov oz. profilov. Hkrati pa naj bi imel tudi možnost, da sočasno objavi isto novico na različnih straneh.

2.3 Predstavitev razvojnih orodij

2.3.1 PowerDesigner

PowerDesigner je programsko orodje za izdelavo podatkovnih, objektnih, procesnih in drugih modelov. Uporablja se ga predvsem za načrtovanje podatkovne in informacijske arhitekture. Deluje na operacijskem sistemu Microsoft Windows kot namizna aplikacija. Uporabljali smo ga predvsem za načrtovanje podatkovne baze in izdelavo diagramov primera uporabe.

2.3.2 Microsoft Visual Studio

Microsoft Visual Studio je integrirano razvojno okolje (angl. Integrated development environment − IDE), ki ga je razvilo podjetje Microsoft. To je aplikacija, ki vsebuje orodja za razvoj in razhroščevanje programske opreme. Omogoča razvoj konzolnih in grafičnih uporabniških vmesnikov, kakor tudi spletne strani, spletne aplikacije ter spletne storitve. [8] Programska oprema, razvita s pomočjo Visual Studia, teče na Microsoftovih operacijskih sistemih in strežnikih. S pomočjo Visual Studia (verziji 2008 in 2010) smo oblikovali grafični uporabniški vmesnik in napisali kodo v programskem jeziku C#, ki teče v ozadju uporabniškega vmesnika.

2.3.2.1 Windows Forms

Windows Forms je ime za grafični vmesnik API, ki je vključen v Microsoftov paket knjižnic, imenovan ".NET Framework" [12] in je del razvojnega okolja Visual Studio. Uporabniku omogoča prilagojen prikaz podatkov s pomočjo komponent, ki so že vgrajene v razvojno okolje, jih razvijemo sami ali pridobimo s kakega drugega vira. Visual Studio s številnimi komponentami velja za najbogatejše razvojno okolje za Windows Forms.

2.3.2.2 Crystal Reports

Crystal Reports je programsko orodje za izdelavo in generiranje poslovnih poročil. Razvilo ga je francosko podjetje Business Objects. Microsoft pa ga je vključil v Visual Studio in s tem je Crystal Reports praktično postal standardno orodje za generiranje poročil v .NET aplikacijah. Vsebuje tudi pregledovalnik poročil, ki med drugim omogoča tiskanje in izvažanje poročil v druge programe.

2.3.3 MySQL

MySQL je sistem za upravljanje s podatkovnimi bazami (SUPB) oz. odprtokodna implementacija relacijske podatkovne baze, ki za delo s podatki uporablja jezik SQL. [9] Deluje po principu odjemalec − strežnik. V našem primeru bo program Pulsmaster deloval kot odjemalec, medtem ko bo MySQL v vlogi podatkovnega strežnika. To povezavo omogoča MySQL gonilnik za ADO.NET. ADO.NET je množica komponent za dostop do podatkov in podatkovnih storitev in so del okolja Visual Studio. [1]

2.3.4 Programski jezik C# in objektno programiranje

C# je objektno orientiran programski jezik, ki so ga razvili pri Microsoftu in pretežno izvira iz jezikov C++, Visual Basic in Java. Namenjen je pisanju programov v okolju .NET.

Objektno programiranje je način računalniškega programiranja, ki uporablja objekte kot podatkovne strukture. Objekt poleg atributov (spremenljivk) vsebuje tudi metode ali funkcije. V objektnem programiranju razred označuje objekte sorodnega tipa.

2.3.5 Povpraševalni jezik SQL

SQL je strukturirani povpraševalni jezik za delo s podatkovnimi bazami (ang. Structured Query Language − SQL). Je najbolj razširjen in standardiziran povpraševalni jezik za delo s podatkovnimi zbirkami. [11] Za povpraševanje uporablja programske stavke, ki posnemajo ukaze v naravnem jeziku. Podatki, ki jih glede na povpraševanje vrne SUPB, navadno prikazujemo v obliki tabel.

2.4 Tehnološke zahteve in omejitve

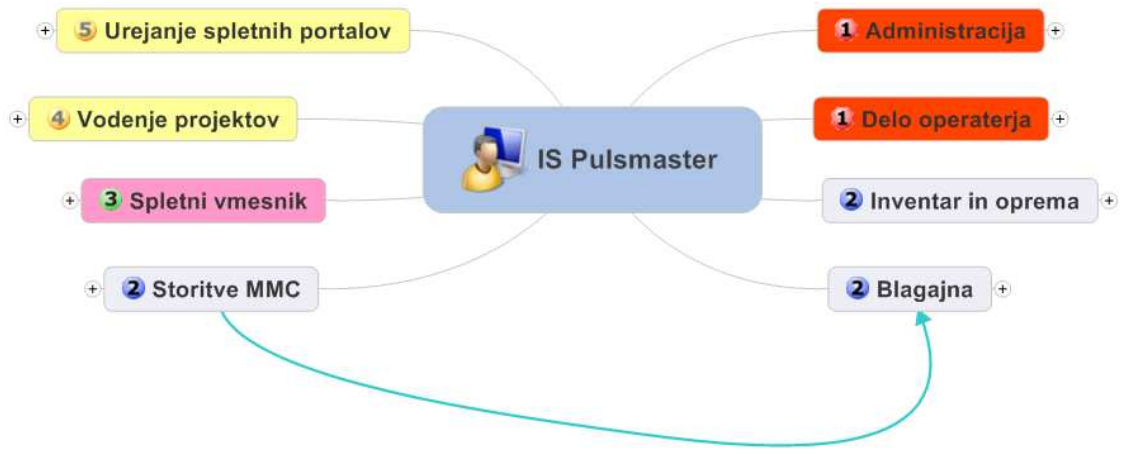
Naš informacijski sistem je sestavljen iz namizne aplikacije za operacijski sistem Microsoft Windows, ki je lahko nameščena na enem ali več računalnikih. Drugi del IS pa je centralni podatkovni strežnik oz. podatkovna baza MySQL. Za pravilno delovanje bo tako vsaka instanca aplikacije potrebovala dostop do podatkovnega strežnika bodisi preko lokalnega omrežja ali preko interneta. Podatkovni strežnik bo moral biti nastavljen tako, da bo omogočal obe vrsti povezav, hkrati pa bo moralo biti poskrbljeno tudi za varnost podatkov med prenosom in hranjenjem. Verjetno najbolj preprosta in učinkovita rešitev pri varnosti je, da je dostop do strežnika iz zunanjega omrežja (interneta) omogočen le preko navideznega zasebnega omrežja (Virtual Private Network − VPN), saj ta protokol poskrbi za kodiranje podatkov med prenosom. Za pravilno delovanje vseh funkcionalnosti sistema je v vsakem primeru potrebna tudi povezava aplikacije na svetovni splet.

2.5 Funkcionalnost in modularnost sistema

Planirana končna oblika IS bo vsebovala veliko različnih funkcionalnosti. Za potrebe te diplomske naloge jih bomo razvili le nekaj od teh. Vendar smo pri načrtovanju upoštevali vse uporabniške zahteve in jih razvrstili po pomembnosti, tako da bomo najprej razvili najpomembnejše funkcionalnosti sistema. Pri določanju pomembnosti posameznih modulov smo se ozirali na prioriteto, ki jo je določil naročnik, poleg tega pa še na soodvisnost posameznih modulov in težavnost implementacije. Če se programiranja lotimo v pravilnem vrstnem redu, nam bo inkrementalni razvoj omogočil, da bo IS prišel v uporabo, še preden bodo implementirane vse funkcionalnosti.

Pri načrtovanju smo sistem razdelili na osem poglavitnih skupin funkcionalnosti, te pa smo razvrstili med pet prednostnih razredov:

- 1. razred
  - Administracija
  - Delo operaterja
- 2. razred
  - Inventar in oprema
  - Blagajna
  - Storitve Multimedijskega centra
- 3. razred
  - Spletni vmesnik
- 4. razred
  - Vodenje projektov
- 5. razred
  - Urejanje spletnih portalov



Slika 3: Prikaz skupin funkcionalnosti sistema

2.6 Arhitektura podatkovne baze

Načrtovanje in izdelava modela podatkovne baze je pomembno in tudi zelo zahtevno opravilo. To je bilo prvo opravilo, takoj zatem, ko smo sprejeli in analizirali zahteve uporabnika. Iz zahtev je bilo potrebno izločiti entitete, jim določiti atribute in poiskati povezave med njimi. Povezavam je bilo potrebno določiti tudi pravilno kardinalnost oz. številnost. Tako smo dobili konceptualni model podatkovne baze. Programsko orodje PowerDesigner pretvori konceptualni v logični model. Logični model je načrt končne podatkovne baze. V principu ta pretvorba pomeni, da se entitete spremenijo v tabele oz. relacije, atributi postanejo stolpci tabel, povezave pa se pretvorijo v sistem primarnih in tujih ključev ali pa v vmesne tabele, odvisno od kardinalnosti. Če atributom nismo določili tipa (če gre za celo oz. decimalno število, niz znakov, datum itd.) že v konceptualnem modelu, moramo to storiti v logičnem. V nadaljevanju z določeno funkcijo v PowerDesignerju generiramo ukaze v jeziku SQL, ki jih nato zaženemo na sistemu MySQL in tako dobimo podatkovno bazo.

Sicer smo konceptualni model večkrat med razvojem tudi spreminjali ali dopolnjevali. Večkrat se šele med delom opazi, da smo izpustili kak kritični atribut ali pa dobimo idejo, kako bi funkcionalnost izboljšali s spreminjanjem tipa ali dodajanjem novega atributa. Po že opisanem postopku smo vsako spremembo modela prenesli v podatkovno bazo.

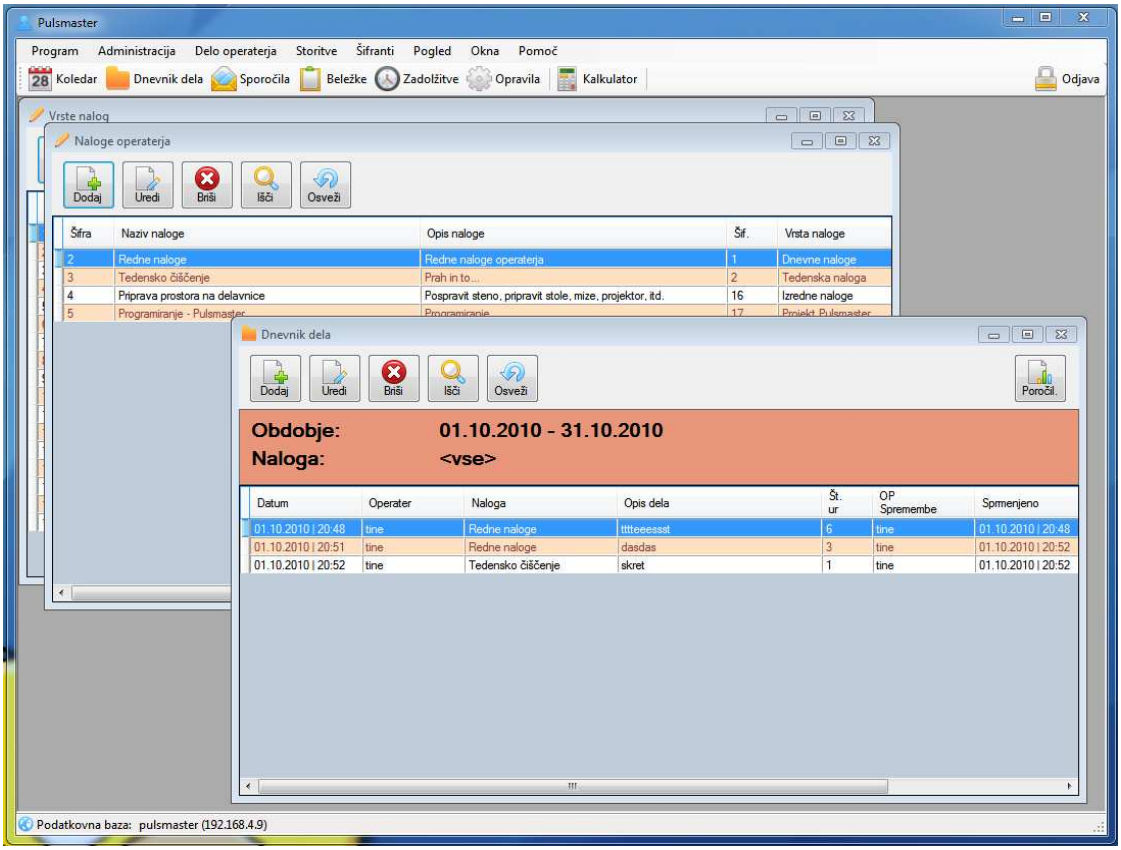
2.7 Razvoj programskih modulov

2.7.1 Večokenski grafični uporabniški vmesnik

Kot že rečeno, smo z razvojnim orodjem Visual Studio oblikovali in sprogramirali grafični uporabniški vmesnik aplikacije. Začetni pogled oz. osnovno okno programa je večokenski vmesnik (Multiple Document Interface − MDI), kar pomeni, da se lahko druga okna aplikacije prikazujejo znotraj osnovnega. V primerjavi z enookenskimi vmesniki (Single Document Interface − SDI) ima taka vrsta vmesnika svoje prednosti in slabosti. Med večjimi prednostmi MDI je gotovo možnost, da z enim klikom uporabnik zapre, skrije, ali poveča velikost vseh oken programa. Prav tako omogoča avtomatično razporeditev oken aplikacije, tako da se vsebina ne prekriva in ima uporabnik boljši pregled. Tudi z vidika porabe pomnilnika naj bi bile aplikacije MDI manj potratne. Malce težje pa je navigirati med okni, če so povečani čez celo osnovno okno ali če so pomanjšani, saj v orodni vrstici operacijskega sistema vidimo le osnovno okno programa. MDI lahko uporabniku povzroča probleme tudi takrat, ko ima uporabnik več zaslonov, saj težje razporedi okna na različne ekrane.

V osnovnem oknu programa imamo tudi menije, ki so zaradi lastnosti MDI uporabniku vedno na voljo ne glede na to, katero okno aplikacije je znotraj vmesnika odprto. Zato je to glavna navigacija po aplikaciji oz. njenih funkcionalnostih. V obstoječi verziji sta to dva menija. Prvi je standardni programski meni, kjer iz tekstovnih seznamov izberemo želeno opravilo ali funkcijo. Drugi meni pa je opremljen tudi z grafičnimi simboli oz. ikonami in je namenjen hitremu izboru najpogostejših opravil ali funkcij, ki jih bo

uporabnik pri svojem delu s programom potreboval. Čisto v spodnjem delu osnovnega okna pa se nahaja tako imenovana vrstica stanja. V tej vrstici je napisan status povezave s centralnim podatkovnim strežnikom.



Slika 4: Večokenski vmesnik programa Pulsmaster

2.7.2 Pomožni razredi

Za potrebe preglednega objektnega programiranja smo ustvarili nekaj statičnih razredov, ki jih bomo uporabljali na različnih mestih in v različnih modulih aplikacije. Statični razredi so tisti razredi, ki vsebujejo le statične metode in atribute, kar posledično pomeni, da iz takega razreda ne moremo ustvariti različnih objektov. Praktično se cel razred obnaša kot en objekt. To nam omogoča, da nam ni potrebno ustvarjati svoje instance razreda (se pravi, objekta), ampak metode in atribute kličemo kar direktno.

2.7.2.1 Razred Uporabnik

Razred Uporabnik hrani atribute oz. lastnosti trenutno prijavljenega uporabnika. Atributi se naložijo med prijavo, o kateri več v nadaljevanju. Ti atributi se namreč večkrat potrebujejo v različnih modulih in funkcijah. V principu je ta razred prepis določene vrstice iz tabele uporabnikov (v podatkovne bazi) v lokalni pomnilnik. Tako se izognemo večkratnemu ponovnemu dostopanju do podatkov v bazi, prav tako pa lahko izpustimo kakšno povezavo (stavek "JOIN") v poizvedbah SQL.

Pomembnejše lastnosti, ki jih hrani ta razred, so identifikacijska številka oz. šifra uporabnika, uporabniško ime in nivo dostopa. Šifro uporabnika namreč potrebujemo v več funkcijah programa, predvsem pri komunikaciji s podatkovno bazo in pri pogojih SQL ("WHERE"). Uporabniško ime ima predvsem vizualen pomen, saj uporabniku prikazujemo ta podatek namesto človeku neprijazne šifre. Nivo dostopa je vrednost, ki jo preberemo, ko uporabnik odpre določen modul ali zahteva določeno funkcionalnost programa. V nastavitvah, ki so shranjene v podatkovni bazi, nato program preveri, katere pravice so mu s to vrednostjo dodeljene. Glede na to se potem določene funkcije programa uporabniku omogočijo ali onemogočijo.

2.7.2.2 Razred za komunikacijo s podatkovno bazo

Ta razred vsebuje atribute, ki so potrebni za povezavo na podatkovno bazo. To so uporabniško ime in geslo za prijavo v podatkovno bazo, IP naslov in vrata podatkovnega strežnika ter ime baze podatkov. Ti atributi se prav tako nastavijo pri prijavi uporabnika. Iz njih sestavimo povezovalni niz znakov (angl. Connection String) ter ga pošljemo v funkcijo za povezovanje. Če in ko se povezava uspešno vzpostavi, lahko začnemo pošiljati poizvedbe SQL na podatkovni strežnik.

Poleg atributov in funkcije za povezovanje z bazo so v tem razredu tudi funkcije, ki izvedejo povpraševanja glede na njihove parametre in vrnejo podatke, zapakirane v objektu, katerega potem lahko prikažemo na preglednici. Te funkcije so v tem razredu zaradi boljše preglednosti in lažjega spreminjanja kode. Še posebej se izkaže za priročno, kadar pride do kakšnih strukturnih sprememb v podatkovni bazi, saj je potrebno popraviti le ta razred.

2.7.2.3 Razredi za skupne funkcije in konstante

Ko programiramo kompleksno aplikacijo, se izkaže, da potrebujemo kako funkcijo ali konstanto večkrat v različnih modulih, v različnih razredih. Ker želimo program napisati pregledno in na tak način, da ga bo laže vzdrževati, smo take funkcije napisali v tri razrede skupnih funkcij.

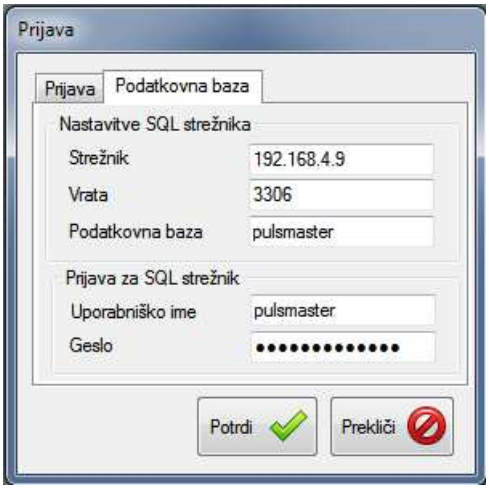
V en razred smo napisali vse konstante. To so predvsem sporočila, ki jih program izpiše uporabniku v programu na različnih mestih. Po potrebi bi lahko tu bile še kakšne druge konstante.

V drugem razredu so napisane funkcije za kontrolo vnosa podatkov. Te funkcije se kličejo, ko uporabnik potrdi obrazec za vnos podatkov. Za vsako vnosno polje program s pomočjo teh funkcij preveri, če so podatki v taki obliki, kot jih pričakujemo. Na primer: tekstovno polje ne sme biti prazno ali pa ne sme vsebovati samo presledkov; polje, kjer se pričakuje številčni vnos, ne sme vsebovati drugih znakov kot cifre; vnos za datum mora biti pravilno oblikovan itd. Če podatki niso v pravilni obliki, program obvesti uporabnika, naj jih popravi.

Tretji razred je razred različnih funkcij, ki si niso med seboj podobne, vendar se jih v programu večkrat uporablja. To so lahko matematične ali pa čisto specifične funkcije.

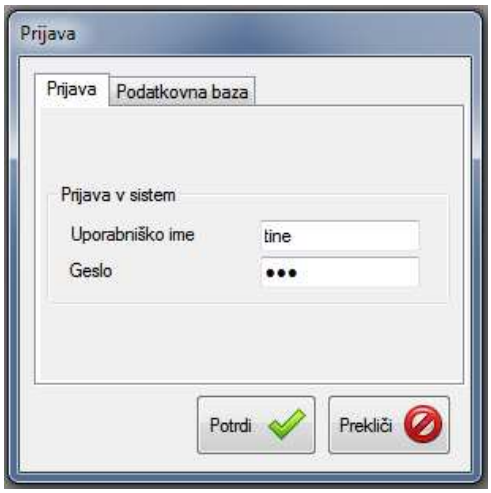
2.7.3 Prijava uporabnika

Prijava je prvi proces programa, skozi katerega mora iti uporabnik, ko aplikacijo zažene. Ima dva namena: zbrati potrebne podatke za povezavo na podatkovno bazo in pa identifikacijo ter avtentikacijo uporabnika v informacijski sistem.



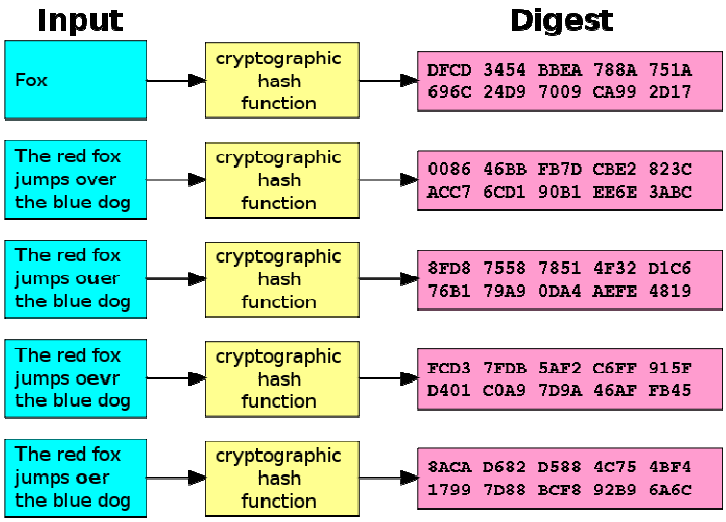
Slika 5: Prijavno okno s podatki za povezavo na podatkovno bazo

Podatki za povezavo z bazo so naslov strežnika, vrata in ime podatkovne baze. Poleg tega mora uporabnik vnesti tudi uporabniško ime in geslo, s katerim ima aplikacija dostop do podatkov v tej podatkovni bazi. Kot že rečeno, se ti podatki ob potrditvi zapišejo v razred, ki skrbi za dostop do podatkovne baze. Tu se obdržijo, dokler programa ne ugasnemo. Vse te podatke bi bilo zamudno izpolnjevati vsakokrat, ko se program Pulsmaster zažene, zato so že avtomatsko vpisani oz. "zapečeni" v programsko kodo. To lahko upravičimo s tem, da je baza podatkov centralna in enotna za vse instance programa, prav tako pa je program pisan za točno določen računalnik oz. mrežo računalnikov v multimedijskem centru. Tu bi morale te nastavitve delovati na vsaki instanci programa. V prihodnjih verzijah bo najverjetneje potrebno shranjevati nastavitve v datoteko nekje na lokalnem disku in s tem omogočiti lažji prenos programa na druga omrežja oz. druge sisteme. Pri shranjevanju nastavitev v lokalne datoteke pa je velik problem z varovanjem podatkov, saj bi do vsebine te datoteke lahko prišla nepooblaščena oseba s slabimi nameni. S temi podatki bi namreč lahko vdrla v podatkovno bazo in ukradla, spremenila ali izbrisala ključne podatke. Zato bi morali to datoteko zaščititi s kodiranjem. Poudariti je tudi treba, da tudi "zapečeni" podatki v programski kodi niso dovolj varno shranjeni, saj obstaja dokaj preprost način, kako priti do njih, zato je ta problem še vedno odprt. V tem trenutku bi bilo najbolj varno (vsaj iz stališča razvijalca programske opreme), da bi uporabnik vsakič znova vpisoval vse podatke oz. vsaj uporabniško ime in geslo. S tem bi problem, kako varno shraniti ta dva podatka, prenesli na uporabnika samega, hkrati pa bi mu otežili oz. podaljšali proces prijave. Ker so ti podatki lahko enaki za več uporabnikov sistema, je v tem primeru nevarnost, da pridejo v roke nepooblaščeni osebi, še toliko večja.



Slika 6: Prijavno okno zahteva podatke za prijavo v sistem

Drugi del ali drugi namen prijavnega okna zahteva od uporabnika unikatno uporabniško ime za identifikacijo ter geslo za avtentikacijo. Po potrditvi program primerja vnesena podatka s podatki, ki so shranjeni v podatkovni bazi. V kolikor se ujemata, je bila prijava v sistem uspešna. V tem primeru se podatki iz baze prepišejo v razred, ki hrani podatke trenutnega uporabnika, in smo ga že omenili.



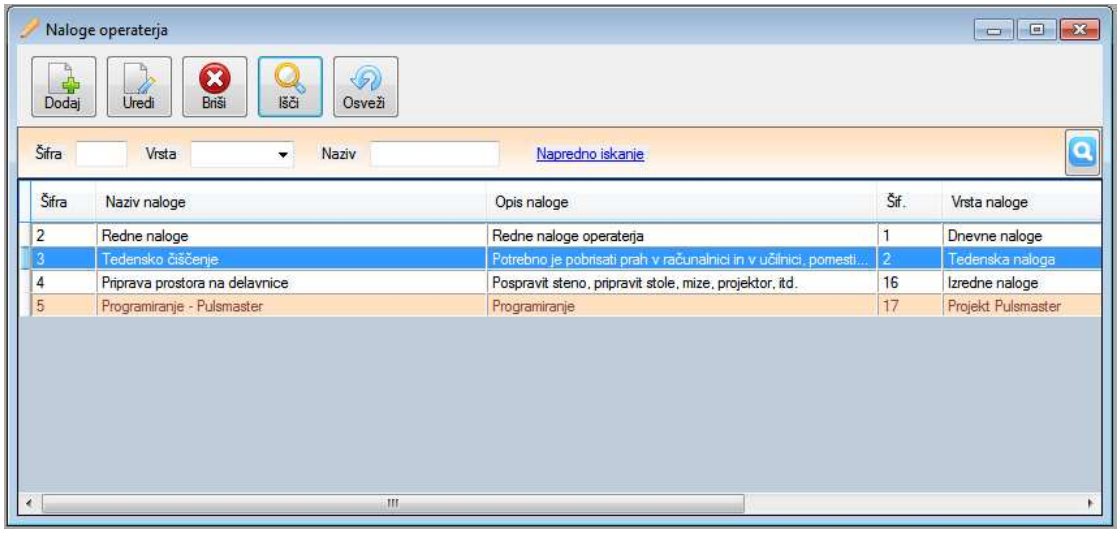
Slika 7: Prikaz delovanja kriptografske zgostitvene funkcije. [3]

Tudi v tem primeru program še ni optimalno zavarovan. Do podatkov, shranjenih v podatkovni bazi, namreč lahko dostopa vsak, ki ima dostop do baze. To pomeni, da bi lahko nekdo z dostopom do baze prišel do podatkov drugega uporabnika in se prijavil v sistem z njegovim uporabniškim imenom. Rešitev za ta problem je enkripcija gesla s kriptografsko zgostitveno funkcijo (angl. Cryptographic hash function). Ta funkcija pretvori podatek v fiksno velik niz bitov. Iz tega niza je izjemno težko pridobiti prvoten podatek, saj je ta funkcija enosmerna. Ima pa to lastnost, da se enak podatek vedno pretvori v enako pretvorjeno vrednost. Zato to lahko uporabimo pri avtentikaciji, in

sicer tako, da vneseno geslo pri prijavi program pretvori z zgostitveno funkcijo in dobljen niz primerja z nizom v podatkovni bazi. Če se niza ujemata, je geslo pravilno. Sicer tudi kodiranje z zgostitveno funkcijo ni stoodstotno varno. Napadalec bi namreč lahko s poizkušanjem uganil, kateri niz je zakodiran. Vendar ima zgostitvena funkcija še to lastnost, da za majhne spremembe v vhodnem nizu znakov povzročijo veliko spremembo rezultata (slika 7). Tako napadalec nikoli ne ve, kako blizu rešitve je, dokler je dejansko ne ugane. Tak nivo varnosti je zadovoljiv za sisteme, kot je naš, z relativno manj vrednimi podatki in manjšo verjetnostjo napada.

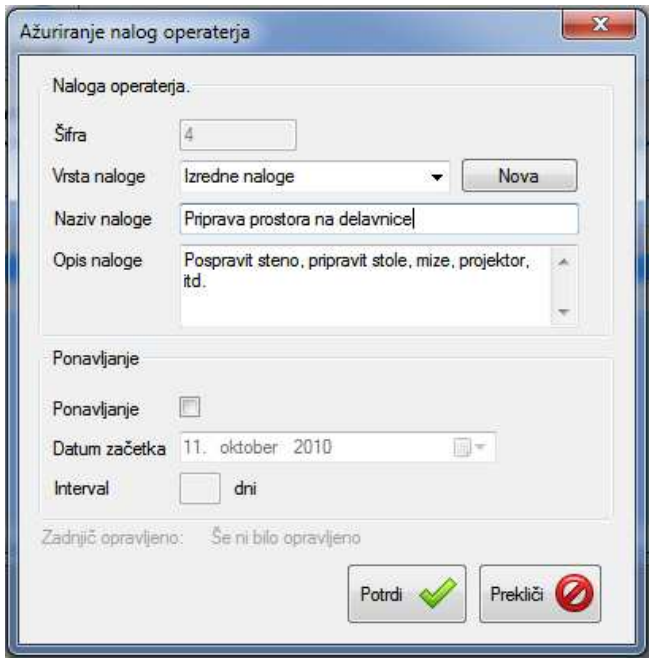
2.7.4 Šifranti

V transakcijskih informacijskih sistemih je število šifrantov relativno veliko. Za vsak šifrant smo naredil svoje okno, ki se odpira znotraj večokenskega vmesnika. V njem prikazujemo podatke v preglednici. Preglednico napolnimo s klicem primerne funkcije za komunikacijo s podatkovno bazo, ki pošlje na strežnik ustrezno povpraševanje SQL. V zgornjem delu okna se nahajajo akcijski gumbi, s katerimi začnemo postopek za določeno akcijo v šifrantu. Po večini so te akcije dodajanje, urejanje in brisanje podatkov ter iskanje po šifrantu.



Slika 8: Primer šifranta s prikazano iskalno vrstico

S klikom na gumb "Dodaj" se nam odpre pogovorno okno, ki zahteva od nas, da v polja vpišemo ali izberemo podatke za nov vnos v šifrant. Ko kliknemo na gumb "Potrdi", program pokliče že prej opisane kontrolne funkcije, ki preverijo pravilnost vnosov. Včasih so za uspešno potrditev potrebni vnosi v vsa polja, včasih pa so obvezna le nekatera, odvisno od narave šifranta. Šifro vnosa, če ni drugače določeno, generira SUBP. Ob uspešni potrditvi se vrednosti polj pošljejo funkciji za komunikacijo s podatkovno bazo, kjer se generira stavek SQL. Tega pa pošlje naprej na podatkovni strežnik. Če ne pride do napake, se podatki vpišejo v podatkovno bazo.



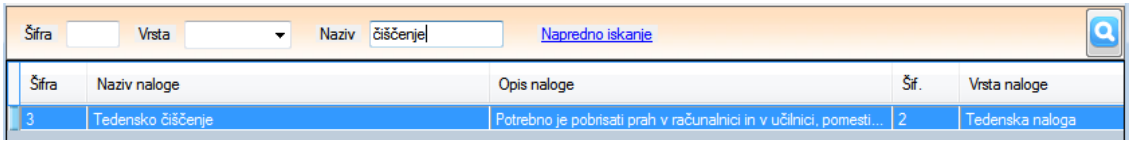
Slika 9: Pogovorno okno za urejanje šifranta

Gumb "Uredi" prikliče na zaslon isto pogovorno okno kot gumb "Dodaj", le da so v tem primeru vrednosti trenutno izbrane vrstice v preglednici že vnesena v vnosna polja. Uporabnik ima možnost, da te podatke spremeni, z izjemo šifre. Ob potrditvi se enako kot prej vnosi preverijo. Tudi potek vpisa podatkov v bazo je podoben, le da se tokrat generira malce drugačen stavek SQL, saj ne gre za dodajanje, temveč za urejanje vnosa pod določeno šifro.

Za akcijo "Briši" je dejansko veliko več problemov, kot bi si sprva mislili. Namreč v večini šifrantov vnosov ne moremo enostavno kar izbrisati iz baze, saj so z njimi lahko povezani drugi vnosi v drugih tabelah v bazi in tako pride do izgube integritete podatkov. Rešitve za te probleme so sicer že vgrajene v SUBP z omejitvami tujih ključev. Te omejitve nam v takem primeru lahko preprečijo brisanje, lahko izbrišejo vse povezane vnose ali pa izbrišejo zgolj povezave. Nobena od teh rešitev ni najboljša, saj bi v najslabšem primeru lahko izgubili precej podatkov. Zato v tem trenutku v nekaterih šifrantih brisanje ni omogočeno, v naslednji verziji pa lahko ta problem rešimo s statusi. To pomeni, da ima vsaka vrstica v šifrantu tudi polje status, ki nam pove, ali je podatek še aktiven ali je ukinjen. Po potrebi in če je povezava na druge tabele pomembna, lahko tudi na vseh povezanih vnosih hkrati spremenimo status na ukinjen ali pa dodamo status "potreben popravka", ki bi uporabnika opozoril, da je povezava postala neveljavna in da mora poiskati drugo.

Gumb "Briši" torej prikliče na zaslon pogovorno okno, ki uporabnika vpraša, če je prepričan o izvedbi te akcije. To je seveda standardni pristop, ki preprečuje, da bi uporabnik po nesreči izbrisal podatek. Ob potrditvi se tako, kot že opisano, pošlje primeren stavek SQL na bazo, kjer se vnos ustrezno spremeni ali izbriše.

Iskanje po šifrantih je implementirano tako, da se s klikom na gumb "Išči" prikaže (ali skrije) dodatna orodna vrstica nad preglednico, v kateri so vnosna polja, ki implicirajo na stolpce, po katerih je omogočeno iskanje. Med pisanjem uporabnika v ta polja se že sproti izvaja iskanje. V nekaterih šifrantih, kjer bi bilo uporabno iskanje po več kot treh poljih, je v iskalni vrstici poleg nekaj iskalnih polj še možnost naprednega iskanja. Ta odpre novo pogovorno okno z večjim številom vnosnih polj, po katerih je možno iskanje. Število vnosnih polj v iskalni vrstici je omejeno, ker mora biti program prilagodljiv tudi na manjše zaslone. Na zaslonih z majhno resolucijo bi pri velikem številu polj v iskalni vrstici namreč lahko prišlo do tega, da se nekaterih polj ne bi videlo.



Slika 10: Primer iskanja v orodni vrstici

Programsko iskanje poteka tako, da se na bazo pošiljajo SQL stavki s pogoji iz iskalnih polj. Rezultat se potem prikaže v preglednici. Ta način, ki sicer trenutno deluje v redu, lahko v primeru, da je v šifrantu veliko podatkov, postane počasen in preformančno zahteven. Sicer ni verjetno, da bi kateri od šifrantov postal ogromen, vendar bi bilo potrebno v naslednjih verzijah iskanje vseeno optimizirati. Boljša rešitev bi bila, če bi podatke filtrirali v lokalnem pomnilniku. V .NET okolju obstaja integriran poizvedovalni jezik LINQ (Language Integrated Query), ki bi nam pri tem prišel prav. Na ta način poizvedb ne bi pošiljali na podatkovni strežnik, temveč bi jih obdelovali lokalno znotraj objekta, ki hrani podatke.

2.7.5 Administracija

Slika 11: Shema administrativnih funkcionalnosti sistema

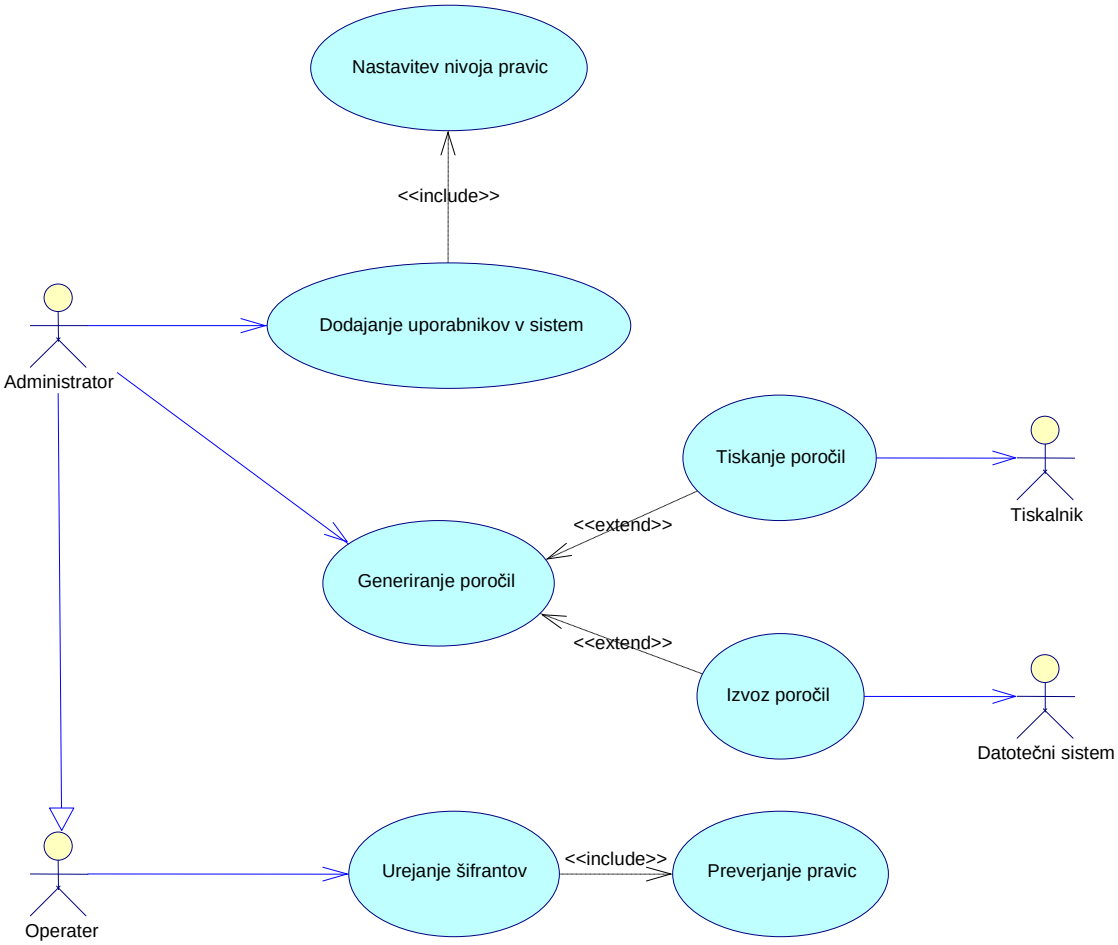
2.7.5.1 Dodelitev pravic

Ena od pomembnejših lastnosti aplikacije Pulsmaster je večnivojski dostop za uporabnike. V osnovi je mišljeno, da bomo imeli dva nivoja: administrator in operater. Vendar je program napisan tako, da bo možno tudi več nivojev. Pomen večnivojskega dostopa je v tem, da imajo različne vrste uporabnikov različne pravice in s tem omogočene različne akcije v programu.

Katere akcije bodo omogočene uporabnikom, bo administrator lahko nastavil v posebnem šifrantu pravic. Tu gre za akcije dodajanja, urejanja in branja podatkov v

modulih. Za vsak modul se vsaki akciji nastavi določen nivo dostopa. Če je uporabnik oz. operater član skupine operaterjev, ki ima nivo dostopa večji ali enak, kot je zahtevan za določeno akcijo, se mu ta akcija omogoči. Nivo dostopa posamezne skupine operaterjev je nastavljiv v svojem šifrantu.

Urejanje podatkov šifranta pravic ima zgolj skupina uporabnikov z najvišjim nivojem dostopa, se pravi administrator ali bolj konkretno koordinator MMC-ja.



Slika 12: Diagram primera uporabe administrativnih funkcij

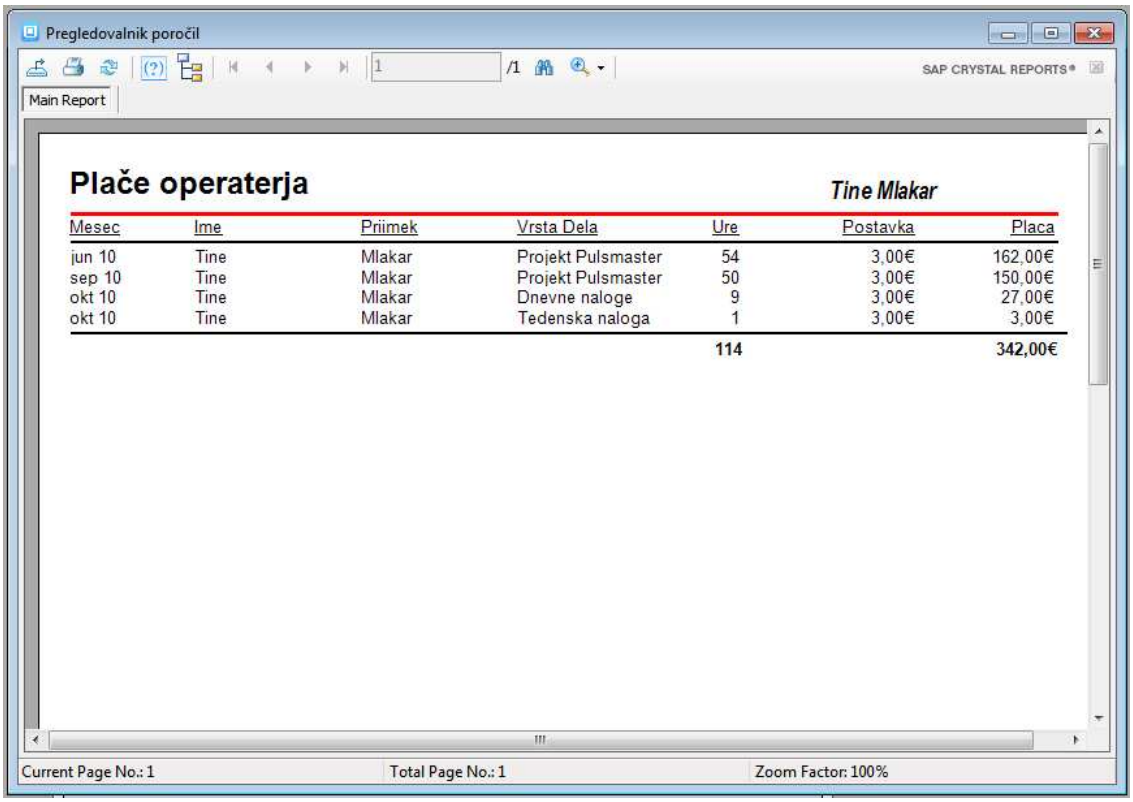
2.7.5.2 Urejanje šifrantov

Kot del administracije sistema je tudi urejanje šifrantov. O šifrantih smo že veliko povedali, zato sedaj bolj na kratko. Večino šifrantov bo glede na dodeljene pravice po vsej verjetnosti urejal tudi operater. Nekaj pa bo takih, bolj varovanih, ki jih bo urejal le administrator oz. operater. Poseben primer je šifrant pravic, kjer dodajanje novih vnosov ne bo možno v programu samem (možno bo le spreminjanje obstoječih vnosov), temveč bo to storil programer direktno v bazi, ko bo dodal v sistem nov modul.

2.7.5.3 Poročila

Generiranje poročil je ena bolj naprednih funkcionalnosti informacijskega sistema Pulsmaster. Z njo presega mejo transakcijskih in posega po prvinah upraviteljskih informacijskih sistemov. Poročila bodo v pomoč predvsem koordinatorju MMC-ja in

upravi Zavoda O pri sprejemanju strateških odločitev ali zgolj vodenju statistike in evidenc. Ker se MMC po večini financira iz raznih javnih razpisov, ki zahtevajo ob koncu določenega projekta temeljito programsko in finančno poročilo, računam, da bo ta funkcionalnost sistema tudi v tem primeru zelo koristna.



Slika 13: Pregledovalnik poročil in primer poročila

Kot rečeno smo za generiranje in prikaz poročil uporabili orodje Crystal Reports, ki je vključeno v Visual Studio. To orodje omogoča dokaj preprosto generiranje poročil, čeprav ni popolnoma prilagojen za delo z MySQL-om. Crystal Reports knjižnice vsebujejo tudi pregledovalnik generiranih poročil. Ta pregledovalnik ima zelo uporabne funkcionalnosti. Poleg možnosti tiskanja je zelo uporabna možnost izvoza poročila v različne priročne formate datotek, kot so:

- xls (Microsoft Excell preglednice),
- pdf (Človeku prijazna oblika datoteke),
- doc (Microsoft Word dokument) ter
- xml (oblika, primerna za elektronski prenos podatkov).

Glede na vrsto izvoznega formata lahko to poročilo naknadno urejamo v izbranem urejevalniku (format xls ali doc), ga pošljemo preko elektronske pošte ali objavimo na internetni strani (format pdf), lahko pa ga pošljemo oz. uvozimo v kak drug program (format xml).

Veliko generatorjev poročil še ni implementiranih v sistem, saj pridejo v naslednjih verzijah sistema skupaj s planiranimi moduli.

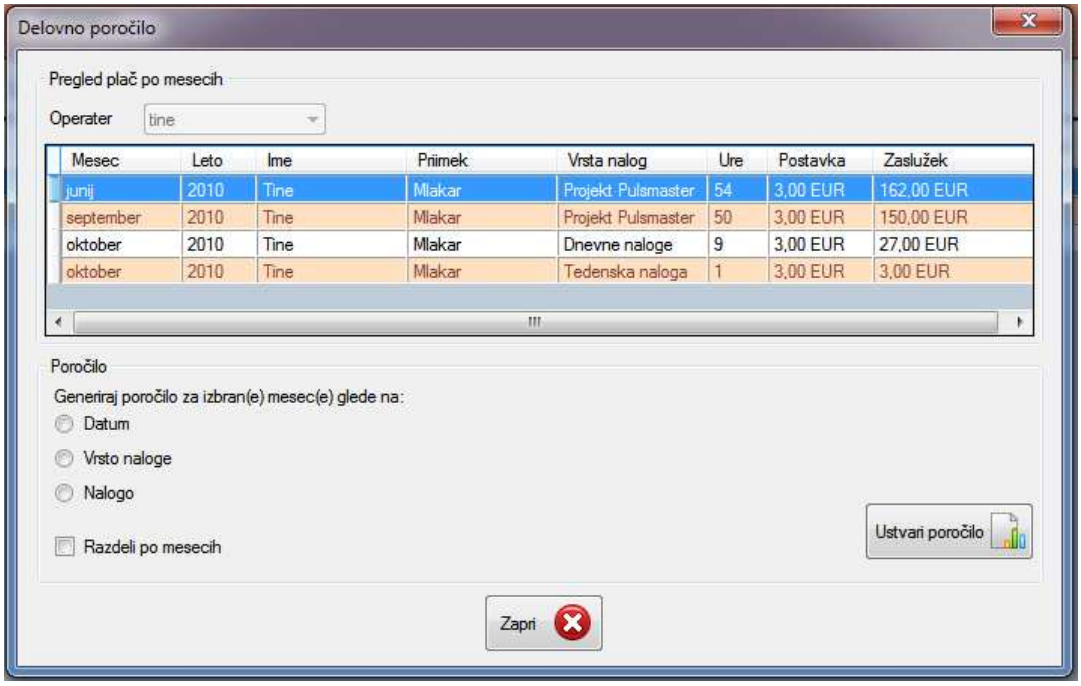
2.7.6 Delo operaterja

Primarna naloga IS Pulsmaster je olajšati in izboljšati operativno delovanje multimedijskega centra. To bo v veliki meri dosegel s tem, da bo nudil operaterju pri njegovih vsakdanjih opravilih informacijsko podporo.

Slika 14: Prikaz funkcionalnosti sistema za podporo dela operaterja

2.7.6.1 Dnevnik dela in izračun plače

Ena najpomembnejših funkcionalnosti sistema je hranjenje podatkov o delu operaterja. Te podatke vpisuje operater v dnevnik dela. Potrebno je hraniti podatke o količini dela in poteku tega dela, ki to količino opraviči. Poleg tega operater izbere tudi vrsto dela in glede na to se določi tudi urna postavka. Operater ima vedno možnost preveriti, koliko dela je opravil v določenem mesecu in kakšno plačilo lahko za to pričakuje. Iz teh podatkov lahko tudi generira poročilo.



Slika 15: Pogovorno okno za pregled zaslužka

Modul dnevnik dela je sprogramiran podobno kot šifranti, le da ima dodano akcijo, ki odpre pogovorno okno, v katerem je pregled opravljenih ur po mesecu ter vrsti dela in možnost generiranja poročil.

Sicer pa je generiranje poročil namenjeno predvsem koordinatorju oz. osebi, ki je zadolžena za izplačilo plač, in pa programskemu vodji, saj ima tudi možnost izpisa poročila glede na projekt. Tako dobi nekatere informacije o določenem projektu oz. vrsti dela, kot so število ur, opravljenih na projektu, stroški projekta itd.

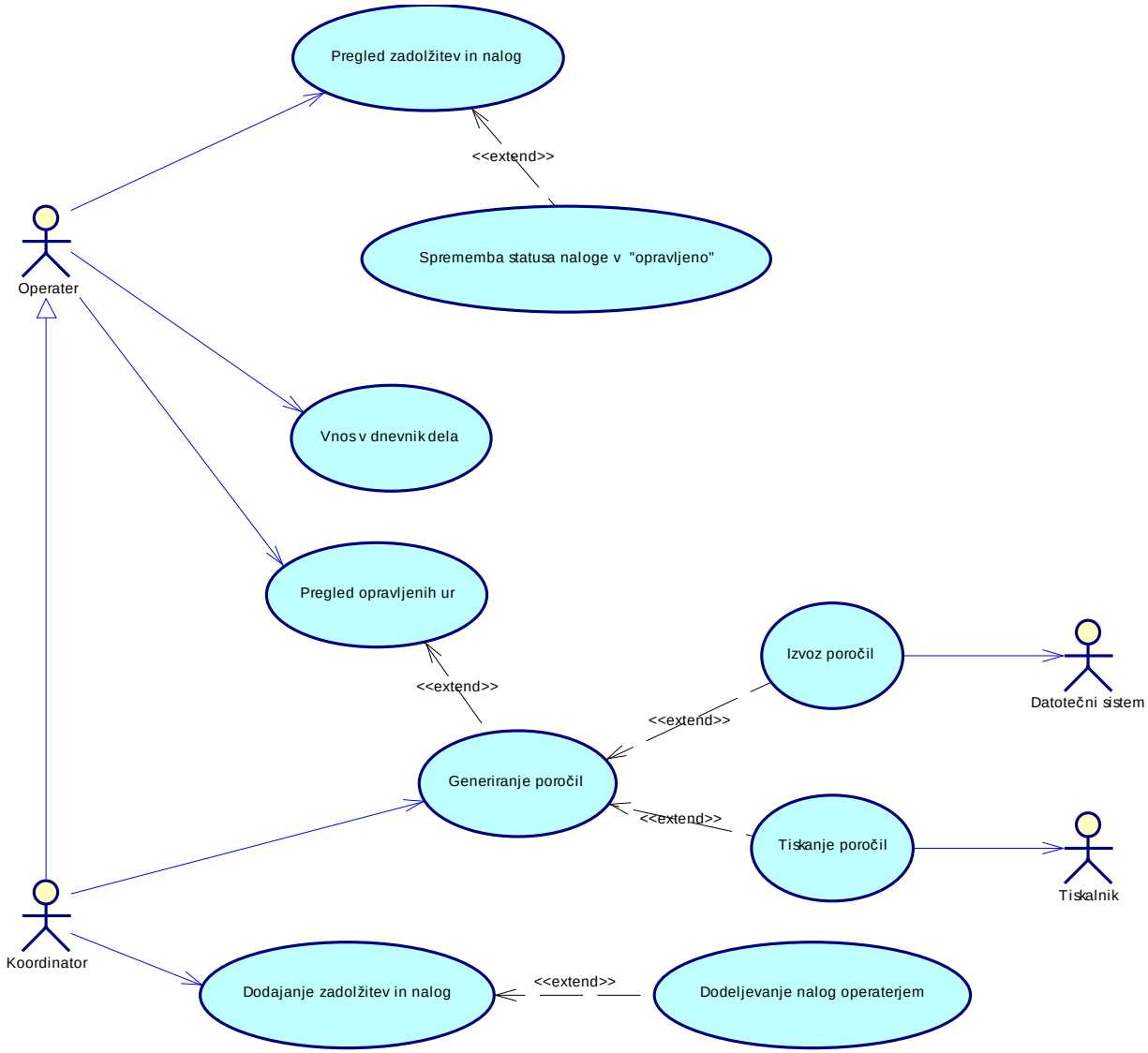
2.7.6.2 Zadolžitve

Operater ima na delovnem mestu določene zadolžitve. Zadolžen je na primer za red in čistočo v prostoru, za pripravo učilnice na predavanje, za urejanje spletnih strani, za strokovno pomoč obiskovalcem itd. Poleg raznih rednih nalog se pojavijo naloge oz. opravki čisto specifične narave. Gre npr. za kakšno opravilo znotraj tekočega projekta ali zgolj nekaj, kar je tisti trenutek potrebno storiti in kar mu je dodelil koordinator ali drugi operater. V preteklosti so si te naloge operaterji zapisovali na list papirja in tega obesili na zid ali pa so si dopisovali po elektronski pošti.

S pomočjo Pulsmasterja bo operater imel natančen pregled nad svojimi zadolžitvami v preglednici. Program mu bo prikazoval naloge, urejene po prioriteti in bližini roka izvedbe. Tako bo delovanje izboljšano, saj se bodo vedno najprej opravila nujna dela, posledično pa se bo zmanjšalo število neopravljenih nalog in zamujenih rokov.

Modul zadolžitev omogoča dodajanje in označevanje zadolžitev kot opravljene. Novo zadolžitev uporabnik doda tako, da izbere nalogo iz šifranta nalog ter rok za opravilo. Poleg tega lahko določi tudi operaterja, ki mu bo ta naloga zaupana. Če ne izbere operaterja, je zadolžitev vidna vsem uporabnikom, sicer pa le določenemu. Ko operater nalogo opravi, jo označi kot opravljeno. S tem se spremeni prikaz te naloge v preglednici.

Nekatere zadolžitve, ki se ciklično ponavljajo, bo program dodajal sam. Primer take zadolžitve je tedensko brisanje prahu. Interval ponavljanja naloge se določi pri vnosu operaterjevih nalog v šifrant.



Slika 16: Diagram primera uporabe modulov dnevnika dela in zadolžitev

2.7.6.3 Osebni organizator − beležke in opravila

Ponavadi ima vsak operater v MMC-ju tudi svoje projekte, ki so bili zaupani le njemu, ali pa si po svoje želi razdeliti določena opravila. Tu mu bo na pomoč priskočil modul "osebni organizator", kjer si bo lahko po svoje sestavljal seznam opravkov, jim določal roke, si nastavljal opomnike, hkrati pa si bo lahko shranjeval določene zapiske.

Ker je trenutno na voljo zelo veliko prostodostopne programske opreme in spletnih aplikacij, ki tako ali drugače podpirajo te funkcionalnosti, in ker nima kritičnega pomena za sistem, tega modula še nismo implementirali.

2.7.6.4 Sporočila

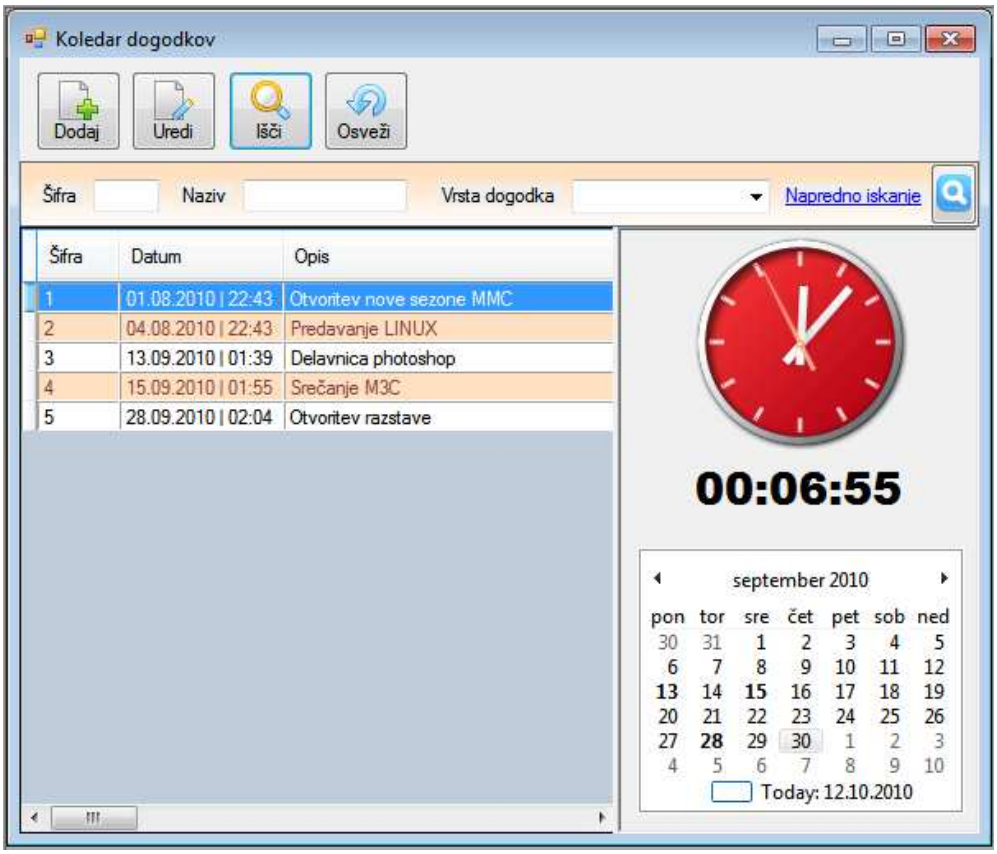
Modul sporočila omogoča pošiljanje sporočil med operaterji. Modul je namenjen predvsem komunikaciji med operaterji. Tudi tega modula še nismo implementirali, saj trenutno dokaj dobro poteka komunikacija preko elektronske pošte. Osnovna ideja je sicer dokaj podobna, le da je komunikacija možna le med uporabniki sistema.

Razmisliti bi bilo dobro tudi o možnosti združitve z obstoječim komunikacijskim kanalom, saj bi uporaba več kot enega načina komunikacije lahko pripomogla le k večji zmedi in ne k učinkovitosti. Že sedaj obstaja poleg elektronske pošte tudi spletni forum, kjer komunikacija med operaterji in zunanjimi sodelavci poteka združeno glede na teme. Iz izkušenj vemo, da se forum ne spremlja tako pogosto kot e-pošto in to včasih pripelje do tega, da sporočilo ne pride do prejemnika v pravem času.

Opcija je tudi implementacija spletnega foruma v program Pulsmaster. S tehnologijo RSS bi lahko uporabnika sproti obveščali o novih aktivnostih na forumu in s tem omogočili hitrejši odziv na sporočilo.

2.7.6.5 Koledar dogodkov

Koledar dogodkov omogoča uporabniku, da v podatkovno bazo dodaja dogodke, ki jih prireja MMC ali druge organizacijske enote Zavoda O. Praktično je to navaden šifrant, ki poleg preglednice prikazuje vnose tudi na koledarju. Na te dogodke bo v sklopu storitev operater lahko prijavljal udeležence. Poleg takih dogodkov bo možno vnašati tudi dogodke, ki zadevajo operaterje znotraj MMC-ja. Ti dogodki bodo predvsem informativne narave (npr. rok za oddajo razpisne dokumentacije).

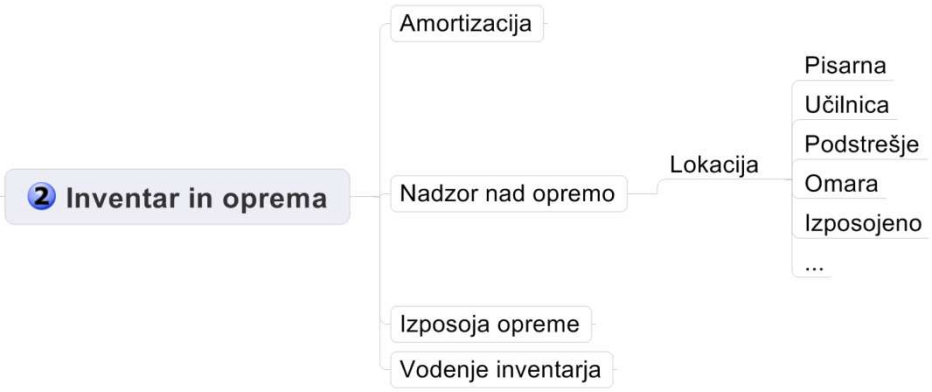


Slika 17: Koledar dogodkov

2.8 Planirani programski moduli

Velikega dela sistema nam žal še ni uspelo razviti. Nekateri moduli so že v fazi načrtovanja in so že predvideni tudi v načrtu podatkovne baze. V nadaljevanju bomo na kratko pogledali ideje o modulih, ki bodo v naslednjih inkrementih sistema.

2.8.1 Inventar in oprema

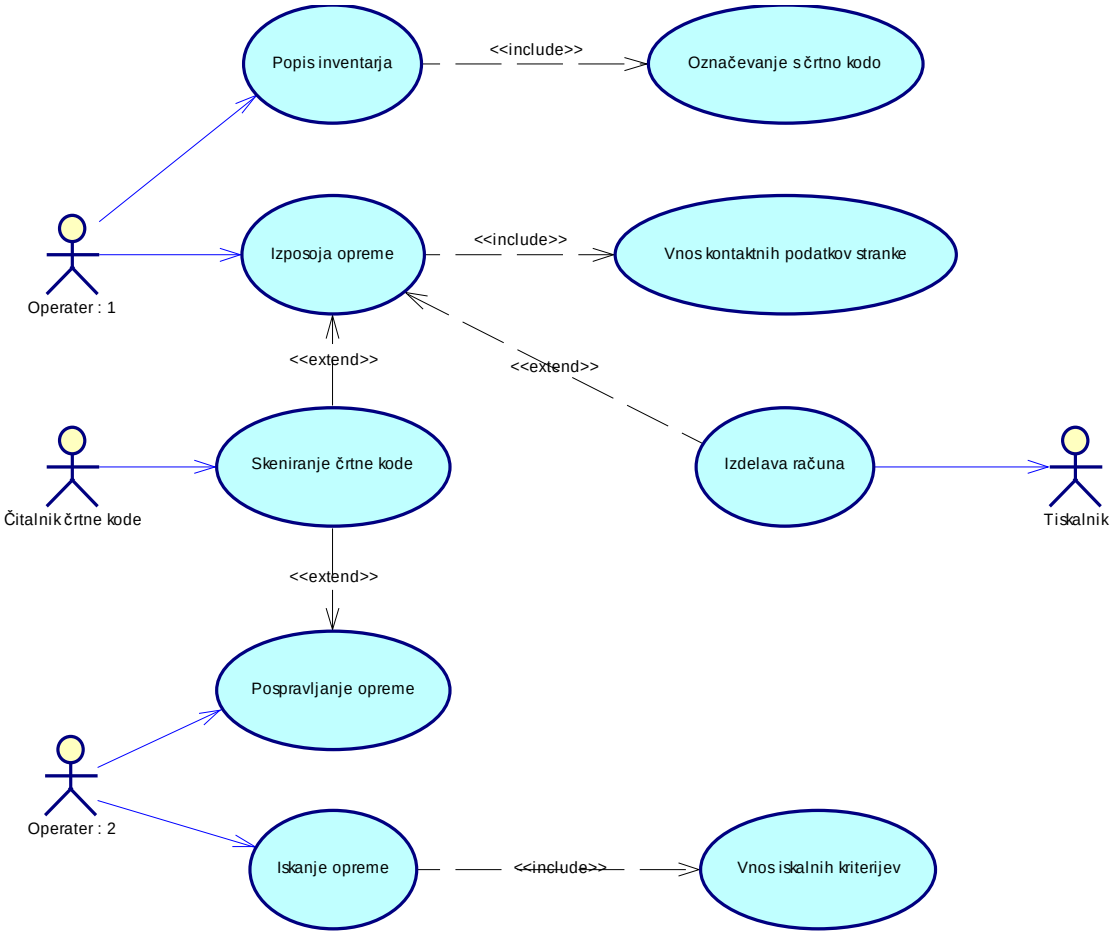


Slika 18: Prikaz funkcionalnosti inventarnega modula

V MMC-ju je veliko opreme oz. inventarja, ki ga je potrebno evidentirati. Ves inventar bo potrebno polepiti z nalepkami s šifro in črtno kodo, ter pod isto šifro vpisati predmet v sistem. V ta zapis bomo dodali naziv, opis, ceno, starost, lokacijo predmeta itd. V glavnem: čim več koristnih podatkov. Iz teh podatkov bomo lahko razbrali, kje naj bi se predmet nahajal, izračunali amortizacijo in še kaj.

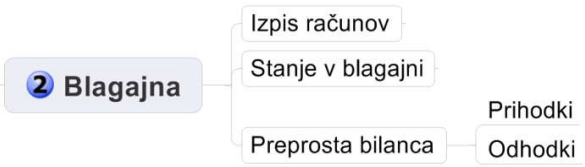
V primeru iskanja nekega predmeta, na primer omrežnega kabla, bo uporabnik v sistemu poiskal zapis iz baze ter prebral lokacijo. In ko bo želel določen predmet pospraviti, bo s čitalcem črtne kode odčital nalepko in sistem mu bo prikazal ustrezen zapis in lokacijo, kamor stvar spada.

Če bi si neka stranka želela izposoditi opremo, recimo projektor, bi operater zopet odčital črtno kodo in na zapisu označil, da je predmet izposojen. Vnesel bi tudi kontaktne podatke stranke in po potrebi lahko izdal račun.



Slika 19: Diagram primera uporabe inventarnega modula

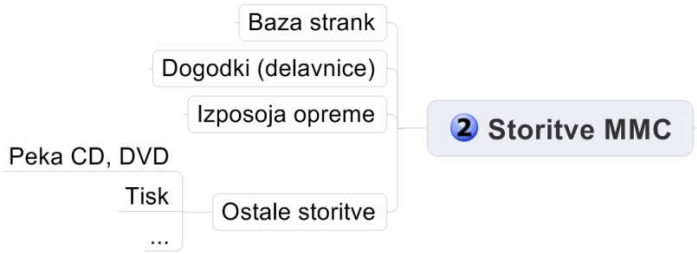
2.8.2 Blagajna



Slika 20: Prikaz funkcionalnosti blagajniškega modula

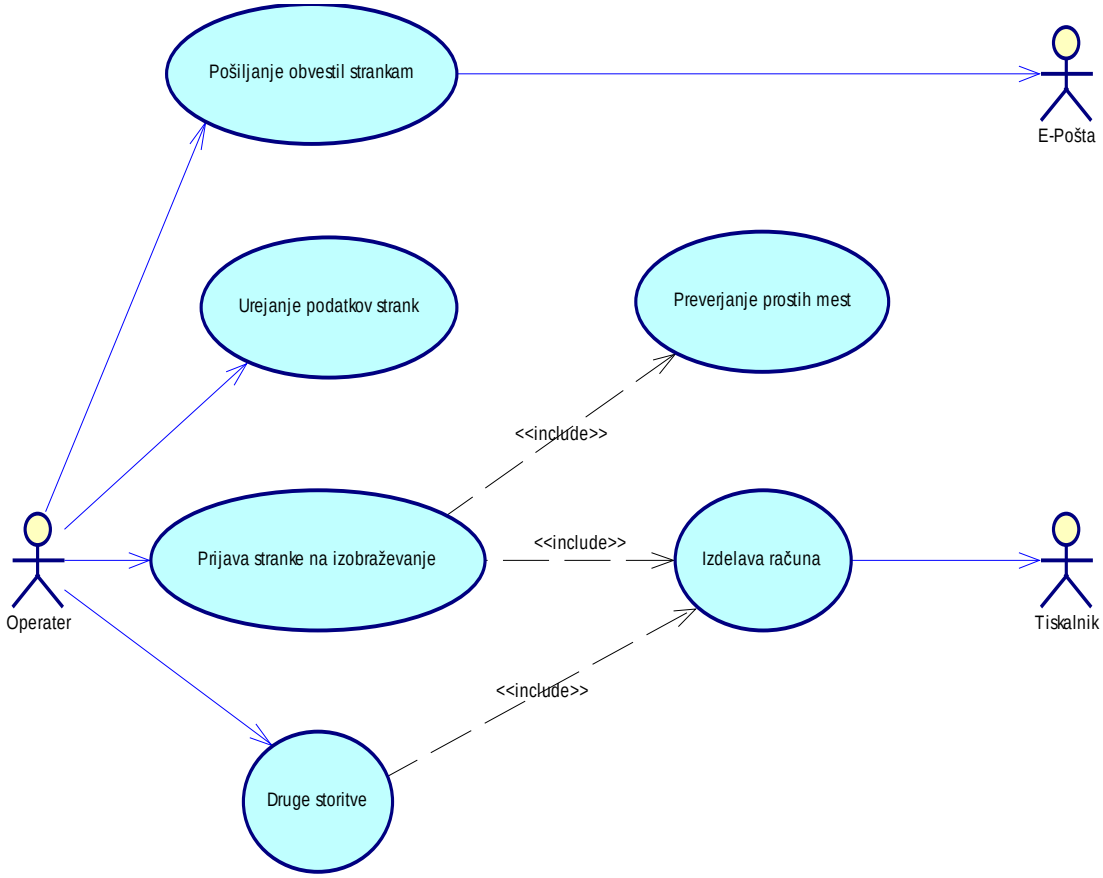
Za začetek je blagajniški modul planiran bolj preprosto. Glavna naloga bi bila izstavljanje in vodenje evidence računov za storitve, poleg tega pa tudi evidenca trenutnega stanja v blagajni. Kasneje bi bilo možno razviti bolj profesionalno računovodsko funkcionalnost, vendar ta za trenutne potrebe sistema ni smiselna.

2.8.3 Storitve



Slika 21: Prikaz funkcionalnosti modula storitev

Najprej bi bilo potrebno voditi bazo strank oz. uporabnikov storitev multimedijskega centra, pa tudi potencialnih novih strank. Sistem naj bi imel opcijo pošiljanja elektronskih sporočil, vsem strankam hkrati, kar bi bilo zelo koristno za promocijske namene, ali pa strankam, ki uporabljajo določeno storitev, na primer vsem udeležencem LINUX delavnice.



Slika 22: Diagram uporabe modula storitev

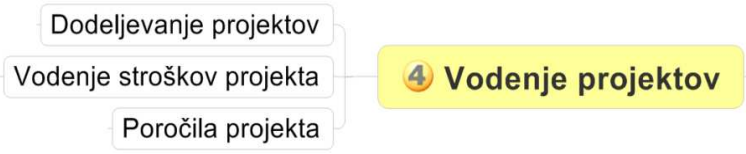
Pomembna storitev multimedijskega centra so izobraževalne vsebine, kot so delavnice, tečaji, predavanja itd. Ko bi se stranka prijavila na izobraževanje, bi jo operater vnesel v sistem. Če bi sistem ugotovil, da so še prosta mesta, bi stranko prijavil in ji preko blagajniškega modula izstavil račun. Podobno bi izstavil račun tudi za ostale storitve, kot so peka CD-jev, tiskanje, izposoja opreme itd.

2.8.4 Spletni vmesnik

Slika 23: Prikaz funkcionalnosti spletnega vmesnika

Spletni vmesnik sistema bi bila povsem samostojna spletna aplikacija, ki bi bila povezana na isto podatkovno bazo kot namizna aplikacija. Njen namen bi bil omogočanje komunikacijo med uporabnikom in sistemom preko interneta. Med pomembnejšimi funkcionalnostmi spletnega vmesnika bi bila možnost operaterja, da označi termine za delo na MMC-ju. Naj spomnimo, da so operaterji predvsem študentje, ki opravljajo to delo v svojem prostem času. Poleg tega bi implementirali tudi pregled in pošiljanje sporočil, spremljanje različnih komunikacijskih kanalov s pomočjo RSS tehnologije, povezava s sistemom varnostnih kamer itd.

2.8.5 Vodenje projektov



Slika 24: Prikaz funkcionalnosti modula za vodenje projektov

Želja naročnika je, da bi v našem sistemu operater oz. vodja nekega projekta imel omogočen čim boljši pregled nad delom in stroški projekta. Zato bi morali razviti modul, ki bi omogočal ustvarjanje projektov, določanje sodelavcev projekta in prikazovanje poteka projekta. Za računalniško podporo vodenja projektov danes obstaja kar nekaj programske opreme. Ti programi so relativno zapleteni sistemi. Zato bi v Pulsmaster poizkusili vključit kakšno primerno odprtokodno rešitev.

2.8.6 Urejanje spletnih portalov

Slika 25: Prikaz funkcionalnosti modula za urejanje portalov

Dandanes je glavna komunikacijska žila svetovni splet. Da bi pokrili čim večjo ciljno publiko, ima MMC Pulsar, skupaj z bratskimi sekcijami znotraj Zavoda O, na internetu veliko spletnih strani in profilov. Tu so običajne spletne strani, blogi, wiki strani, socialne mreže, forumi itd. Če želi danes operater objaviti neko novico na vseh primernih mestih, mu vzame veliko časa. Poleg tega mora imeti še znanje za urejanje vsebin v različnih sistemih in pa različna gesla za administracijske pravice.

Da bi rešili ta problem, bi razvili modul, ki bi uporabniku omogočil objavo prispevkov na različnih spletnih mestih le z nekaj kliki.

3 Zaključne ugotovitve

3.1 Problemi in izzivi projekta

Z multimedijskim centrom Pulsar tako ali drugače sodelujem že od srednješolskih let. Ideja o izdelavi oz. posodobitvi informacijskega sistema se nam je porodila že pred časom. Najtežje je bilo poiskati čas in voljo za delo. Ker se že več kot dve leti preživljam s programiranjem, prosti čas raje preživljam kako drugače. Za diplomsko nalogo sem si ravno zato izbral ta izziv, saj me je diploma dodatno motivirala. Vendar bolj ko sem se v projekt poglabljal, bolj je postal kompleksen in težaven.

Poskusil sem pridobiti tudi ekipo sodelavcev, a veliko učinka od tega ni bilo. Problemi pri skupinskem programiranju se zelo hitro pojavijo. Prva stvar, s katero pride do konflikta, so standardi. Če standardi niso dobro definirani, je izjemno težko programirati. Drug problem pa je hierarhija. Če je družba enakovredna, začne vsak delati po svoje. Zato je nujno, da obstaja vodja skupine, ki sicer posluša predloge članov skupine, a odloča po lastni presoji. Kljub vsemu pa mi je uspel izdelek, ki bo morda začetni korak k odličnemu informacijskemu sistemu.

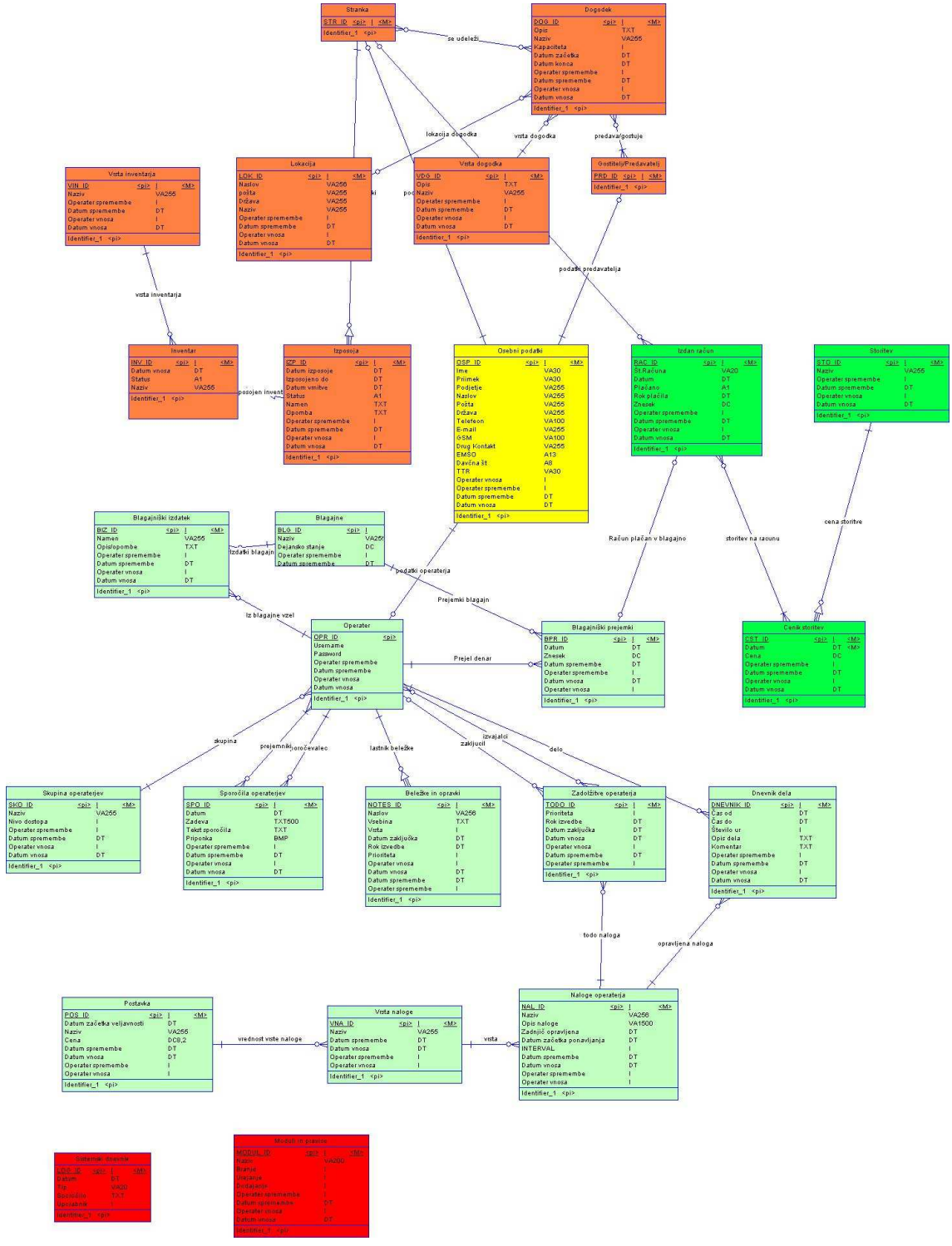
Kot izziv sem si ta projekt izbral tudi zato, da bi drugim, predvsem pa sebi, dokazal, da sem si med študijem na Fakulteti za računalništvo in informatiko pridobil vsa potrebna znanja za izvedbo takih projektov. In kot zaključno ugotovitev lahko rečem, da znanje je, izkušenj pa še malo manjka.

3.2 Nadaljnji razvoj

V mojem informacijskem sistemu vidim velik potencial. Upam, da mi bo uspelo ustvariti tim sodelavcev, s katerimi bomo Pulsmaster še razvijali. Z razvojem novih tehnologij, predvsem spletnih in mobilnih sistemov, so opcije za razvoj tako rekoč neskončne. Upam, da bo nekoč Pulsmaster razvit do te mere, da bo postal tudi tržno zanimiv. S tem mislim predvsem na to, da bom lahko za podobno organizacijo, kot je MMC Pulsar, z malo popravki obstoječega sistema izdelal svoj sistem in ga za primerno ceno tudi prodal. Moj dolgoročni cilj je, da bom nekega dne lastnik podjetja, ki bo razvijalo informacijske sisteme. S to diplomsko nalogo sem naredil prvi korak na poti do cilja.

Dodatek A

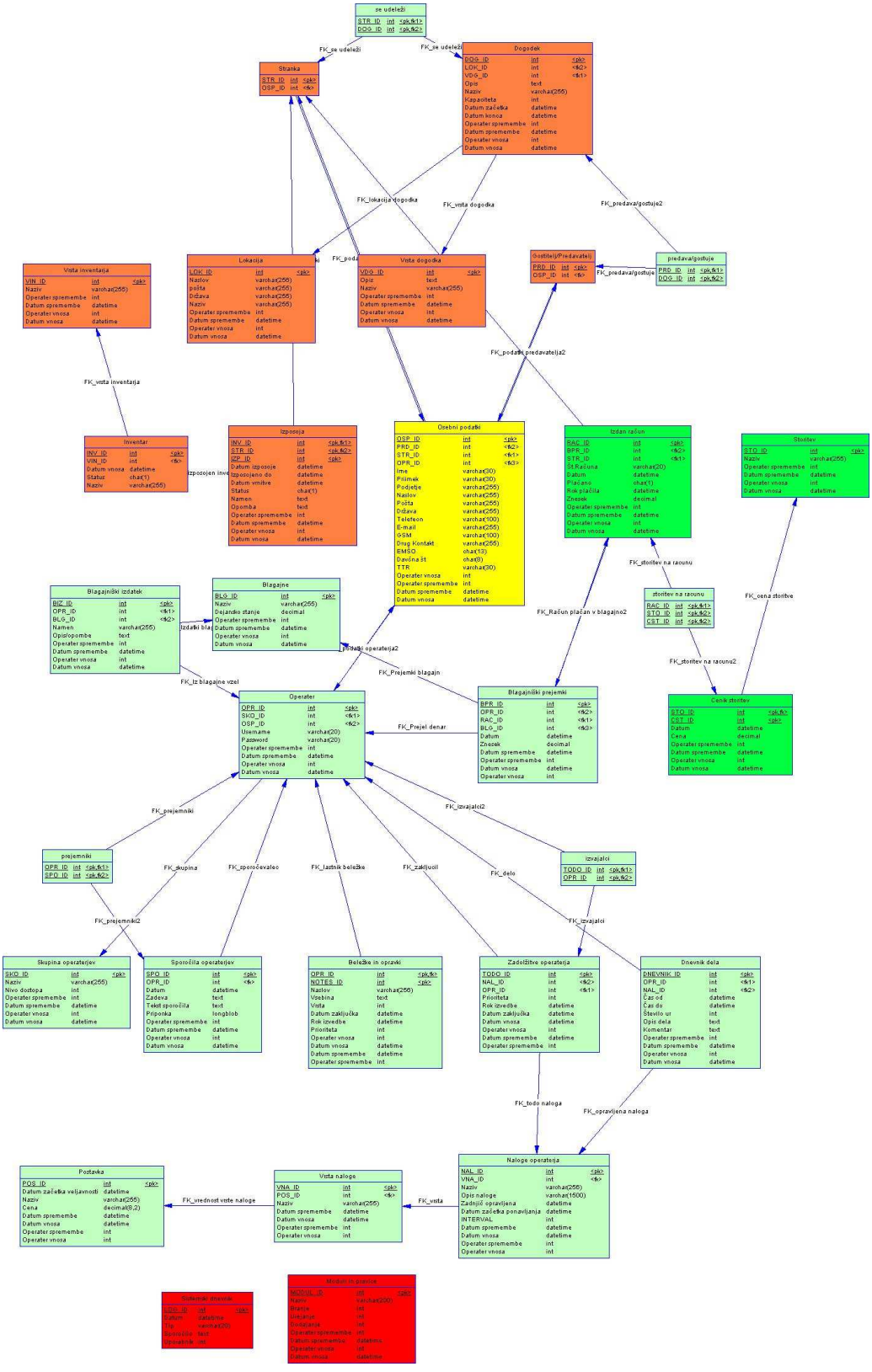
Konceptualni model podatkovne baze



Slika 26: Konceptualni model podatkovne baze

Dodatek B

Logični model podatkovne baze



Slika 27: Logični model podatkovne baze

Seznam slik

Literatura

[1] (2010) ADO.NET - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/ADO.NET

[2] D. Avison, G. Fitzgerald, *Information Systems Development, 4th Edition* ,
Berkshire: McGraw-Hill Education, 2006

[3] (2010) Cryptographic hash function - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Cryptographic_hash_function

[4] (2010) Enterprise resource planning - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Enterprise_resource_planning

[5] (2010) Information science - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Information_science

[6] (2010) Information system - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Information_system

[7] (2010) Iterative and incremental development - Wikipedia, the free
encyclopedia. Dostopno na:
http://en.wikipedia.org/wiki/Iterative_and_incremental_development

[8] (2010) Microsoft Visual Studio - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Microsoft_Visual_Studio

[9] (2010) MySQL - Wikipedija, prosta enciklopedija.
Dostopno na: http://sl.wikipedia.org/wiki/MySQL

[10] M. Silič, dr. M. Colnar, *Enotna metodologija razvoja informacijskih sistemov*, 4.
zvezek, Ljubljana: Center Vlade RS za informatiko, 2000

[11] (2010) SQL - Wikipedija, prosta enciklopedija.
Dostopno na: http://sl.wikipedia.org/wiki/SQL

[12] (2010) Windows Forms - Wikipedia, the free encyclopedia.
Dostopno na: http://en.wikipedia.org/wiki/Windows_Forms