Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aleksander Rebula

Zasnova celovite informacijske podpore delu strokovnega delavca za varnost pri delu

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: viš. pred. dr. Alenka Kavčič

Ljubljana 2013

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00372/2013
Datum: 01.03.2013

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

Kandidat: **ALEKSANDER REBULA**

Naslov: **ZASNOVA CELOVITE INFORMACIJSKE PODPORE DELU STROKOVNEGA DELAVCA ZA VARNOST PRI DELU**

DESIGN OF A COMPREHENSIVE INFORMATION SYSTEM IN SUPPORT OF THE OCCUPATIONAL SAFETY OFFICER

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

V diplomski nalogi izdelajte ogrodje informacijskega sistema, ki pokriva vse aspekte dela strokovnega delavca za varnost pri delu in omogoča celovito informacijsko podporo pri njegovem delu. Sistem naj bo sestavljen čimbolj modularno in naj omogoča enostavno razširljivost. Pri zasnovi upoštevajte dobre prakse in principe načrtovanja uporabniških vmesnikov, da bo sistem zasnovan intuitivno in bo deloval prijazno tudi za računalnika manj vešče uporabnike.
Prototipno implementirajte enega izmed modulov zasnovanega sistema, in sicer modul za usposabljanje, ki pokriva predvsem pripravo izpitnih pol. Sistem naj bo realiziran v obliki spletne aplikacije in naj bo dosegljiv tudi z mobilnimi napravami s terena.

Mentor:
Alenka Kavčič
viš. pred. dr. Alenka Kavčič

Dekan:
prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Aleksander Rebula, z vpisno številko **63080311**, sem avtor diplomskega dela z naslovom:

Zasnova celovite informacijske podpore delu strokovnega delavca za varnost pri delu

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom viš. pred. dr. Alenke Kavčič,
- so elektronska oblika diplomskega dela, Zasnova celovite informacijske podpore delu strokovnega delavca za varnost pri delu, povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblilke diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 11. septembra 2013 Podpis avtorja:

Hvala moji dragi Azri za vso podporo, ki sem je bil deležen skozi življenje in študij ter sinu Žigi, ki me vedno znova spomni, kako lepo je življenje.

Hvala mami Ani in očimu Bojanu, ki sta me vseskozi podpirala, tako moralno kot finančno in mi tako omogočila, da sem prišel do tega pomembnega mejnika v življenju. Očimu gre tu še posebna zahvala za svetovanje pri strokovnih zadevah iz področja varnosti pri delu in požarne varnosti, ki je bilo ključno za izvedbo te diplomske naloge. Hvala tudi sestri Sari za podporo in sodelovanje pri zbiranju idej in pregledovanju besedila.

Iskreno se zahvaljujem tudi tetama Nadji in Mariji ter pokojnima babici Emi in stricu Ludviku, ki so vseskozi in brezpogojno podpirali vsako mojo odločitev in mi stali ob strani, ko sem jih najbolj potreboval.

Zares hvala vsem zgoraj omenjenim, ta diploma je tudi vaša.

Za nesebično moralno in strokovno pomoč ter vse nasvete, bi se rad zahvalil tudi prijateljem, sošolcem in nekdanjim sodelavcem: Janiju Poljšaku, Binetu Gorjancu, Eriku Cerkveniku, Dougu Boudu, Urošu Jorasu, Marku Milostu in Mitji Maružinu.

Zahvaljujem se vsem asistentom in predavateljem, ki so študij omogočili. Predvsem bi, zaradi dodatne popestritve, med temi rad izpostavil viš. pred. dr. Igorja Rožanca, doc. dr. Petra Peera, uni. dipl. ing. Andreja Krevla, doc. dr. Boštjana Slivnika in mag. Igorja Škrabo.

Posebna zahvala gre viš. pred. dr. Alenki Kavčič za kakovostno in potrpežljivo mentorstvo.

To diplomsko delo posvečam dekletu Azri in sinu Žigi.

Kazalo

Povzetek

Abstract

1 Uvod **1**

2 Analiza problematike **5**

2.1 Opis dejavnosti varnosti pri delu 6

2.2 Splošne naloge strokovnega delavca 7

2.3 Težavnost dela strokovnega delavca 8

2.3.1 Informacijski sistem v podporo strokovnemu delavcu . 9

2.4 Glavne naloge in problemi strokovnega delavca 10

2.4.1 Natančnejša analiza nalog in problemov 11

2.5 Uporabljene programske rešitve 17

2.5.1 Problemi obstoječe programske programske opreme . . 18

2.5.2 Želene izboljšave trenutno uporabljene programske opreme 21

2.6 Določitev kriterijev za dobro programsko rešitev 23

2.6.1 Celovitost 24

2.6.2 Enostavna dostopnost (lokalno/globalno) 24

2.6.3 Enostavna možnost preizkusa v primeru postavitve v oblak 25

2.6.4 Odlična uporabniška izkušnja s kvalitetnim grafičnim uporabniškim vmesnikom 25

2.6.5 Učinkovit sistem obvestil in sporočil 28

2.6.6 Enostavna registracija novega uporabnika in prijava v sistem 29
2.6.7 Ustrezne varnostne zahteve 30
2.6.8 Večnivojska arhitektura programske rešitve 30
2.7 Analiza obstoječih programskih rešitev na trgu 31
2.7.1 Neizpolnjeni kriteriji 31

3 Rešitev problema **35**
3.1 Zajem zahtev naročnika 36
3.1.1 Zahteva 1: Namen informacijskega sistema 37
3.1.2 Zahteva 2: Enostavna obdelava podatkov in dostop do njih 37
3.1.3 Zahteva 3: Prijaznost sistema s krajšimi časi izvedbe . 38
3.1.4 Zahteva 4: Onemogočanje podvajanja podatkov 38
3.1.5 Zahteva 5: Enostaven vpogled v podatke 38
3.2 Ureditev zahtev po področjih - modulih 38
3.2.1 Glavni moduli 40
3.2.2 Podporni moduli 50
3.3 Načrtovanje arhitekture sistema 52
3.3.1 Podatkovni nivo 54
3.3.2 Poslovni nivo 60
3.3.3 Predstavitveni nivo 60
3.3.4 Skupni nivo 61

4 Informacijski sistem Helmet IS **63**
4.1 Varnost in prijava v sistem 64
4.2 Trenutna lokacija sistema in izbira lokacije spletnega strežnika na dolgi rok 65
4.2.1 Lokalni strežnik podjetja s povezavo VPN 65
4.2.2 Več namestitev sistema na različnih fizičnih lokacijah . 66
4.2.3 Običajen spletni strežnik z zaščito SSL in certifikati . . 67
4.2.4 Informacijski sistem kot servis v oblaku 67

4.3 Ogrodje ali zbirka modulov 70
4.3.1 Glavna razporeditev HTML elementov grafičnega uporabniškega vmesnika 71
4.4 Modul Usposabljanje 73
4.4.1 Testne pole 74
4.4.2 Vprašanja 77
4.4.3 Kategorije 78
4.5 Ostali moduli 80
4.5.1 Uporabniški račun 80
4.5.2 Media 81
4.6 Prednosti novega informacijskega sistema 81
4.6.1 Moderen grafični uporabniški vmesnik za boljšo uporabniško izkušnjo 81
4.6.2 Večjezičnost 82
4.6.3 Socialni dejavniki 82
4.6.4 Enostavna razširljivost in integracija z drugimi sistemi 83
4.6.5 Eno samo orodje za vse administrativne naloge in centraliziranost virov 83
4.6.6 Ni podvajanja virov 84
4.6.7 Varnost podatkov 84
4.6.8 Dostop 84
4.6.9 Boljša medsebojna obveščenost uporabnikov o pomembnih dogodkih 84
4.7 Uporabljena programska oprema in ostala tehnologija 85

5 Sklepne ugotovitve **87**

Slike **88**

Literatura **91**

Seznam uporabljenih kratic in simbolov

CSS	Cascading Style Sheets
EF	Entity Framework
ERP	Enterprise resource planning
HIS	Helmet Information System
HTML	HyperText Markup Language
JS	JavaScript
LDAP	Lightweight Directory Access Protocol
MS SQL	Microsoft Structured Query Language
MVC	Model View Controller
MVVM	Model-View-ViewModel
ORM	Object-relational mapping
PV	Požarna varnost
SaaS	Software as a Service
SEO	Search Engine Optimization
SSL	Secure Socket Layer
VPD	Varnost pri delu
VPN	Virtual private network
WCF	Windows Communication Foundation
XML	Extensible Markup Language

Povzetek

Cilj tega diplomskega dela je opisati in izdelati informacijski sistem, katerega glavni namen je narediti korenite spremembe v delu strokovnih delavcev za varnost pri delu in izboljšati njihovo učinkovitost. Delo se osredotoči na razvoj informacijskega sistema v obliki spletne aplikacije. Za razvoj sistema je uporabljena metodologija strukturnega razvoja. Ker pa se zaradi časovnih in drugih omejitev ne moremo v celoti držati pravil te metodologije, lahko v samem potopku srečamo tudi elemente agilnih metodologij. Pri posameznih delih sistema se namreč še vedno ne ve, kakšen bo končni cilj in se nekateri problemi rešujejo sproti. Preko natančne analize problema poslovne domene, torej delovnega okolja strokovnega delavca, njegovih nalog in trenutno uporabljenih ter razpoložljivih programskih orodij, v drugem poglavju, določimo kriterije za kvalitetno rešitev, ki bi jo strokovni delavec potreboval pri svojem delu. V tretjem poglavju se na podlagi omenjene analize posvetimo reševanju problema. Po zajemu in ureditvi zahtev se lotimo načrtovanja arhitekture sistema, na podlagi katerega nastane informacijski sistem, ki ga poimenujemo "Helmet Information System"ali HIS. Ko je izdelan zadosten del sistema, da pokrije zahteve diplomskega dela, ga bralcu v četrtem poglavju predstavimo in opišemo njegovo varnost, v katero okolje bo sistem postavljen in na kakšen način bodo uporabniki deležni njegovih storitev. V istem poglavju predstavimo dele sistema - module, od katerih pa podrobno obravnavamo le modul "Usposabljanje", ki služi kot prototip, preko katerega posredno spoznamo, kakšno vlogo bo imel HIS v prihodnosti in kateri moduli bodo sistem še gradili in se med seboj povezovali.

Ključne besede

arhitektura, HIS, informacijski sistem, podpora delu strokovnega delavca, moduli, svetovni splet, varnost pri delu.

Abstract

The goal of this thesis is to describe and develop an Information System, the main purpose of which is to improve the efficiency of Occupational Safety Officers by fundamentally altering the way in which they perform their duties. This paper focuses on the development of the subject Information System in the form of a Web Application. This application's architecture applies a hybrid development approach between the standard Structured Development and Agile methodologies, and additionally will incorporate custom solutions that evolve throughout the development lifecycle. This application's business requirements, rules, logic, and User Interface design are derived from a thorough study and analysis of the Occupational Safety Officer's domain. Regarding the previously mentioned analysis, we address this challenge in the third chapter. After gathering and organizing the user's requirements, we begin planning the Information System that we call "Helmet Information System" (HIS). When a sufficient part of the system that covers the needs of this thesis is completed, we present it in chapter four. In that chapter describe its safety measures, the environment that it is being placed into, and the means by which the system will offer its services to the user. In that same chapter we also present the parts of the system that we call Modules. Though the system will be composed of many Modules, this thesis properly addresses only the "Training" Module, which serves as a prototype that indirectly explains other Modules that will be added in the future. In this step-by-step process we get a better idea of how HIS will look like and what role will have in the future.

Ključne besede

Architecture, HIS, Information System, Support for the work of the Occupational Safety Officer, Modules, Internet, Occupational Safety.

Poglavje 1

Uvod

Začenši s prvo polovico 90-ih let prejšnjega stoletja smo bili priča izrazitemu vzponu informacijskih, natančneje spletnih tehnologij, ki so danes pomemben del našega zasebnega, kot tudi poslovnega življenja[4]. Nastajali so novi izzivi in panoge, ki so postale popolnoma odvisne od teh tehnologij oziroma so na njih celo temeljile; poglejmo samo primere neštetih podjetij tako v Sloveniji kot tudi v tujini, katerih posel temelji na storitvah, kot so izdelava spletnih strani, optimizacija strani za spletne brskalnike (SEO), omogočanje spletnega gostovanja, spletno oglaševanje in mnoge druge. Druge, že obstoječe, starejše panoge, med katerimi je tudi varnost pri delu, pa so iz različnih razlogov relativno pozno začele izkoriščati prednosti, ki jih prinaša razvoj tehnologij svetovnega spleta.

Pričujoče diplomsko delo se osredotoča na področje varnosti pri delu in požarne varnosti, ki bo naša poslovna domena. Strokovnemu delavcu na tem področju bomo z novim informacijskim sistemom skušali popolnoma spremeniti delovne navade. Njegovo administrativno delo, ki zajema več kot polovico njegovega delovnega časa, bomo skušali bistveno skrajšati. To bomo storili z avtomatizacijo mnogih administrativnih nalog, ki so do sedaj zmanjševale produktivnost.

Glavno orodje za dosego tega cilja bo svetovni splet. Želimo si namreč oziroma vsaj eden izmed namenov tega diplomskega dela je, da bi spletni

brskalnik postal pravzaprav edino orodje strokovnega delavca pri izvajanju administrativnih nalog, ki so se do sedaj izvajale s celim naborom različnih programskih orodij. To število orodij želimo zmanjšati, jih združiti v eno celoto. Z drugimi besedami, cilj našega raziskovanja je vzpostavitev enotnega sistema, ki bo uporabniku omogočil, da bo za administrativne naloge porabil manj časa in se tako bolj posvetil, za posel relevantnim nalogam, kot je naprimer delo na terenu.

Če opazujemo samo podjetja v naši ciljni panogi in njihovo dosedanje izkoriščanje svetovnega spleta za svoje poslovanje, lahko ugotovimo, da bi lahko nekatera bolje izrabljala možnosti, ki jih le-ta prinaša. Do njegovega vzpona, okrog leta 1994, so podjetja še lahko shajala brez spletnih rešitev, kot so spletne strani, v današnjem poslovnem svetu pa temu ni več tako. Pri mnogih podjetjih, ki so bila dejavna že pred tem obdobjem in so do takrat sicer dobro poslovala, veliko časa sploh niso videli potrebe po usmerjanju svojih naložb v spletne storitve in niti v preproste spletne strani. Danes pa si ne moremo več predstavljati, da bi podjetje ne imelo najmanj slednje. Za nekatera manjša podjetja je bilo še okrog leta 2000 to nekaj čisto navadnega, toda časi so se spremenili.

Seveda gre taka podjetja razumeti, saj so informatizacijo svojega dela razumevala zgolj skozi nakup in uporabo različnih programskih rešitev, s katerimi so si pomagali pri vsakdanjem delu, čemur pa spletne strani takrat niso bile namenjene, vsaj ne v večini. Spletna stran je igrala pač samo vlogo "vizitke" in nič drugega. Podjetja so takrat z vidika informatizacije poznala in uporablja predvsem namizne aplikacije, kot so računovodski programi, programi za evidence zaposlenih, Microsoft Office pisarniška orodja itd., takšni produkti pa so predstavljali velik strošek. Le malokdo si je takrat predstavljal, da se bodo v kakem desetletju tudi takšne storitve preselile v splet (v oblak) oziroma, da bo spletni brskalnik postal glavno orodje pri delu in ne več samo kot pripomoček za iskanje informacij.

Svetovni splet se je, zahvaljujoč izjemnemu napredku tehnologij tako na strani strežnika kot odjemalca, od nastanka pa do danes tako zelo razvil,

da se ne uporablja več samo za spletne predstavitve, portale in podobne storitve, temveč za prave aplikacije, ki lahko služijo neštetim namenom. Uporabniški vmesniki so postopoma postajali čedalje bolj sofisticirani, privlačnejši in zabavnejši za uporabo. Tu se lahko zahvalimo predvsem izrazitemu razvoju spletnih brskalnikov in tehnologij na strani odjemalca, kot so HTML, CSS, JavaScript (ECMAScript). Tako lahko danes v spletnem brskalniku igramo kompleksne računalniške igrice, urejamo družinske foto albume ali spremljamo statistične podatke o obiskanosti naših spletnih strani (Google Analytics ipd.).

Tako se sedaj direktor nekega naključnega podjetja (na primer v panogi varnosti pri delu in požarne varnosti) lahko upravičeno sprašuje, kako naprej: ali ostati pri trenutnem stanju in zastarelih informacijskih pripomočkih, ki so že na voljo (in ki sicer še vedno relativno dobro služijo svojemu namenu ter na katere so se zaposleni že privadili), ali pa preiti na nek popolnoma nov, spletni informacijski sistem.

Izdelava in nakup nove, sodobne rešitve, zagotovo ni zastonj, a s seboj nosi veliko prednosti: sodobnejši uporabniški vmesnik, vsi viri na enem mestu in vedno na dosegu roke, brez nevarnosti podvajanja, boljše, hitrejše in enostavnejše sodelovanje med zaposlenimi ...

V tem diplomskem delu bomo najprej na kratko predstavili področje varnosti pri delu in požarne varnosti, kjer bomo definirali delo in potrebe strokovnega delavca za varnost pri delu, v nadaljevanju "strokovnega delavca"ter ugotovili zahteve, na podlagi katerih bomo izpeljali postopek informatizacije poslovnih procesov z uporabo izključno internetnih tehnologij. Pomagali si bomo z določitvijo kriterijev za dobro programsko rešitev, katere končno ogrodje, sestavljeno iz modulov, bo tudi predstavljeno v tej diplomski nalogi. Po izdelavi ogrodja bomo podali smernice za nadalnji razvoj.

Dosego cilja povečanja produktivnosti strokovnega delavca bomo skušali doseči z informacijskim sistemom, v obliki spletne aplikacije, katere uspeh bo zelo odvisen od tega, v kolikšni meri ga bomo uspeli približati strokovnemu delavcu oziroma, koliko mu bo pisan na kožo. Informacijski sistem, imenovan

"Helmet IS"(v nadaljevanju HIS) bo uporabljal zadnje spletne tehnologije in bo s svojim intuitivnim ter prilagodljivim grafičnim uporabniškim vmesnikom na enoten način strokovnim delavcem omogočil hitro in učinkovito delo. Omogočal bo tudi lažje sodelovanje med zaposlenimi in popolnoma izničil nepotrebno podvajanje določenih skupnih virov.

Z izdelavo tega sistema pričakujemo dolgoročno povečanje hitrosti in učinkovitosti posameznega strokovnega delavca, in s tem bistveno zmanjšanje stroškov dela in boljše poslovne rezultate celotnega podjetja.

Poglavje 2

Analiza problematike

Na podlagi dolgoletnega, posrednega stika s stroko varnosti pri delu in požarne varnosti, predvsem pa po posvetu s strokovnjaki s tega področja, je postalo samoumevno, da njihov način dela potrebuje sodobnejši pristop. Potrebujejo boljša in učinkovitejša administrativna orodja, ki bi izboljšala celoten poslovni proces.

Narava dela je namreč taka, da se morajo strokovni delavci veliko posvečati učenju uporabe delovnih orodij in zbiranju virov ter njihovemu usklajevanju, namesto da bi večji del pozornosti posvečali konkretnim delovnim nalogam na terenu, ki izboljšujejo kvaliteto storitev in podjetju nenazadnje prinašajo dobiček. Delo je poleg tega pogosto enolično in motivacija za delo je v upadu, s tem pa tudi rezultati strokovnih delavcev in podjetja.

Po podrobni in dolgotrajni analizi delovnih nalog smo ugotovili, kateri so tisti problemi oziroma pomankljivosti, ki najbolj zmanjšujejo učinkovitost posameznega strokovnega delavca in se na njih najbolj osredotočili pri nadaljnem strukturnem razvoju programske rešitve.

Preden bralcu predstavimo naloge in probleme strokovnega delavca, ki bodo temelj naše informacijske rešitve, je najprej potrebno bolje spoznati dejavnost oziroma našo poslovno domeno.

2.1 Opis dejavnosti varnosti pri delu

Vsaka aktivnost človeka po svoji materialni naravi prinaša s seboj določene nevarnosti in tveganja za nastanek nezgod in/ali okvar zdravja. Aktivnosti se med seboj razlikujejo samo po različnih vrstah nevarnosti, ki so prisotne in po stopnji tveganja za nastanek nezgode ali okvare zdravja. Področje varnosti in zdravja pri delu in požarne varnosti je urejeno z vrsto mednarodnih konvencij, direktiv in smernic. Slovenija kot članica EU to področje ureja v okviru evropske zakonodaje. V ta namen je Slovenija sprejela in uveljavila vrsto zakonov in podzakonskih aktov. Izvajanje nalog na področju varnosti in zdravja pri delu in požarne varnosti ni izbira ampak obveznost vsakega delodajalca.

Vsak delodajalec je dolžan svojo dejavnost organizirati tako, da pri delu zaposlenim zagotavlja varno in zdravo delo ves čas opravljanja svoje dejavnosti. Delodajalec je dolžan naloge na področju varnosti pri delu poveriti strokovnemu delavcu, naloge zdravja pa pooblaščenemu zdravniku.

Naloge na področju varnosti pri delu in požarne varnosti opravlja strokovni delavec, ki mora imeti določeno izobrazbo, izkušnje in opravljene določene strokovne izpite, s katerimi dokazuje svojo strokovnost.

Strokovni delavec je lahko zaposlen pri delodajalcu ali pa delodajalec v ta namen najame zunanje strokovnjake, ki imajo ustrezna znanja in strokovnost.

V našem primeru obravnavamo strokovnega delavca, ki je zaposlen v podjetju, ki svojim strankam ponuja strokovne naloge s tega področja.

Glavna značilnost dejavnosti podjetja za varnost pri delu je izvajanje in zagotavljanje nalog na področju varnosti pri delu in požarne varnosti tistim, ki v svoji strukturi zaposlenih nimajo ali ne morejo zagotoviti strokovnjakov s tega področja in zato teh obveznosti ne zmorejo zagotoviti sami.[1]

2.2 Splošne naloge strokovnega delavca

Področje dela je zelo kompleksno in zahteva poznavanje širokega spektra strokovnih področij, od izobraževanja, poznavanja tehnike na različnih področjih, kot so industrija, gradbeništvo, gostinstvo, promet, zdravstvo, izobraževanje, itd. ter številne obrtne in storitvene dejavnosti. V splošnem, strokovni delavec opravlja naslednje naloge:

- usposablja delavce in preizkuša njihovo usposobljenost,
- izvaja preglede delovnega okolja (objekte, delovno opremo, materiale in snovi), ugotavlja pomanjkljivosti ter predlaga ukrepe za odpravo le-teh,
- spremlja delovni proces, analizira pogoje dela, škodljivosti, ki se pojavljajo na delovnih mestih, le-te ocenjuje in določi ukrepe za odpravo nevarnosti in obvladovanje tveganja za nastanek nezgode ali okvare zdravja,
- izvaja meritve z namenom ugotavljanja kvalitete delovnega okolja in prisotnosti morebitnih škodljivosti za delavca,
- sodeluje s službami delodajalca in zunanjimi institucijami,
- izdeluje interne akte, kot so strokovne podlage za izjavo o varnosti, program usposabljanja, požarni red s pripadajočimi izvlečki, načrti, shemami, pravila o preprečevanju uživanja alkohola, drog in drugih prepovedanih substanc, o preprečevanju mobinga na delovnem mestu,
- za delodajalca vodi evidence o izvajanju nalog, ter rokih, v katerih je potrebno naloge obnoviti,
- raziskuje nastale nezgode, ugotavlja vzroke za nastanek nezgod in predlaga ukrepe za odpravo pomanjkljivosti,
- svetuje delodajalcu pri načrtovanju, izbiri, nakupu in vzdrževanju sredstev za delo,

- svetuje delodajalcu glede opreme delovnih mest in glede delovnega okolja,
- usklajuje ukrepe za preprečevanje psihosocialnih tveganj,
- opravlja obdobne preglede in preizkuse delovne opreme,
- opravlja notranji nadzor nad izvajanjem ukrepov za varno delo,
- izdeluje navodila za varno in zdravo delo,
- pripravlja in izvaja usposabljanje delavcev za varno delo,
- sodeluje z izvajalcem medicine dela.

2.3 Težavnost dela strokovnega delavca

Čeprav mora strokovni delavec poznati pravila in varnostna sredstva, ki pridejo v poštev pri mnogih drugih različnih delovnih mestih, se mora zelo dobro zavedati tudi pogojev na svojem lastnem delovnem mestu. Pri svojem delu mora obvladovati mnogo različnih delovnih orodij, med katere spadajo številne programske aplikacije, drage merilne naprave itd..

Strokovni delavec mora veliko svojega časa preživeti tako na terenu, kot v pisarni, kjer ga čaka kup dokumentov, ki jih mora pregledovati, izpolnjevati, znova sestavljati in iz katerih se mora vedno znova učiti. Poznati mora namreč mnoga pravna pravila, ki pa se pogosto spreminjajo. Prisotno je veliko usklajevanja s sodelavci in potrebno je skrbeti za skupno in enotno "odlagališče" različnih dokumentov, ki nastajajo v procesu dela in so velikokrat končni produkt.

Iz tega lahko zaključimo, da ne samo, da takšno delo prinaša veliko odgovornosti, ampak mora strokovni delavec vsak dan obvladovati premnoga orodja in vire. Sklepamo lahko torej, da je zaposleni v takem podjetju neprestano podvržen stresu. Tudi nek zunanji opazovalec, ki pride v naključno

podjetje za varnost pri delu, kmalu ugotovi, da so nekatere naloge strokovnega delavca enolične, kar je nenazadnje tudi mnenje nekaterih strokovnih delavcev v tej stroki.

Uvedba računalniške tehnologije in možnosti, ki jih omogoča, je povzročila, da se je struktura dela strokovnega delavca korenito spremenila. Pričakovali bi, da bo delo postalo enostavnejše in v začetku je tako tudi bilo, a je bil s porastom števila računalniških orodij, strokovni delavec vse bolj primoran sam opravljati vsa administrativna opravila, ki prej niso bila v njegovi domeni. Namesto, da bi tehnologija delo olajšala, se je zaradi naraščajočega nabora različnih programskih orodij, težavnost še dodatno povečala.

Dodatno breme za strokovnega delavca pa so v zadnjih trinajstih letih prinesle spremembe zakona o varnosti in zdravju pri delu. Prinesle so ogromno število dodatnih aktivnosti, ki so vse povezane z delom na računalniku, ta trend pa se samo še povečuje. Lahko bi rekli, da obseg in količina dela na računalniku že nekoliko omejuje normalen razvoj strokovnega delavca in obenem poslovni proces, namesto, da bi ta tehnologija delo olajšala. Zaradi tega čas izvajanja administrativnih nalog narašča, vse manj pa je časa za izobraževanje, pridobivanje znanja in nenazadnje za izvajanje relevantejših aktivnosti. Zato je ključnega pomena, da se lotimo izdelave aplikacij, ki morajo biti čim bolj enostavne in učinkovite. Po pričanju strokovnih delavcev na tem področju, take aplikacije še niso ustvarili.

Vprašamo se lahko torej, ali res ne obstaja kakšen način, s katerim bi tehnologija dejansko olajšala delo, ki bi tako postalo preprostejše in s tem seveda učinkovitejše. Kako torej rešiti zgoraj omenjene težave? Kako zaposlenemu v takšnem podjetju olajšati njegovo delo?

2.3.1 Informacijski sistem v podporo strokovnemu delavcu

Zagotovo lahko informatiki, s pravo idejo, veliko pripomoremo k zmanjševanju naštetih težavnosti. Kot že nakazano, dejstvo je, da je dandanes računalnik najpomembnejše delovno orodje pri skoraj vsakem poklicu in podjetja za var-

nost pri delu in požarno varnost niso pri tem nikakršna izjema. Drži, da se v tej stroki pomemben del nalog izvaja na terenu, kjer je v tej fazi težko najti razloge za uporabo spletne podporne aplikacije, kot je informacijski sistem, ki bo predmet tega diplomskega dela. Vseeno pa se več kot polovica nalog izvede in zaključi v pisarni, pred računalnikom. Prav ta administrativni del je tisti, katerih probleme bo skušal reševati naš informacijski sistem. Ta bo naredil delo zaposlenih preprostejše, hitrejše in predvsem učinkovitejše.

Preden pa se lotimo izdelave nove programske rešitve, je nujno potrebno, da postavimo trdnejše temelje in natančneje analiziramo zadolžitve in naloge strokovnega delavca ter natančno opredelimo oziroma formaliziramo probleme, ki nam bodo služili kot osnova pri zajemu zahtev in nenazadnje gradnji tega informacijskega sistema.

2.4 Glavne naloge in problemi strokovnega delavca

Počasi a vztrajno se pomikamo k temelju tega diplomskega dela in hkrati temelju končnega produkta. Ta temelj so problemi, ki jih je potrebno rešiti, da bi izboljšali poslovne procese ciljnega podjetja. Preden pa se jih lotimo, moramo opredeliti glavne naloge, da ugotovimo, kje tičijo pravi problemi, ki jih bo reševal nov informacijski sistem, in le tako lahko zmanjšamo prepad med razumevanjem različnih udeležencev, ki pri razvoju sistema sodelujejo.

V ciljnem podjetju za varnost pri delu in požarno varnost je vsak strokovni delavec v splošnem zadolžen za neko število strank iz množice vseh strank podjetja. Lahko rečemo tudi, da je zadolžen za vodenje evidence strank, v okviru tega pa delovno opremo in delavce, ki so v njegovi domeni. Naloge strokovnega delavca smo sicer že našteli v poglavju 2.2, a jih bomo tu zožili in razvrstili v 5 glavnih skupin, ki bodo podlaga za nadalnjo analizo.

Če torej povzamemo vse naloge, naštete v poglavju 2.2, nastanejo naslednje glavne skupine nalog:

1. usposabljanje delavcev za varno delo,
2. pregledovanje in preizkušanje delovne opreme,
3. priprava napotnic za zdravstveni pregled,
4. izdelava strokovnih podlag za oceno tveganja,
5. vodenje evidenc zaposlenih in delovne opreme pri strankah.

2.4.1 Natančnejša analiza nalog in problemov

Vse glavne naloge, ki smo jih pravkar našteli, se izvede na podlagi:

- vsakokratnega naročila stranke ali
- periode, katero zadolženemu delavcu sporoči planer dogodkov.

Vsakokratno naročilo stranke se vnese v knjigo naročil, podatke o periodi zapadlih nalog pa dobimo iz planerja dogodkov. Planer dogodkov se tiska mesečno in se ga uporabi za mesečni razpored nalog posameznim strokovnim delavcem. Knjigo naročil se tiska tedensko in je ravno tako podlaga za razdelitve nalog posameznim strokovnim delavcem. Tako planer dogodkov kot knjigo naročil je možno uporabiti kadarkoli, kot pomoč pri planiranju delovnih nalog.

V nadaljevanju sledi seznam nalog strokovnega delavca. Pri vsaki nalogi so naštete aktivnosti, z zvezdico pa so izpostavljene problematične, ki so pravzaprav ključne za določitev programskih modulov informacijskega sistema.

Aktivnosti vsake naloge, ki so opisane v nadaljevanju, si kronološko sledijo po označenem vrstnem redu. Tiste, ki niso označene, nov informacijski sistem zaenkrat ne bo obravnaval, ker to zaradi narave aktivnosti, za sedaj še ni mogoče.

Pri vsaki nalogi so še dodatno opisani izpostavljeni problemi.

Naloga 1: Usposabljanje delavcev za varno delo

Aktivnosti:

1. priprava testnih pol *,
2. izvedba usposabljanja,
3. ocenjevanje testnih pol,
4. izdaja potrdil v urejevalniku besedil MS Word *,
5. arhiviranje zapisnikov usposabljanja (fotokopija potrdila v register) *.

Problemi:

- neažurnost vsebine (npr. vsebina lahko ni usklajena s spremembami zakonskih predpisov...),
- podvajanje istih testnih pol v različnih mapah strokovnih delavcev v mreži,
- neusklajenost testnih pol med posameznimi strokovnimi delavci (oblika, vsebina),
- stihijski pristop k spremembam vsebine testnih pol (popravlja strokovni delavec vsak zase, kar bi lahko poenotili s centralizacijo),
- decentraliziranost vira podatkov (se veže tudi na prejšnjo alinejo),
- Wordove datoteke so neprimerne oziroma zastarel pristop, ker zahtevajo ogromno discipline in truda, da se podatki (dokumenti) pravilno administrirajo (poimenujejo, klasificirajo, uredijo ...),
- dobro definirane administrativne konvencije podjetja glede poimenovanja, klasificiranja, oblike itd. testnih pol, vendar se jih strokovni delavci ne držijo povsem, prevelika odvisnost od natančnosti strokovnega delavca,

- ročna izdelava potrdil (podobna težava kot pri testnih polah),
- ročno arhiviranje zapisnikov in potrdil (slaba preglednost, ki zahteva ogromno časa pri iskanju).

Naloga 2: Pregledovanje in preizkušanje delovne opreme

Aktivnosti:

1. priprava za delo na terenu (priprava merilne opreme, tiskanje zapisnika v primeru periodičnega pregleda),
2. pregled in preizkus na terenu pri stranki,
3. izdelava zapisnika (v aplikaciji za izdelavo zapisnikov o pregledu delovne opreme) *.

Problemi:

- nezmožnost združevanja posamezne delovne opreme iz dveh ali več zapisnikov v enega,
- programske omejitve glede kreiranja zapisnikov (nezmožnost prikaza nekaterih znakov, formul itd.),
- programska togost, ki se izraža v onemogočenem prilagajanju izgleda zapisnika s strani uporabnika.

Naloga 3: Priprava napotnic za zdravstveni pregled

Aktivnosti:

1. izdelava napotnice *.

Problemi:

- nezmožnost povezave s podatki iz ocene tveganja,

- ročno prenašanje podatkov v obrazec (stranka, delavec, obremenitve, zdravstveni pogoji, ki jih mora delavec izpolnjevati),
- pomanjkljiv vnos podatkov (nekaterih podatkov se pogosto ne vnaša),
- prevelika poraba časa.

Naloga 4: Izdelava strokovnih podlag za oceno tveganja

Aktivnosti:

1. analiza stanja na terenu in proučitev obstoječe dokumentacije,
2. ročni vnos podatkov v obrazec,
3. izdelava strokovnih podlag za oceno tveganja *.

Problemi:

- ogromna količina podatkov, ki jih je potrebno urediti in upoštevati,
- nezmožnost avtomatskega prenosa podatkov iz sorodnih ocen tveganj s podobno dejavnostjo,
- zamudno prenašanje podatkov iz drugih datotek (na način kopiraj/prilepi),
- zamudno iskanje sorodnih vsebin v drugih datotekah (čeprav je omogočeno tudi centralno hranjenje datotek, pa to zahteva tudi skrbnika sistema, ki ga podjetje pogosto nima),
- zamudno in drago skrbništvo sistema,
- stihijski pristop k spremembam vsebine ocene tveganja - posodobitve,
- neusklajenost pristopov reševanja posameznega tveganja med strokovnimi delavci.

Naloga 5: vodenje evidenc zaposlenih in delovne opreme pri strankah

Vodenje evidenc zaposlenih

Za zaposlene delavce pri stranki vodimo dve evidenci, in sicer za usposabljanje za varno delo in evidenco zdravstvenih pregledov. Obe evidenci se vodi ločeno z isto bazo podatkov delavcev.

Aktivnosti:

1. vnos podatkov o novo zaposlenem delavcu in izbris ob odhodu *,
2. s pomočjo filtra izpis delavcev, katerim bo v kratkem pretekla veljavnost usposabljanja ali zdravstvenega pregleda *,
3. kopiranje v Wordovo datoteko z dodatkom teksta - obvestilo stranki,
4. pošiljanje Wordovega dokumenta preko el. pošte stranki v vednost.

Problemi:

- zamudno vnašanje podatkov,
- zamudna obdelava podatkov,
- nezdružljivost oziroma nezmožnost prenosa podatkov v druge aplikacije na drug način kot s kopiranjem,
- vsako tabelo, ki je malo drugačna, je potrebno znova kreirati ali obstoječe spreminjati,
- nepreglednost pri pregledovanju arhivskih podatkov, ker je potrebno vsaki dve leti kopirati bazo podatkov zaradi preglednosti. Tako se v 10 letih nabere za vsako stranko 5 podobnih datotek. Problem nepreglednosti se precej poveča, kadar ima stranka 50 ali več zaposlenih in povečano fluktuacijo zaposlenih.

Vodenje evidenc delovne opreme pri strankah

Za vsako stranko, ki ima delovno opremo, se vodi evidenca pregledov in preizkusov.

Evidenca se vodi v Excelovi datoteki. Evidenca je obsežna, saj vsebuje številne podatke, in sicer o:

- nazivu stroja,
- proizvajalcu,
- tipu stroja,
- tovarniški številki,
- letu proizvodnje,
- inventarni številki,
- datumu pregleda,
- izdaji potrdila o brezhibnosti,
- datumu zapadlosti veljavnosti potrdila.

Aktivnosti:

1. iz zapisnika o pregledu strojev se podatke vnese v Excelovo tabelo,
2. tabelo se ažurira ob vsaki spremembi (nov stroj ali odprodan) *,
3. s filtrom se izpisuje seznam strojev, katerim je že ali bo v kratkem, pretekla veljavnost potrdila *,
4. izpis evidence za stranko in nadzorne organe - inšpekcijo.

Problemi:

- podvojeno vnašanje podatkov - prvič v zapisnik, nato še posebej v tabelo,
- nezdružljivost baze podatkov v evidenci s podatki v zapisniku,
- nepregledni arhivski podatki - množica tabel,
- zamudna obdelava podatkov,
- stihijsko urejeno arhiviranje, možnost različnih tabel.

2.5 Uporabljene programske rešitve

V primeru podjetja za varnost pri delu in požarno varstvo iz katerega črpamo izkušnje in informacije (v nadaljevanju naročnika), si poglejmo, katere so tiste programske rešitve, ki jih trenutno uporabljajo pri svoji vsakdanji dejavnosti:

- glavna knjiga (kjer so vnešene stranke in dogodki),
- planer dogodkov (uporablja podatke, vnešene iz glavne knjige),
- program za izdelavo zapisnikov o pregledu delovne opreme (uporablja podatke, vnešene iz glavne knjige),
- knjiga naročil (ob realizaciji naročila se ga zaključi s potrdilom, ki se knjiži v planer dogodkov),
- Microsoft Word,
- Microsoft Excel.

V naslednjih dveh podpoglavjih si oglejmo še analizo težav pri uporabi obstoječih programskih rešitev in seznam želenih izboljšav, ki naj bi jih prinesel nov informacijski sistem.

2.5.1 Problemi obstoječe programske programske opreme

Planer dogodkov

Ta aplikacija služi za evidenco izvedenih storitev pri posamezni stranki. V program lahko dostopajo vsi zaposleni. Vnos storitve poteka tako, da se v šifrantu izbere standardiziran opis naloge ter vnese morebitne opombe in podatke o datumu izvedbe naloge in rok zapadlosti. Rok zapadlosti predlaga program sam. Uporabnik ga potrdi ali spremeni.

Problemi:

- vnos podatkov o opravljeni nalogi je odvisen od skrbnosti strokovnega delavca. V kolikor opravljena naloga ni vnešena, ob zapadlosti roka tega ne bomo vedeli, kar je zelo problematično in lahko povzroči škodo in finančne posledice tako pri stranki kot tudi pri izvajalcu storitve,
- vnos v planer je ročen, se pravi ne omogoča avtomatskega ažuriranja,
- program ne opozarja avtomatsko na zapadle naloge, ampak je to zopet prepuščeno uporabniku. V kolikor uporabnik neredno pregleduje zapadlost dogodkov, storitve lahko niso pravočasno izvedene.

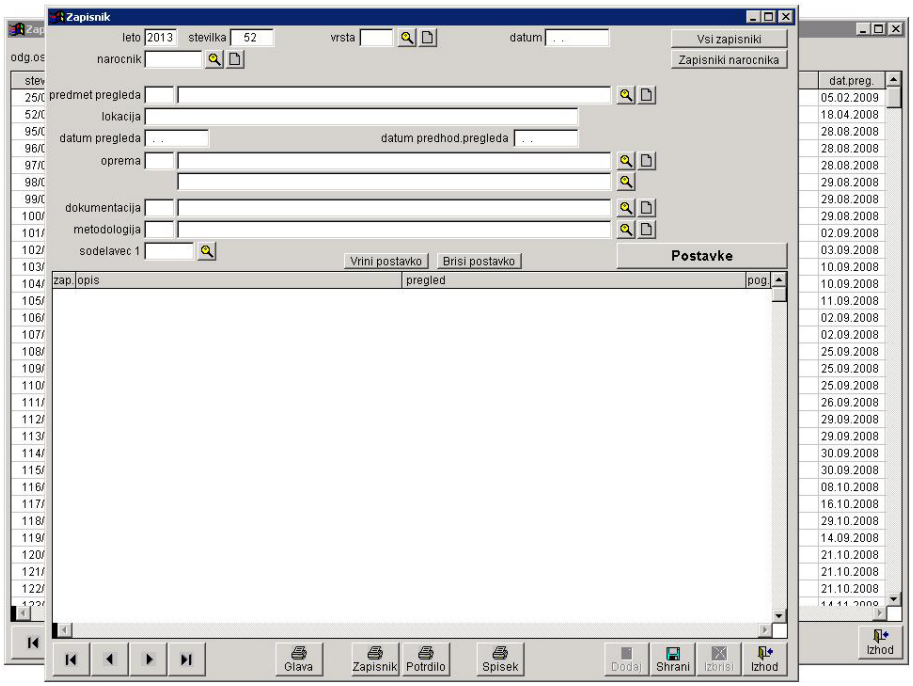
Program za izdelavo zapisnikov o pregledu delovne opreme

Program omogoča izdelavo zapisnika o opravljenem pregledu strojev, arhiviranje in pregled nad opravljenimi pregledi in izpis seznama strojev pri posamezni stranki (slika 2.1).

Problemi:

- nezmožnost popravljanja standardiziranega dela vsebine s strani uporabnika,
- nezmožnost združevanja dveh že izdelanih zapisnikov v enega,

- kdorkoli, ki ima dostop do programa, lahko briše vsebine zapisnikov, ki so bili že izdelani,
- nezmožnost avtomatskega vnosa podatkov strojev v evidenčno tabelo,
- togost programa - nezmožnost uporabe nekaterih znakov in oblik pisave.

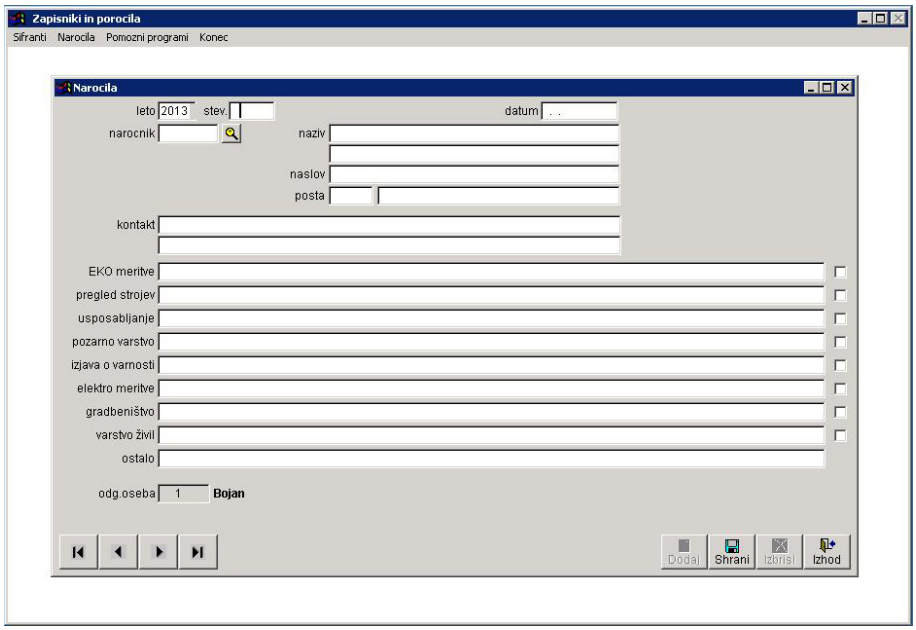


Slika 2.1: Program za izdelavo zapisnikov.

Knjiga naročil

Knjiga naročil (slika 2.2) služi dokumentiranju posameznih naročil strank. Program je povezan z glavno knjigo oziroma podatki obstoječih strank. V kolikor stranke še ni v glavni knjigi, se podatke začasno vnese v program ročno. Po izbiri stranke iz glavne knjige ali vnosu nove stranke se vpiše naročeno storitev, rok izvedbe in morebitne opombe (ceno, kraj izvedbe, datum, uro itd.). Odgovorno osebo za izvedbo storitve se določi na tedenskem

sestanku in se jo po sestanku ročno zabeleži v programu. Novo naročilo vnese kdorkoli od zaposlenih, ki naročilo sprejme. Običajno se to zgodi preko telefona.



Slika 2.2: Knjiga naročil.

Problemi:

- program naročil je povezan s planerjem dogodkov z namenom zaključitve naloge z vnosom izvedene naloge v planer, vendar se tega ne uporablja zaradi nedoslednosti uporabnikov, vzrok pa je tudi v kompleksnem načinu dela, ki od uporabnika zahteva preveč časa,

- program od strokovnih delavcev zahteva dnevno skrb za kontrolo vnosa novih naročil, česar pa zaradi prevelike porabe časa za to opravilo ne izvajajo redno. Tako se morajo strokovni delavci kljub vsej množici programov še precej pogovarjati med seboj. Pri tem se izguba časa množi.

Microsoft Word in Excel

Problemi:

- nezdružljiva s programi, ki jih trenutno uporabljamo,
- obvladovanje dokumentov zahteva veliko doslednost pri arhiviranju datotek in zelo dodelan sistem shranjevanja ter upoštevanje konvencij, kar se ne dogaja,
- delno stihijski način arhiviranja, ki je pogosto neusklajen med sodelavci,
- pogosto dolgotrajno iskanje ustreznih dokumentov,
- preveč časa se porabi za oblikovanje dokumentov,
- nezmožnost direktnega prenosa podatkov,
- zamudno delo z dokumenti,
- zamudno iskanje obstoječih datotek.

2.5.2 Želene izboljšave trenutno uporabljene programske opreme

Najbrž ni potrebno posebej poudarjati, da nam je že prejšnje poglavje precej dobro predstavilo probleme, ki jih lahko neposredno prevedemo v želje za izboljšavo. Pa vendar je smiselno, da storimo še korak dlje in na podlagi te analize naredimo še bolj sistematičen pregled želja naročnika. Analiza problemov v prejšnjem poglavju je služila kot nekakšno neformalno viharjenje možganov, na podlagi katere lahko sedaj bolje formaliziramo bodoča opravila in tako še bolj zmanjšamo prepad med razumevanjem končnega cilja med naročnikom in razvijalci.

Gremo sedaj po vrsti od aplikacije do aplikacije in pri vsaki zapišimo seznam želenih sprememb.

Planer dogodkov

Program mora na prijazen način sam obveščati uporabnika o zapadlih dogodkih, vendar ne prepogosto. Mogoče bi bilo ustrezneje, če bi program avtomatsko obveščal samo o nalogah, ki so zapadle v določenem mesecu in v naslednjem niso bile zaključene oziroma izvedene. Tako bi bil vnos dogodka v planer avtomatski, ko bi neko nalogo izvedli.

Program za izdelavo zapisnikov o pregledu delovne opreme

Naj omogoči:

- spremembo standardizirane vsebine s strani uporabnika,
- ob izdelavi zapisnika izpis evidence strojev,
- možnost ročnih sprememb podatkov v evidenci,
- izbira strojev, ki bodo v novem zapisniku, ne glede na to, v katerem zapisniku se nahajajo,
- večjo izbiro znakov in pisav.

Knjiga naročil

Naj omogoči:

- prijaznejši in enostaven postopek zaključitve naročila,
- samodejno obveščanje odgovornih oseb za izvedbo naloge.

Microsoft Word in Excel

Nov informacijski sistem naj v zvezi s pomankljivostmi, opisanih pri teh dveh programih, omogoča:

- da uporabnik določi vsebino, sistem pa skupno obliko - uporabnik naj se z obliko dokumenta ne ukvarja,

- zmanjšanje potrebe po uporabi teh programov,
- prenos nalog iz tega okolja v nov informacijski sistem,
- dostopnost in obvladljivost obstoječih dokumentov v novem informacijskem sistemu,
- povezanost sorodnih dokumentov in podatkov,
- sorodni dokumenti in podatki se ne smejo podvajati,
- podatki v tem delu sistema morajo biti združljivi z drugimi deli sistema,
- samodejno arhiviranje,
- hitrejši in preglednejši način arhiviranja,
- usmerjanje uporabnika pri delu in upoštevanje konvencij,
- pravočasno obveščanje uporabnika o napakah, podvojenih vnosih in obstoječih vnosih,
- onemogočanje stihijskega vnosa podatkov.

2.6 Določitev kriterijev za dobro programsko rešitev

Sedaj, ko poznamo želje naročnika, se moramo, preden se podamo v načrtovanje in razvoj sistema, vprašati, kakšna mora biti programska oprema, ki bo nastala. V tem poglavju bomo torej določili kriterije, ki se jih bomo skušali držati pri izdelavi naše programske rešitve.

Jasno je, da je programska rešitev, ki nastaja, namenjena določenim podjetjem v naši ciljni stroki. Uporabniki aplikacije bodo zaposleni nekega podjetja za varnost pri delu.

Sedaj je potrebno le še določiti, ali bomo izdelali informacijski sistem, ki bo koristil enemu samemu, točno določenemu podjetju, ki bi z njegovo

pomočjo doseglo večjo konkurenčno prednost, ali tudi vsem ostalim konkurentom v tej stroki. Odgovor na to vprašanje je: oboje, a postopoma.

V prvi fazi bomo namreč izdelali aplikacijo, ki jo bo preizkušalo in uporabljalo samo naše ciljno podjetje, torej naš naročnik. To bo informacijski sistem, ki bo uporabnikom, v fazi testiranja prototipov, najprej dostopen preko lokalnega omrežja in povezave VPN, kasneje, ko bo bolj dodelan, pa preko oblaka.

Prvotni interes je namreč pomagati našemu ciljnemu podjetju pri izboljšavi poslovnih procesov in povečanju konkurenčnosti v svoji panogi, dolgoročni cilj pa je v oblaku ponuditi kvalitetno in preizkušeno aplikacijo, od katere bi imela korist tudi ostala podjetja v isti panogi.

V nadaljevanju si poglejmo kriterije, ki se jih bomo držali.

2.6.1 Celovitost

Programska rešitev mora zadoščati vsem potrebam za delo strokovnega delavca. Vse, kar slednji potrebuje pri administrativnih nalogah, mora biti na enem mestu, na dosegu roke. To pomeni, da mora sistem vključevati vse, kar strokovni delavec potrebuje za svoje delo. Dodajanje uporabnikov in modulov mora biti enostavno in uporabnikom ne sme onemogočati normalnega dela. Seznam modulov, ki so nepogrešljivi za zadostitev tega kriterija, bo natančneje definiran v tretjem poglavju, saj ga ne moremo pravilno izdelati, dokler nismo izvedli temeljite analize trenutnega stanja.

2.6.2 Enostavna dostopnost (lokalno/globalno)

Uporabnikom je potrebno zagotoviti enostaven dostop do aplikacije, ne glede na to, kje izvajajo svojo dejavnost. Možnosti izvedbe je veliko, končna odločitev pa je odvisna od omenjenega podjetja.

Ena možnost je ta, da aplikacijo namestimo kot spletno aplikacijo na lokalnem strežniku podjetja (na intranet). Tako bi bila aplikacija dostopna samo znotraj domene podjetja. Vsak uporabnik bi zaradi sledljivosti imel

svoje uporabniško ime in geslo za prijavo v HIS.

V primeru, da se izkaže, da bi podjetje potrebovalo dostop do aplikacije na terenu, se lahko uporabnikom omogoči povezavo VPN, s čemer rešimo problem lokalnosti.

Slabost tega načina seveda tiči v slabostih, ki jih s seboj nosi vsak lokalni strežnik, vsaka lokalna mreža računalnikov (stroški vzdrževanja). Če bi aplikacija bila postavljena v oblak, bi se sicer znebili tega problema, a so tudi tam težave, s katerimi bi se prej ali slej soočili. Prva težava je varnost zasebnih podatkov in usklajenost le-tega z zakoni, druga težava pa je ta, da si podjetje ne bi moglo privoščiti izpada internetne povezave, saj bi tako lahko delo obstalo. Povečajo pa se tudi odgovornosti na strani izvajalca sistema.

2.6.3 Enostavna možnost preizkusa v primeru postavitve v oblak

V primeru da se sistem ponudi širšemu trgu in ne samo enemu podjetju, potencialni uporabnik ne sme imeti težav z dostopom in preizkusom aplikacije na internetu. To je sicer delno res stvar merketinga in bi lahko rekli, da ne spada v to poglavje, a je tu potrebno izpostaviti, da moramo mi kot informatiki že na začetku dobro razmisliti in predvideti arhitekturo aplikacije, ki bo omogočila zadostitev tega kriterija. Ta odločitev torej igra pomembno vlogo, ko se, kot v tem primeru, odločamo za postavitev aplikacije v oblak, torej v obliki programja kot storitve (SaaS).

2.6.4 Odlična uporabniška izkušnja s kvalitetnim grafičnim uporabniškim vmesnikom

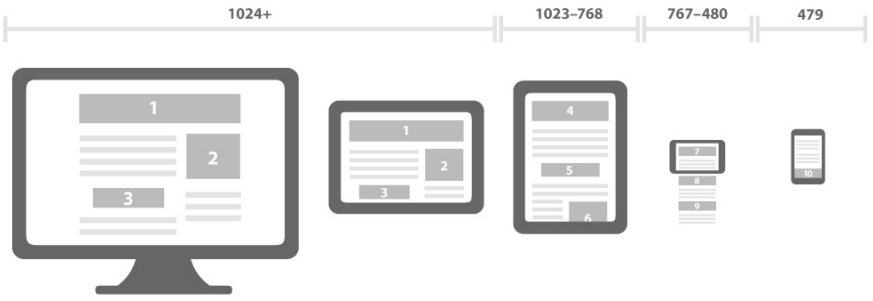
Uporabniška izkušnja je ključnega pomena za uspeh produkta. Stik uporabnika z aplikacijo mora biti takšen, da uporabnika dejansko motivira k uporabi programske rešitve. Ta mora uporabnika prepričati in mu vzbuditi zaupanje ter željo po ponovni uporabi.

Enostaven grafični uporabniški vmesnik, ki podpira hitrost odločanja in konvencije

Grafični vmesnik celotnega sistema mora biti tako zelo intuitiven in enostavno zastavljen, da uporabniku ni potrebno preveč razmišljati, kako priti do cilja. Sistem mora uporabnika na enostaven način voditi skozi administrativne naloge in ga spodbujati pri sledenju in uporabi konvencij ter tako zmanjšati stihijski način dela. Vsak uporabnik začne tako na spontan način slediti enakim postopkom kot ostali uporabniki sistema.

Opravila oziroma postopki uporabniških akcij, morajo biti poenostavljena do te mere, da se ne ponavljajo. Aplikacija mora biti zastavljena tako, da če uporabnik v neko vnosno polje vnese nek podatek, mora biti ta podatek v realnem času na voljo tudi v drugih delih programa, kjer se ta podatek potrebuje. To skratka pomeni, da mora aplikacija vsebovati v sistem, ki bo na enostaven način zagotavljal ažurnost in konsistentnost podatkov v realnem času. Ena izmed možnosti implementacije takega sistema je uporaba vzorca MVVM [9]. Primer implementacije slednjega je prisoten v odprtokodni JavaScript knjižnici Knockout[10].

Odzivna razporeditev elementov grafičnega uporabniškega vmesnika

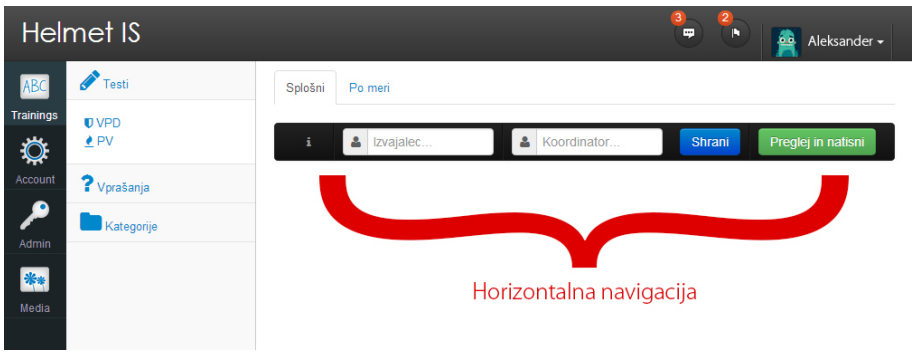


Slika 2.3: Odziven grafični uporabniški vmesnik.

Zaradi poplave različnih mobilnih naprav in posledično zaslonskih velikosti in resolucij mora biti grafični uporabniški vmesnik oziroma grafična postavitev primerno odzivna (angl. Responsive Layout). To pomeni, da se mora grafični uporabniški vmesnik smiselno prilagajati velikosti oziroma resoluciji trenutnega zaslona, a uporabniku še vedno ponujati vse najnujnejše informacije in funkcionalnosti, ki jih ponuja pri največjih zaslonih in resolucijah (slika 2.3).

Tako se mora na primer element HTML horizontalne navigacije ob največji resoluciji prikazati v celoti, ko pa zaslon pomanjšamo ali isto vsebino odpremo na mobilnem telefonu, se mora vsebina te navigacije "skriti"in biti na voljo v spustnem seznamu.

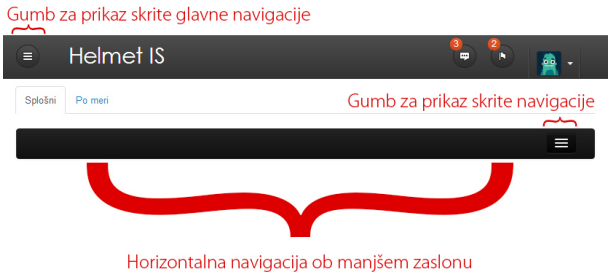
Sliki 2.4 in 2.5 primer prilagajanja grafičnega uporabniškega vmesnika v odvisnosti od velikosti oziroma resolucije zaslona.



Slika 2.4: Horizontalna navigacija, prikazana v celoti, ko je zaslon dovolj velik.

Hitro in učinkovito iskanje želenih podatkov

Aplikacija mora imeti omogočeno infrastrukturo in tak grafični vmesnik, ki omogoča hiter in učinkovit način iskanja podatkov.



Slika 2.5: Horizontalna navigacija, ko zaslon ni dovolj velik.

Samodejno zaključljiva vnosna polja

Sistem mora omogočati, da vsa vnosna polja, pri katerih je to mogoče, ob vnašanju teksta, črpajo podatke iz podatkovne baze in tako avtomatsko predlagajo obstoječo vsebino. To naj se, kot je v navadi, izvaja preko poizvedb AJAX na podatkovni bazi.

2.6.5 Učinkovit sistem obvestil in sporočil

Glede na to, da je v sistemu, ki je predmet te diplomske naloge, prisoten močan socialni faktor, je potrebno poskrbeti za učinkovit sistem obveščanja uporabnikov o dogodkih v sistemu in za medsebojno komuniciranje uporabnikov v obliki sporočil, ki bodo do uporabnika lahko prišla po različnih kanalih, odvisno od uporabnikovih nastavitev.

Dober primer takega načina obveščanja je prisoten v večjih socialnih omrežjih, kot sta Facebook in Google+, kjer nam ob vsakem prejetem sporočilu ali obvestilu "zažari lučka" na določenem mestu, ki je za to predvideno. Tak sistem mora vsebovati tudi naš informacijski sistem. Dodatna možnost oziroma razširitev teh funkcionalnosti, je obveščanje uporabnikov na njihove elektronske naslove ali telefone preko sporočil SMS.

Vse to mora biti nastavljivo za posameznega uporabnika posebej, saj moramo uporabniku, kljub nekim vsiljenim konvencijam, še vedno dovoliti neko mero svobode pri nastavitvah.

2.6.6 Enostavna registracija novega uporabnika in prijava v sistem

Registracija je vsekakor nujna, kljub temu, da so uporabniki večinoma znani vnaprej, saj se zaposleni v podjetju s časom menjajo, pa tudi z vidika varnosti in morebitnih nesporazumov je vsekakor pomembno, da ima vsak uporabnik svoj lasten račun oziroma dostopne podatke. Pri predpostavki, da je sistem HIS tak, pri katerem samoregistracija uporabnika ne sme biti omogočena, poglejmo, kakšne so ostale možnosti, ki pridejo v poštev.

V odvisnosti od tega, ali končni produkt postavimo v oblak ali na lokalni strežnik podjetja, je potrebno razmisliti, na kakšen način bomo omogočili registracijo. V primeru intranet aplikacije (lokalni strežnik), kjer so uporabniki znani vnaprej, je potrebno imeti sistem, ki omogoča različne vloge uporabnikov, tako da lahko obstoječi uporabnik z vlogo administratorja ustvari oziroma registrira novega uporabnika v sistem.

Na tem mestu velja omeniti tudi možnost, ki bi jo v zvezi z uvozom uporabnikov lahko uporabili v prihodnosti. Govorimo o protokolu LDAP [12], ki se ga med drugim velikokrat uporablja za uvoz objektov, povezanih z uporabniki.

Če se odločimo, da aplikacijo postavimo v oblak, v obliki programja kot storitve, kjer uporabniki niso vnaprej znani, je potrebno ustvariti sistem registracije, ki je hiter, enostaven in predvsem varen. Število korakov za prvo registracijo mora biti minimalno, da se uporabnika ne odžene že pri prvem stiku s programsko rešitvijo. Vendar pa je potrebno v določenem trenutku, na "takten način", povečati stopnjo varnosti in uporabniku nevsiljivo ponuditi nadaljnje možnosti in "obveznosti". Poglejmo si primer, kako bi potekala registracija uporabnika, če bi bil sistem na voljo v obliki programja kot storitve, torej bi bil na voljo mnogim različnim (med seboj konkurenčnim) podjetjem.

Denimo, da želi direktor podjetja A prvič preizkusiti programje v obliki storitve, ki je na voljo vsem uporabnikom svetovnega spleta. Na spletnem naslovu sistema se preko registracijskega obrazca prijavi v sistem, tako da vnese svoj elektronski naslov, geslo in njegovo potrditev. Direktor podjetja A

(v nadaljevanju uporabnik X) na svoj elektronski naslov prejme aktivacijske podatke, s pomočjo katerih aktivira svoj račun v sistemu. Uporabnik X se nato prijavi v sistem in po želji sistem preizkusi ter izpolni podatke v nastavitvah svojega uporabniškega profila. Ker je uporabnik X prvi uporabnik podjetja A, ki se je kadarkoli prijavil v sistem, postane ta uporabnik administrator svoje delovne skupine, ki ima trenutno samo enega člana. Na podlagi povabila uporabnika X se v sistem lahko prijavijo drugi uporabniki istega podjetja. Sistem jih avtomatsko uvrsti v isto skupino, torej skupino podjetja A. Ta skupina ima za sedaj samo enega administratorja, dokler uporabnik X (ki ima do tedaj edini administratorske pravice v okviru svoje skupine) ne dodeli administratorskih pravic drugim uporabnikom, izključno skupine podjetja A.

Primeri takega načina registracije so vedno pogosteje v uporabi in vidni v mnogih spletnih sistemih, ki imajo skupen podoben delovni tok registracije.

2.6.7 Ustrezne varnostne zahteve

Uporabnik, ki se poslužuje storitev sistema, mora za dostop uporabljati veljaven digitalni podpis/certifikat in sistem mora to seveda podpirati. Ob validaciji s formami moramo nujno uporabljati zavarovan protokol HTTPS ali Secure HTTP. Obstaja tudi možnost intranetne aplikacije, kjer igra pomembno vlogo pri varnosti tudi dostop VPN.

2.6.8 Večnivojska arhitektura programske rešitve

Zaradi kompleksnosti problemov in programske rešitve, je smiselno uporabiti več-nivojsko arhitekturo (angl. N-Tier Architecture) in komponente, ki so med seboj šibko povezane (angl. loosely-coupled), kar omogoča lažje vzdrževanje, testiranje in razširljivost sistema.

2.7 Analiza obstoječih programskih rešitev na trgu

Do sedaj smo opredelili probleme, ki so prisotni pri delu strokovnega delavca, predvsem pa smo določili kriterije, ki se jih bomo držali pri izdelavi sistema. Na podlagi tega lahko sedaj ocenimo, ali na trgu mogoče že obstajajo programske rešitve, ki delno ali v celoti zadoščajo naštetim kriterijem, oziroma če lahko kaj od tega ponovno uporabimo v našem sistemu. Nesmiselno bi namreč bilo na novo razvijati nekaj, kar nam je že na voljo.

Ko pa smo opravljali raziskavo in skušali najti obstoječe programske rešitve, sorodne informacijskemu sistemu, ki ga želimo ustvariti, smo kmalu ugotovili, da med razpoložljivimi niti ena ni tako celovita, kot je načrtovani sistem. Ker nobena aplikacija ni zadostila večini v tem diplomskem delu naštetih kriterijev, smo se odločili, da ne bomo nobene obstoječe programske rešitve posebej izpostavljali, temveč bomo v naslednjem podpoglavju raje opisali splošno stanje. Izpostavili bomo kriterije, katerim ni zadoščeno in so med drugim tudi razlog za nastanek novega sistema.

Vse trditve, navedene v naslednjem podpoglavju, temeljijo na raziskavi, pri kateri so sodelovali strokovni delavci s 30-letnimi izkušnjami na področju varnosti pri delu.

2.7.1 Neizpolnjeni kriteriji

Celovitost

Glede na obstoječe informacije nobena obstoječa programska rešitev ni celovita (poglavje 2.6.1) in ne zadošča vsem administrativnim potrebam strokovnega delavca. Vse razpoložljive rešitve rešujejo le del celotnega problema. To pomeni, da se mora strokovni delavec posluževati več različnih aplikacij, od katerih vsaka služi svojemu namenu, avtorji aplikacij pa so različni, kar pomeni velik strošek vzdrževanja. Obstoječe rešitve so v večini zastarele in niso zgrajene modularno, tako da ne omogočajo enostavne nadgradnje v primeru,

da se pojavi potreba po novem orodju (modulu).

Enostavna dostopnost

Kriterij enostavne dostopnosti (poglavje 2.6.2) pomeni, da lahko uporabnik dostopi do programske rešitve hitro, povsod in na katerikoli sodobnejši mobilni napravi. Torej, potrebno je izdelati rešitev, ki temelji na svetovnem spletu, v obliki spletne ali mobilne aplikacije, pa naj bo ta dostopna globalno ali lokalno. Tu smo takoj naleteli na težave. Mobilnih aplikacij na tem področju ni, tiste spletne aplikacije, ki pa so na voljo, so izjemno redke in jih strokovni delavci, s katerimi smo se pogovarjali, niso nikoli želeli uporabljati. Razlog za to je, da so obstoječe rešitve predrage, neučinkovite, postopek za pridobitev in preizkus pa je preveč kompleksen.

Možnost enostavnega preizkusa

Nobena obstoječa programska rešitev ne zadostuje kriteriju o možnosti enostavnega preizkusa (poglavje 2.6.3) oziroma postavitve programske rešitve v oblak. Pri pregledu ponudbe na slovenskem in tudi širšem evropskem trgu smo večinoma srečevali ponudno zastarelih aplikacij za vodenje evidenc zaposlenih in za izdelavo ocene tveganja. Nobena ni bila enostavno dostopna preko oblaka. V večini primerov je šlo za namizne aplikacije, pri katerih sta enostaven dostop in brezplačen preizkus težje zagotovljiva.

Odlična uporabniška izkušnja

Obstoječe programske rešitve tudi ne zadovoljijo kriteriju kvalitetne uporabniške izkušnje (poglavje 2.6.4). Grafični uporabniški vmesniki so zastareli, niso dovolj intuitivni in niso sposobni prilagajati se različnim velikostim zaslonov. Grafična odzivnost, glede na napravo, je slaba. Delo s takimi aplikacijami je za strokovnega delavca naporno.

Sistem obvestil

Ker so aplikacije zastarele, ne upoštevajo socialnega vidika, torej medsebojne komunikacije med uporabniki (poglavje 2.6.5). Ne omogočajo zadostnega obveščanja med uporabniki, s tem pa se zanemarja pomen timskega dela. To pomeni, da sodelavci med seboj pogosto premalo komunicirajo in ne dobivajo stimulacije, ki bi jo sicer lahko dobili pri bolj socialno usmerjenih sistemih. Produktivnost tima se tako premalo vzpodbuja.

Enostavna registracija v sistem

Glede kriterija enostavne registracije ne moremo veliko povedati, saj tega vidika nismo uspeli objektivno preizkusiti, a glede na zastarelost programskih rešitev, ki smo jih lahko preizkusili, lahko sklepamo, da je v splošnem zanemarjen tudi ta vidik.

Poglavje 3

Rešitev problema

V prejšnjem poglavju smo temeljito analizirali področje varnosti pri delu in požarne varnosti, predvsem pa naloge strokovnega delavca ter definirali probleme, ki nastajajo v praksi in ki jih je potrebno rešiti z nastajajočim informacijskim sistemom. Določili smo tudi kriterije za dobro programsko rešitev, ki jim bomo sledili, skratka, postavili smo temelje, na katerih lahko sedaj načrtujemo rešitev.

Ker si ne želimo, da bi razvoj sistema potekal stihijsko, je postopke potrebno kar se da formalizirati, pa čeprav to v začetku vzame dragocen čas. Tako dosežemo sledljiv razvoj in kakovosten končni produkt, ki je enostaven za vzdrževanje. Se je pa potrebno tudi zavedati, da je projekt tako obsežen, da je za njegov razvoj ena oseba nedvomno premalo. Dejstvo je tudi, da po pravilih in zdravemu razumu, za vsako fazo razvoja sistema, ne more in ne sme biti zadolžena samo ena oseba. Med drugim tudi zaradi morebitnih konfliktov interesov posameznih vlog. Ker gre pri našem projektu zgolj za diplomsko nalogo in smo časovno omejeni, smo namenoma naredili izjemo pri omenjenih pravilih.

Na začetku smo se odločili, da bomo sledili metodologiji strukturnega razvoja, a smo se hkrati zavedali, da se bomo srečevali tudi z značilnostmi agilnih metodologij. Okvirni cilj končnega informacijskega sistema je bil namreč znan že na začetku, toda je bil ta ves čas nekoliko zabrisan in ne

čisto natančno določen. Do tistega pravega in končnega cilja nas dejansko vodi sproti sam razvoj, kar je izrazita značilnost agilnih metodologij.

Omeniti je potrebno tudi, da smo se pri sami fazi analize opirali na standardne postopke, ki spremljajo to fazo metodologije strukturnega razvoja informacijskih sistemov. Postavili smo si vprašanja o tem, kako bomo sistem zasnovali, kakšno arhitekturo bomo uporabili, kako bomo morali zasnovati podatkovno bazo ter v kateri tehnologiji. Zavedali smo se, da je podatkovni model izjemnega pomena in je temelj končne programske rešitve, zato smo morali dobro razmisliti o tem, kaj bomo hranili v podatkovni bazi ter kako bomo entitetne tipe (ali enititete) med seboj povezali, ali z drugimi besedami, kakšne bodo relacije oziroma asociacije med njimi. Pojavila so se tudi vprašanja, ali bomo morda lahko za rešitev problema uporabili že obstoječo kodo ali tehnologijo ter kako bomo poskrbeli, da bo moja koda lahko v prihodnosti ponovno uporabljena (angl. code reusability). Na podlagi tega smo se morali odločiti tudi o tem, katero programsko ogrodje za razvoj aplikacij bomo uporabili in okolje, v katerem bo programska rešitev "živela", da bi kar se da najbolje zadostila učinkovitosti poslovnih procesov.

Podatkovni model smo izdelali s pomočjo vizualnega orodja Microsof Server SQL Management Studio. Uporabili smo tudi orodje ORM Entity Framework in izbrali tehniko "najprej model", s pomočjo katere se poenostavi dostop do podatkovne baze. Nastane nov nivo v trinivojski arhitekturi, to je podatkovni nivo.

3.1 Zajem zahtev naročnika

Da smo lahko določili zahteve, smo morali z naročnikom najprej ugotoviti, kateri so problemi, s katerimi se strokovni delavec srečuje pri opravljanju vsakdanjih nalog. To smo storili v okviru temeljite analize v drugem poglavju. V nadaljevanju je seznam zahtev naročnika, ki natančneje opisuje, kaj slednji oziroma uporabnik sploh pričakuje od sistema. Zahteve so splošne, neobdelane in napisane z vidika naročnika.

3.1.1 Zahteva 1: Namen informacijskega sistema

Namen informacijskega sistema je preko sodobnega spletnega grafičnega vmesnika poenotiti in poenostaviti ter narediti vsakdanja opravila zaposlenega oziroma strokovnega delavca zanimivejša in učinkovitejša. Sistem naj omogoča avtomatizacijo večine delovnih procesov na področju varstva pri delu in požarnega varstva, ki so se do sedaj izvajali ročno. Končni namen je povečanje učinkovitosti uporabnikov in skrajšanje časa posameznih delovnih procesov.

Preko centraliziranih podatkov naj sistem omogoča enostavnejši pregled nad strokovnimi podatki in dokumenti, bistveno enostavnejši naj bosta tudi njihovo iskanje ter uporaba. V okviru sistema naj bo s pomočjo enotnega grafičnega vmesnika ustvarjeno enotno delovno okolje za vsakega uporabnika. Poenotene naj bodo metode dela uporabnikov, prav tako naj ne bo več nekonsistentnosti dokumentov med računi posameznih uporabnikov.

Sistem mora izboljšati in olajšati komunikacijo med uporabniki, obenem pa ponuditi tudi pregled vseh opravil, iz katerega bo natančno razvidno, kateri uporabnik je zadolžen za določeno opravilo. Eden izmed končnih ciljev naj bo med drugim tudi ukinitev uporabe datotečnega sistema preko lokalne mreže kot glavnega vira podatkov. Bistveno naj se zmanjša ali v celoti odstrani potreba po uporabi lokalnega strežnika in s tem tudi stroški vzdrževanja slednjega. Ta del naj se prenese na splet. Sistem naj se izdela v prvi fazi za namestitev na lokalni strežnik z možnostjo namestitve na zunanji strežnik ali po možnosti v “oblak”.

3.1.2 Zahteva 2: Enostavna obdelava podatkov in dostop do njih

Pri obdelavi podatkov mora sistem omogočiti enostavnost vnosa podatkov s čim manj koraki. Omogočiti mora enostavno iskanje in hiter dostop do izbranega področja. Program naj omogoči predogled nameravanega tiskanja dokumenta.

3.1.3 Zahteva 3: Prijaznost sistema s krajšimi časi izvedbe

Sistem naj bo prijazen do uporabnika. Omogočiti mora krajši čas za izdelavo naročene naloge - izdelka.

3.1.4 Zahteva 4: Onemogočanje podvajanja podatkov

Sistem naj prepreči podvajanje podatkov na različnih lokacijah. Kjerkoli v sistemu naj uporabnik vnese določen podatek le enkrat, ta pa mora biti na razpolago v vseh potrebnih delih sistema.

3.1.5 Zahteva 5: Enostaven vpogled v podatke

Sistem naj uporabniku omogoči vse možne vpoglede v podatke in sicer, da lahko uporabnik takoj dobi podatke o delavcu, kdaj in kolikokrat je bil na usposabljanju in katerih ter kdaj in kolikokrat je bil na zdravniškem pregledu. Na področju strojev naj omogoči vpogled v zgodovino dogodkov o posameznem stroju, kdaj je bil pregledan, kdo je pregled opravil oziroma kakšne so bile pripombe ob pregledu. Skratka, infrastruktura sistema mora biti dovolj dobro izdelana, tako da ima lahko uporabnik v splošnem grafični uporabniški vmesnik, ki omogoča pridobivanje kompleksnih podatkov.

3.2 Ureditev zahtev po področjih - modulih

Zajete zahteve, naštete v podpoglavju 3.1, so očitno neurejene in precej splošne. Pa vendar gre tudi v praksi pričakovati, da bodo zahteve naročnika do neke mere nejasne. Prav v tem je razlog, da se pri razvoju sistema, kot je HIS, še naprej posvetimo natančnejši analizi in obdelavi pridobljenih informacij in se vmes večkrat ponovno posvetujemo z naročnikom.

Tu se torej pokaže pravi namen vseh dosedanjih korakov, ki so del metodologije strukturnega razvoja - torej natančne analize, zajema zahtev in njihova ureditev. Proces je sicer morda res dolgotrajen in naporen, a je

dolgoročno gledano zgolj manjši strošek pri izdelavi kakovostne programske rešitve. Namen izbrane metodologije je zmanjšati prepad v medsebojnem razumevanju vseh udeležencev pri razvoju projekta.

Na podlagi problemov, ki smo jih identificirali v drugem poglavju , lahko sedaj precej natančno določimo vse končne zahteve nastajajočega informacijskega sistema.

Kako bi to najlažje naredili? V veliko pomoč nam je Microsoftovo programsko ogrodje ASP.NET MVC4, v katerem bomo izdelali naš končni produkt oziroma njegov predstavitveni del. Lahko bi izbrali tudi kakšno drugo ogrodje za obličje sistema (na primer Microsoft Web Forms), a smo se zaradi množice odlik spletnega programskega ogrodja ASP.NET MVC4 odločili za uporabo slednjega.

Med odlike tega ogrodja štejemo:

- pomoč pri izdelavi zmogljivih dinamičnih spletnih aplikacij, ki temeljijo na vzorcih in konvencijah,
- ločevanje posameznih skrbi v sorodne skupine,
- lažje vzdrževanje in testno usmerjen razvoj,
- možnost izdelave šibko povezanih, torej medsebojno neodvisnih programskih komponent,
- enostavna možnost integracije ogrodja v trinivojsko arhitekturo,
- ogrodje omogoči popolno kontrolo nad upodabljanjem kode HTML,
- enostavna integracija z ogrodji JavaScript ...

To programsko ogrodje nas namreč vodi k temu, da naloge za posamezna področja kategoriziramo na poseben način. To se sicer neposredno tiče same arhitekture, o kateri bo govora v naslednjih podpoglavjih. Na tem mestu je dovolj, da bralcu pojasnimo le, da bomo zahteve razporedili po področjih

(angl. Areas) oziroma modulih. Vsak modul bo lastna, samostojna celota, ki bo namenjena reševanju skupine sorodnih problemov. Kaj moduli so in kakšna je njihova vloga, bomo bralcu predstavili v naslednjem poglavju, ko bomo predstavili rezultat našega dela.

V tej diplomski nalogi opisujemo sicer zasnovo celega informacijskega sistema z vsemi moduli, vendar bomo zavoljo časa, ki je na razpolago, zares podrobno opisali le modul "Usposabljanje", ki bo tudi edini realiziran modul za namene te diplomske naloge. Vsi ostali moduli bodo sicer opisani, a zagotovo ne bodo vsebovali vseh zahtev, temveč le tiste osnovne, ki samo nakazujejo, v kateri smeri se bomo pri njih gibali.

3.2.1 Glavni moduli

Module lahko razdelimo na dve vrsti - glavni in podporni. V tem poglavju bomo opisali prvo vrsto.

Glavni moduli so tisti, ki neposredno služijo ciljem sistema. Z njimi bomo dejansko izdelali nekaj, kar doprinese k boljšemu poslovemu učinku. Skratka, rezultati teh modulov so oprijemljivi oziroma merljivi z učinki. Primer tega je lahko natisnjena testna pola, pripravljena za usposabljanje delavcev.

Glede na zahteve naročnika smo naloge razporedili na posamezne glavne skupine oziroma glavne module:

1. Usposabljanje,
2. Delavci,
3. Stroji,
4. Evidenca pogodbenih strank,
5. Izjava o varnosti.

Končni produkt bo seveda vseboval več modulov, saj bomo k že naštetim morali dodati podporne module (poglavje 3.2.2), katerih namen bo predvsem podpora glavnim petim modulom.

V nadaljevanju sledi razširitev naročnikovih zahtev iz poglavja 3.1, ki smo jih pripravili na podlagi analize, opisane v drugem poglavju, in njihova ureditev po področjih.

Modul Usposabljanje

Pri tem modulu gre v osnovi za sestavo izpitnih pol pri usposabljanju delavcev za varno delo in požarno varnost. Cilj tega je hitra izdelava ustreznih testnih pol, preden se strokovni delavec odpravi na delo na terenu, torej na usposabljanje delavcev pri določeni stranki.

Uporabnik naj ima možnost izbire izdelave dveh različnih tipov testnih pol:

- tip VPD - varnost pri delu in
- tip PV - požarna varnost.

Testna pola tipa VPD naj vsebuje prvo stran zapisnika ter vprašalnik, ki je lahko sestavljen samo iz splošnih vprašanj ali tudi posebnih, dodatnih vprašanj, ki so vezana na določeno delovno mesto.

Modul naj uporabnikom, ki izberejo področje VPD, omogoča hitro nastavitev oziroma sestavo in tiskanje:

- splošnega testa,
- testa glede na delovno mesto,
- testa glede na delavca.

Testna pola tipa VPD naj ima drugačno prvo stran zapisnika in samo vprašalnik s posebnim delom.

V nadaljevanju so navedene funkcionalne zadeve, ki se nanašajo na modul Usposabljanje.

Sestava splošnega testa, tipa VPD

- V vpogled mora biti prednastavljen zapisnik in vsa splošna vprašanja, tako da lahko uporabnik hitro izvede predogled celotne pole splošnega testa in natisne izbrano število pol.
- Uporabnik lahko v posebno vnosno polje vnese ime izvajalca teoretičnega usposabljanja in koordinatorja praktičnega usposabljanja. Ko to izbere, se morajo spremembe v realnem času odražati na predogledu testne pole in na na polah za tisk.
- Zapisnika in splošnih vprašanj v tem pogledu ni mogoče spreminjati in je vsem prijavljenim uporabnikom viden na popolnoma enak način, v popolnoma enaki obliki. Skratka, posamezen uporabnik nima možnosti prilagajanja splošne testne pole, saj je slednja standardizirana.
- Splošna testna pola se lahko spreminja samo v temu namenjenem podpornem modulu. Vsaka sprememba se mora takoj odražati vsem prijavljenim uporabnikom na enak način.
- Spremembe splošne testne pole se navadno dogajajo zelo poredko. Vsaka sprememba splošne testne pole sproži pojavo obvestila, ki ga vidijo vsi prijavljeni uporabniki.
- Pravico do sprememb splošne testne pole ima samo uporabnik s priviligirano uporabniško vlogo administratorja.
- Vsaka nova verzija testne pole mora biti vidna na seznamu sprememb oziroma seznamu verzij.

- Seznam verzij testnih pol mora vsebovati naslednje podatke:

  - datum in ura verzije,
  - opis spremembe, ki razlikuje to verzijo od prejšnje,
  - ime uporabnika, ki je avtor verzije.

- Administrator naj ima možnost povračila splošne testne pole v eno izmed prejšnjih verzij.

Sestava testa tipa VPD glede na delovno mesto

- Osnova te testne pole je sestavljena iz osnovne testne pole tipa VPD, kar pomeni, da ima enak zapisnik in splošna vprašanja.
- Sistem naj ponudi uporabniku še dodatna vprašanja, specifična za to delovno mesto.
- Poleg vseh ponujenih vprašanj mora imeti uporabnik naslednje možnosti:
  - brisanje neželenih vprašanj,
  - dodajanje novih vprašanj,
  - spreminjanje vrstnega reda vprašanj,
  - shranjevanje trenutne verzije za to izbrano delovno mesto.
- Vsakič, ko se uporabnik na novo prijavi v sistem in izbere določeno delovno mesto, se mu mora prikazati zadnja verzija testne pole za izbrano delovno mesto.
- Uporabnik naj ima na voljo seznam verzij testnih pol za to delovno mesto.
- Uporabnik lahko povrne trenutno testno polo v eno izmed testnih pol na seznamu verzij.

Sestava testa tipa VPD, glede na delavca

- Podobno kot pri testni poli, glede na delovno mesto, je tudi tu osnova te testne pole sestavljena iz osnovne testne pole tipa VPD, kar pomeni, da ima enak zapisnik in enaka splošna vprašanja, le da se tu pojavijo še dodatna vprašanja, ki so odvisna od izbranega delavca (enega ali več).

- Ta pogled omogoča izpis testa iz "kartonov" posameznih delavcev, enega ali več.

- Da sestavimo to testno polo, je potrebna povezava delavca z delovnim mestom, in sicer preko kartona delavca, kjer je označeno, s katerimi delovnimi mesti ima delavec opravka. Na podlagi le-teh se uporabniku ponudi primerno sestavljeno testno polo.

- Ponujena testna pola je za izbranega delavca osnovna, vendar jo lahko uporabnik sistema še dodatno prilagaja, in sicer z dodajanjem in odvzemanjem vprašanj ter spreminjanjem vrstnega reda le-teh.

- Uporabnik lahko v posebno vnosno polje vnese ime izvajalca teoretičnega usposabljanja in koordinatorja praktičnega usposabljanja. Ko to izbere, se morajo spremembe v realnem času odražati na predogledu testne pole in polah za tisk.

- Vsaka verzija testne pole, ki jo uporabnik sistema shrani, mora biti vidna in dostopna na seznamu verzij testnih pol za izbranega delavca.

- Sama izbira delavca poteka tako, da sistem sam izbere grafično komponento (filter), kjer ima uporabnik možnost izbire enega ali več različnih delavcev nekega poljubnega podjetja. V filtru se obkljuka ali kako drugače označi vse delavce, za katere se mora izvesti izpis testne pole. Glede na to, da gre najverjetneje za delavce na različnih delovnih mestih, je vsaka testna pola. Vsak delavec tako dobi samo njemu primeren test.

Dodajanje novih vprašanj in njihovo razvrščanje po skupinah

- Omogočeno naj bo dodajanje novih vprašanj v nabor obstoječih ter njihovo razvrščanje v določeno skupino (ena skupina je eno delovno mesto).
- Razvrstitev posameznega vprašanja v skupino bo nujna ob vnosu/kreiranju vprašanja. Vprašanje pripada dodeljleni skupini oziroma delovnemu mestu, dokler uporabnik eksplicitno ne zahteva nove razvrstitve.
- Ko dodamo novo delovno mesto, mora obvezno slediti tudi vsaj eno vprašanje, ker je nesmiselno imeti vnešeno delovno mesto, ki nima dotičnih vprašanj.
- Obstajati mora šifrant skupin oziroma delovnih mest.

Modul Delavci

Pri stroki varnosti pri delu in požarne varnosti se vse vrti okrog zaposlenih v podjetjih - delavcih. V tem diplomskem delu uporabljamo izraz "delavec"v širšem smislu in označuje zaposlene vseh vrst dejavnosti. Lahko je to frizer, gasilec, policist, mesar ali pa kakršenkoli drug poklic.

Podjetje za varnost pri delu in požarno varnost mora hraniti evidenco delavcev podjetij, pri katerih izvaja svoje storitve. Za vsakega delavca mora hraniti informacije o usposobljenosti za varnost pri delu, o zdravstvenih omejitvah, zapadlosti usposabljanj in zdravstvenih pregledov itd. Osrednji del tega modula bo tako imenovani elektronski karton ali kartoteka, ki bo uporabniku sistema nudila skupek vseh najpomembnejših informacij o posameznem delavcu iz seznama vseh delavcev. Informacije se bodo s časom dopolnjevale, s tem pa se bo povečevalo znanje glede posameznega delavca, v kontekstu varnosti pri delu in požarne varnosti. Med drugim bo na podlagi kartoteke delavca in njegovih podatkov sistem sposoben obveščati uporabnike sistema o potečeni veljavnosti usposabljanja in avtomatsko izdelati testno polo za njegovo ponovno usposabljanje. Na voljo bo tudi zgodovina dogodkov za

posameznega delavca.

V nadaljevanju si podrobneje poglejmo, kaj bo modul Delavci še vseboval.

- Obstajati mora možnost vpogleda v kartoteko delavca, v kateri bodo zbrani vsi podatki, ki se jih sedaj vodi v številnih Excelovih datotekah.
- Vsak delavec bo imel povezavo na delodajalca, torej se iz te povezave lahko izpisuje različne sezname in podatke:
  - zaposlenih pri delodajalcu,
  - zapadlosti usposabljanja,
  - zapadlost zdravstvenih pregledov,
  - zdravstvenih omejitev (tako v preteklem obdobju kot v trenutno stanje).
- Kartoteka delavca bo vsebovala naslednje atribute:
  - ime,
  - priimek,
  - podatki o rojstvu (samo letnica),
  - delovno mesto,
  - dodatna znanja, ki so dejansko potrebna za to delovno mesto,
  - datum usposabljanja,
  - veljavnost (rok poteka) usposabljanja,
  - datum zdravniškega pregleda,
  - rok zdravniškega pregleda,
  - morebitne zdravstvene omejitve delavca (obstajati mora šifrant omejitev),
  - status invalida (da/ne) ter katere kategorije,

  - kateremu delodajalcu - stranki pripada.

- Za vsako kartoteko delavca moramo zagotoviti dostop do vpogleda zgodovine sprememb. Možne spremembe:
  - menjava delovnega mesta delavca,
  - zdravniški pregled je bil narejen za eno delovno mesto in je zato ob menjavi delovnega mesta potrebno narediti nov pregled,

- Če gre delavec v pokoj, mora kartoteka postati neaktivna, vsebovati mora status.

- Obstajati mora možnost izpisa samo aktivnih ali samo neaktivnih delavcev.

- Omogočen mora biti status v primeru invalidnosti, sistem mora vsebovati tudi šifrant omejitev in mora omogočati njihovo sprotno urejanje, dodajanje in brisanje.

- Ko delavci opravijo izpit, je potrebno vnesti podatke o opravljenem usposabljanju. Vnese se le uspešno opravljena usposabljanja in sicer datum ter osebo. Prvo izberemo stranko, določimo datum opravljanja usposabljanja, nato obkljukamo delavce, ki so uspešno opravili usposabljanje.

Modul Stroji

Podobno kot bo imel uporabnik sistema za vsakega delavca možnost vpogleda v kartoteko in bo na podlagi nje izvajati različne aktivnosti, tako bo to lahko počel tudi pri strojih. V vsakem podjetju bo na voljo seznam strojev in pri vsakem bo moč preko "kartona"natančneje pregledovati njegove podatke ter določati bodoče aktivnosti. Na voljo bo tudi zgodovina dogodkov za posamezni stroj.

Poglejmo, kakšne so podrobnejše zahteve pri tem modulu.

- Možna naj bo izdelava zapisnikov.
- Možen mora biti izpis strojev pri določeni stranki.
- Za vsak stroj mora biti na razpolago seznam vseh do sedaj narejenih pregledov.
- En zapisnik vsebuje podatke o pregledih strojev.
- Sistem naj omogoči pregled in urejanje ter neposreden vnos podatkov pregledanih strojev iz modula za pregled strojev. Iz tega modula bo razvidno, katerim strojem smo podelili potrdilo o brezhibnosti, kje se nahajajo in kaj se je s strojem dogajalo v preteklosti (npr. sprememba lokacije).

Modul Evidenca pogodbenih strank

V tem modulu bo uporabnik lahko dostopal do seznama podjetij, torej pogodbenih strank. Ta modul bo po načinu dela zelo podoben modulom Delavci in Stroji. Za vsako stranko bo uporabnik sistema lahko analiziral podatke in se odločal glede prihodnjih akcij ter imel vpogled v zgodovino posamezne pogodbene stranke.

- Nastavi se kartoteka strank, kjer bodo razvidni podatki:
  - osnovni podatki podjetja,
  - vrsta pogodbe,
  - naloge, ki jih moramo zagotavljati,
  - možni so izpisi strank po različnih kriterijih (določeni nalogi, specifičnih zahtevah stranke).
- Sistem omogoči elektronsko komunikacijo z vsemi strankami hkrati ali samo z določenimi preko elektronske pošte.
- Vsaka stranka ima oznako kvalitete stranke (redni plačniki, kompliciranе stranke, neplačniki).

Modul Izjava o varnosti

Ta modul bo uporabnikom omogočal hitro izdelavo dokumenta o oceni tveganja. To je dokument, ki je namenjen opredelitvi vrst nevarnosti in stopnje tveganja na delovnih mestih ter določitvi varnostnih ukrepov za preprečevanje poškodb in okvar zdravja.

Modul naj omogoči:

- preglednost in združljivost enakih dejavnosti,
- poenotenost vrst nevarnosti,
- poenotenje ukrepov za enaka dela,
- sistem za obveščanje o spremembah med uporabniki sistema,
- uporabnik s primernimi pravicami, dodeljenimi s strani administratorja, naj ima pregled nad vsemi spremembami, ki so se zgodile v kateremkoli modulu, na kateremkoli dokumentu,
- obstajati mora primeren in učinkovit način avtomatskega obveščanja o določenih spremembah v določenih modulih, na določenih dokumentih,
- za vsak dokument v vsakem modulu mora obstajati zgodovina sprememb, dostopna vsakemu uporabniku,
- vsak uporabnik bo imel svoj uporabniški račun (delovni prostor je enak za vsakega uporabnika, le nastavitve uporabnika se shranjujejejo,
- filtriranje po različnih postavkah,
- preprečitev različne interpretacije oziroma izdelke za enaka delovna mesta.

3.2.2 Podporni moduli

Podporni moduli so tisti moduli, ki le posredno služijo končnim ciljem sistema in katerih rezultati niso merljivi, niti niso temu namenjeni. To so moduli, s katerimi le podpiramo delovanje prvih petih modulov, ki dejansko neposredno služijo končnim ciljem sistema.

Spodnji seznam nakazuje le nekaj podpornih modulov, za katere se že v tej fazi zavedamo, da bodo potrebni, saj so pri spletnih aplikacijah nekaj popolnoma standardnega:

- Uporabniški račun (urejanje uporabniških nastavitev trenutno prijavljenega uporabnika),
- Administracija (mesto, kjer uporabniki administratorji urejajo vse, kar zadeva ostale uporabnike in nastavitve celotnega sistema),
- Media (modul, kjer bo možno urejati medijske vsebine, predvsem slike).

Potreba po dodatnih podpornih modulih se bo pokazala med samim razvojem in preizkušanjem sistema, zato lahko sklepamo, da zgornji seznam ni dokončen.

V naslednjem poglavju si oglejmo zahteve za posamezen podporni modul, čeprav se je potrebno zavedati, da je pri takih modulih težko vnaprej dovolj natančno predvideti vse zahteve.

Uporabniški račun

Ta modul bo omogočal spreminjanje vseh vnaprej določenih nastavitev uporabnika, kot je profil z osebnimi podatki, sliko (avatar) in geslo.

Uporabnik naj ima vpogled v zgodovino dogodkov, ki si z njim povezani. V okviru tega naj ima uporabnik možnost ogleda zgodovine dogodkov v obliki tabele in pa zgodovino vseh prejetih ter poslanih sporočil.

Vsak uporabnik bo imel svoj delovni prostor v sistemu, ki bo prikazan s pomočjo grafičnega uporabniškega vmesnika, njegova grafična struktura pa

bo za vsakega uporabnika enaka, le podatki se bodo od uporabnika do uporabnika spreminjali. V delovnem prostoru bo uporabnik odpiral module. Ena izmed komponent delovnega prostora uporabnika bo zadolžena za prikaz za sporočila, ki je prispelo s strani sistema ali drugega uporabnika (sodelavca).

Primeri, v katerem nek uporabnik prejme obvestilo:

- nekdo spremeni dokument, katerega sam ni avtor,
- avtor dokumenta je obveščen.

Administracija

Ta modul bo dostopen samo uporabnikom sistema, ki imajo dodeljeno vlogo administratorja.

Administrator naj ima možnosti:

- pregledovanja seznama obstoječih uporabnikov,
- dodajanja novih uporabnikov,
- brisanja obstoječih uporabnikov,
- dodajanja in urejanja vlog in pravic ostalih uporabnikov.

Media

- Določeni moduli sistema naj vsebujejo dodajanje slik in drugih različnih medijskih datotek. V tem modulu naj ima uporabnik možnost dostopa in urejanja le-teh.
- Uporabniku naj bo omogočen dostop do vseh medijskih datotek, ki jih je bodisi sam naložil v sistem, bodisi so jih v sistem naložili drugi uporabniki sistema, ki so v isti skupini.

- Uporabnik naj ima možnost nalaganja medijskih datotek iz svojega računalnika.

- Nalaganje datotek naj bo možno tudi na način "povleci in spusti" (drag-and-drop).

- Uporabnik naj ima možnost pregledovanja vseh medijskih datotek v "knjižnici" medijskih datotek.

3.3 Načrtovanje arhitekture sistema

V tem diplomskem delu smo do te točke precej natančno opisali zahteve celotnega sistema, pa vendar je za kakovosten razvoj tega sistema potrebno še nekaj več. Potrebno je načrtovanje arhitekture, saj smo do sedaj spoznali šele, KAJ bo sistem potreboval. Jasno je torej, da bo pred dejansko izvedbo sistema sedaj potrebno formalizirati tudi to, KAKO bomo izvedli zahteve, ki smo jih zajeli in uredili v poglavjih 3.1 in 3.2.

Pred dejanskim pričetkom razvoja, bomo predstavili načrt arhitekture sistema. Govorili bomo o tem, kako bomo programski del celotnega problema razbili na več manjših, programskih problemov oziroma nivojev, te pa še dalje razdelali na komponente in module oziroma "področja" - če gledamo strogo z vidika Microsoftovega ogrodja MVC4.

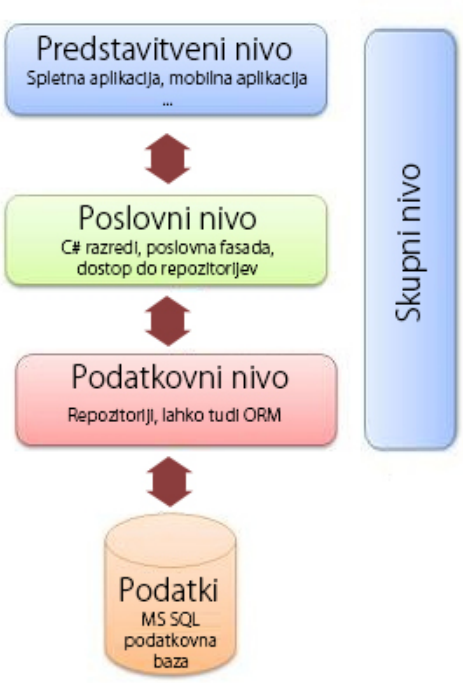
Zaradi prednosti, ki jih ločevanje programskih problemov na nivoje prinaša trinivojska arhitektura, smo se odločili, da se bomo pri izdelavi sistema držali arhitekturnega vzorca n nivojev (angl. N-Tier), v našem primeru treh, kjer višje oštevilčeni nivo nudi storitve nižje oštevilčenemu:

1. predstavitveni,

2. poslovni in

3. podatkovni.

V tej arhitekturi sicer ponavadi nastopa še četrti, skupni nivo. Ta se vertikalno "razteza" preko vseh treh, saj vsebuje storitve, ki se jih lahko poslužuje vsak od glavnih treh nivojev. To je razvidno tudi na sliki 3.1.



Slika 3.1: Trinivojska arhitektura z vertikalnim skupnim nivojem.

Že od samega začetka smo se dobro zavedali, da bo razvita programska rešitev kompleksna aplikacija in da bo takšna razdelitev programskih problemov na različne nivoje nujna, če bomo želeli izdelati uspešen informacijski sistem, ki bo enostaven za vzdrževanje, testiranje in nadgradnjo. Poleg tega nismo natančno vedeli, kam nas bo razvoj popeljal v prihodnosti. Zato smo se odločili za tako arhitekturo, katerih nivoji (in tudi komponente znotraj leteh) so lahko med seboj neodvisni in nam z lahkoto dopuščajo spremembe.

Pri večnivojski arhitekturi nam taka šibka povezanost posameznih nivojev in programskih komponent omogoča zmanjševanje kompleksnosti posameznega omenjenega elementa, saj lahko celoten problem razbijemo na več

posamičnih, lažje obvladljivih elementov (nivojev, komponent), ki so same po sebi lahko tudi samostojne in prenosljive aplikacije ali celo programski vmesniki. Ko aplikacijo razdelimo na podatkovni, poslovni in predstavitveni nivo, s tem poenostavimo vzdrževanje celotne aplikacije, med drugim tudi zato, ker pri vsaki komponenti natanko vemo, kaj slednja vključuje pri trinivojski arhitekturi za vsakega od nivojev tudi ni važno, v kateri tehnologiji je izdelana ali na kateri platformi temelji, kar je nedvomno velika arhitekturna prednost in dobra naložba za prihodnost.

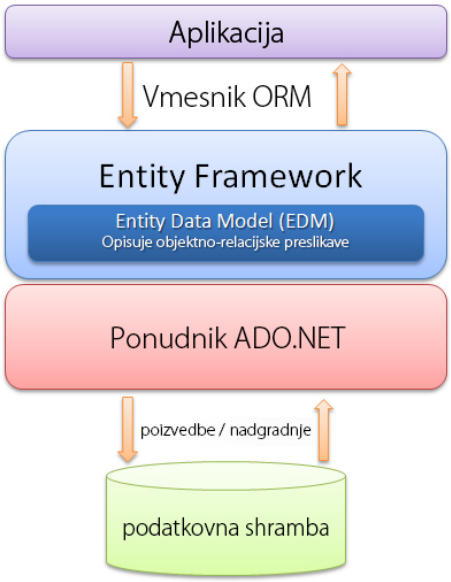
Posamezne nivoje bomo podrobno razdelali v nadaljevanju, izpostavili pa bomo tudi novosti in prednosti sistema, ki so jih strokovni delavci na področju varnosti pri delu in požarne varnosti v obstoječih programskih rešitvah do sedaj najbolj pogrešali.

3.3.1 Podatkovni nivo

V realiziranem sistemu bi podatkovni nivo lahko črpal in obdelovala podatke iz MySQL, MSSQL ali katerekoli druge relacijske podatkovne baze. Podatke bi nenazadnje lahko pridobivali tudi preko spletnih servisov in formatov za izmenjavo podatkov, kot sta XML in JSON, ali bi uporabljali danes vedno bolj priljubljene podatkovne baze NoSQL. Vse to pa se na višjem, torej poslovnem nivoju, ne bi čisto nič poznalo, saj bi za to poskrbelo primerno orodje ORM.

Vemo, da obstaja veliko različnih podatkovnih baz, v katerih lahko hranimo podatke in z njimi "napajamo"našo programsko rešitev. Neposredno dostopanje do podatkov v podatkovni bazi s poizvedbami SQL lahko zaradi svoje kompleksnosti zelo oteži in upočasni programerjevo delo na višjem nivoju arhitekture ter zmanjša produktivnost. Podatki, ki jih na tak način dobimo, pa so lahko v nepreglednih seznamih, v obliki spremenljivk primitivnih podatkovnih tipov.

Na podatkovnem nivoju se je torej pojavila potreba po dodatni abstrakciji med fizično podatkovno bazo in poslovnim nivojem, ki omogoča delo s podatkovnimi objekti, namesto s surovimi, netipiziranimi seznami podatkov.



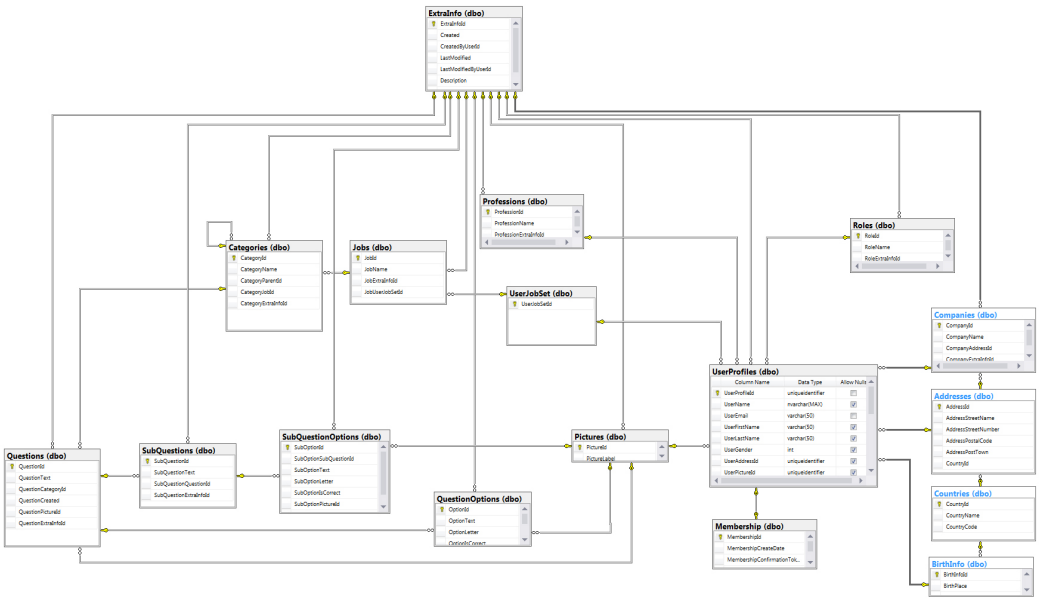
Slika 3.2: Ogrodje ADO.NET Entity Framework.

Namesto, da iz fizične podatkovne baze dobimo sezname podatkov primitivnih tipov, nam nivo ORM sedaj ponuja podatkovne objekte, torej tipizirane podatke. Preprost primer tega je lahko preslikava ene vrstice v fizični podatkovni tabeli "Delavec"v instanco podatkovnega tipa (ali modela) "Delavec".

Konkretno bo naš sistem črpal podatke iz podatkovne baze MSSQL s pomočjo orodja ORM, v našem primeru odportokodnega Microsoftovega ogrodja ADO.NET Entity Framework (v nadaljevanju entitetno ogrodje), kot prikazuje slika 3.2. Entitetno ogrodje je v .NET skupnosti najbolj priljubljena in samoumevna izbira ogrodja ORM.

Na en in isti podatkovni nivo se tako lahko priključi poslovni nivo, ki lahko temelji bodisi na ASP.NET, Android ali kateri drugi platformi. Pomembno je le, da izberemo primerno orodje ORM, kompatibilno s platformo tako na višjem, kot tudi nižjem nivoju arhitekture. V našem primeru smo se pri implementaciji take arhitekture, zaradi zrelosti, odločili za uporabo Microsoftovih programskih orodij.

Podatkovni modeli



Slika 3.3: Fizični podatkovni model.

Na podlagi zajema zahtev lahko sedaj identificiramo vse entitetne tipe, ki bodo v sistemu nastopali, vse njihove atribute in definiramo njihove medsebojne relacije. Tako bomo lahko ustvarili konceptualni podatkovni model, iz njega logični model, vse dokler ne bomo ustvarili najpomembnejši programski temelj našega sistema - fizično podatkovno bazo.

Orodje ORM, ki ga imam na razpolago, nam ponuja več načinov dela s podatki. S pomočjo razvojnega okolja lahko z vgrajenim vizualnim orodjem najprej izdelamo logični podatkovni model. Postopki so standarni. Najprej iz sveta oziroma poslovne domene ustvarimo mentalni model, iz njega izdelamo konceptualni model, ki ga nato v omenjenem vizualnem orodju enostavno narišemo in opremimo z vsemi atributi oziroma meta podatki. EF nato avtomatsko poskrbi tako za izdelavo fizične podatkovne baze, kot tudi programskega konteksta, v katerem so vsi potrebni razredi in metode za delo z entitetami entitetnega ogrodja. Temu pristopu pravimo “najprej model”

(angl. Model First).

Druga možnost, ki je v zadnjem času precej priljubljena med programerji, predvsem tistimi, ki imajo opravka z agilnimi metodologijami, je pristop, imenovan "najprej koda" (angl. Code First). Že samo ime pove, kako poteka ustvarjanje podatkovnega modela. Najprej izdelamo razrede, ki jim v rečemo tudi modeli. Dejansko lastnoročno izdelamo podatkovni kontekst tako, da razrede napišemo na roke in tako v praksi modeliramo končni podatkovni model. Nato zaženemo EF, ta pa za nas avtomatsko ustvari tako logični model, kot tudi fizično podatkovno bazo.

Pristop, ki ga bomo izbrali pri izdelavi našega sistema, bo bolj tradicionalen, a še vedno bomo pri tem uporabili entitetno ogrodje. Pristop se imenuje "najprej podatkovna baza" (angl. Database First) in sledi standardnemu postopku svet - mentalni model - konceptualni model - logični model - fizični model. Čeprav gre pri uporabi entitetnega ogrodja ORM za manjše variacije opisanega postopka, se ta ponavadi začne na papirju ali na tabli in se pred izdelavo kode in modela entitet entitetnega ogrodja zaključi z izdelavo fizične podatkovne baze v kakem vizualnem orodju, kot ga vsebuje razvojno okolje Visual Studio 2012 ali MSSQL Server Management Studio. Ko imamo natančno izdelan fizični podatkovni model, lahko začnemo s pisanjem kode. Razlog, da smo se odločili za ta pristop, je zgolj eksperimentalne narave. Načeloma bi se lahko odločili tudi za prvi ali drugi opisani pristop.

V nadaljevanju si poglejmo najprej podatkovne modele, ki so nas pripeljali do fizičnega podatkovnega modela, začenši s konceptualnim modelom.

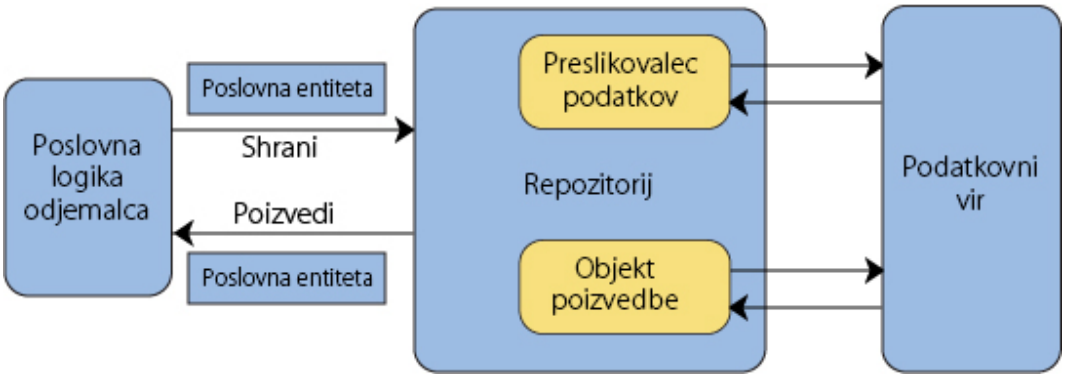
Iz teorije vemo, da obstaja več različnih tehnik konceptualnega načrtovanja, a smo pri tem, zaradi obsežnosti problema in danega časovnega okvira, nekoliko improvizirali. Do entitetnega modela smo prišli tako, da smo najprej določili oziroma poimenovali vse entitetne tipe. Standarden začetek torej. Ko smo bili prepričani, da so zbrani vsi tipi, smo jih skušali povezati s povezavami - relacijami. Skratka, določili smo njihova razmerja. Nato smo določili še atribute in konceptualni model predstavili naročniku.

Ko smo z naročnikom uskladili še zadnje detajle okrog entitetno-relacijskega

modela, smo se na podlagi načrtovanega modela lotili ročnega izdelovanja podatkovne baze z vizualnim orodjem MSSQL Management Studio. Nastal je fizični podatkovni model sistema (slika 3.3), ki pa še ne vsebuje vseh entitet oziroma tabel, saj bomo v prihodnosti dodali še druge module.

Vzorec repozitorijev

Repozitoriji [5] so tiste komponente (razredi) podatkovnega nivoja, ki poslovnemu nivoju nudijo storitev pridobivanja podatkov. Delujejo po principu črne škatle, saj repozitoriji skrijejo podrobnosti o tem, kako se podatki pridobivajo. Pomembno je tisto, kar kot rezultat dobimo iz repozitorija. Rezultat so pravzaprav poslovne entitete, s katerimi v logiki odjemalca ves čas delamo. To so podatkovni objekti oziroma zbirka objektov, pri katerih nam na višjem nivoju pravzaprav ni potrebno vedeti, kako, kje oziroma na kakšen način smo podatke teh objektov pravzaprav hranili. Ni nam potrebno niti vedeti, kako je realizirana podatkovna baza niti to kakšne so poizvedbe. Opisan vzorec ponazarja shema na sliki 3.4.



Slika 3.4: Princip delovanja repozitorija.

Ko želimo iz podatkovne baze pridobiti na primer eno vrstico iz neke fizične podatkovne tabele, uporabimo temu namenjeno, vnaprej načrtovano metodo repozitorija, ki se nanaša na ciljno oziroma želeno entiteto (tabelo) v podatkovni bazi. Tako si lahko vrstico neke podatkovne tabele enostavno

predstavljamo kot en objekt iz zbirke objektov, kjer vsak objekt predstavlja eno vrstico v podatkovni tabeli.

V primeru, da med programiranjem, na poslovnem nivoju, ugotovimo, da nek repozitorij nima želene metode za pridobitev podatkov, lahko oziroma celo moramo slednjo dodati v repozitorij, torej na nižjem, podatkovnem nivoju in jo na višjem, poslovnem nivoju samo poklicati. Bistvo povedanega je to, da si v nekaterih primerih sicer lahko dovolimo takšo ad-hoc spremembo (ni sicer zelo zaželjeno), vendar moramo vedno paziti, da se držimo konvencij, ki nam jih narekujejo nivoji in v vsakega postaviti samo tisto, kar tja spada. Tako ohranimo red in skrbimo za lažje vzdrževanje.

Naj še poudarimo, da si pri repozitorijih lahko enostavno pomagamo, če imamo na voljo določen osnovni razred (angl. base class), katerega implementirajo vsi repozitoriji. S tako hierarhijo razredov se izognemo podvajanju kode. Poslužimo se lahko tudi vmesnika (angl. interface), če želimo programerja prisiliti k uporabi določenih, vnaprej zastavljenih metod ob implementaciji posameznega repozitorija.

Vzorec repozitorijev je pravzaprav zelo pogosto uporabljen način, s katerim se izognemo podvajanju logike za dostopanje do podatkovne baze. V repozitorijih do podatkovne baze dostopamo preko tako imenovanega podatkovnega konteksta. V primeru našega sistema smo se odločili, da bo naš podatkovni kontekst pravzaprav zbirka entitet, ki jih bo iz fizičnega podatkovnega modela sestavilo orodje Entity Framework ORM [6]

Uporaba repozitorijev nam preko podatkovnih entitet orodja ORM enostavno omogoča dostop do podatkov iz baze s poizvedbami, pri katerih nam ni potrebno vsakič podajati ključnih informacij za dostop do baze, kot so na primer naslov podatkovne baze in uporabniški podatki za dostop do baze. Poleg vsega tega pa nam uporaba repozitorijev omogoča boljše testiranje aplikacije.

3.3.2 Poslovni nivo

Poslovno nivo bodo sestavljale komponente oziroma natančneje razredi, ki bodo za svoje delovanje uporabljale dostop do repozitorijev, ki povezujejo podatkovni in poslovni nivo.

Naloga poslovnega nivoja je v našem primeru zgolj in samo ta, da razbremeni tisti del predstavitvenega nivoja, ki je dejansko stičišče med predstavitvenim in poslovnim nivojem - govorim o nadzornikih MVC.

Uporabili bomo priljubljen koncept poslovne fasade (angl. Business Facade), ki predstavlja še dodaten nivo abstrakcije v poslovnem nivoju trinivojske arhitekture. Včasih se res zdi, da je drobljenje na tolikšne nivoje morda neproduktivno, a pri bolj kompleksnih sistemih lahko na tak nalin poenostavimo implementacijo, saj se na nivoju nadzornikov MVC lahko za pridobivanje podatkov zanesemo zgolj in samo na poslovno fasado.

V poslovni fasadi lahko natančno določimo, kaj bomo ponudili dalje našim nadzornikom MVC in tako slednje bistveno razbremenimo. Naša koda v nadzornikih MVC bo na takšen način manj obsežna, bolj sistematična in lažje berljiva.

V naši poslovni fasadi imamo zopet nek urejen sistem razredov. Za vsako entiteto v logičnem podatkovnem modelu imamo v poslovni fasadi skladen razred, kjer določamo, kaj bomo ponudili na razpolago nadzornikom MVC na predstavitvenem nivoju. Če poenostavimo, lahko rečemo, da poslovni nivo predstavitvenemu nudi storitev poslovne fasade.

3.3.3 Predstavitveni nivo

Predstavitveni nivo smo še dodatno razbili z uporabo vzorca MVC in programskega ogrodja ASP.NET MVC4. S slednjim bi pravzaprav lahko dejansko izdelali celoten informacijski sistem, brez uporabe trinivojske arhitekture, saj bi v ogrodju ASP.NET MVC4 lahko pokrili vse komponente te arhitekture, a bi to na dolgi rok pomenilo težko vzdrževanje, ker bi bile programske komponente samega ogrodja MVC (predvsem nadzorniki MVC) preveč kom-

pleksne.

Predstavitveni nivo trinivojske arhitekture vsebuje tako zaledni del kot tudi obličje sistema. To lahko vidimo v vzorcu MVC, ki je prisoten v spletnem programskem ogrodju ASP.NET MVC4. Prav ta nam tudi dejansko omogoča, da lahko programsko rešitev pravzaprav izdelamo tudi brez uporabe večnivojske arhitekture, saj bi ogrodje MVC4 zlahka pokrilo vse nivoje. Ker pa je načrtovan sistem precej kompleksen, smo se raje odločili za njegovo razgradnjo s pomočjo n-nivojske arhitekture.

Bistvo je, da je v vzorcu nadzornik MVC nadzornik izrazit predstavnik zalednega dela in z njim oblikujemo oziroma polnimo predhodno definirane modele (podatkovne strukture), ki se prikazujejo v pogledih kot predstavnikih obličja v predstavitvenen nivoju.

Vzorec MVC pa ni edini vzorec, ki smo ga uporabili pri izdelavi predstavitvenega nivoja sistema. Izključno na strani odjemalca se namreč poslužujemo tudi model-pogled-pogled modela (angl. Model-View-ViewModel ali krajše MVVMM), ki je namenjen zgolj enostavnejšemu pristopu izdelovanja grafičnega uporabniškega vmesnika. Vzorec MVVM prav tako centralizira nadzor nad podatki in bistveno zmanjša količino kode na strani odjemalca.

Mogoče lahko za konec tega podpoglavja omenimo še to, da bomo v kombinaciji z vzorcem MVVM, z namenom optimizacije kode, uporabili tudi predloge JavaScript (angl. JS templates), kar nam bo omogočala knjižnica Knockout[10].

3.3.4 Skupni nivo

Skupni nivo je le dodaten, vertikalen nivo, od koder ostali nivoji trinivojske arhitekture črpajo podatke oziroma se poslužujejo njegovih storitev. Je tudi nivo, kamor ponavadi vključimo komponente sistema, ki neposredno ne služijo samo enemu nivoju, pa čeprav bi take komponente lahko postavili tudi v enega od glavnih nivojev. Tak primer je komponenta, ki vsebuje rezultat uporabe orodja ORM - podatkovne entitete, se pravi entitete podatkovnega

modela, ki so pravzaprav abstrakcija podatkovne baze.

Ko smo iz fizičnega podatkovnega modela z orodjem ORM izdelali logični podatkovni model in obenem tudi podatkovni kontekst, smo se morali odločiti, kam v našo arhitekturo ga bomo umestili. Sprva je bila odločitev samoumevna, in sicer ta, da omenjeni podatkovni kontekst umestimo v podatkovni nivo. To bi po pravilih oziroma konvencijah trinivojske arhitekture pomenilo, da bi do podatkovnega konteksta imele dostop zgolj komponente poslovnega nivoja. Tako bi v podatkovnem nivoju, poleg definicij repozitorijev, imeli še podatkovni kontekst. Pa vendar smo kmalu spoznali, da bomo v predstavitvenem nivoju, v nekaterih primerih občasno potrebovali tudi dostop do podatkovnih modelov oziroma razredov, ki so na voljo v podatkovnem kontekstu.

Poglavje 4

Informacijski sistem Helmet IS

V tem poglavju skušamo predstaviti rezultat oziroma vsaj del rezultata tega diplomskega dela, izdelanega na podlagi obsežne analize in načrtovanja, predstavljenega v prejšnjih poglavjih. Razvili smo informacijski sistem za podporo strokovnim delavcem za varnost pri delu, ki smo ga poimenovali Helmet IS ali krajše HIS. Pojasnili bomo delovanje sistema HIS z zornega kota končnega uporabnika, namen modulov in zakaj sistemu HIS rečemo lahko tudi ogrodje.

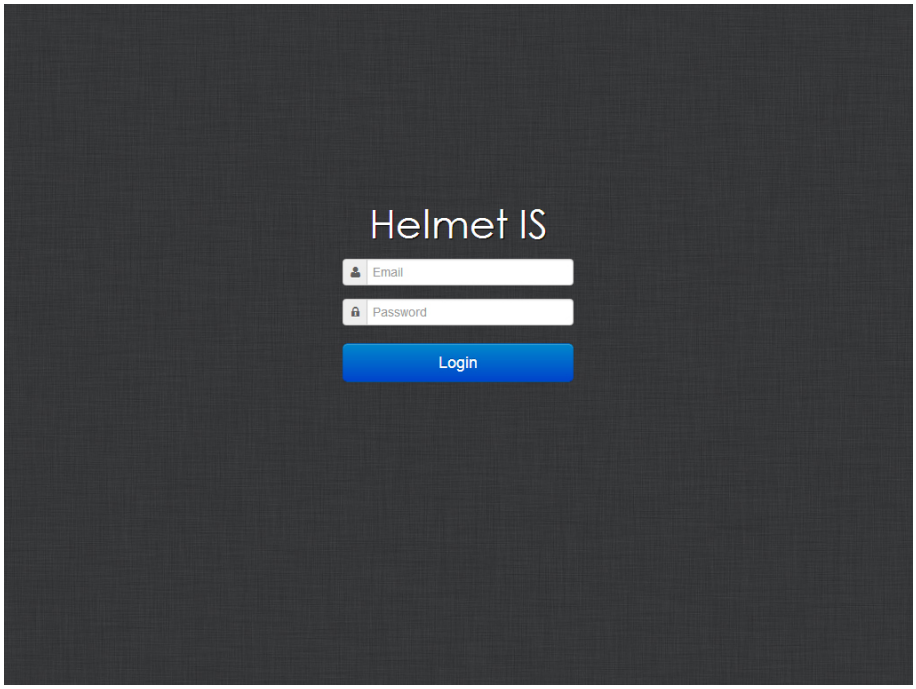
Ko bo govora o modulih, bo glavnina razlage pripadla modulu "Usposabljanje", saj je v okviru te diplomske naloge to več kot zadosten kontekst, s pomočjo katerega bomo bralcu lahko dovolj pojasnili delovanje celotnega ogrodja, se pravi celotnega informacijskega sistema.

Preden pa natančneje opišemo nastali sistem kot rezultat diplomskega dela, moramo prej spregovoriti tudi o tem, kako smo si zamislili varnost in prijavo uporabnika v sistem.

4.1 Varnost in prijava v sistem

Z naročnikom še nismo prišli do končne odločitve o tem, od kje in na kakšen način se bodo uporabniki prijavili v sistem. Nismo še določili, kam bomo sistem postavili. Obstaja več možnosti, ki jih bo potrebno dobro preučiti in se na koncu odločiti za tisto, ki bo, ob upoštevanju marketinških in varnostnih zahtev najbolj ustrezalo naročnika.

Ne glede na izbiro lokacije strežnika, kjer bo živel naš sistem, gre tu v vsakem primeru za spletni strežnik s podporo SSL. To je zelo pomembno, saj sistem uporablja spletne obrazce in poizvedbe tipa POST. Eden izmed obrazcev, s katerim se bo uporabnik srečal najmanj enkrat, je prijavno okno, kjer bo moral uporabnik vnesti svoje uporabniško ime in geslo (slika 4.1).



Slika 4.1: Prijava v sistem preko spletnega brskalnika.

Vnešeni podatki v spletnih obrazcih se preko spletnih zahtev prenašajo po omrežju, kjer so zelo ranljivi. Ker gre tu za občutljive podatke tako o uporabnikih kot tudi o podjetju, je seveda potrebno poskrbeti za primerno zaščito,

v tem primeru z enkripcijo podatkov, kar ponuja protokol SSL. Dodatna zaščita, za katero bo najvernetneje potrebno še poskrbeti, so tudi certifikati, seveda v odvisnosti komercialno strateške odločitve naročnika o tem, kako bomo uporabnikom sistem ponujali oziroma kje bo sistem nameščen.

4.2 Trenutna lokacija sistema in izbira lokacije spletnega strežnika na dolgi rok

V našem primeru je HIS zgolj v predstavitvene namene trenutno postavljen na javno dostopni spletni lokaciji, ki seveda ni končna, torej ni produkcijska, saj se bo z naročnikom potrebno še dogovoriti o primerni strategiji, ki zadeva lokacijo in tudi o drugih marketinških strategijah. Poleg tega je potrebno najprej preizkusiti trenutni prototip sistema in njegove izboljšave, dokler ne dobimo končnega produkta. Uporabnik, ki ima dodeljeno uporabniško ime in geslo, lahko v sistem trenutno vstopi tako, da vnese v spletni brskalnik primeren spletni naslov sistema in svoje uporabniško ime ter geslo.

Se bo pa v prihodnje trenutna lokacija zagotovo spremenila. Poglejmo si nekaj možnosti o morebitnih končnih lokacijah produkcijskega spletnega strežnika, na katerem bo nameščen sistem HIS. V dogovoru z naročnikom bomo eno od teh možnosti uporabili v prihodnosti. Dotaknili se bomo tudi varnosti in skušali pretehtati tako prednosti kot slabosti vseh predlaganih možnosti.

4.2.1 Lokalni strežnik podjetja s povezavo VPN

Kot že omenjeno, je ena možnost ta, da sistem postavimo na lokalni spletni strežnik podjetja naročnika, torej omogočimo prijavo uporabnikov v sistem samo lokalno. V primeru, da uporabniki fizično niso prisotni v okvirih lokalne mreže, lahko uporabnik vzpostavi povezavo VPN do lokalnega omrežja v podjetju. To pomeni, da morajo biti uporabniki oziroma njihovi odjemalci del lokalnega omrežja, pa naj si bo to fizično na lokaciji ali preko VPN pove-

zave. Načeloma bi uporabniki v veliki večini primerov dostopali do sistema HIS samo v svojih pisarnah, v redkih primerih pa tudi preko VPN. Seveda tu ni izključen niti preprost oddaljeni dostop do namizja (angl. Remote Desktop), če predpostavimo, da politika podjetja to dovoljuje.

V vsakem primeru bi bilo primerno in priporočljivo, da ima vsak odjemalec, ki bi želel dostopati do spletnega strežnika, nameščen primeren certifikat. Lahko pa, če se podjetje tako odloči, v tem primeru izpustimo certifikate in se opremo samo na zaščito SSL, brez certifikatov in s prijavo v sistem z uporabniškim imenom in geslom.

Dobra stran tega načina postavitve sistema je, da lahko v primeru VPN povezave ali oddaljenega dostopa do namizja, izvajalec sistema HIS slednjega vzdržuje tudi na daljavo, vse dokler lokalno omrežje podjetja normalno deluje. Slabost je v tem primeru morebiten izpad mreže, ko oddaljen dostop ni mogoč, dokler se mreža ponovno ne vzpostavi. Tu je namreč nujno potrebna fizična prisotnost vzdrževalca. Druge dobre lastnosti tega pristopa se kažejo v večji varnosti, saj bi bil sistem dostopen samo lokalno. Vendar pa se tu ne moremo izogniti stroškom, ki jih prinaša vzdrževanje lokalnega omrežja in strežnika.

V tem primeru bi na enem izmed računalnikov v mreži postavili spletni strežnik in na njega namestili HIS. Vendar pa se iz zornega kota izvajalca takoj postavi vprašanje, kaj bi se zgodilo v primeru, ko bi še kakšno drugo podjetje želelo uporabljati enak sistem, torej novo instanco sistema HIS. Kar naenkrat bi imeli dva popolnoma enaka sistema, katera bi morali ločeno vzdrževati.

V naslednjem podpoglavju si poglejmo ta način postavitve.

4.2.2 Več namestitev sistema na različnih fizičnih lokacijah

Večkratna namestitev istega sistema na različnih fizičnih lokacijah je iz zornega kota izvajalca zagotovo slabost, saj bi v primeru, da imamo več naročnikov sistema, morali na več različnih lokacijah skrbeti za več različnih strežnikov,

vsakega za enega naročnika. Skratka, vsako podjetje bi imelo svoj sistem HIS, ki bi bil popolnoma ločen od ostalih. To bi bile kopije istega sistema na različnih lokacijah, z različnimi podatki, v enaki podatkovni strukturi. Takšna decentraliziranost vsekakor ne olajša vzdrževanja, zgodilo pa bi se zagotovo tudi to, da bi za vsako podjetje morali posebej izvajati integracijo podatkov v obstoječi podatkovni model in ga po potrebi celo spreminjati. Zagotovo bi se pojavila tudi potreba po spremembi v poslovni logiki.

Skratka, v primeru komercializacije in ponujanja sistema HIS večim podjetjem, bi se, zaradi stroškov postavitve in vzdrževanja, cena takega projekta bistveno povečala.

Naslednji dve podpoglavji rešujeta problem večih namestitev istega sistema, a odpirata nova zanimiva vprašanja.

4.2.3 Običajen spletni strežnik z zaščito SSL in certifikati

Sistem HIS lahko namestimo tudi na nek običajen spletni strežnik, tako, kot je izvedena trenutna testna verzija sistema. Lahko rečemo, da se pri tej postavitvi zelo približamo ponujanju storitev sistema v oblaku, ki jo bomo opisali v naslednjem podpoglavju, a bi morali zagotoviti še merila, ki jih naša trenutna postavitev sistema še ne zagotavlja.

4.2.4 Informacijski sistem kot servis v oblaku

Možnost, sistem da HIS postavimo v oblak v obliki storitve (angl. Software as a service ali SaaS), je zelo podobna prejšnji, a gre tu za novejši, širši in kompleksnejši koncept, ne le zgolj zakup spletnega strežnika in postavitev HIS nanj.

SaaS, čemur rečemo tudi programska oprema na zahtevo (angl. on-demand software), je model dostave programske opreme, kjer sta tako aplikacija kot njeni podatki postavljena na spletnem gostitelju, torej centralizirano. Pa vendar gre tu za širši pojem, ki se od prejšnjega primera razlikuje pred-

vsem v modelu ponujanja in zajemanja storitev ter v prenosu velikega dela različnih odgovornosti na ponudnika storitev.

SaaS je postal priljubljen model dostave programske opreme za mnoge poslovne aplikacije oziroma programske storitve.

Slabost tega načina je vezanost na internet, saj so internetne povezave včasih nezanesljive in bi morebiten izpad vsem uporabnikom popolnoma onemogočil dostop do sistema HIS, dokler se internetna povezava zopet ne bi vzpostavila. Lahko se zgodi tudi izpad samega strežnika, zato je potrebno izbrati kvalitetnega in odzivnega ponudnika gostovanja. To sicer niso tako pogoste nevarnosti, so pa za uspeh sistema HIS vsekakor nujno potrebne dobre preučitve.

Več podjetij naročnikov bi se tako lahko poslu̇ževalo storitev sistema HIS, postavljenega na istem strežniku. Podobno kot pri običajnem spletnem strežniku, opisanem v poglavju 4.2.1, bi vsako prijavljeno podjetje lahko ustvarilo svoj HIS račun in svojo ekipo, ki bi uporabljala storitve sistema na enak način, kot ekipe vseh ostalih podjetij. Verjetno pa tu ni treba posebej poudarjati, da je v tem primeru potrebna izjemna mera pazljivosti, da se slučajno ne bi zgodili incidenti pri izmenjavi podatkov med posameznimi ekipami in konkurenčnimi podjetji. V tem primeru bi morala biti varnost sistema prioriteta, potrebno pa bi bilo intenzivno testiranje programske opreme. To bi vsekakor povečalo stroške realizacije, bi pa to bil izjemen vložek v prihodnost samega sistema.

Naj na tem mestu bralca opozorimo, da trenutni fizični podatkovni model še ne predvideva, da bi eno in isto instanco sistema HIS uporabljalo več različnih konkurenčnih podjetij. Potrebna bi bila dodatna nadgradnja na praktično vseh nivojih arhitekture. Pa vendar bi se na dolgi rok postavitev sistema v oblak zagotovo obrestovala, če bi lahko pokrili investicijske stroške.

Vsak naročnik bi tako lahko na neko časovno obdobje ali za določeno količino plačeval za storitve sistema HIS, kar bi za izvajalca pomenilo pokritje stroškov in nenazadnje tudi zaslužek. Obenem bi tako naročnike razbremenili skrbi vzdrževanja in nadgradnje sistema. Ta skrb bi se seveda prenesla na

stran ponudnika SaaS.

Prednosti tega načina so v tem, da lahko uporabnik dostopa do sistema HIS od kjerkoli po svetu,čeprav je to možno tudi pri lokalni mreži in VPN dostopu (podpoglavje 4.2.1).

Seveda so slabosti z vidika uporabnika tu skoraj popolnoma enake, kot v primeru, opisanem v poglavju 4.2.3, a so to precej majhne slabosti v primerjavi s tistimi, ki se v primeru programja kot storitve prenesejo na izvajalca. Ta mora namreč skrbeti za nemoteno delovanje sistema, saj bi izpad občutila vsa podjetja.

Ko enkrat rešimo te probleme, pa postane vzdrževanje sistema lažje, četudi bi šlo za več naročnikov iste instance sistema HIS. To pa nam omogoči prav centralizacija. Poleg tega lahko sistem lažje ponudimo širšemu svetovnemu trgu, če le zadostimo zakonskim merilom posamezne države, v kateri sistem ponujamo.

Še ena slabost, ki jo velja omeniti, je ta, da mora vsako podjetje, ki želi uporabiti sistem HIS kot storitev v oblaku, popolnoma zaupati sistemu (zaradi narave dela in podatkov) in dandanes to doseči vsekakor ni mačji kašelj. Zavedati se moramo namreč, bi se v sistem HIS tako pretakali podatki podjetij, ki so lahko poslovna tajnost. Na strani izvajalca se kar naenkrat ustvari izjemna odgovornost, ki je primerljiva z odgovornostjo izvajalcev spletnih bančnih storitev. Vprašati se moramo tudi, koliko podjetij bi pravzaprav svoje poslovne podatke zaupalo nekemu novemu sistemu v oblaku.

4.3 Ogrodje ali zbirka modulov

Sistemu HIS lahko rečemo tudi delovno ogrodje strokovnega delavca. To lahko storimo iz preprostega razloga, ker skupini strokovnih delavcev oziroma uporabnikov, omogoča delo v nekem kontekstu, okviru, jih vodi in usmerja. Sistem Sistem HIS kot pravo ogrodje uporabniku preko spletnega brskalnika nudi nek omejen nabor možnosti oziroma orodij za delo in ga usmerja s pomočjo konvencij.

Našemu ogrodju, torej sistemu HIS, lahko z vidika uporabnika rečemo tudi zbirka modulov, katerih narava je tesno povezana s spletnim programsikm ogrodjem ASP.NET MVC4. Da bi lažje razumeli, kaj točno moduli so, se moramo spomniti o tem, kaj sestavlja ogrodje MVC, s pomočjo katerega smo realizirali predstavitveno nivo arhitekture sistema.

Zaradi boljše struktuiranosti programskih problemov in zaradi ločevanja odgovornosti, imamo v vsakem programskem ogrodju MVC, ne glede na platformo, tri enote: model, pogled in nadzornik. V programskem ogrodju ASP.NET MVC4 imamo na voljo še dodaten nivo ločevanja - področja (angl. Areas). Vsako področje vključuje svoj vzorec MVC, saj vsebuje mapo za modele, mapo za poglede in mapo za nadzornike.

Lahko rečemo, da se področje programskega ogrodja ASP.NET MVC4 preslika v modul. Drugačno poimenovanje za eno in isto stvar smo izbrali zgolj zato, da bomo lažje prepoznali, o katerem kontekstu se pogovarjamo. Ko govorimo namreč o področjih MVC, imamo v mislih pravzaprav predvsem zaledni del aplikacije (datoteke in kodo ASP.NET ogrodja MVC4), s katerim se uporabnik nikoli ne sreča neposredno, za razliko od programerja, ki s področji upravlja, ko programira in z njimi določa obliko modulov za uporabnika. Torej, ko govorimo o modulih, si lahko predstavljamo to, kar neposredno vidi in občuti uporabnik. Modul torej ni nič drugega, kot tisti element sistema HIS, ki je v neposrednem stiku z uporabnikom.

Včasih, ko bomo govorili o modulih, pa bomo uporabil tudi izraz pogled (iz vzorca MVC), čeprav bi bralca radi opozorili, da sta modul in pogled dve različni stvari. Modul ponavadi vsebuje več pogledov in so njegov sestavni

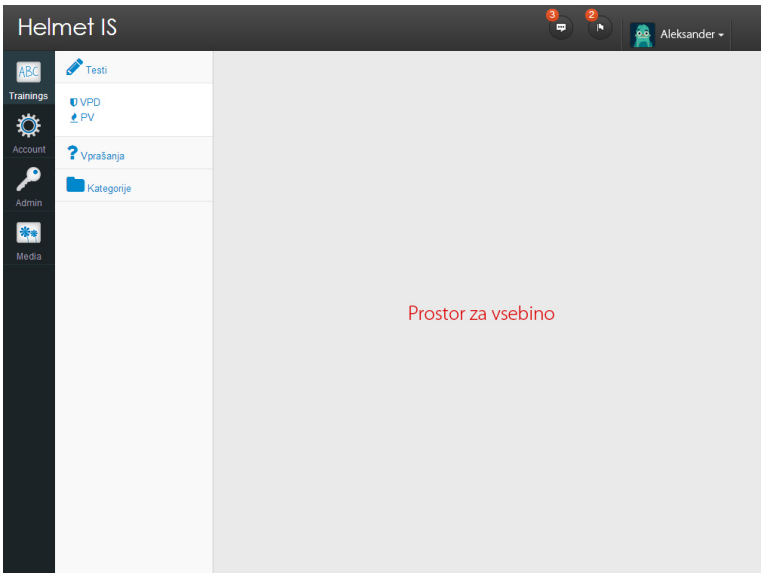
del. Nenazadnje si lahko uporabnik vsak pogled predstavlja kot grafični uporabniški vmesnik nekega modula.

4.3.1 Glavna razporeditev HTML elementov grafičnega uporabniškega vmesnika

Glavna razporeditev elementov HTML (angl. Layout), ki tvorijo grafični uporabniški vmesnik je sestavljena iz treh elementov:

- glavne navigacije,
- pomožne navigacije in
- vsebine.

Na sliki 4.9 je moč videti glavno razporeditev, ki je prisotna ne glede na to, v katerem modulu se kot uporabniki nahajamo. Skratka, vsak modul si deli glavno razporeditev, ki se z njihovo izbiro ne spreminja.

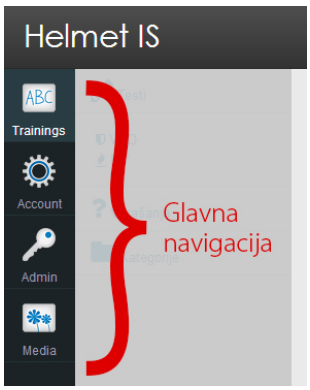


Slika 4.2: Glavna razporeditev HTML.

Vsebina modulov se ob izbiri želenega modula v glavni navigacji prikazuje na sredini zaslona, na označenem delu.

Glavna navigacija

Ob izbiri grafične predloge smo z naročnikom hoteli uporabniku ponuditi kar se da preprosto in enostavno navigacijo, ki uporabnika ne bi mogla zmesti. Glavna navigacija vsebuje povezave do vseh glavnih modulov sistema, predstavljenih v poglavju 3.2.1, od katerih ima vsak svoj predstavitveni simbol. Vsebuje tudi povezave do pomožnih modulov iz poglavja 3.2.2. Ko uporabnik izbere želeni modul, se na desni strani glavne navigacije prikaže primerna pomožna, kontekstna navigacija, v prostoru za vsebino pa izbrani modul oziroma njegov pogled.

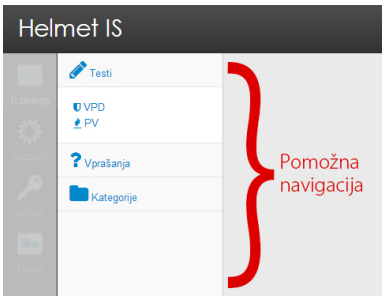


Slika 4.3: Glavna navigacija.

Zanimivost, ki jo tu velja omeniti, je dejstvo, da smo se ob izdelavi sistema skušali striktno držati konvencije, ki pravi, da vsak gumb glavne navigacije (torej vsak modul) predstavlja skladno področje v zalednem delu predstavitvenega nivoja aplikacije.

Pomožna oziroma kontekstna navigacija

Pomožna oziroma kontekstna navigacija (slika 4.9) se spreminja v odvisnosti od izbire modula v glavni navigaciji in vsebuje podmodule izbranega modula. Omogoča dostop do njegovih funkcionalnosti.



Slika 4.4: Pomožna navigacija.

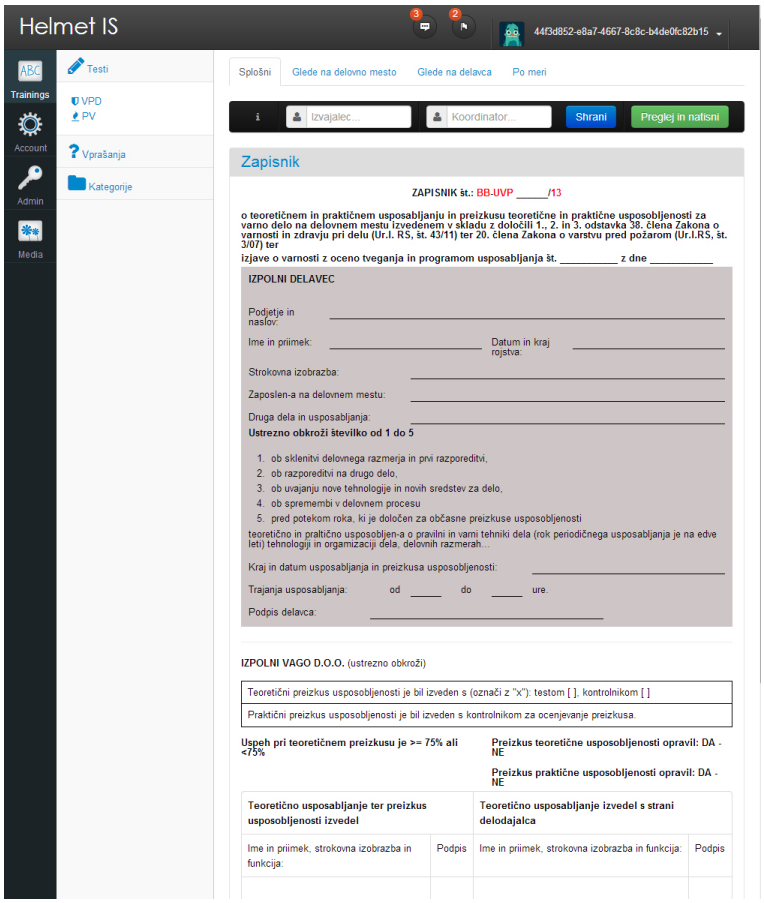
Vsebina

Vsebina je dejansko pogled MVC, ki predstavlja izbrani modul ali podmodul, in se spreminja na podlagi izbrane opcije v glavni ali pomožni navigaciji.

4.4 Modul Usposabljanje

V okviru diplomske naloge smo izdelali prototip modula Usposabljanje. Ta modul je med vsemi moduli sistema sicer morda najenostavnejši za realizacijo, vendar še vedno precej kompleksen. Izdelan prototip modula sicer še ne implementira celotnega nabora možnosti, a nam lahko kljub temu služi kot vzorec za ostale module. Preko njega bomo zagotovo bolje spoznali celotno ogrodje, pa čeprav se bodo posamezni moduli med seboj lahko bistveno razlikovali, tako v namenu kot v obliki.

Ob izbiri modula Usposabljanje v glavni navigaciji nam pomožna, kontekstna navigacija prikaže možnosti oziroma podmodule, ki so nam na razpolago v kontekstu modula Usposabljanje. V prostoru na zaslonu, namenjenem vsebini, se pokaže vsebina izbranega podmodula v pomožni navigaciji. V primeru na sliki 4.5 je pod vsebino prikazana predloga testne pole tipa VPD.



Slika 4.5: Splošni test tipa VPD, s prikazom zapisnika.

4.4.1 Testne pole

Prvi podmodul v pomožni navigaciji modula Usposabljanje vključuje izbiro dveh različnih tipov testnih pol - VPD in PV. V odvisnosti od tega, kakšno usposabljanje bo strokovni delavec izvajal, lahko ta izbere eno ali drugo predlogo. Če izbere tip VPD, se mu prikaže predogled testne pole za varnost pri delu, ki jo lahko po potrebi prilagaja oziroma spreminja nastavitve, preden testno polo natisne.

Enako bo v prihodnosti s tipom predloge PV, le da bo ta prilagojena usposabljanju delavcev za požarno varnost. Podmodul PV pa trenutno še ni implementiran, zato bomo v nadaljevanju opisali le tip testne pole VPD.

Ob izbiri tipa VPD se lahko uporabnik znotraj pogleda odloči za način izdelave testne pole. Na vrhu pogleda ima na izbiro jezičke, od katerih vsak predstavlja enega izmed naslednjih načinov izdelave testne pole:

- izdelava splošne testne pole,
- izdelava testne pole glede na delovno mesto in
- izdelava testne pole glede na delavca.

Poglejmo si sedaj, kaj vsak izmed teh načinov omogoča.

Izdelava splošne testne pole

Na vrhu zaslona, tik pod jezički za izbiro načina izdelave testne pole, ima uporabnik na voljo preprost razdelek za nastavitve, ki se odražajo v končni obliki splošne testne pole. Tu lahko uporabnik nastavi izvajalca teoretičnega dela in pa koordinatorja praktičnega usposabljanja. To stori tako, da v vnosna polja vnese imeni obeh akterjev. Obe nastavitvi se ob spremembi takoj in brez osveževanja pogleda, odražajo v predlogi testne pole, ki jo je moč videti pod nastavitvami.

Obe vnosni polji lahko sicer ostaneta prazni, saj uporabniku pred pregledom testne pole in tiskanjem ni potrebno nastavljati ničesar. V teoriji lahko uporabnik namreč zgolj prispe v modul in testne pole natisne, ne da bi v nastavitvah karkoli spreminjal. Tak primer bi se zgodil, če bi se strokovnemu delavcu mudilo in bi zanj bilo pomembno le tiskanje nekega števila splošnih testnih pol. Zanimivo bo v praksi opazovati, koliko bo dejansko takšnih slučajev. Ta možnost je tudi posledica dejstva, da je splošna testna pola vnaprej strogo določena s strani odgovornih v podjetju in se njena oblika posameznim uporabnikom kaže popolnoma enako, z izjemo nastavljivih polj, ki so odvisni tako od strokovnega delavca, kot tudi od narave usposabljanja.

Seveda bo imel uporabnik z dodeljenimi primernimi pravicami možnost urejanja tudi oblike splošne testne pole, saj se lahko tudi v slednji s časom

pojavi potreba po marsikateri spremembi. Lahko je to del zapisnika ali del z vprašanji. Dober primer je recimo sprememba zakona, ki povzroči spremembo v zapisniku. Tu pa ne smemo mešati med spremembo oblike testne pole in spremembo njenih nastavitev. Medtem, ko se bo vsaka sprememba oblike predloge testne pole odražala vsem uporabnikom enako, se nastavitve testne pole (v nastavitveni vrstici na vrhu pogleda) odražajo samo trenutno prijavljenem uporabniku.

V primeru, da uporabnik kakšno nastavitev (samo zase) spremeni , lahko spremembo shrani s temu namenjenim gumbom. Tako bodo nastavitve shranjene in bodo temu uporabniku takšne na voljo tudi ob naslednjem dostopu do modula.

Zanimiv podatek pri nastavitvenih poljih je ta, da slednja uporabljajo samodejno zaključevanje besedila, medtem, ko uporabnik vpisuje tekst. To je sicer vedno bolj pogosta praksa pri spletnh uporabniških vmesnikih in tudi nujna za tak sistem, kot je HIS, ki temelji tudi na prijaznosti do uporabnika.

Samodejno zaključevanje besedila smo izvedli s pomočjo knjižnice jQuery in njenih vtičnikov, v tem primeru vtičnika jquery.autocomplete. Ob tipkanju vsebine vnosnega polja se v ozadju izvajajo zahteve AJAX na strežnik, ki vračajo seznam podobnih vnešenih tekstovnih rezultatov.

Pod vrstico z nastavitvami je mesto za ključno vsebino. V obliki zgibnega menija se prikaže predogled testa, ki je sestavljen iz dveh delov: zapisnika in zbirke splošnih vprašanj.

Sam zapisnik je sestavljen iz naslednjih segmentov:

- nastavljiva številka zapisnika,
- fiksni pravni del,
- fiksni del, ki ga izpolni delavec pred pričetkom usposabljanja,
- fiksni del za rezultat izpita, ki ga izpolni podjetje po usposabljanju.

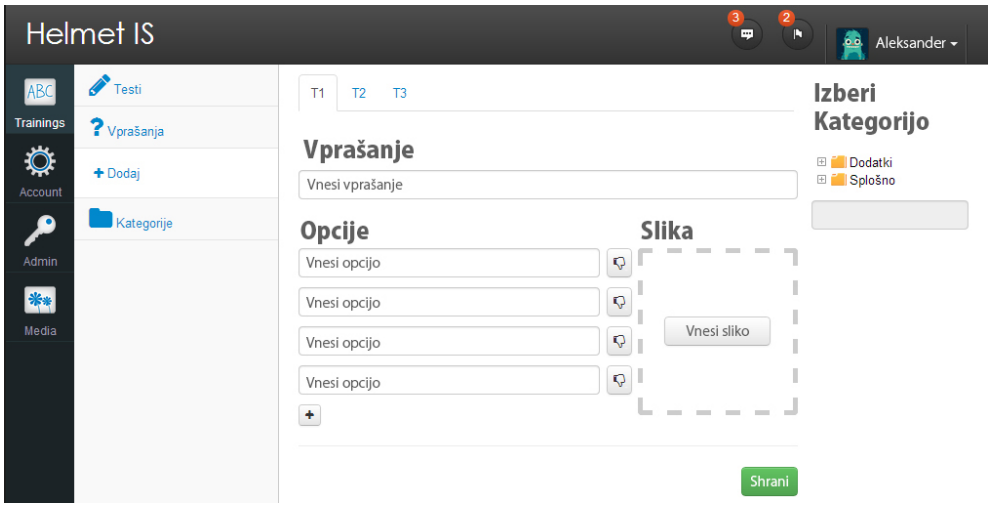
Zbirka splošnih vprašanj je sestavljena iz različnih kategorij ali področij,

od katerih je vsako prikazano v obliki zgibnega menija in od katerih vsako vsebuje določeno število vprašanj.

Uporabniku so v pomoč tudi zaslonski namigi, ki so del ogrodja Twitter Bootstrap[8] za obličje spletnih aplikacij. Z njimi smo želeli uporabniku še bolj olajšati uporabo sistema. Tako lahko ob premiku kurzorja miške na posamezne segmente, opazimo sporočila, ki uporabniku pomagajo pri razumevanju le-teh. V prihodnosti bomo sistem razširili še tako, da bomo na podoben način uporabniku približali sistem z uporabo vtičnikov za vodenje uporabnika skozi pogled (kot ga na primer uporablja vodnik intro.js [13]).

4.4.2 Vprašanja

Če uporabnik med izdelovanjem testne pole ugotovi, da določenega vprašanja ni na seznamu, lahko po potrebi v zbirko vseh vprašanj doda novo. Na razpolago je več predlog vprašanj, ki so na izbiro v jezičkih na vrhu pogleda (slika 4.6).



Slika 4.6: Dodajanje vprašanja z uporabo prve predloge.

Predloga vprašanja je zelo podobna končnemu izgledu vprašanja, da uporabniku olajšamo proces vnosa. Na voljo je tudi gumb za predogled vprašanja

preden se vprašanje shrani. Skrajno desno zgoraj je na voljo tudi hierarhija kategorij, med katerimi mora uporabnik eno izbrati. Tako se to vprašanje avtomatsko pojavi v dodatku oziroma izbrani skupini vprašanj.

Zanimivost tega pogleda je v dejstvu, da so vse predloge med seboj na nek način povezane. To smo izvedli s pomočjo vzorca MVVM in povezovalnih atributov HTML (angl. data-binding), s katerimi lahko označujemo istoležne elemente HTML, kot so na primer vnosna polja. Knjižnica JavaScript Knockout, ki svoje podatke črpa iz svojega modela MVVM pogleda, poskrbi, da se ob morebitni spremembi določenega atributa tega podatkovnega modela vsi elementi istočasno osvežijo. Tako lahko na primer v realnem času dosežemo, da če v polje za vprašanje vnesemo tekst "abc", se ob izbiri druge predloge v polju za vnos vprašanja pojavi popolnoma enak tekst. Podobno se zgodi, če neko opcijo označimo kot pravilno in se ta obarva v zeleno, bomo v drugih predlogah prav tako opazili isto spremembo na pravem mestu. Tudi če vprašanju dodamo novo opcijo, se ta takoj prikaže tudi v ostalih predlogah. Vse to je moč knjižnice Knockout, ki svoje delo opravlja zgolj na strani odjemalca, torej v brskalniku, kar pomeni, da za to ni potrebnih poizvedb na strežnik, uporabniku pa se tako ponudi odziven grafični vmesnik.

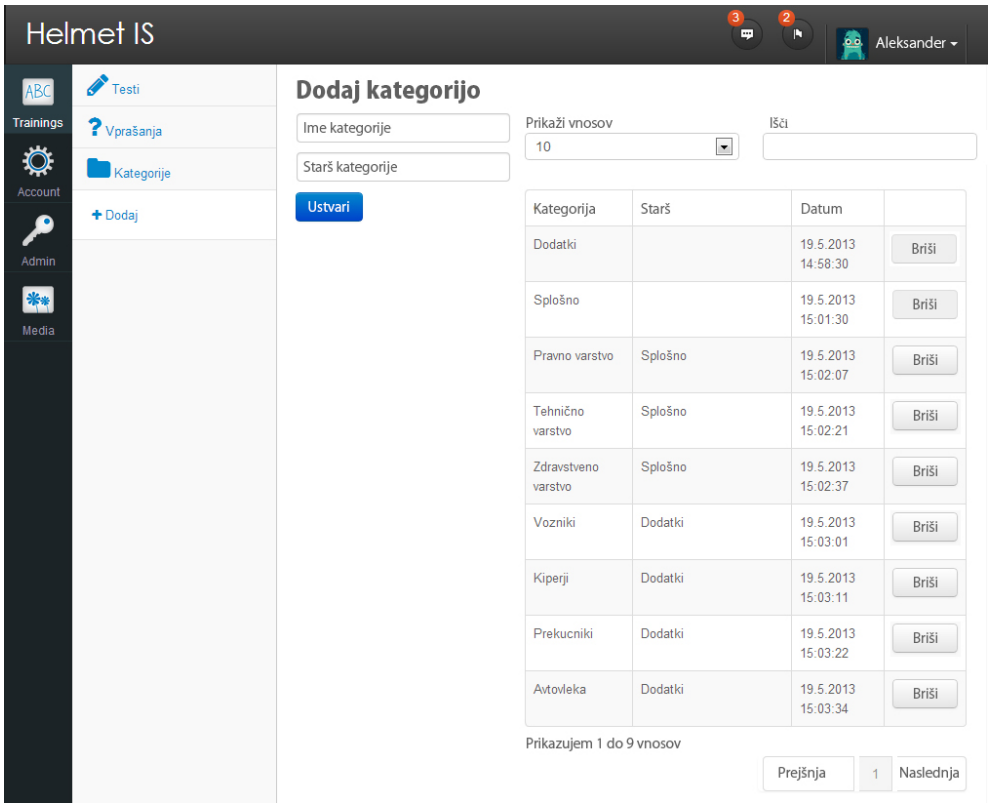
Nekatere predloge vprašanj omogočajo vnos simbola (slike), ki se bo pojavil v testni poli poleg določenega vprašanja ali poleg posamezne opcije. Uporabnik lahko naloži simbol tako, da iz računalnika naloži slikovno datoteko. To lahko stori na tradicionalen način s pritiskom gumba in izborom datoteke na lokalnem disku ali pa datoteko enostavno zagrabi z miško, jo potegne na zaslon in jo, v temu namenjenem polju, izpusti. Za implementacijo tega načina smo uporabili odprtokodni vtičnik jQuery File Upload [7].

4.4.3 Kategorije

Tabela, v kateri je seznam kategorij, ponuja različne možnosti, med katerimi so:

- interakcije s tabelo se izvedejo z zahtevami AJAX in ne zahtevajo osveževanja pogleda,

- iskanje ključnih besed s poizvedbami AJAX,
- pregled celotnega seznama kategorij po straneh,
- nastavitev velikosti ene strani,
- urejanje vrstnega reda v stolpcih s klikom na posamezen stolpec.



Slika 4.7: Dodajanje nove kategorije.

Tako ta tabela, kot skoraj vse podobne tabele v sistemu, so in še bodo izvedene na podoben način, torej z uporabo vtičnika jquery.datatable.

Tako kot lahko dodajamo nova vprašanja, tako lahko to počnemo tudi s kategorijami, kamor vprašanja uvrščamo. Če izberemo podmodul dodajanja kategorij, se nam v pogledu, in sicer v odzivni tabeli, prikaže seznam vseh kategorij, ki so že v sistemu, lahko mu tudi dodajamo nove kategorije ali

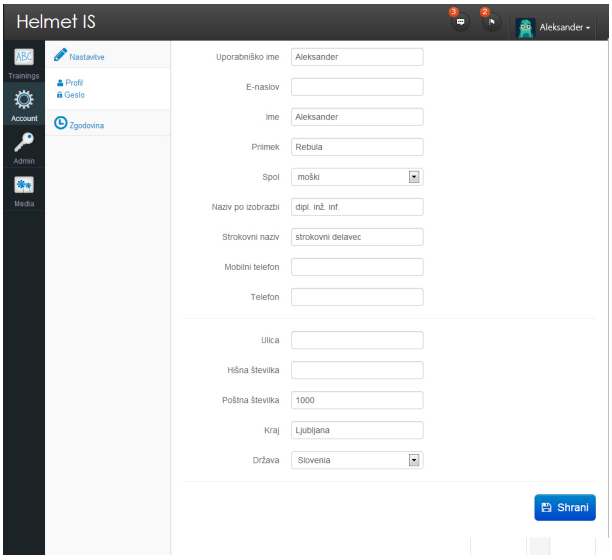
brišemo tiste, ki še ne vsebujejo vprašanj ali podkategorij. Zopet imamo na voljo prijazen grafični vmesnik z zaslonskimi namigi. Preden kategorijo izbrišemo, nas sistem seveda prosi, da izbiro potrdimo.

4.5 Ostali moduli

To podpoglavje je namenjeno krajšemu opisu ostalih modulov sistema, ki so v opisanem sistemu vsaj delno implementirani. To sta dva pomožna modula, in sicer Račun in Media.

Tistih modulov, ki so v navedeni na seznamu zahtev v tretjem poglavju in jih sploh nismo uspeli implementirati, v tem diplomskem delu ne bomo opisovali. Mankajoči moduli bodo prišli na vrsto v prihodnosti, ko bo za to nastopil primeren čas.

4.5.1 Uporabniški račun



Slika 4.8: Nastavitve profila.

To je modul, kjer lahko trenutno prijavljeni uporabnik nastavlja podatke

profila, kot so osebni podatki, prijavno geslo itd. Primer urejanja profila uporabnika prikazuje prikazuje slika 4.8.

4.5.2 Media

V tem modulu je moč pregledovati vse naložene slikovne datoteke, jih dodajati ali brisati obstoječe. Poskrbeli smo, da je nalaganje slik kar se da prijazno, zato smo s pomočjo odprtokodne knjižnice JavaScript jquery.fileupload uredili tako možnost nalaganja več datotek hkrati, kot nalaganje slik na način povleci in spusti (angl. drag & drop).

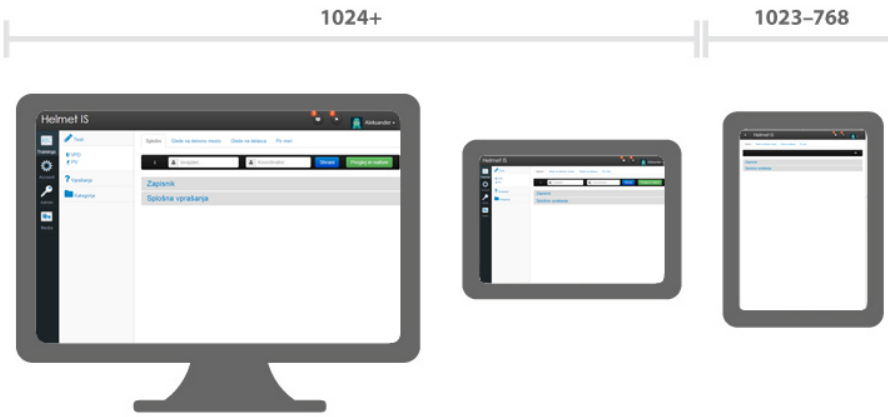
4.6 Prednosti novega informacijskega sistema

4.6.1 Moderen grafični uporabniški vmesnik za boljšo uporabniško izkušnjo

Grafični uporabniški vmesnik je realiziran z ogrodjem za obličje, imenovanim Twitter Bootstrap [8], ki v kombinaciji s stili CSS in knjižnice JavaScript omogoča hiter razvoj kvalitetnega in predvsem odzivnega grafičnega vmesnika. Ta se lahko prilagaja različnim zaslonom oziroma napravam. Na sliki 4.9 lahko vidimo, kako se pri različno velikih zaslonih prilagaja grafični vmesnik sistema HIS na primeru pogleda splošne testne pole.

Z uporabo takšnih grafičnih ogrodij se zmanjša tudi potreba po grafičnih uslugah, kar pomeni hitrejši razvoj in njegovo nižjo ceno. Obstoječa odprtokodna ogrodja za obličje nam namreč omogočajo uporabo že pripravljenih elementov HTML, ki so že primerno opremljeni s stili CSS in že vsebujejo ogromno število simbolov oziroma ikon, ki so za tak sistem potrebne. Elementi so oblikovani tako, da ponudijo kvalitetno uporabniško izkušnjo.

Tu gre predvsem za izboljšane elemente, kot so gumbi, polja spletnih obrazcev, ikone, navigacije, spustni meniji, drobtinice, značke, sporočila, zaslonski namigi, tipografski elementi, vrstice napredka, modalna okna, jezički, pregibni meniji ...



Slika 4.9: Glavna razporeditev HTML v sistemu HIS.

S strani programerja je ustvarjanje pogledov sistema tako hitrejše, saj se mu ni potrebno več toliko ukvarjati z izgledom, ampak se lahko bolj posveti vsebini. Omenjeno grafično ogrodje pa tudi omogoča hiter razvoj morebitnih novih prototipov, ki so včasih lahko zelo blizu končnega produkta.

4.6.2 Večjezičnost

V primeru, da se odločimo storitve sistema prodajati tudi na tujem, lahko z lahkoto uredimo večjezičnost, saj je lokalizacija že predvidena v sistemu in potrebno je le še vnesti prevode.

4.6.3 Socialni dejavniki

Uporabnik se bo lahko preko sistema HIS enostavneje povezoval z ostalimi člani ekipe, ki si bodo lahko med seboj pošiljali sporočila, naloge in ustvarjali viharjenje možganov, ki bodo pomagale pri pridobivanju in uresničevanju novih idej.

4.6.4 Enostavna razširljivost in integracija z drugimi sistemi

Zaradi uporabe trinivojske arhitekture nismo omejeni s platformami, na katere bi se sistem HIS morebiti še razširil v prihodnosti. En primer, ki morda celo pride v poštev v prihodnosti, je uporaba istega fizičnega podatkovnega modela za izdelavo mobilne aplikacije, ne glede na to, ali gre za Android, iPhone ali katerokoli drugo mobilno platformo. Naslednji primer, ki ga lahko omenimo, je izdelava še ene oblike predstavitve, in sicer v obliki spletnih servisov. Uporabnik, ki bi v spletni brskalnik vnesel določen spletni naslov, bi kot odgovor na zahtevo prejel podatke v obliki XML ali JSON. Lahko pa bi na predstavitvenem nivoju morda celo izdelali navadno namizno aplikacijo, ki bi se prav tako posluževala enega in istega podatkovnega in poslovnega nivoja, kot se ju trenutno poslužuje spletno ogrodje MVC. Skratka, trenutna arhitektura nam omogoča res veliko možnosti pri morebitni razširitvi in integraciji trenutnega sistema z novimi, zunanjimi sistemi.

4.6.5 Eno samo orodje za vse administrativne naloge in centraliziranost virov

Ko bodo enkrat izdelani vsi moduli sistema HIS, bomo lahko govorili o centralizirani dejavnosti strokovnega delavca. To pomeni, da bo imel vsa administrativna delovna orodja na enem mestu. Sčasoma se bo zmanjšala ali celo popolnoma izginila potreba po uporabi starih administrativnih orodij, kot sta MS Word in Excel. Ne bo mu potrebno iskati orodij na različnih mestih, prav tako pa bo lahko z enostavnim iskalnikom ali z uporabo modula na enostaven in hiter način našel željene podatke. Tudi za delo pomembna sporočila bodo centralizirana na enem mestu in predvidevam lahko, da bo ob redni uporabi sistema, usklajenost ekipe večja. Vsak uporabnik bo imel možnost preusmerjanja sporočil na različne odjemalce. Lahko si bo nastavil, da sporočila dobiva samo v poštni predal, preko sporočil SMS na svoj telefon, ali pa jih bo pregledoval kar v sistemu HIS.

4.6.6 Ni podvajanja virov

Ker se podatki shranjujejo v podatkovno bazo, jih lahko prikažemo in osvežujemo na mnogo načinov. Sistem HIS poskrbi, da ko je nek podatek nekje vnešen, se ta lahko pojavi in uporabi na različnih mestih v celotnem sistemu. Tudi podvajanje enakih testnih pol pri posameznih uporabnikih ne bo več mogoče, saj bo za to skrbel sistem HIS.

4.6.7 Varnost podatkov

S pomočjo protokola SSL in certifikatov so dostop do sistema in podatki v njem dobro zavarovani. Ker je HIS storitev na spletu oziroma v prihodnosti verjetno celo v oblaku, imamo zagotovljeno tudi varnost v primeru izgube podatkov.

4.6.8 Dostop

Ker gre pri HIS za spletno aplikacijo, je dostop do nje izjemno enostaven, če le ima uporabnik spletni brskalnik in če s certifikatom dokaže svojo pravo identiteto.

4.6.9 Boljša medsebojna obveščenost uporabnikov o pomembnih dogodkih

S pomočjo učinkovitega sistema obveščanja se ne bo moglo več zgoditi (razen iz malomarnosti), da bi nek uporabnik pozabil na nek pomemben dogodek.

4.7 Uporabljena programska oprema in ostala tehnologija

Za razvoj sistema HIS oziroma njegovega zalednega dela (angl. back end) smo uporabili predvsem Microsoftove produkte in večino časa posvetili razvoju sistema v razvojnem okolju Visual Studio 2012. Izkazal se je za izredno učinkovito orodje in primerno rešitev za tovrstne kompleksne probleme. Za vsako nivo arhitekture ima Microsoft odlična orodja (tudi vizualna) in programske vmesnike, ki izjemno olajšajo programerjevo delo in vzdrževanje aplikacije same.

V Visual Studiu 2012 smo tako skrbeli ne samo za zaledje, temveč tudi za obličje (HTML, pogledi ...), a smo za slednje nujno potrebovali tudi orodja za učinkovit razvoj tega dela; potrebovali smo orodja za delo s programsko kodo na strani odjemalca. Za razhroščevanje dela, ki je neposredno v stiku z uporabnikom, smo uporabili zadnji verziji spletnih brskalnikov Firefox in Chrome. Predvsem je tu potrebno omeniti razhroščevanje programske kode JavaScript z dodatkom (za Firefox) Firebug.

Za zaledni del smo uporabili programski jezik C#, za obličje pa skriptne jezike HTML, CSS in JavaScript.

Poglavje 5

Sklepne ugotovitve

Končni rezultat tega diplomskega dela je nov spletni informacijski sistem, imenovan HIS oziroma njegovo ogrodje. V njem smo realizirali enega izmed vseh predvidenih glavnih modulov, ki smo ga poimenovali Usposabljanje. Ko bodo vsi ostali moduli na svojem mestu, bo to pomenilo realizacijo programskega ogrodja, ki bo v prihodnosti po vsej verjetnosti dodobra pretreslo trenutni način dela strokovnih delavcev na področju varnosti pri delu. Strokovni delavci se bodo, če bodo želeli doseči boljši poslovni učinek, morali namreč postopoma privaditi na uporabo sodobnejših delovnih prijemov in orodij, ki jih bo ponudil sistem HIS. Ker pa vemo, da ljudje v splošnem ne maramo sprememb v svojih navadah, ki smo jih izoblikovali skozi čas, smo med razvojem sistema HIS zelo upoštevali tudi ta vidik. Zato verjamemo, da bo sistem HIS s svojo prijaznostjo do uporabnika in učinkovitostjo zagotovo dobro in hitro sprejet s strani strokovnih delavcev.

Kot je bilo v tem diplomskem delu že večkrat nakazano, smo si kot končni produkt zastavili izjemno obsežen projekt, z mnogimi moduli. Razlog gre iskati v našem trdnem prepričanju, da ima sistem HIS izjemen potencial v komercialnem smislu. Verjamemo, da lahko prinese izjemno pomembne pozitivne spremembe, ne samo v ciljnem podjetju, temveč tudi na celotnem področju varnosti pri delu in požarne varnosti tako v Sloveniji, kot v tujini. Verjamemo, da lahko sistem HIS povzroči pravo malo revolucijo v načinu

dela strokovnega delavca v naši ciljni stroki.

Da pride do tega preboja, pa je najprej potrebno, da sistem HIS dozori. To pomeni, da ga je potrebno temeljito preizkusiti v vsaj enem podjetju. Strokovni delavci se morajo nanj navaditi, predvsem pa mu začeti zaupati, kar se lahko zgodi le z rezultati. Preprosto povedano, sistem HIS se mora izkazati. Prispevati mora k krajšemu času izvajanja aktivnosti strokovnega delavca in posledično povečati produktivnost delavcev, kar se na koncu pokaže tudi na boljših poslovnih rezultatih podjetja.

sistem HIS se lahko namreč hitro izkaže za neuspešnega in nepotrebnega, če ga ne bomo znali primerno predstaviti in ga strokovni delavci ne bodo sprejeli za svojega ali če v njem ne bodo videli možnosti za hitrejše in lažje opravljanje svojega dela, kar bi s seboj nosilo boljše poslovne rezultate. Za informacijski sistem, kot je sistem HIS, je torej nujno potrebno, da ga izdelujemo po meri strokovnega delavca za varnost pri delu in požarno varnost. Udobje strokovnega delavca je torej, poleg varnosti seveda, na prvem mestu. Prav zato se od vsega začetka uspešno trudimo, da bi končni uporabnik ves čas tesno sodeloval pri razvoju.

Nas pa ob povedanem čudi dejstvo, da se, glede na razpoložljive informacije, v tej stroki do sedaj še ni pojavil nihče, ki bi si tak celovit sistem že zamislil in ga realiziral ter ponudil v oblaku. Res je, kot smo videli v poglavju 2.7, da obstajajo rešitve, ampak nobena ni tako celovita, kot je oziroma bo postal sistem HIS.

Ni potrebno preveč poudarjati, da je HIS poln razvojnih potencialov, na katere bi se bilo v bodočnosti zagotovo vredno osredotočiti, zato imamo namen sistem razvijati tudi izven okvira tega diplomskega dela.

Slike

2.1 Program za izdelavo zapisnikov. 19
2.2 Knjiga naročil. 20
2.3 Odziven grafični uporabniški vmesnik. 26
2.4 Horizontalna navigacija, prikazana v celoti, ko je zaslon dovolj velik. 27
2.5 Horizontalna navigacija, ko zaslon ni dovolj velik. 28

3.1 Trinivojska arhitektura z vertikalnim skupnim nivojem. 53
3.2 Ogrodje ADO.NET Entity Framework. 55
3.3 Fizični podatkovni model. 56
3.4 Princip delovanja repozitorija. 58

4.1 Prijava v sistem preko spletnega brskalnika. 64
4.2 Glavna razporeditev HTML. 71
4.3 Glavna navigacija. 72
4.4 Pomožna navigacija. 73
4.5 Splošni test tipa VPD, s prikazom zapisnika. 74
4.6 Dodajanje vprašanja z uporabo prve predloge. 77
4.7 Dodajanje nove kategorije. 79
4.8 Nastavitve profila. 80
4.9 Glavna razporeditev HTML v sistemu HIS. 82

Literatura

[1] "Zakon o varnosti in zdravju pri delu (ZVZD-1)", Uradni list RS, št. 43/11.

[2] doc. dr. D. Vavpotič, "Razvoj informacijskih sistemov", izročki s predavanj, 2010.

[3] A. Freeman, S. Sanderson, "Pro ASP.NET MVC 3 Framework", Third Edition, 2011.

[4] (2012) "The Evolution Of The Web". Dostopno na: http://www.evolutionoftheweb.com.

[5] (2009) Julie Lerman, "Agile Entity Framework 4 Repository: Part 1- Model and POCO Classes". Dostopno na: http://thedatafarm.com/blog/data-access/agile-entity-framework-4-repository-part-1-model-and-poco-classes.

[6] (2012) Remondo blog, "The Repository Pattern Example in C#". Dostopno na: http://www.remondo.net/repository-pattern-example-csharp.

[7] Blueimp, "JQuery File Upload". Dostopno na: https://github.com/blueimp/jQuery-File-Upload.

[8] "Twitter Bootstrap". Dostopno na: http://getbootstrap.com/2.3.2.

[9] (2012) Addy Osmani, "Understanding MVVM – A Guide For JavaScript Developers". Dostopno na:
http://addyosmani.com/blog/understanding-mvvm-a-guide-for-javascript-developers/.

[10] "Knockout". Dostopno na:
http://knockoutjs.com.

[11] "jQuery". Dostopno na:
http://jquery.com.

[12] (2006) RFC 4511, "Lightweight Directory Access Protocol (LDAP)". Dostopno na:
http://tools.ietf.org/html/rfc4511.

[13] "Intro.js". Dostopno na:
http://usablica.github.io/intro.js.