UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Uroš Kastelic

Izdelava aplikacij s podporo delovnih tokov za okolje SharePoint Server

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Ljubljana, 2011

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Uroš Kastelic

Izdelava aplikacij s podporo delovnih tokov za okolje SharePoint Server

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor: viš. pred. dr. Igor Rožanc

Ljubljana, 2011

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: 00548/2011
Datum: 01.09.2011

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

Kandidat: **UROŠ KASTELIC**

Naslov: **IZDELAVA APLIKACIJ S PODPORO DELOVNIH TOKOV ZA OKOLJE SHAREPOINT SERVER**

THE WORK FLOW SUPPORTED APPLICATION CONSTRUCTION FOR SHAREPOINT SERVER ENVIRONMENT

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

SharePoint strežnik služi za hranjenje in upravljanje vsebin dokumentov, ki so lahko vezani na ustrezne delovne tokove poslovnega procesa. V diplomski nalogi najprej predstavite SharePoint stražnik, nato pa predstavite razvoj poslovne aplikacije s podporo delovnega toka za okolje SharePoint na tri načine: z opisom delovnega toka s programsko kodo, z uporabo ogrodja Windows Workflow Foundation in z uporabo orodja K2. Nalogo zaključite s kratko primerjavo vseh treh načinov izdelave.

Mentor:

viš. pred. dr. Igor Rožanc

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

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

diplomskega dela

Spodaj podpisani/-a Uroš Kastelic,

z vpisno številko 63070267,

sem avtor/-ica diplomskega dela z naslovom:

Izdelava aplikacij s podporo delovnih tokov za okolje SharePoint Server.

S svojim podpisom zagotavljam, da:

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

  viš. pred. dr. Igorja Rožanca.

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

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

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

ZAHVALA

Zahvaljujem se svojim domačim za podporo skozi vsa študijska leta.

Prav tako se zahvaljujem viš. pred. dr. Igorju Rožancu za strokovne nasvete in pomoč pri nastajanju tega diplomskega dela.

Hvala tudi vsem ostalim, ki ste mi skozi vsa leta študija stali ob strani.

KAZALO

SEZNAM KRATIC IN SIMBOLOV

AJAX *Asynchronous Javascript and XML* – asinhroni Javascript in XML.

CAML - *Collaborative Application Markup Language* – sodelovalni aplikacijski označevalni jezik.

CMS - *Content Management System* - sistem za upravljanje z vsebinami.

CSS - *Cascading Style Sheets* – kaskadna slogovna predloga.

DLL - *Dynamic-link Library* – dinamično povezovalna knjižnica.

DMS - *Document Management System* - sistem za upravljanje z dokumenti.

HTML - *HyperText Markup Language* – označevalni jezik za gradnjo spletnih strani.

HTTP - *HyperText Transfer Protocol* – protokol za prenos informacij na spletu.

IIS – *Internet Information Services* (včasih tudi *Internet Information Server*) – spletne informacijske storitve (včasih tudi spletno informacijski strežnik).

LINQ – *Language Integrated Query* – poizvedbeni jezik, vgrajen v programski jezik.

SAP – *Systems, Applications and Products* – poslovni sistemi, aplikacije in produkti za procesiranje podatkov.

SSP – *Shared Service Providers* – skupne storitve ponudnikov.

XML – *Extensible Markup Language* – razširljiv označevalni jezik.

XSLT - *EXtensible Stylesheet Language Transformations* – razširljiva predloga za pretvorbo programskega jezika XML.

POVZETEK

Vedno več podjetij se odloča za prenovo informacijskih sistemov, kakor tudi za informatizacijo in optimizacijo poslovnih procesov. Naj bo malo, srednje ali veliko podjetje, vsa se srečujejo z veliko količino dokumentov tako v fizični kot v virtualni obliki. SharePoint strežnik ponuja možnost shranjevanja in upravljanja z vsebinami dokumentov in ostalih podatkov. V kolikor shranjeno vsebino podpremo še z delovnimi tokovi poslovnega procesa, zadovoljimo želje marsikaterega podjetja. Tako lahko uporabniku prikrijemo zelo zahteven delovni tok in mu ponudimo interakcijo z aplikacijo le, kadar je to potrebno.

Cilj diplomske naloge je izdelati poslovno aplikacijo s podporo delovnega toka za okolje SharePoint. Pred vsako izdelavo aplikacij je potrebno, da je razvijalec seznanjen s produkti in možnimi rešitvami istega problema na več načinov. Le tako je mogoče, da v nekem trenutku izbere pravo rešitev za problem. V diplomskem delu so predstavljeni trije načini izgradnje delovnega toka: od simulacije delovanja delovnega toka s programsko kodo, preko temeljnega ogrodja Windows Workflow Foundation, do t.i. orodij tretjih strank. Vsi produkti za razvoj kakor tudi SharePoint portal so namenjeni platformi Microsoft Windows, zaradi česar je izgradnja aplikacije potekala s pomočjo tehnologij .NET.

Ključne besede

SharePoint, delovni tok, Workflow Foundation, K2, Workflow Designer, informatizacija poslovnih procesov.

SUMMARY

More and more companies are deciding for the renewal of information systems, as well as the computerization and the optimization of business processes. Small, medium or large companies, all are faced with a large amount of documents in both physical and virtual form. SharePoint Server provides an opportunity for storage and content management of the documents and other data. We satisfy the needs of many businesses if we provide the support to the stored contents by the business process workflows. That is how we can mask a very complex workflow to a user and offer him interaction with the application when it is required

only.

The aim of the thesis is to create a business application with a workflow support for the SharePoint environment. Before any application construction it is necessary that the developer is familiar with the products and the possible solutions to the same problem in several ways. Only then it is possible that at some point developer chooses the right solution for the problem. Featured are three methods of construction, from the simulation of workflow effect with the program code, through the core of the Windows Workflow Foundation framework to other so-called third-party tools. All development products as well as the SharePoint Server are designed for the Microsoft Windows platform, which is why applications were built with the .NET technologies as well.

Keywords

SharePoint, workflow, Workflow Foundation, K2, Workflow Designer, computerization of the business processes.

1. UVOD

Vsa podjetja izvajajo tako notranje kot tudi zunanje poslovanje na podlagi poslovnih procesov. Skozi poslovanje se v podjetju hrani veliko prepotrebne dokumentacije. Hranjenje starejših dokumentov poteka v arhivih, medtem ko novejše dokumente prepogosto predolgo zadržujejo v mapah po predalih, omarah, preden tudi njih spravijo v arhiv, pa naj gre za potne naloge, pošto, pogodbe ali kaj drugega. Običajno je potrebno arhiv urediti, da je iskanje po njem lažje in bolj pregledno, saj neustrezno shranjena poslovna dokumentacija lahko povzroči marsikatero nevšečnost. Prav tako je dokumentacijo zelo težko zavarovati pred nepooblaščenimi vpogledi. Ravno zaradi tega se v podjetjih vedno bolj kaže potreba po centralnem nadzoru in upravljanju z dokumenti.

Sistemov za elektronsko shrambo dokumentov je veliko. V diplomski nalogi si bomo pogledali rešitev, ki se imenuje Microsoft SharePoint [2]. Gre za več kot samo sistem za elektronsko shrambo dokumentov. V drugem poglavju bomo SharePoint podrobneje spoznali. Razložili bomo kaj Microsoft SharePoint sploh je in kakšne različice poznamo. Prav tako bomo razložili topologijo strežnikov in arhitekturo SharePoint portala. Podrobneje bomo razložili tudi strukturo SharePoint strani.

Podjetja potrebujejo informatizirane poslovne procese, saj bi s tem lahko pohitrili izvajanje le-teh ter tako povečali produktivnost. Vendar pa večji poslovni procesi potrebujejo natančno zasnovane in izdelane delovne tokove. V tretjem poglavju bomo zato na primeru aplikacije za podporo nastanka pogodbe pogledali, kakšne so možnosti integracije delovnega toka znotraj SharePoint portala. S pomočjo različnih orodij in tehnologij bomo ne le simulirali, temveč tudi zgradili dejanski delovni tok in ga nato povezali v celovito rešitev skupaj s SharePoint portalom.

2. SPLETNO OKOLJE MICROSOFT SHAREPOINT

Microsoft SharePoint je spletno okolje, katerega je razvil Microsoft za manjša, srednja in večja podjetja. Microsoft opisuje SharePoint kot sistem za upravljanje z vsebinami (CMS) in sistem za upravljanje z dokumenti (DMS). SharePoint je dejansko skupek Microsoftovih tehnologij in omogoča ustvarjanje intranet ter ekstranet portalov, upravljanje z dokumenti, datotekami, avtomatiziranje poslovnih procesov in še marsikaj. Beseda SharePoint se velikokrat uporablja samostojno in označuje platformo oz. strežnik, ki vsebuje Microsoftove SharePoint storitve, katerih prvoten namen je deljenje informacij med zaposlenimi, shranjevanje in organizacijo dokumentov ter še marsikaj. Zaposlenim torej omogoča, da naredijo delovne prostore, kjer delijo informacije o projektih, svoje lastne strani, sezname za arhiviranje dokumentov ter še marsikaj in to s samo nekaj kliki z miško. Razvijalcem pa zaradi svoje razširljivosti in fleksibilnosti omogoča poseg v skoraj vsako lastnost portala, s čimer se lahko portal tako vizualno kot tehnično prilagodi željam podjetju. Prav tako pa je razvijalcem omogočena izdelava novih SharePoint aplikacij, delovnih tokov, seznamov, spletnih delov itd.

SharePoint se deli v več prodajnih produktov, od tehnologij v oblaku (angl. cloud) oz. spletnih storitev (angl. online services), do strežniških produktov, ki se nahajajo na strežnikih, do katerih imamo praviloma fizičen dostop. Zadnja različica je seveda dražja od spletnih različic, obstaja pa tudi brezplačna različica, namenjena manjšim podjetjem.

Z izdajo nove različice platforme Microsoft SharePoint 2010, Microsoft prinaša veliko novosti [5]. Med drugimi uvedba traku (angl. ribbon), ki je uporabnikom znan že iz programskega paketa Microsoft Office 2007 [25] in novejšim. Trak omogoča hitro izvajanje opravil, ki ustrezajo trenutnemu delu uporabnika, saj so ukazi razporejeni v logične skupine in prikazani pod zavihki. Največja novost je, da je možno upravljati s številnimi spletnimi brskalniki, kot so na primer Internet Explorer [22], Firefox [20], Safari [21] in ostali. S tem se omogoči uporabnikom, da uporabljajo njihov priljubljeni brskalnik, kar v prejšnjih različicah ni bilo mogoče, in je bil uporabnik omejen le na brskanje po SharePoint portalih s pomočjo Internet Explorer-ja. Tako naj bi odpravili napačno prikazovanje spletnih obrazcev, nepravilno izvajanje skript in podobno, do katerih je prihajalo pri prejšnjih različicah SharePoint portala, v kolikor uporabnik ni uporabljal Internet Explorer.

SharePoint Foundation je brezplačen strežniški produkt in je osnova za SharePoint Server 2010. Namenjen je manjšim in srednjim podjetjem. Njegov zastonjski predhodnik se imenuje Windows SharePoint Services (verzija 3.0), ki pa je osnova za Microsoft Office SharePoint Server 2007. Vsebuje objekte in gradnike, s katerimi upravlja z dokumenti, seznami ipd. Največja kritika pri SharePoint strežniku 2007 je bil ravno uporabniški vmesnik, tako da z

novo različico Microsoft prinaša v uporabo nov uporabniški vmesnik, katerega v ozadju poganja AJAX tehnologija [26]. S tem so se znebili nepotrebnih t.i. "postback" klicev s spletnim strežnikom.

SharePoint Online [23] prav tako prinaša vse prednosti SharePoint tehnologij s pomočjo tehnologij v oblaku. Gre torej za strežniško različico, pri kateri pa nimamo vpogleda v fizični strežnik, temveč le v storitev SharePoint Online. Ciljna množica so srednja podjetja, s približno 25 zaposlenimi, in omogoča izgradnjo intranetov, izdelavo obrazcev, prav tako omogoča avtomatiziranje poslovnih procesov s pomočjo Microsoft Dynamics ter seveda 24 urni dostop vsak dan. Za dostop potrebuje vsak zaposleni svojo licenco, za katero pa je potrebno plačati naročnino za uporabo. SharePoint Online spada v spletni paket Office 365 [24]. Gre za spletno različico znanega Microsoft Office paketa, vendar vsebuje Office Professional Plus, Exchange Online, SharePoint Online ter Lync Online. Razvoj SharePoint Online še poteka in bo v prihodnje prinesel še več funkcionalnosti.

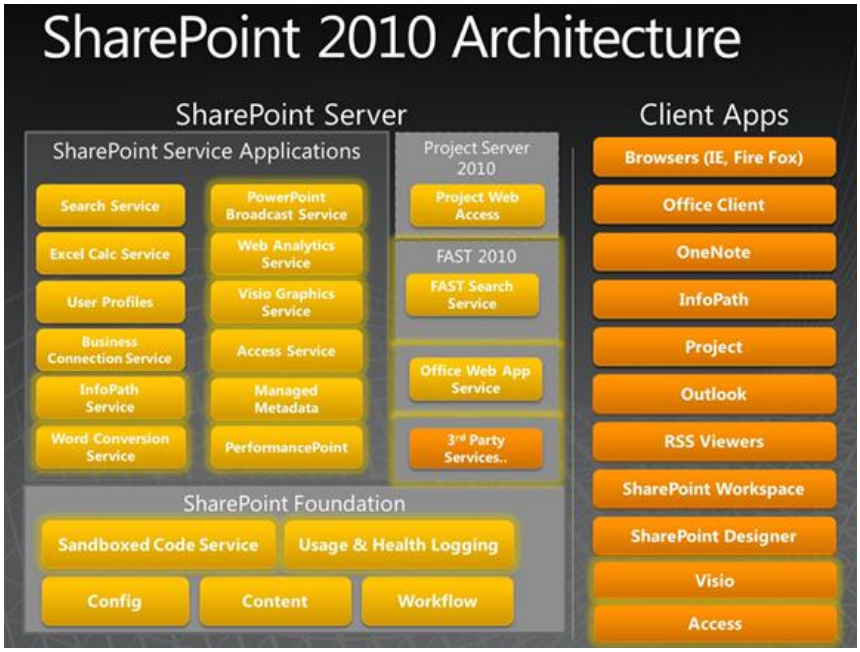
SharePoint Server je najbolj razširjena strežniška različica orodja SharePoint pri večjih podjetjih. Za uporabo je potreben nakup strežniške licence ter tudi namenski strežnik oz. več le-teh, kar bo tudi pojasnjeno v naslednjem poglavju. SharePoint Server je namenjen večjim podjetjem, zaradi česar se od SharePoint Foundation močno razlikuje v številu storitev, ki jih ponuja. Storitve v SharePoint Server 2010 so prav tako posodobljene in izpopoljnene. Tako strežniška različica vsebuje novo funkcijo iskanja, ki omogoča hitrejše in bolj učinkovito iskanje informacij. Storitve Excel Services izboljšujejo poslovno obveščanje, saj omogočajo vizualno analizo podatkov z dodajanjem grafikonov ipd. Office Web App storitev omogoča spletno ustvarjanje delovnih zvezkov, tabel, zapisov in še marsikaj.

Pri uporabi sistema za upravljanje z dokumenti SharePoint uporablja tehnologije Microsoft Office paketa. S pomočjo paketa Office omogočimo uporabnikom SharePoint sistema lažjo povezovanje z dokumenti (Word, Excel,...), podatki (Excel, Access,...) kar znotraj SharePoint portala. Novejše različice SharePoint portala omogočajo izdelavo obrazcev s pomočjo orodja InfoPath, izdelavo delovnih tokov s pomočjo orodja Visio in podobno. Tako vidimo, da se Microsoft SharePoint poslužuje ter integrira uporabo nekaterih Microsoftovih produktov, vse to za lažjo uporabo končnemu uporabniku. To pa je seveda veliki plus za podjetja, saj tako zaposleni ne potrebujejo dodatnega časa za spoznavanje in izobraževanje novega produkta. Uporaba je tako lažja in hitrejša.

Potrebno je omeniti, da SharePoint Foundation 2010 in SharePoint Server 2010 lahko najdemo samo v 64-bitni različici in ju lahko namestimo samo na 64-bitno različico Windows Server 2008 ali Windows Server 2008 R2. Obstaja tudi možnost namestitve SharePoint Foundation ter SharePoint Server 2010 na 64-bitno različico operacijskega sistema Windows 7 ali Windows Vista. Na osprednjem strežniku SharePoint Foundation uporablja IIS 7.0 za

poslušanje prihajajočih HTTP zahtev. SharePoint Foundation je zgrajen na ogrodju .NET 3.5 in ASP.NET 3.5 s servisnim paketom 1.

Slika 1 prikazuje spekter storitev in aplikacij, ki jih SharePoint strežnik podpira.



Slika 1: Arhitektura SharePoint 2010.

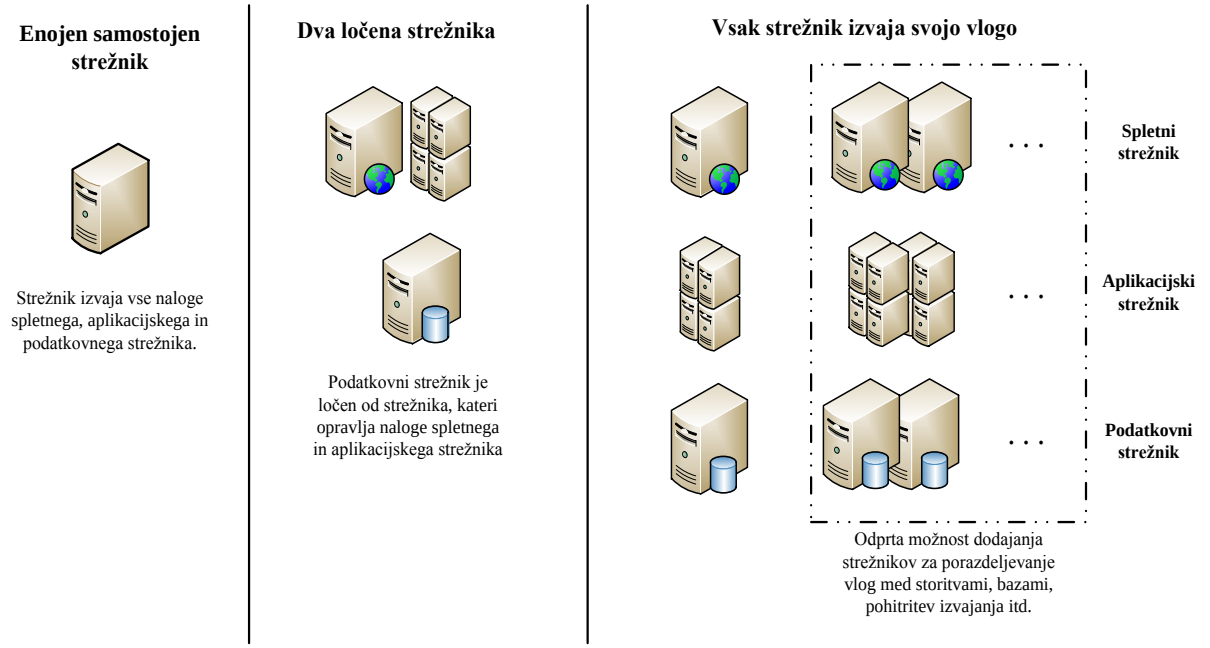
2.1. Topologija namestitve SharePoint portala

Uporabniku pri uporabi storitve SharePoint Online ni potrebno skrbeti glede topologije fizičnih strežnikov in namestitvi storitve. Vsi skrbniki strežnikov pa se morajo zavedati, da je pri namestitvi okolja SharePoint Foundation ali Server potrebno poznati vse možne postavitve in namestitve fizičnih strežnikov.

SharePoint Server uporablja trislojno nivojsko arhitekturo (angl. three-tier roles). Tako opravlja vlogo spletnega strežnika, aplikacijskega strežnika in podatkovnega strežnika. Tehnični diagrami za SharePoint strežnik, kakor tudi diagrami topologije strežnikov, se nahajajo v [7]. Skupino strežnikov, ki sodelujejo pri poganjanju portala imenujemo farma (angl. farm). Namestitev portala je možna na en strežnik (angl. one-server farm), kar je priporočljivo pri manjših podjetjih oz. pri številu uporabnikov manjših od 100. Čim pa se številka uporabnikov poveča, vendar še ne presega približno 20.000 uporabnikov, je priporočljivo fizično ločiti strežnike. Vloge strežnikov lahko združujemo. Tako lahko na

primer ločimo spletni strežnik, a ohranimo enojen aplikacijsko-podatkovni strežnik. Tako se pri ločevanju vlog med fizično različne strežnike znebimo obremenjenosti strežnikov. Pri velikem številu dokumentov, objav in uporabnikov oz. pri večjih podjetjih je priporočljivo vse vloge strežnikov ločiti in po možnosti celo uporabiti več strežnikov pri vsaki vlogi izvajanja (torej več skupnih strežnikov za aplikacijski nivo, več skupnih strežnikov za podatkovni nivo, ipd.). Slika 2 prikazuje opisane možne topologije SharePoint strežnikov.

Omenili smo že, da mora strežnik, v kolikor želimo nanj namestiti SharePoint produkt, uporabljati Microsoftove produkte. Tako mora biti na strežniku nameščen Windows Server 2008 (lahko tudi 2003 v kolikor nameščamo starejšo različico SharePoint produkta). Prav tako je potrebno na podatkovni strežnik namestiti podatkovno bazo Microsoft SQL Server 2008 (lahko tudi 2005, z najnovejšim servisnim paketom).



Slika 2: Možne topologije SharePoint strežnikov.

Spletni strežnik gostuje spletne strani, spletne storitve, SharePoint spletne dele (angl. Web Parts), ki so potrebni za procesiranje vseh zahtev sprejete v okviru farme. Prejete zahteve spletni strežnik nato preusmerja primernemu aplikacijskemu strežniku. Uporabniki se pri delu s SharePoint portalom povezujejo samo s spletnim strežnikom. Tako uporabniki nimajo dostopa do aplikacijskega ali podatkovnega strežnika. Zaradi tega je spletni strežnik najbolj uporabljen strežnik v farmi. Tako je pri večjem številu uporabnikov potrebno razmisliti o večjem številu uporabljenih spletnih strežnikov, da breme porazdelimo.

Aplikacijski strežnik ima naloge povezane s storitvami (angl. services), ki so lahko razporejene na fizičnem računalniku. Vsaka storitev predstavlja različno aplikacijsko storitev, ki jo je možno ločiti na določen aplikacijski strežnik. Prav tako lahko storitve, ki imajo podobno uporabo in lastnosti združimo skupaj in jih po potrebi tudi premaknemo na samostojen strežnik (primer: storitve povezane z uporabnikom lahko združimo v lastno skupino storitev). Primer dobre prakse je, da po uspešni namestitvi vseh komponent SharePoint portala, poiščemo storitev, ki jemlje preveč resursov in jo prestavimo na ločen fizični strežnik.

Podatkovni strežnik ima nalogo shranjevanja podatkov vseh nastavitev, dokumentov, zapisov ipd. V manjših farmah so lahko vse podatkovne baze razporejene na lastnem fizičnem strežniku, medtem ko je v večjih farmah smiselno baze združiti po njihovih vlogah in jih ločiti na več fizičnih strežnikov.

Število storitev (in skladno s tem podatkovnih baz) je v novejši različici SharePoint 2010 večje kot v SharePoint 2007 in starejših različicah, zaradi česar se priporoča razvrščanje storitev in podatkovnih baz s podobnimi karakteristikami na lasten strežnik.

2.2. Arhitektura SharePoint portala

Tako ASP.NET [27] kot tudi SharePoint storitve se zanašajo na IIS za oskrbo z osnovnim mehanizmom za poslušanje in procesiranje prihajajočih HTTP zahtev in za oskrbo upravljavske infrastrukture za postavitev in poganjanje delovnih procesov na spletnem strežniku. IIS spletna stran je definirana kot vstopna točka v infrastrukturo IIS spletnega strežnika. Vsaka IIS spletna stran je nastavljena, da posluša in procesira vhodne HTTP zahteve, ki ustrezajo določenim kriterijem. Vsaka IIS spletna stran je povezana s korenskim imenikom, to je fizični imenik na datotečnem sistemu znotraj gostujočega spletnega strežnika [1].

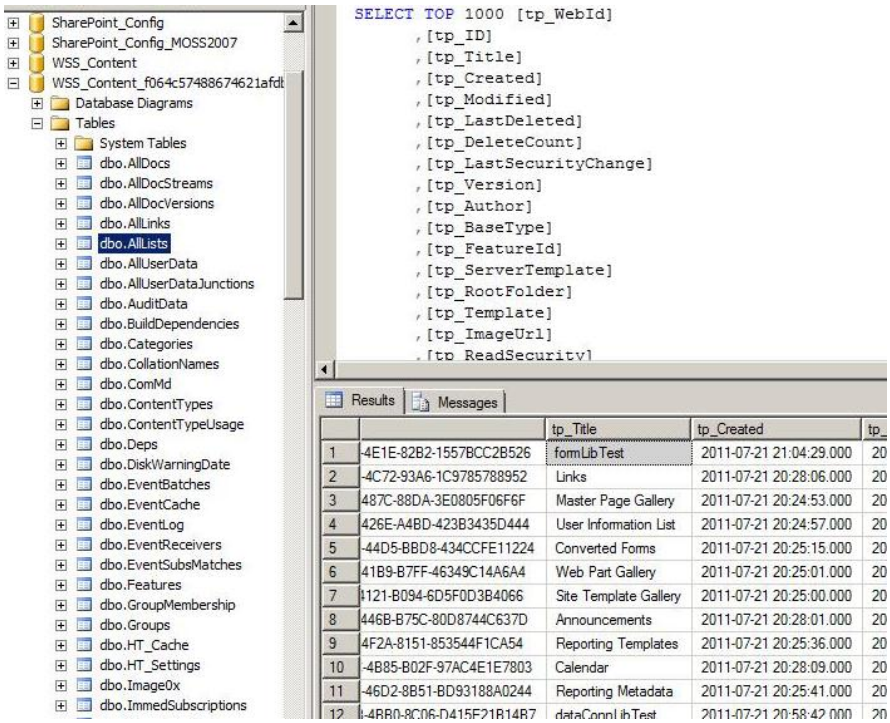
Korenski imenik, znotraj katerega so imeniki ostalih dodanih IIS spletnih strani, se po privzetih nastavitvah nahaja na `C:\Inetpub\wwwroot`, znotraj spletnega strežnika, na katerem imamo nameščen IIS. V kolikor bo prišla zahteva po spletni strani `page1.html`, bo strežnik preveril ali se v korenskem imeniku nahaja datoteka `page1.html`, vsebino datoteke naložil v pomnilnik in to posredoval nazaj odjemalcu. Prav tako IIS spletna stran preverja, ali prihajajoče zahteve potrebujejo avtentikacijo in kateri avtentikacijski protokol potrebujejo. Obstaja več načinov avtentikacije, za kar poskrbijo avtentikacijski protokoli kot so na primer Basic Authentication, Integrated Windows Authentication in drugi, dovoljen pa je lahko tudi anonimni dostop (angl. anonymous access).

Skrbnik IIS strežnika lahko pregleduje, ustvarja in nastavlja IIS spletne strani s pomočjo skrbniškega orodja IIS Manager.

IIS shranjuje namestitvene informacije o svojih IIS spletnih straneh in spletnih imenikih zbirki, znani kot IIS metabaza (v kolikor uporabljamo IIS 6.0) ali IIS konfiguracijska datoteka (v kolikor uporabljamo IIS 7.0). IIS metabaza ali konfiguracijska datoteka je shranjena na datotečnem sistemu osprednjega spletnega strežnika (angl. front-end web server), ki poganja IIS. Tako se vsakič, ko skrbnik naredi in nastavi IIS spletno stran z uporabo skrbniškega orodja, IIS naredi zapis o teh spremembah v lokalno IIS metabazo. Pri IIS 6 je IIS metabaza predstavljena kot navaden XML zapis, pri IIS 7, pa so namestitvene informacije shranjene v datoteki `applicationHost.config`. Zapis v datoteki je tipa XML.

SharePoint se povezuje z ASP.NET na nivoju IIS spletne strani. Vsaka IIS spletna stran, v kateri gostujemo SharePoint strani, mora iti skozi enkraten proces transformacije, v katerem se v korenski imenik znotraj gostujoče IIS spletne strani na datotečnem sistemu doda specifična SharePoint namestitvena datoteka imenovana `web.config`. Prav tako se skozi proces transformacije v IIS metabazo doda zapis o t.i. spletni aplikaciji, kot jo pojmuje SharePoint terminologija. Zatem, ko se proces transformacije konča, SharePoint s pomočjo IIS strežnika in ASP.NET skrbi za preusmerjanje vseh prihajajočih zahtev v času izvajanja.

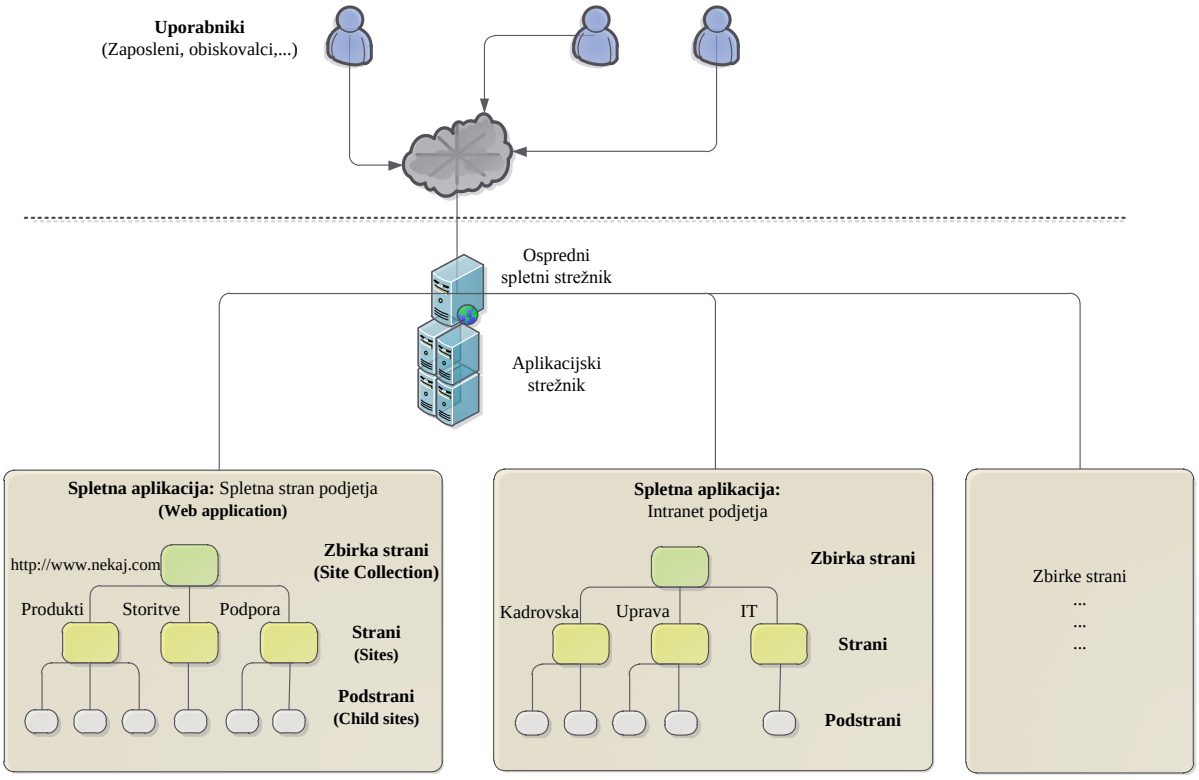
Ustvarjanje SharePoint spletne aplikacije je zelo velik skrbniški proces in za izvedbo le-tega so potrebne skrbniške pravice na ravni SharePoint farme. Po uspešni izdelavi SharePoint spletne aplikacije ni več potrebno manipulirati z datotečnim sistemom oz. IIS metabazo (ali konfiguracijsko datoteko). Od tega trenutka dalje SharePoint arhitektura omogoči dodajanje, popravljanje in brisanje strani kakor tudi zbirk strani preko dodajanja zapisov v namestitveno podatkovno bazo (angl. configuration database) in vsebinsko podatkovno bazo (angl. content database). Primer vsebinske podatkovne baze je prikazan na sliki 3.



Slika 3: Prikaz SharePoint vsebinske podatkovne baze.

Ena od prednosti SharePoint portala je možnost izdelave in upravljanja s stranmi znotraj spletne aplikacije, brez posegov v lokalni datotečni sistem osprednjega spletnega strežnika (angl. front-end web server). SharePoint tako poskrbi za shranjevanje prilagojenih oblik `aspx` datotek in master datotek, ki predstavlja neke vrste predlogo spletne strani (angl. template), znotraj vsebinske podatkovne baze in pridobivanje le-teh na zahtevo, ko jih vhodne spletne zahteve zahtevajo za prikaz. V kolikor želi razvijalec popraviti HTML obliko spletne strani s pomočjo orodja SharePoint Designer in popravljeno shraniti, SharePoint zapiše vso vsebino popravljene strani v vsebinsko bazo. Ko pride zahteva za prikaz, jo SharePoint tako naloži iz podatkovne baze in jo posreduje ASP.NET Runtime (Common Language Runtime), ki poskrbi, da se v času izvajanja prevede izvorna koda programskega jezika C# ali Visual Basic. Za prehode v in iz podatkovne baze poskrbi razred `SPVirtualPathProvider`, ki je integriran v vsako SharePoint spletno aplikacijo.

Zelo pomembno je, da razvijalec direktno ne popravlja podatkov v vsebinski ali nastavitveni podatkovni bazi. Namesto tega mora razvijalec delo izvesti preko uporabe SharePoint objektnega modela ali spletnih storitev.



Slika 4: Primer arhitekture SharePoint portala.

2.2.1. SharePoint strežniška farma

SharePoint strežniška farma oz. kmetija se uporablja za opisovanje enega ali več SharePoint strežnikov, ki sodelujejo med sabo in izvajajo SharePoint funkcionalnost odjemalcem. V najbolj osnovnem primeru SharePoint farma predstavlja en sam strežnik, ki se odziva tako kot osprednji spletni strežnik kot tudi SQL podatkovni strežnik. Bolj kompliciranа SharePoint farma je sestavljena iz več osprednjih spletnih strežnikov in posebej namenjene nastavitvene podatkovne baze. Nastavitvena podatkovna baza hrani informacije o celotni farmi.

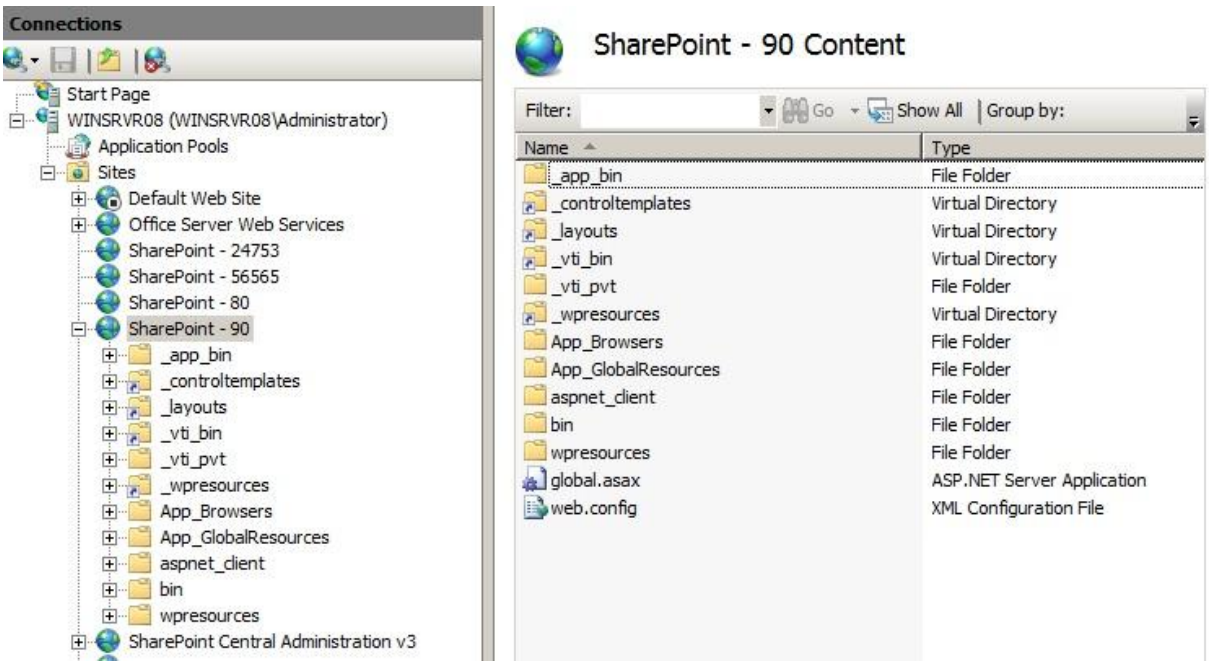
Znotraj strežniške farme so največkrat različne spletne aplikacije (angl. Web Applications). Slika 4 prikazuje primer arhitekture SharePoint portala in primer zgrajenih spletnih aplikacij. Za upravljanje z vsemi spletnimi aplikacijami znotraj strežniške farme (kakor tudi z ostalimi nastavitvami) se uporablja centralna administracija, ki jo SharePoint poganja kot lastno spletno aplikacijo. Ob kreiranju novih spletnih aplikacij strežniška farma ustvari za vsako spletno aplikacijo vsebinsko podatkovno bazo. V kolikor farmo sestavlja samo en strežnik, je vsebinska baza prav tako na istem strežniku, drugače pa je mogoča zahtevnejša namestitev (npr. vsaka spletna aplikacija ima ločene podatkovne baze).

2.2.2. Spletna aplikacija

Spletna aplikacija se lahko definira kot IIS spletna stran (angl. IIS Web Site), nameščena posebej za poganjanje SharePoint strani.

Privzeta spletna stran posluša za HTTP zahteve na vratih 80. Znotraj IIS strežnika lahko naredimo več spletnih strani, ki poslušajo promet na različnih vratih (npr. vrata 8080 ipd.).

SharePoint spletna aplikacija postavljena na IIS strežniku torej gostuje zbirke strani, ta pa hierarhično določa spletne strani.



Slika 5: Spletne aplikacije na IIS strežniku.

Vsaka spletna aplikacija vsebuje nastavitveno datoteko (`web.config`), katero prebirajo tako aplikacije kot storitve tekom izvajanja. Struktura spletne aplikacije znotraj IIS strežnika je razvidna s slike 5.

Ob kreiranju spletne aplikacije se znotraj IIS strežnika naredi virtualni imenik, znotraj katerega se avtomatično naredijo imeniki *_vti_bin*, *_vti_pvt*, ipd. Ti imeniki opravljajo specifične vloge, tako na primer *_vti_bin* skrbi za uporabo SharePoint spletnih storitev in DLL-jev, *_controltemplates* je shramba za vse ASP.NET uporabniške kontrole, ki se lahko uporabijo na spletni strani ipd.

2.2.3. Skrbniški modul centralne administracije

Skrbniški modul centralne administracije (angl. Central Administration) je namenjen izvajanju skrbniških nalog v zvezi z spletnimi aplikacijami, zbirkami spletnih strani ter SharePoint strežnikom iz centralizirane lokacije. Skrbniški modul je razdeljen na štiri dele: na glavno stran, stran z operacijami, aplikacijsko upraviteljsko stran in administrativne strani za skupne storitve ponudnikov (SSP).

S pomočjo centralne administracije (na glavni strani) vidi skrbnik naloge, ki jih še mora dokončati z vidika namestitve. Prav tako ima vpogled v vse strežnike v farmi, vidi pa tudi storitve, ki se izvajajo na strežniku. Omogoča izdelovanje novih ter upravljanje z že obstoječimi spletnimi aplikacijami, izdelavo zbirk spletnih strani, nalaganje, posodabljanje storitev in še marsikaj. SharePoint skrbnik tako lahko izvaja vse skrbniške naloge iz ene lokacije, kar močno olajša nadzor.

2.2.4. Zbirka strani

Zbirka strani (angl. Site Collection) je množica hierarhično odvisnih SharePoint strani, katere so združene v celoto. Strani iste zbirke imajo nekaj skupnih značilnosti, med drugimi si delijo iste pravice (razen če jih ponastavimo), skupne galerije šablon (angl. template galleries), spletne dele, najpogosteje pa si delijo tudi navigacijski menu.

Vsaka zbirka strani vsebuje eno glavno stran na najvišji hierarhični ravni (angl. top-level site), kar že zadostuje, da razglasimo zbirko strani. Glavna stran se nam prikaže ob prihodu na zbirko strani. V praksi se v zbirki strani poleg glavne strani nahajajo tudi podstrani, te pa imajo lahko tudi pod strani. Tako je skorajda možno neomejeno število podstrani, ki so hierarhično strukturirane, kot je to prikazano na sliki 4.

Ustvarjanje nove zbirke strani lahko naredi samo skrbnik (administrator), medtem ko podstrani lahko naredi tudi pooblaščen uporabnik.

2.2.5. SharePoint stran

SharePoint stran (angl. SharePoint site) je zbirka strani (angl. pages), seznamov (angl. lists), knjižnic (angl. libraries), ki so ustvarjene s posebnim namenom znotraj SharePoint zbirke strani (angl. site collection). SharePoint stran lahko prav tako vsebuje podstrani, te pa lahko ponovno vsebujejo podstrani. Držati se je potrebno hierarhičnega strukturiranja.

SharePoint ponuja veliko vnaprej nastavljenih šablon, katere omogočijo hitro izdelavo SharePoint strani. Vsekakor pa je mogoče SharePoint stran izdelati tudi iz popolnoma prazne strani. Na prazno stran tako lahko po lastni želji nanesemo spletne dele, besedilo in druge kontrole. Prav tako lahko že vnaprej nastavljene šablone uredimo po lastni želji ali pa naredimo svoje šablone.

SharePoint ponuja naslednje vnaprej nastavljene šablone za izgradnjo SharePoint strani:

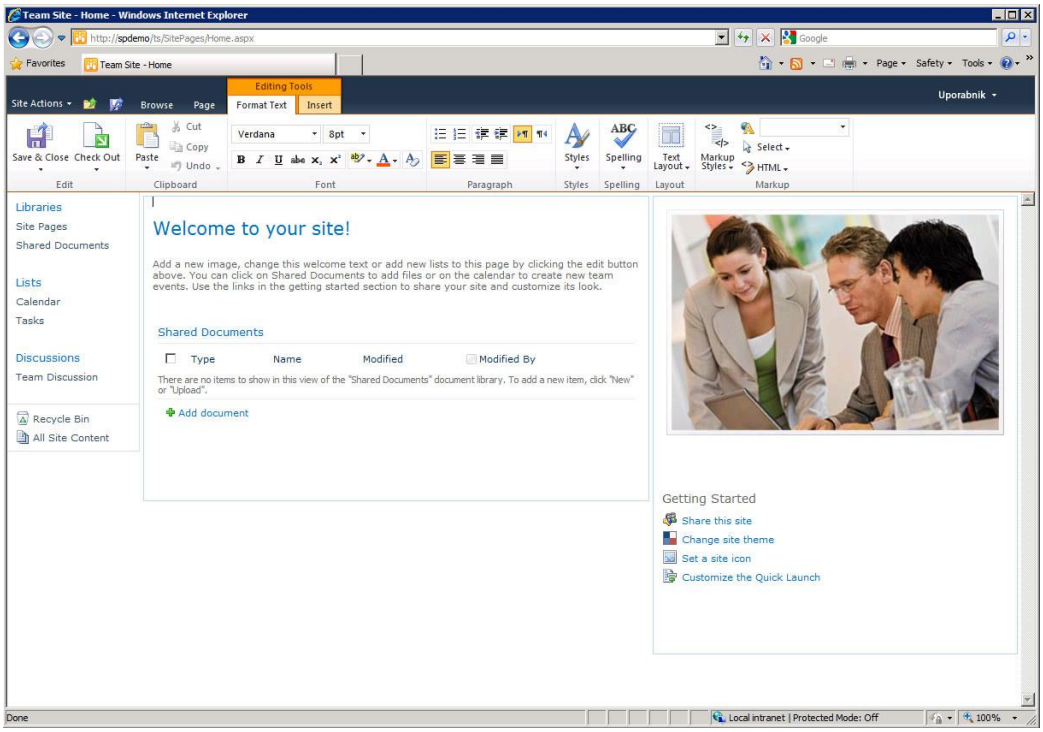
- Skupinska stran (angl. Team Site),
- prazna stran (angl. Blank Site),
- blog,
- wiki stran (angl. Wiki Site),
- dokumentni delovni prostor (angl. Document Workspace),
- osnoven / prazen delovni prostor sestanka (angl. Basic/Blank Meeting Workspace),
- odločitveni delovni prostor sestanka (angl. Decision Meeting Workspace),
- socialni delovni prostor sestanka (angl. Social Meeting Workspace),
- dokumentni center (angl. Document Center),
- iskalni center (angl. Search Center),
- ...

Stran lahko večino nastavitev podeduje od nadrejene zbirke strani, s čimer je nastavitev strani poenostavljena, vendar pa lahko vse te nastavitve tudi sami ponastavimo. Tako lahko ponastavimo navigacijski meni, temo strani (vključuje spremembo barv, postavitev ipd.), pravice na strani, delovne tokove ipd.

2.3. Struktura SharePoint strani

SharePoint portal omogoča uporabnikom znotraj SharePoint strani izgradnjo seznamov, knjižnic, preproste strani brez ali s spletnimi deli, podstrani ali spletna mesta (angl. Workspaces). Seznami se uporabljajo za hranjenje in izmenjavo informacij med zaposlenimi, knjižnice pa za hranjenje datotek (dokumentov, slik, ostalih datotek).

SharePoint stran tako predstavlja končno točko, ki je dostopna iz kjerkoli preko omrežja pa naj bo to internet, intranet ali ekstranet. SharePoint stran je prav tako shramba, preko katere lahko uporabniki shranjujejo in upravljajo z vsebino, kot je to mogoče z vnosi v seznamu ali dokumenti v knjižnici. Uporabniki, ki imajo primerne pravice, lahko tako dodajajo strani, sezname in podstrani. Podatki na straneh so dostopni le vnaprej nastavljeni skupini uporabnikov ali posameznikov.



Slika 6: Osnovna stran SharePoint portala.

Zgornja slika 6 prikazuje osnovno SharePoint stran. Uporabnik takoj v zgornjem delu opazi trak s kontrolniki za upravljanje z lastnostmi strani. Trak si podobnosti deli z ostalim Office programjem. Na levi strani je prikazan meni z navigacijo. V osrednjem delu pa je v tem primeru pas s spletnimi deli, lahko pa je vse kar si podjetje oz. razvijalec zaželita. Oblika

SharePoint strani je polno prilagodljiva s pomočjo datoteke CSS. Datoteko CSS lahko prilagodimo ali izdelamo svojo in jo uporabimo na želeni SharePoint strani ali kar na celotni SharePoint zbirki strani.

2.3.1. Seznam

SharePoint seznami (angl. SharePoint Lists) predstavljajo ključni del portala in bi lahko rekli, da so kar jedro SharePoint arhitekture. Za razvijalca je SharePoint seznam glavni mehanizem, preko katerega se podatki shranjujejo in prikazujejo uporabniku. V skupino SharePoint seznamov spadajo tudi SharePoint knjižnice, vendar jih bomo tokrat opisali ločeno.

Seznam znotraj SharePoint strani predstavlja skupek informacij, ki jih uporabniki portala na enostaven način delijo med seboj. Seznam vsebuje stolpce oz. polja, ki definirajo zapise v seznamu. Tako ima vsak zapis seznama, enako shemo. Na sliki 7 je prikazan SharePoint seznam z nekaterimi lastnimi stolpci. Na seznamu je uporabljen pogled, ki združevanje zapise po nazivih.

Po obliki in obnašanju je SharePoint seznam podoben tabelam v podatkovnih bazah. Seznam je sestavljen iz več polj, ti pa so lahko različnih podatkovnih tipov. Prav tako je mogoče seznamu dodati razne dogodke (angl. events), ki se izvedejo, ko se v seznam doda nov zapis, se popravi obstoječega ali se zapis izbriše. Pri iskanju zapisa v seznamu lahko podatke sortiramo, filtriram ali celo združujemo na podlagi določenih lastnosti. Seznamu je mogoče dodati tudi delovni tok, kar omogoči kompleksnejšo manipulacijo z zapisi v seznamu.

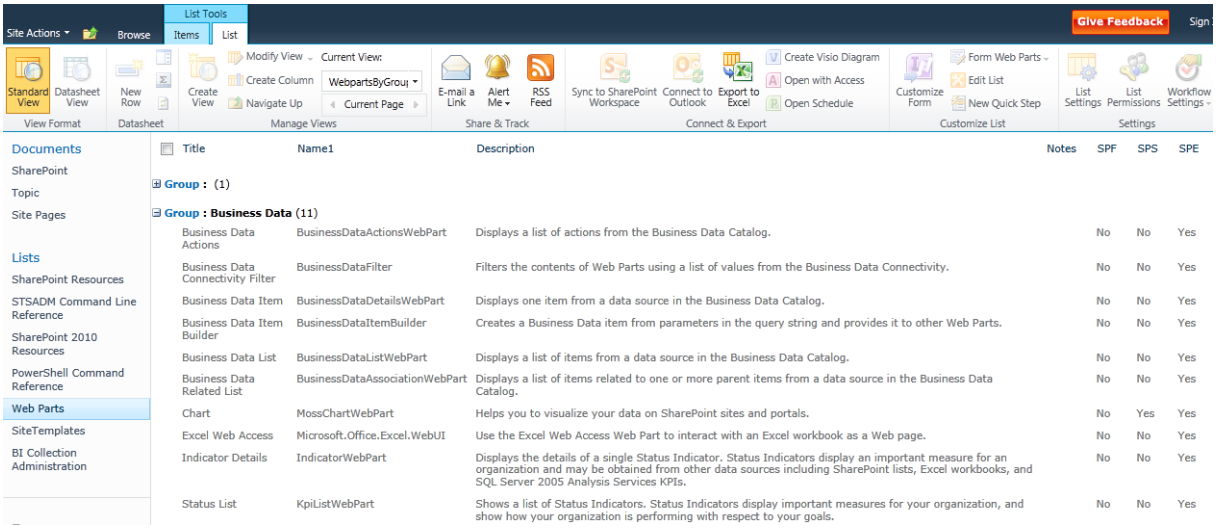
V kolikor ima uporabnik lokalno nameščen Office paket in ima potrebne pravice oz. je ta lastnost uporabniku omogočena, lahko podatke v seznamu popravlja ali dodaja kot v tabeli. SharePoint tako s pomočjo orodja Access lahko prikaže celoten seznam kot podatkovno tabelo in omogoči direkten vnos ali popravljanje zapisov. To omogoča hitrejše vnašanje in je priporočljivo predvsem pri vnašanju večje količine podatkov. Prav tako pa lahko izvozimo seznam v Excel ali naredimo nov seznam s pomočjo preglednice. Omogočeno je celo ustvariti nov seznam iz Excel preglednice. SharePoint bo tako pretvoril takšno preglednico v seznam, vendar pa mora biti na strežniku nameščen program za delo s preglednicami, ki je kompatibilen z Windows SharePoint storitvami.

Kot že omenjeno seznami vsebujejo podatkovna polja oz. stolpce. Sezname lahko skrbnik portala popravlja, jim dodaja stolpce ali seznam tudi izbriše. Prav tako lahko ustvari tudi nov lasten seznam. Lastne sezname je možno zasnovati od začetka. Tako jim lahko dodamo in poimenujemo stolpce, ki jih želimo oz. potrebujemo. Izbiramo lahko med podatkovnimi polji različnih tipov, med drugimi sem spadajo besedilna polja, časovna polja, številčna polja,

izbirna polja, povezave, iskalna polja in še marsikaj. V vsak seznam, se poleg lastnih stolpcev dodajo sistemski stolpci, kot npr. časovna polja (Narejeno, Spremenjeno), polje tipa "Uporabnik ali skupina" (angl. People or Group), v katerem je shranjeno, kdo je naredil ali kdo popravil vnos ipd. Sistemskih polj uporabniki ne morejo spreminjati, kar omogoči boljšo transparentnost in sledljivost.

Vnos v seznamu imenujemo zapis ali predmet (angl. item) in ga lahko popravimo ali zbrišemo, prav tako pa lahko v seznam dodamo nov vnos. Uporabniki lahko zapisu v seznamu pripnejo datoteke, v kolikor se te navezujejo na omenjeni vnos ali je to želeno. SharePoint vsebuje vnaprej narejene obrazce za dodajanje in popravljanje vnosov. Razvijalec lahko prav tako preoblikuje obstoječi obrazec ali naredi svojega s pomočjo orodja SharePoint Designer. V kolikor je potrebno izdelati lasten obrazec, se s pomočjo ASP.NET in SharePoint objektnega modela ali spletnih storitev obrazec integrira v SharePoint okolje.

Pravice za dostop zaposlenih do seznamov ali predmetov v seznamu lahko nastavljamo na celotnem seznamu ali na vsakem predmetu ločeno. Nastavljanje pravic na celotnem seznamu je primerno, v kolikor želimo, da skupina zaposlenih vidi vse predmete v seznamu. Lahko pa prav tako znotraj seznama nastavimo ločene pravice na posameznih predmetih, s čimer omogočimo vpogled do predmetov samo izbranim zaposlenim.



Slika 7: Primer SharePoint seznama.

2.3.2. Knjižnica

SharePoint knjižnico (angl. SharePoint Library) si je mogoče predstavljati kot hibridni seznam, ki razširja isti mehanizem in model shranjevanja kot ga imajo standardni seznami. V knjižnico se prav tako lahko zapisujejo podatki preko stolpcev oz. polj, vendar gre tukaj za

razliko od seznama za t.i. metapodatke. Tako dejanski zapis v knjižnici predstavlja datoteka ali dokument, katerega opisujejo še metapodatki.

SharePoint knjižnica dovoljuje hrambo dokumentov, slik, datotek, med drugimi tudi obrazcev. Za vsako hrambo ima vnaprej določene knjižnice, te so:

- Dokumentna knjižnica (angl. Document Library),
- knjižnica obrazcev (angl. Form Library),
- knjižnica Wiki strani (angl. Wiki page Library),
- slikovna knjižnica (angl. Picture Library),
- knjižnica s podatkovnimi povezavami (angl. Data Connection Library),
- knjižnica za urejanje prevodov (angl. Translation Management Library),
- knjižnica za prosojnice (angl. Slide Library),
- knjižnica za poročila (angl. Report Library).

Dokumentna knjižnica je najbolj uporabljena knjižnica. Ta dovoli shranjevanje in deljenje datotek z ostalimi uporabniki. Dokumente in druge datoteke lahko hierarhično uredimo s pomočjo dodajanja map. Dokumente lahko uporabniki odjavijo (angl. check-out) in jih uredijo lokalno, tako da med njihovim posegom ne pride do prepisovanja datotek, kar bi privedlo do neskladnosti. SharePoint tudi knjižnicam omogoča dodajanje funkcij, kot so npr. diagrami poteka, ki določajo potek obdelave dokumenta po naložitvi. Prav tako SharePoint omogoča shranjevanje zgodovine različic (angl. version history) naloženih dokumentov. Uporabniki imajo tako vpogled v podatke kdo je naložil dokument in kdaj.

Knjižnica obrazcev nam omogoči shranjevanje obrazcev na osnovi XML programskega jezika (npr. naročilnica, poročila o statusu...), ki jih zaposleni pogosto uporabljajo. Ta knjižnica potrebuje za upravljanje XML urejevalnik, ki ga podpirajo Windows SharePoint storitve (npr. Office InfoPath).

Knjižnica Wiki strani se uporablja, ko želimo imeti povezano zbirko Wiki strani. Knjižnica Wiki strani podpira slike, tabele, povezave in wiki povezave.

Slikovna knjižnica omogoča izmenjavo slik. Prav tako knjižnica omogoča posebne lastnosti in sicer prikazovanje slik v stilu majhnih sličic (angl. thumbnails), predvajanje slike ter shranjevanje slik.

Knjižnica s podatkovnimi povezavami nam omogoči shranjevanje datotek, v katerih so shranjene informacije o zunanjih podatkovnih povezavah.

Knjižnica za urejanje prevodov se uporablja, kadar želi uporabnik narediti več dokumentov v različnih jezikih in urediti prevode. Ta knjižnica namreč vsebuje delovni tok, ki ureja proces prevajanja.

Knjižnica za prosojnice se uporablja, ko želimo deliti prosojnice narejene s pomočjo programa Office PowerPoint oz. podobnimi kompatibilnimi aplikacijami. Prav tako so te knjižnice zelo uporabne za iskanje, urejanje in ponovno uporabo prosojnic.

Knjižnica za poročila se uporablja za izdelavo, urejanje in dostavo spletnih strani, dokumentov, ki vsebujejo meritve, cilje in ostale informacije o poslovni inteligenci.

V sami knjižnici pogosto želimo dodati stolpce metapodatkov, ki jih morajo uporabniki izpolniti pri nalaganju dokumentov. S tem omogočimo, da se želeni podatki in dokument nahajajo na istem mestu in jih ni potrebno iskati ločeno. Nad vsakim dokumentom v knjižnici, prav tako nad knjižnico samo, je mogoče nastavljanje pravic, da nepooblaščeni zaposleni nimajo vpogleda v dokument, zbirko dokumentov znotraj knjižnice ali nasploh v knjižnico.

2.3.3. Spletni deli

Stran s spletnimi deli (angl. Web Part Page) se uporablja, kadar želimo na spletni strani prikazati enega ali več spletnih delov. Spletni deli omogočajo lahko pot za izgradnjo bogatih spletnih strani, preko katerih je možno prikazati sezname, knjižnice ali tudi lastne aplikacije s pomočjo SharePoint spletnih storitev.

Spletni del je ASP.NET strežniška kontrola, v katero se doda v pas za spletne dele (angl. Web Part Zone) na strani s spletnimi deli. Dobra lastnost je, da lahko uporabnik spletne dele premika med conami v času delovanja in razvijalcu zaradi tega ni potrebno popravljati programske kode spletne strani. Kontrola omogoča uporabniku, da spremeni vsebino in obliko kar znotraj spletnega brskalnika.

Osnovna oblika SharePoint spletnega dela je sestavljena iz naslovne vrstice, ogrodja in vsebine. Gre torej za modularno enoto informacij, namenjena ponastavitvi uporabniškega vmesnika končnega uporabnika.

2.3.4. Pogledi

SharePoint lahko vsebuje veliko količino podatkov v seznamih ali raznih datotek v knjižnicah. V kolikor želimo zaposlenim olajšati iskanje podatkov po seznamih je najhitreje in najlažje narediti lasten pogled (angl. view).

Prav tako lahko z uporabo orodja SharePoint Designer naredimo lasten pogled podatkov, ki ga imenujemo podatkovni pogled (angl. Data View). Podatkovni pogled se uporablja za pregledovanje podatkov iz različnih virov, npr. poizvedb iz podatkovnih baz, XML dokumentov, spletnih storitev, SharePoint seznamov in knjižnic ter drugih. SharePoint podatke pridobi iz podatkovnih virov v XML obliki ter te podatke prikaže s pomočjo XSLT.

Podatkovni pogledi predstavljajo pogled podatkov v živo v vsakem trenutku. Podatke je možno filtrirati, sortirati in grupirati. Prav tako se lahko spremeni grafična oblika, doda pogojno formatiranje podatkov ipd.

Vsak seznam in knjižnica imata že narejenih nekaj vnaprej pripravljenih pogledov. Vsak od pogledov ima svojo isto imensko imenovano `aspx` spletno stran (npr. `AllItems.aspx`). Poglede se lahko izbira tudi preko spustnega menija v desnem zgornjem kotu spletnega dela.

Pogled je mogoče tudi popraviti ali izdelati s pomočjo programske kode preko SharePoint objektnega modela in razreda *`SPView`* ali jezika CAML [15], slednji pa omogoča veliko večjo manipulacijo.

3. SHAREPOINT PORTAL V PRAKSI

3.1. Aplikacija za podporo potrjevanja nastanka pogodb

Podjetja imajo pri sklepanju pogodb svoja interna pravila poslovanja, katerih se morajo zaposleni držati. Vsaka pogodba oz. posel, ki se sklene v podjetju, gre skozi proces usklajevanja in potrjevanja predlogov za nastanek pogodbe. Ta proces ima svoja pravila kakor tudi predpisan delovni tok.

Za potrebe podjetja smo izdelali aplikacijo potrjevanja nastanka pogodbe, ki nadomesti prenašanje fizičnega dokumenta po podjetju. Do sedaj je bilo zahtevano, da nosilec posla do vseh akterjev v procesu, prinese fizičen dokument, na katerega so se napisali predlogi, pripombe in zahteve vseh vpletenih pri odobritvi, ter šele po odpravi vseh pripomb ali zahtev, ta dokument tudi podpisali in dovolili pripravo pogodbe. Zaradi počasnosti izvajanja obstoječega načina dela in tudi izgub dokumentov, se je porodila ideja o informatizaciji procesa. Tako je bil namen je bil izdelati elektronsko aplikacijo za potrjevanje potrditvenih papirjev in pogodb. Glavni cilji izdelave elektronske različice so pohitritev procesa potrjevanja, oz. skrajšanje odzivnega časa vseh akterjev v procesu. S pomočjo elektronske različice se bo omogočila večja transparentnost, torej preglednost in sledljivost. Prav tako se s tem omogoči elektronska hramba dokumentov, s čimer se želi preprečiti morebitne izgube dokumentov, podvajanja ali zamenjevanje potrditvenih papirjev oz. pripomb v le-tem. Prav tako se z nastavljanjem pravic prepreči dostop do podatkov nepooblaščenim osebam.

Aplikacija je bila narejena s pomočjo orodja Microsoft Visual Studio 2008 ter ogrodja .NET Framework 3.0. Uporabili smo programski jezik C# in ASP.NET. Za potrebe preverjanj vnosov s strani odjemalcev oz. dogodkov s strani uporabnikov (angl. client-side events) pa smo uporabili tudi programski jezik JavaScript. Aplikacija je bila narejena, testirana in uporabljena na portalu Microsoft SharePoint Server 2007, saj je podjetje v tistem času uporabljalo to različico. Pri komunikaciji aplikacije s SharePoint portalom se uporablja SharePoint objektni model ter tudi spletne storitve. Aplikacijo smo tudi preizkusili na SharePoint Server 2010 in je uspešno prestala vse teste, ter je tako pripravljena na migracijo v novejše okolje, ko bo podjetje naredilo prehod na SharePoint 2010.

Pri izgradnji poslovnih aplikacijah se v veliki meri srečujemo z dosedanjimi poslovnimi procesi, po katerih želimo, da se aplikacija premika od začetka do konca. Podjetja se vedno bolj odločajo za prenovo informacijskih sistemov in informatizacijo ter optimizacijo poslovnih procesov. S pomočjo izgradnje aplikacije z delovnim tokom želimo delovni proces standardizirati na vseh ravneh, uporabnikom pa zakriti kompleksnost samega procesa ter mu

omogočiti prikaz le tistih informacij, ki jih kot akter potrebuje v nekem koraku znotraj poslovnega procesa.

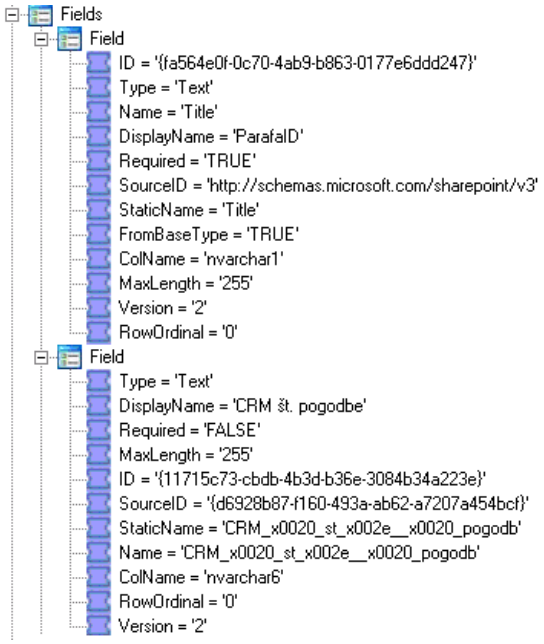
3.1.1. Uporaba SharePoint seznama

Vsi potrditveni dokumenti se shranjujejo v prilagojen SharePoint seznam, ki vsebuje namensko narejena polja. Namen seznama in prilagojenih stolpcev oz. polj (angl. column) je identičen izgradnji podatkovne baze. Aplikacija tako namesto zapisovanja v podatkovno bazo, komunicira s SharePoint portalom s pomočjo SharePoint objektnega modela ali SharePoint spletnih storitev (angl. Web Services). S pomočjo objektnega modela ali spletnih storitev dobimo tako programski nadzor nad SharePoint seznamom in lahko opravljamo vse administrativne funkcije, kot je npr. dodajanje vnosov v seznam, popravljanje, brisanje ipd.

Pri izgradnji seznama je potrebno stolpce oz. polja vnaprej načrtovati, tako kot pri izgradnji podatkovne baze. Potrebno je določiti ime in tip polja. Vsak tip polja ima nato še dodatne možnosti konfiguracije, npr. dolžina vnosnih znakov, ali je polje obvezno, oblika datuma, seznam izbirnih vrednosti, na katerem mestu v seznamu naj se pojavijo itd.

Pri izgradnji SharePoint seznama je tako potrebno paziti, da v čim večji meri seznam pravilno načrtujemo že v začetku, tako da vsak vnos ali predmet (angl. item) v seznamu vsebuje vsa polja. V primeru kasnejšega dopolnjevanja polj v seznamu, stari vnosi v seznamu ne vsebujejo novih polj, kar lahko posledično privede do ničelne reference (angl. null reference), ko pri vnosu iščemo novo dodano polje. To je še posebej nadležno pri zankah, kjer pregledujemo vse vnose po vseh poljih. Tako imajo nekateri vnosi več polj, nekateri manj. Zaradi tega se priporoča, da se seznam, tako kot tabelo v podatkovni bazi, v čim večji meri, naredi še preden vanj vpisujemo vnose.

Pri ustvarjanju seznama, SharePoint vsakemu polju pripiše podatke, ki jih potrebuje za dostopanje in manipuliranje s temi polji. Vsako polje ima tako svojo lastno identifikacijsko številko, ime, tip, prikazno ime in še ostale podatke. Iz slike 8 so razvidne lastnosti SharePoint polj.



Slika 8: Pregled SharePoint polj.

Pri ustvarjanju stolpcev znotraj seznama, je potrebno paziti na poimenovanje polj. Uporabnik, ki izdeluje seznam, lahko v prikazno ime stolpca hitro vpiše šumnike, presledke ali kakšne druge znake, katerega pa nato SharePoint pretvori v njemu bolj prijazno obliko, t.j. brez šumnikov, presledkov ipd. Tako se na primer najbolj uporabljan presledek pretvori v znak `_x0020_`. Razvijalec mora tako nujno skrbno pogledati, kaj je zapisano v stolpčeve lastnosti `Name` ali `Static Name`, ki se lahko razlikujeta od lastnosti `DisplayName`.

SharePoint seznami so tako kot podatkovne baze ločene od same aplikacije. SharePoint portal in seznami delujejo neodvisno od aplikacij, tako da je tudi delo skrbnika SharePoint portala in razvijalca aplikacij ločeno. Skrbnik SharePoint portala pripravi spletno mesto ter seznam s potrebnimi stolpci in te podatke posreduje razvijalcu. Razvijalec pa lahko razvija na ločenem strežniku, v izbranem .NET programskem jeziku in za komunikacijo s SharePoint portalom uporablja SharePoint objektni model ali SharePoint spletne storitve.

V kolikor je potrebno, mora razvijalec poskrbeti, da se nad vnosi v seznamu nastavijo pravice le tistim uporabnikom, ki so udeleženi v obdelavo tega vnosa, s čimer se onemogoči nepooblaščeni vpogled v vnos.

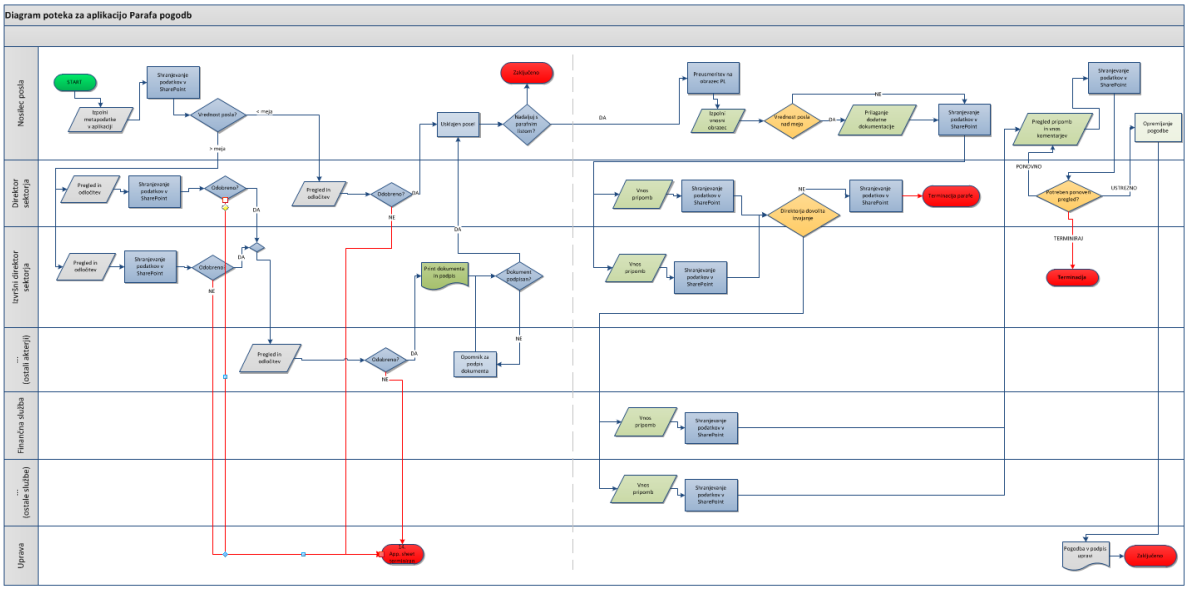
3.1.2. Izdelava SharePoint aplikacije s simulacijo delovnega toka

SharePoint Services omogoča izdelavo delovnega toka (angl. workflow), ki bo skrbel da se proces oz. aplikacija izvaja po specifičnem diagramu poteka. Aplikacijo, ki smo jo v podjetju razvijali za potrebe potrjevanja nastanka pogodbe, mora slediti določenemu diagramu poteka. V samem diagramu poteka je orisan celoten potek procesa, od začetka do konca, ki potuje do vseh akterjev v procesu, kakor so to do sedaj izvajali zaposleni s fizičnim papirjem. Pred izdelavo aplikacije se je potrebno dogovoriti, da mora aplikacija slediti omenjenemu diagramu poteka, ni pa nujno, da v ozadju dejansko poteka delovni tok, možna je tudi simulacija izvajanja delovnega toka.

Pri izgradnji aplikacije mora razvijalec aplikacije najprej poskrbeti, da pred samim začetkom izdelave, dobi usklajen diagram poteka procesa in da je dogovorjeno, kateri so potrebni podatki, ki se bodo zapisovali oz. uporabljali skozi proces. Prav tako je potrebno določiti akterje v procesu in ostale lastnosti kot so pravice, lokacija seznama ipd. Šele nato se lahko razvijalec varno loti izdelave delovnega toka, ki bo tekel v ozadju aplikacije za potrjevanje pogodb. Delovne tokove je možno narediti na različne načine, z uporabo različnih orodij, odvisno od zahtevnosti procesa pa je potrebno preveriti, katera rešitev je najboljša.

Izdelave delovnega toka procesa se lahko lotimo na najbolj klasičen način: celoten diagram poteka sprogramiramo v aplikacijo. Pri tem dejansko ne zgradimo delovnega toka, temveč obnašanje delovnega toka le simuliramo preko programske kode. Na ta način dobimo klasično aplikacijo, ki s pomočjo odločitvenih stavkov (`if`, `switch` stavki) in zank (`for` zanka, razni iteratorji...) ustvari aplikacijo, ki sledi diagramu poteka. Problem pri takšni aplikaciji je, da je potrebno vnaprej računati na vse možne izjeme in te izjeme tudi zajeti v programski kodi, kajti kasnejše večje reprogramiranje aplikacije je lahko kompliciranо in zahtevno opravilo. Zaradi tega je veliko lažje uporabiti orodja za izgradnjo delovnih tokov, ki so namenska in se s pomočjo teh znebimo problem, ki bi lahko nastali ob kasnejšem dodajanju aktivnosti v programsko kodo.

Trenutno razvita aplikacija v podjetju je narejena na način, da je celotna logika poteka procesa sprogramirana in le simulira izvajanje delovnega toka, ki je prikazan na sliki 9. Seveda pa se aplikacija kljub temu do potankosti drži vseh zahtev in dogovorov glede opisanega diagrama poteka. Razlog za takšno odločitev v podjetju, je bilo pomanjkanje časa in virov, zaradi česar so se vodje v podjetju odločile, da se naredi najhitrejša in najlažja rešitev v tistem trenutku.



Slika 9: Diagram poteka za razvito aplikacijo.

Pri izdelavi aplikacije, ki ne bo uporabljala delovnega toka, temveč bo le sledila diagramu poteka, je potrebno računati na to, da mora aplikacija vedeti, v katerem koraku procesa se trenutno nahaja. Zato je potrebno podatek v katerem koraku je trenutno aplikacija shranjevati in osveževati skozi proces. Ta podatek imenujemo status zahtevka. Podatek o statusu zahtevka omogoča ustrezno vejitev aplikacije, s čimer se omogoči odpiranje in zapiranje vnosnih polj, prikazovanje potrebnih podatkov v trenutnem koraku procesa, pošiljanje obvestil na podlagi statusa in podobno. Prav tako skozi proces beležimo že izvedene statuse, tj. informacijo, ki pove, po kateri poti je šel zahtevek.

SharePoint seznami so namenjeni shranjevanju podatkov, zaradi tega jih lahko uporabimo kot prostor, v katerih bomo hranili razne šifrante. Na takšen način je za potrebe aplikacije, narejen tudi seznam, ki hrani seznam vseh možnih statusov skozi izvajanje procesa. Prav tako so narejeni naslednji lastni seznami: šifrant tipa pogodbe, šifrant besedila elektronskih sporočil, zavarovanja posla ipd. Takšen način shranjevanja šifrantov omogoči hitro in pregledno popravljanje obstoječih ali dodajanje novih vnosov. V primeru dodajanja novih statusov v seznam statusov, je v našem primeru potrebno aplikacijo tudi prilagoditi, da bo uporabila nov status in izvedla potrebno delo za ta status. Primer šifranta je viden na sliki 10. Zaradi tega je potrebno aplikacijo reprogramirati in ji dodati nove vejitve ter programsko kodo, ki bo opisovala izvajanje ob tem statusu. Dodajanje novih statusov je največja slabost pri izgradnji aplikacij na takšen način, zaradi česar je potrebno dobro predhodno načrtovanje.

Slika 10: Primer seznama statusov.

V kolikor ne poskrbimo za zasebnost seznama, imajo vpogled vanj vsi zaposleni oz. vsi uporabniki SharePoint spletne aplikacije in SharePoint strani (v kolikor nimamo drugače nastavljenih pravic in vlog uporabnikov). Sezname, ki so povezani z aplikacijo, je tako potrebno zaščiti, da nepooblaščeni uporabniki ne bi spreminjali njene vsebine. Prav zaradi tega je potrebno spremeniti dovoljenja oz. pravice na seznamih šifrantov in dovoliti vnos samo skrbniku portala ali skrbniku aplikacije.

Pri dodeljevanju SharePoint pravic je potrebno omeniti, da obstaja več vrst dovoljenj oz. pravic, na podlagi katerih omogočamo nadzor in zaščito informacij. Spodaj je napisanih nekaj bolj pomembnih vrst in vlog dovoljenj:

- **Polni nadzor**: s tem dovoljenjem je uporabniku omogočeno branje, urejanje, kopiranje, shranjevanje in spreminjanje upravljalske vsebine.
- **Načrtovanje**: uporabniku se omogočijo iste pravice kot pri polnem nadzoru. Ne dovoli se mu le spreminjanja upravljalskih vsebin.
- **Prispevanje**: omogočeno pregledovanje, dodajanje, popravljanje in brisanje vsebine znotraj seznama.
- **Branje**: uporabniku je omogočeno samo prebiranje vnosov v seznamu. Uporabnik ne more kopirati ali spreminjati njegove vsebine.

Dostopanje do seznamov preko programske kode je možno na dva načina:

1. ***Dostopanje preko SharePoint spletnih storitev***
2. ***Prebiranje podatkov s pomočjo SharePoint objektnega modela***

Dostopanje preko SharePoint spletnih storitev je razmeroma preprosto. SharePoint vsebuje polno zbirko že pripravljenih spletnih storitev, ki omogočajo manipulacijo s skoraj vsemi

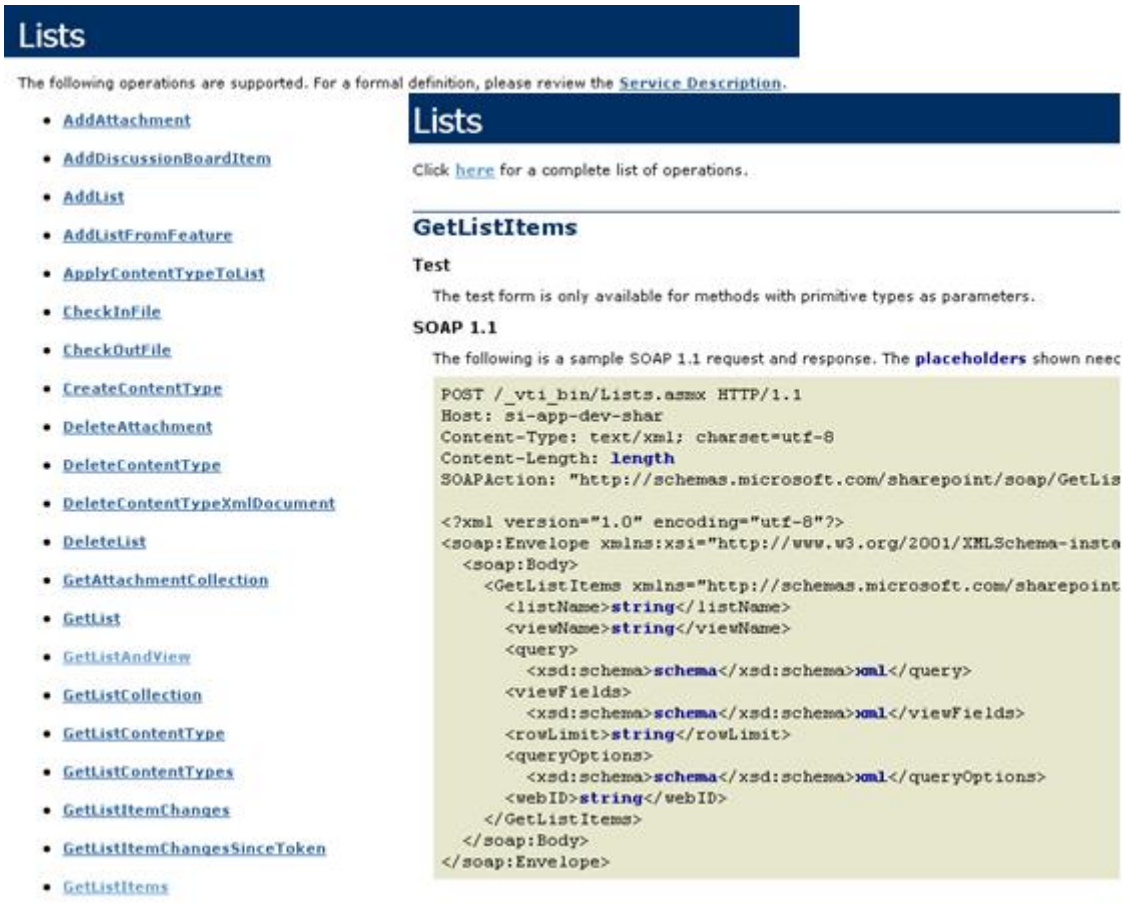
nastavitvami znotraj SharePoint portala. Spletne storitve so shranjene v mapi *_vti_bin*, znotraj vsake zbirke strani posebej, razlikuje se le spletna storitev z administrativnimi metodami, ki je shranjena v mapi *_vti_adm* in jo je potrebno nasloviti z url-jem, ki ima zraven številko vrat, preko katerih dostopa do spletne aplikacije SharePoint centralna administracija.

Seznam vseh spletnih storitev (izvzeto iz *Microsoft Office SharePoint Server 2007*):

- `http://streznik:12345/_vti_adm/Admin.asmx` - Administrativne metode, med drugimi tudi izdelava in brisanje strani.
- `http://`*`streznik`*`/_vti_bin/Alerts.asmx` – Metode za upravljanje opomnikov.
- `http://`*`streznik`*`/_vti_bin/DspSts.asmx` – Metode za pridobivanje shem in podatkov.
- `http://`*`streznik`*`/_vti_bin/DWS.asmx` – Metode za upravljanje z dokumentnimi delovnimi prostori (angl. Document Workspaces).
- `http://`*`streznik`*`/_vti_bin/Forms.asmx` – Metode za upravljanje z uporabniškimi interaktivnimi obrazci.
- `http://`*`streznik`*`/_vti_bin/Imaging.asmx` – Metode za delanje z slikovnimi knjižnicami.
- `http://`*`streznik`*`/_vti_bin/Lists.asmx` – Metode za delanje s seznami.
- `http://`*`streznik`*`/_vti_bin/Meetings.asmx` – Metode za upravljanje z delovnimi prostori sestankov (angl. Meeting Workspaces).
- `http://`*`streznik`*`/_vti_bin/Permissions.asmx` – Metode za delanje s SharePoint varnostnimi storitvami.
- `http://`*`streznik`*`/_vti_bin/SiteData.asmx` – Metode, ki jih uporablja Windows SharePoint Portal strežnik.
- `http://`*`streznik`*`/_vti_bin/Sites.asmx` – Vsebuje metodo za pridobivanje predlog strani.
- `http://`*`streznik`*`/_vti_bin/UserGroup.asmx` – Metode za delovanje z uporabniki in skupinami.
- `http://`*`streznik`*`/_vti_bin/versions.asmx` – Metode za upravljanje z različicami datotek.
- `http://`*`streznik`*`/_vti_bin/Views.asmx` – Metode za delanje s pogledi seznamov.
- `http://`*`streznik`*`/_vti_bin/Webs.asmx` – Metode za upravljanje s stranmi in podstranmi.

- `http://streznik/_vti_bin/Copy.asmx` – Metode za kopiranje datotek znotraj SharePoint strani.
- `http://streznik/_vti_bin/People.asmx` – Vsebuje metode, ki jih lahko uporabljamo za asociacijo uporabnikov s pomočjo identifikatorja.
- ...

Kot vidimo, obstaja za skoraj vsako funkcionalnost znotraj SharePoint portala svoja spletna storitev, katera ima deklarirane metode, ki jih lahko razvijalci uporabijo pri izdelavi svoje aplikacije. V kolikor spletne storitve nimajo želene funkcionalnosti, pa mora razvijalec poseči po SharePoint objektnem modelu in sam napisati metodo, ki bo opravila želeno delo.



Slika 11: Primer SharePoint spletnih storitev.

Za interakcijo s spletnimi storitvami je potrebno predhodno referenciranje v sami programski kodi. V projekt, v katerem razvijamo aplikacijo, je potrebno najprej dodati referenco na izbrano spletno storitev, kar lahko naredimo tudi s pomočjo čarovnika za dodajanje spletnih referenc. Povezavo lahko preverimo preko spletnega brskalnika. Slika 11 prikazuje vstopno

stran znotraj ene od spletnih storitev ter opis metode znotraj spletne storitve. Po uspešno dodani spletni referenci je potrebno v aplikaciji narediti še deklaracijo in inicializacijo spletne storitve. Nov objekt naredimo s pomočjo razreda, ki se imenuje isto kot smo poimenovali spletno referenco. Spletni storitvi je potrebno nastaviti še ustrezne t.i. poverilnice (angl. credentials), ki bodo dovoljevale, da klic spletne storitve, preko uporabe aplikacije, uporabi tudi kateri koli drug uporabnik, ki nima skrbniških pravic. Z uporabo poverilnic vpišemo uporabniško ime in geslo uporabnika, v imenu katerega želimo poganjati spletno storitev. Šele nato sledi klic želene metode iz spletne storitve.

Spodnji del programske kode prikazuje klic metode *ResolvePrincipals*. Parametri, ki jih metoda sprejme so tabela uporabnikov (ali skupin) s prikaznimi imeni, tip uporabnikov (ali skupine) in ali naj se uporabnika doda v SharePoint seznam uporabnikov na strani. V našem primeru tabela uporabnikov *users*, predstavlja prikazna imena uporabnikov, kot tip smo izbrali uporabniki, saj gre za uporabnike in ne za skupino. Kot zadnji parameter izberemo, da ne potrebujemo vnosa v seznam uporabnikov. Vrnjena vrednost je tabela z informacijami o uporabnikih (uporabniško ime, naziv, e-poštni naslov, itd.). Vsak uporabnik predstavlja svojo vrstico v tabeli. Te podatke lahko nato kasneje izkoristimo za pošiljanje elektronskega sporočila uporabnikom ali preprosto za preverjanje informacij o uporabnikih.

```
string[] users = new string[] { uporabnik , uporabnik2, uporabnik3 };
sharepointWS.people.People pe = new sharepointWS.people.People();
pe.Credentials = System.Net.CredentialCache.DefaultCredentials;

sharepointWS.people.PrincipalInfo[] principleInfoUser =
pe.ResolvePrincipals(users, sharepointWS.people.SPPrincipalType.User,
false);
```

Programska koda 1: Primer klica metode znotraj SharePoint spletne storitve.

Pridobivanje podatkov iz SharePoint seznamov je mogoče prav tako s pomočjo SharePoint objektnega modela. Tako kot pri SharePoint spletnih storitvah, je tudi tu potrebno posredovati poizvedbo, s pomočjo katere pridobimo podatke. Pisanje poizvedb je mogoče s pomočjo programskega jezika CAML [15]. CAML oz. "*Collaborative Application Markup Language*" temelji na jeziku XML. Korenski element je *Query*, znotraj njega pa sta mogoča dva elementa, in sicer element *OrderBy* ter element *Where*. Element `Where` je zahtevan element, za razliko od elementa `OrderBy`, ki je opcijski. Potrebno je paziti, kajti `Where` stavek lahko zelo hitro postane zelo zahteven, zato je dobro, če za boljšo preglednost oblikujemo stavke s praznim prostorom. V kolikor imamo več primerjalnih pogojev, te združujemo med seboj s pomočjo logičnih operatorjev (logičnih vrat) `And` in `Or`. Logični operatorji morajo biti na prvem mestu, takoj za `Where` elementom. Dovoljeno je gnezdenje

logičnih operatorjev, vendar je potrebno paziti, da bodo vsi ostali logični operatorji znotraj logičnih vrat *And* ali *Or*, v nasprotnem primeru pridemo do sintaktično nepravilnega stavka. Pomen logičnih operatorjev je opisan v tabeli 1. Prav tako je potrebno paziti, da vse operatorje na koncu z istoimensko značko tudi zapremo. S pomočjo elementa *FieldRef* lahko naslovimo katerikoli stolpec znotraj SharePoint seznama, s pomočjo elementa `Value` pa naslovljeni stolpec iz seznama preverjamo z vrednostjo. Elementu `FieldRef` je potrebno dodati atribut Name ali ID, s čimer naslovimo stolpec.

Logični operatorji znotraj `Where` stavkov:

Tabela 1: Pomen CAML logičnih operatorjev.

Izraz	Pomen
And	In
Or	Ali
Eq	Enako
Neq	Ni enako
Gt	Večje kot
Geq	Večje kot ali enako
Lt	Manjše kot
Leq	Manjše kot ali enako
IsNull	Je ničelne vrednosti
BeginsWith	Začne se s/z
Contains	Vsebuje

```
<Where>
   <And>
      <And>
         <And>
            <Eq>
               <FieldRef Name='Produkt' />
               <Value Type='Text'>Jogurt</Value>
            </Eq>
            <Geq>
               <FieldRef Name='TezaGram' />
               <Value Type='Number'>250</Value>
            </Geq>
         </And>
         <Contains>
            <FieldRef Name='Vsebnost' />
```

```
            <Value Type='Number'>Vitamin A</Value>
         </Contains>
      </And>
      <Eq>
         <FieldRef Name='DrzavaIzdelave' />
         <Value Type='Lookup'>Slovenija</Value>
      </Eq>
   </And>
</Where>
```

Programska koda 2: Primer sintaktično pravilne CAML poizvedbe.

Programska koda 2 prikazuje sintaktično pravilno poizvedbo, kadar imamo v poizvedbi več logičnih operatorjev. Pri pisanju CAML poizvedb pa je potrebno še posebej paziti, kajti vse značke zahtevajo točno sintakso in zato napačno pisanje velikih in malih črk (angl. case-sensitive) lahko privede do nedelujoče poizvedbe. Več informacij o sintaktično pravilno napisani CAML poizvedbi se dobi na spletnem naslovu [15].

Programska koda 3 prikazuje pridobivanje podatkov iz SharePoint seznam s pomočjo programskega jezika CAML preko SharePoint spletnih storitev. Pri gradnji aplikacije smo prišli do koraka, ko je potrebno iz SharePoint seznama pridobiti šifrante ali kakšne druge podatke. Po tem, ko smo uspešno dodali spletno referenco ter naredili tako deklaracijo kot tudi inicializacijo le-te, je potrebno oblikovati CAML poizvedbo. CAML jezik je vsesplošno uporaben programski jezik tipa XML. Zaradi tega, se poizvedbe pišejo s pomočjo objekta tipa *`XmlDocument`*. V poizvedbi smo najprej napisali, da naj se podatki sortirajo po atributu *`Version`*, s pomočjo opcijske značke *`OrderBy`*. V kolikor bi želeli sortirati po več različnih stolpcih, bi morali napisati več `OrderBy` stavkov. Lastnost `Ascending` lahko izpustimo, kajti privzeta vrednost je sortiranje po naraščajoči vrednosti. S pomočjo pogojev znotraj značke `Where` smo določili, katere vnose iz seznama sploh potrebujemo. Načeloma je CAML poizvedba končana in bi na tem koraku lahko končali, vendar pa ne potrebujemo vseh stolpcev iz seznama, zato je potrebno napisati še katere stolpce iz seznama želimo in tako omejimo rezultat. Tako dopišemo še dodatni CAML stavek, v katerem s pomočjo značke `FieldRef` napišemo vse stolpce, ki jih zahtevamo iz seznama. Spletni metodi *`GetListItems`* kot vhodni parameter tako sedaj posredujemo obe CAML poizvedbi, število vrnjenih vrstic, ime SharePoint pogleda, ime seznama itd. Spletna metoda podatke pridobi in v kolikor ni prišlo do napake, nam jih vrne v obliki, kot jo deklarira razred `XmlNode`.

```
sharepointWS.lists.Lists lists = new sharepointWS.lists.Lists();
lists.Credentials = System.Net.CredentialCache.DefaultCredentials;
XmlDocument camlDoc = new XmlDocument();
XmlNode queryNode = camlDoc.CreateElement("Query");
queryNode.InnerXml = @"<OrderBy>
```

```
                              <FieldRef Name='Version' Ascending='True'/>
                         </OrderBy>
                        <Where>
                             <Contains>
                                  <FieldRef Name='Title' /><Value
Type='Text'>Stranka</Value>
                             </Contains>
                         </Where>";
XmlNode viewNode = camlDoc.CreateElement("ViewFields");
viewNode.InnerXml = @"<FieldRef Name='Title' />
                      <FieldRef Name='ID_Pos' />
                      <FieldRef Name='Version' />";
XmlNode queryOptions = camlDoc.CreateElement("QueryOptions");
XmlNode resultNode = lists.GetListItems(listName, viewName, queryNode,
viewNode, rowLimit, queryOptions, webID);
```

Programska koda 3: Pridobivanje podatkov iz seznama s poizvedbo CAML preko spletnih storitev.

Zgornji primer bi lahko na enaki način preoblikovali v pridobivanje podatkov iz seznama s pomočjo SharePoint objektnega modela. Medtem, ko spletna metoda namesto nas opravi s povezovanjem na stran in seznam, moramo pri uporabi objektnega modela to narediti sami. Tako se najprej povežemo na SharePoint stran, nato je potrebno poiskati želeni seznam in narediti poizvedbo. Spodnja poizvedba se malo razlikuje od zgornjega primera, kajti tukaj zahtevamo vse zapise iz seznama, vendar nas zanimajo samo podatki zapisani v stolpcih *`Title`* in *`ID_Pos`*, vrnjene podatke pa želimo urejene po atributu *`ID_Pos`*. Vrnjene podatke lahko iterativno pregledujemo s pomočjo `foreach` zanke ali LINQ poizvedbe.

```
using (SPSite site = new SPSite(siteURL)) {
      using (SPWeb web = site.OpenWeb()) {
            SPList list = web.Lists[listName];
            SPQuery query = new SPQuery();
            query.ViewFields = @"<FieldRef Name='Title' />
                                 <FieldRef Name='ID_Pos' />";
            query.Query = @" <OrderBy>
                               <FieldRef Name='ID_Pos' />
                             </OrderBy>";
            if (list != null) {
                  SPListItemCollection items = list.GetItems(query);
                  foreach (SPListItem curItem in items) {
                        curItem["Title"].ToString();
                        curItem["ID_Pos"].ToString();
                  }
            }
      }
}
```

Programska koda 4: Pridobivanje podatkov iz seznama s poizvedbo CAML preko objektnega modela.

Pri poizvedbi s pomočjo SharePoint objektnega modela, kot prikazuje programska koda 4, je vrnjen seznam zahtevanih vnosov v obliki zbirke podatkov (*SPListItemCollection*) in ne kot *XmlNode*. Potrebno je omeniti, da mora razvijalec imeti dostop do SharePoint strežnika, v kolikor razvija aplikacijo iz druge lokacije. V kolikor razvijalec ne razvija na strežniku z nameščenim SharePoint portalom, je potrebno pri uporabi SharePoint objektnega modela dodati referenco na *Micorosft.SharePoint.dll* datoteko. Pomembno je tudi dejstvo, da če končna aplikacija ne bo gostovala na strežniku, kjer je nameščen SharePoint portal, je potrebno vso manipulacijo s SharePoint portalom razviti v obliki spletnih storitev, kajti dostop do SharePoint objektnega modela iz ločenega strežnika ni mogoč. Tako je praksa pripeljala do tega, da v primeru razvoja aplikacije za gostovanje na ločenem strežniku, poskrbimo za vso komunikacijo s SharePoint portalom z izgradnjo lastnih spletnih storitev. Te nato nastavimo na strežnik s SharePoint portalom, za tem pa jih po potrebi kličemo iz aplikacije. V kolikor se razvijalec odloči za uporabo SharePoint spletnih storitev, pa razvoj lastnih spletnih storitev ni potreben, zahteva se le, da bo aplikacija imela dostop do SharePoint spletnih storitev preko omrežja.

Dodajanje novega vnosa v seznam in pripenjanje datotek k temu vnosu je tako s pomočjo SharePoint objektnega modela narejena s spletno metodo znotraj lastne spletne storitve. Metoda *WriteNewItem* tako kot parametre sprejme naziv SharePoint strani, naziv SharePoint seznama, podatkovno tabelo `dtTable`, ki predstavlja vnos z izpolnjenimi polji, ki pa morajo biti poimenovani isto kot v seznamu, ter podatkovno tabelo `dtFiles`, ki predstavlja tabelo datotek, shranjenih v tabeli tipa `byte`.

```
[WebMethod]
        public int WriteNewItem(string _siteURL, string _listName,
DataTable dtTable, DataTable dtFiles) {
      try {
          using (SPSite site = new SPSite(_siteURL)) {
               using (SPWeb web = site.OpenWeb()){
                         web.AllowUnsafeUpdates = true;
                         SPList list = web.Lists[_listName];
                         SPListItem newItem = list.Items.Add();
                         newItem["Title"] =
dtTable.Rows[0]["PogodbaID"].ToString();
                         newItem["St_x0020_pogodbe"] =
dtTable.Rows[0]["StPogodbe"].ToString();
                         newItem["Stranka"] =
dtTable.Rows[0]["Stranka"].ToString();
                          ....

                         //Dodajanje priponk k trenutnemu vnosu v seznam
```

```
                        if (dtFiles != null) {
                            foreach(DataRow row in dtFiles.Rows) {
                                byte[] fileBytes =
                                    (byte[]) row["byteAttachment"];
            newItem.Attachments.Add(row["attachmentPath"].ToString(),
fileBytes);
                            }
                        }

                        newItem.Update();
                        web.AllowUnsafeUpdates = false;
                        return newItem.ID;
                    }
                }
            }
            catch(Exception ex) { ... }
        }
```

Programska koda 5: Primer lastne spletne metode.

V aplikaciji smo naredili kar nekaj lastnih metod, s katerimi smo pridobili želeno funkcionalnost. Uporabili smo tudi kar nekaj SharePoint spletnih storitev, predvsem za pridobivanje podatkov o uporabnikih in manipulacijo s seznamom. Primer lastne spletne metode prikazuje programska koda 5.

V nadaljevanju bo opisano kako s pomočjo orodij za izgradnjo delovnih tokov pridemo do končne rešitve za isto aplikacijo, vendar tokrat aplikacija sledi dejanskemu delovnemu toku.

3.1.3. Izdelava aplikacije z delovnim tokom s pomočjo Workflow Foundation

Na trgu se pojavljajo različna orodja, s pomočjo katerih je mogoče izdelati delovne tokov za portal SharePoint. Najbolj znano orodje se imenuje K2 [4], izdeluje ga skupina SourceCode Tehnology. Od izida SharePoint Server 2010 dalje, na trg prihajajo mlada in nova orodja. Predvideva se, da bo teh vedno več, kajti zadeva je trenutno še v razcvetu. Omenimo samo še nekaj orodij, to so na primer Nintex Workflow 2010 podjetja Nintex [17] ali Workbox for SharePoint podjetja DataPolis [18] in še kaj bi se našlo. Potrebno je preveriti kaj ta orodja omogočajo, kajti ne omogočajo vsa orodja implementacijo lastne programske kode znotraj delovnega toka. Vendar pa v kolikor imamo nameščen SharePoint strežnik in orodja kot sta Visual Studio 2010 ali SharePoint Designer 2010, lahko delovne tokove pripravimo kar sami - s pomočjo Windows Workflow Foundation. Delovne tokove lahko pripravimo tudi pri starejših različicah tako SharePoint strežnika kot tudi orodja Visual Studio in SharePoint Designer.

Workflow Foundation ni produkt, temveč gre za skupek tehnologij, narejenih v ogrodju .NET Omogoča izgradnjo delovnih tokov, ki so opisani kot zbirka programiranih korakov oz. faz, ki jih Workflow Foundation poimenuje aktivnosti. Ogrodje .NET omogoča zbirko aktivnosti iz katere je mogoče izbirati in graditi delovni tok, prav tako pa je mogoča izgradnja lastnih programiranih aktivnosti. Pogon Workflow Foundation tako skrbi za izvedbo delovnih tokov in aktivnostih v njih, prehode med aktivnosti, shranjevanje delovnega toka v podatkovno bazo in tako sprostitev spomina. Prav tako skrbi za upravljanje s podatki za trenutne aktivnosti v izvajanju, sledenje izvajanja delovnega toka in še marsikaj.

SharePoint delovni tok tako ni isto kot Windows Workflow Foundation. Windows Workflow Foundation predstavlja osnovo za SharePoint delovne tokove. SharePoint delovni tok je opisan kot zbirka nalog, ki proizvedejo nek izid. Gre za avtomatizirano premikanje dokumentov ali zapisov znotraj seznama, skozi sekvenco akcij ali nalog, ki so povezane s poslovnim procesom.

V tem razdelku bomo pogledali izgradnjo aplikacije in delovnega toka s pomočjo Windows Workflow Foundation, ki ga uporablja SharePoint Server. S pomočjo orodja Visul Studio 2010 Workflow Designer lahko naredimo delovne tokove, ki vsebujejo tudi programsko kodo za bolj kompleksne aplikacije. SharePoint Designer 2010 omogoča izdelavo preprostejših in deklarativno usmerjenih delovnih tokov, kjer v ozadju nimamo lastne programske kode. Prav tako je razlika pri uporabi Visual Studio 2010 Workflow Designer in SharePoint Designer 2010 v tem, da pri slednjem kot izid izdelamo skupek datotek na podlagi XML programskega jezika, medtem ko v Visual Studio gradimo projekt in na koncu kot rezultat dobimo zbirko programske kode.

Prejšnje SharePoint različice prav tako omogočajo namestitev lastnih delovnih tokov, vendar pa orodja za izgradnjo še niso bila vgrajena direktno v sam Visual Studio. Tako je bilo potrebno za orodje Visual Studio poiskati in namestiti razširitev, katera je nato omogočila izgradnjo delovnih tokov. Brez te razširitve, je bila mogoča izgradnja samo s pomočjo orodja SharePoint Designer oz. preko različnih t.i. orodij tretjih strank (angl. third party tools).

Razvijalec lahko izdela delovne tokove, pri katerih uporabnik sproži začetek procesa ali takšne, pri katerih se začetek procesa sproži avtomatično na podlagi izvedenega dogodka, npr. dodana ali spremenjena datoteka. Ob namestitvi delovnega toka na seznam, se tako nastavi t.i. prejemnik dogodkov (angl. event receiver), ki posluša za določen tip dogodkov. V primeru delovnega toka, se tako nastavi dogodek "vnos dodan" (`itemAdded`) ali "vnos popravljen" (`itemUpdated`) in v kolikor imamo delovni tok tako nastavljen, se ob oddaji novega vnosa v seznam naredi in požene nova instanca delovnega toka.

SharePoint podpira dva osnovna tipa delovnih tokov:

- **Sekvenčni oz. zaporedni delovni tok** (angl. Sequential workflow) predstavlja delovni tok, ki ima začetek, konec ter zaporedni tok dogodkov od začetka do konca. Ti delovni tokovi lahko vsebujejo ponavljanje dogodkov, gnezdenje in paralelno vejitev. Načeloma ti delovni tokovi niso čisto sekvenčni oz. zaporedni v polnem pomenu besede, saj lahko skozi proces prejmejo zunanje dogodke ali vsebujejo paralelni tok izvajanja, kar je razlog, da se lahko potek aktivnosti spreminja. Sekvenčni delovni tokovi so najbolj uporabljani in se jih, kot bomo videli kasneje, poslužujejo tudi druga orodja za izgradnjo delovnih tokov.

- **Delovni tok stanja stroja** (angl. State machine workflow) predstavlja skupek stanj, prehodov in akcij. Neko stanje je označeno kot začetek stanja. Vsako stanje lahko prejme dogodek. Na podlagi izvedenega dogodka se izvede prehod na drugo stanje. Tako proces prehaja med stanji do stanja, ki opisuje končno stanje.

Pred izdelavo delovnega toka se je potrebno odločiti, kakšen delovni tok bomo izdelovali. Delovni tok je pripet samo k enemu seznamu ali knjižnici, zaradi tega je potrebno, da še pred izdelavo delovnega toka skrbnik portala ali razvijalec izdela SharePoint seznam ali knjižnico z vsemi potrebnimi stolpci.

Pri izdelavi aplikacije za potrjevanje nastanka pogodb smo izbrali sekvenčni delovni tok. S pomočjo orodja Visual Studio odpremo nov projekt in že se pred nami prikaže čarovnik, preko katerega naredimo povezavo na SharePoint stran. Čarovnik nam ponudi izbiro seznama ali knjižnice, kjer se bo nahajal delovni tok, seznama zgodovine vseh instancah delovnih tokov (angl. History List) ter seznama nalog (angl. Task List). Izberemo še način, kdaj želimo, da se naredi in požene nova instanca delovnega toka. Možno je izbirati med ročnim zagonom delovnega toka s strani uporabnika, zagon delovnega toka, ko uporabnik odda nov zapis v seznam ter, ko uporabnik spremeni oz. popravi zapis v seznamu. Razvijalec se sam odloči kakšen zagon delovnega toka želi, lahko je izbranih več možnosti, ni pa mogoče izvoziti delovni tok, v kolikor nimamo izbrane vsaj ene možnosti..

Pri izgradnji SharePoint delovnega toka velik pomen predstavlja korelacijski žeton (angl. correlation token), ki določa enolično identifikacijsko oznako, katera omogoča razpoznavanje med objekti znotraj delovnega toka in okoljem, kjer gostuje Windows Workflow Foundation. Na podlagi izvedenega dogodka pride zahteva z žetonom v Workflow Foundation izvajalca, kjer se na podlagi žetona odloči h kateremu delovnemu toku je potrebno poslati to zahtevo. Windows Workflow Foundation se tako obnaša kot neke vrste posrednik za vse komunikacije s programsko opremo zunaj delovnega toka. Vsak delovni tok ima tako svoj korelacijski žeton. Naloge znotraj delovnega toka lahko uporabljajo isti korelacijski žeton kot delovni tok,

razen, če želimo neko nalogo (angl. task) nasloviti iz druge naloge - takrat naredimo za to nalogo lasten korelacijski žeton.

```
public sealed partial class Workflow1 : SequentialWorkflowActivity
    {
      ...
        public Guid workflowId = default(System.Guid);
        public SPWorkflowActivationProperties workflowProperties = new
SPWorkflowActivationProperties();
```

Programska koda 6: Lastnosti delovnega toka

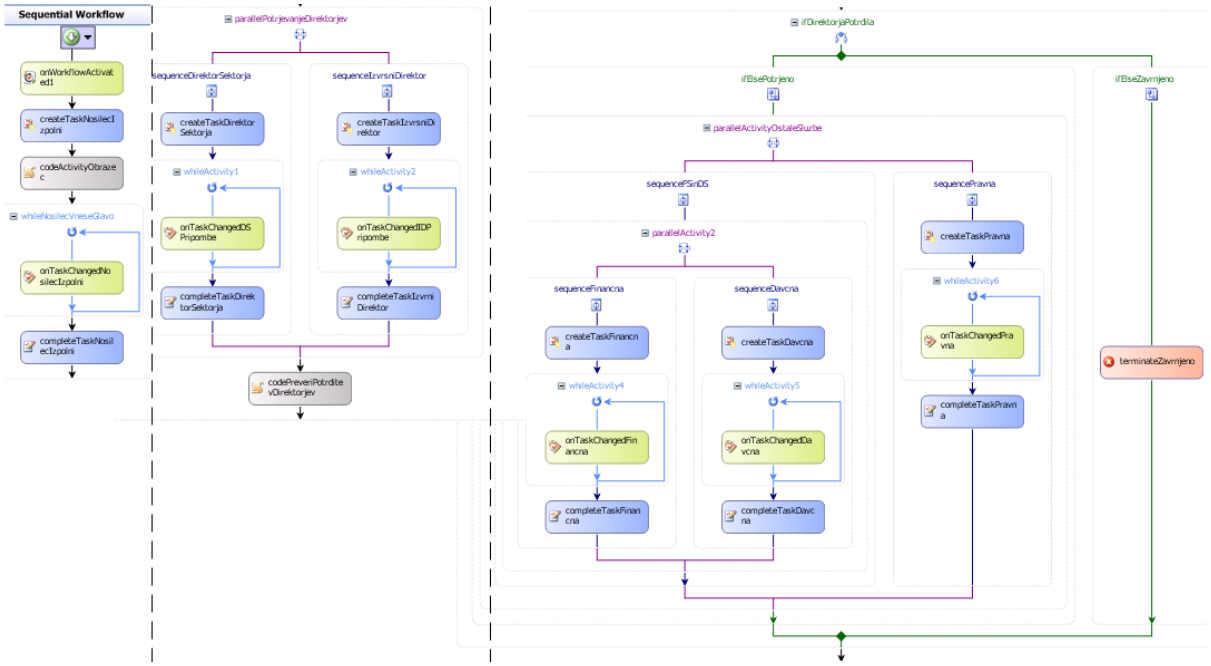
Zelo pomembno je omeniti lastnosti delovnega toka, ki so shranjene v razredu *SPWorkflowActivationProperties*, kot prikazuje programska koda 6. Vsak delovni tok ima instanco tega razreda in notri shranjene podatke o SharePoint spletni aplikaciji, SharePoint strani, seznamu, vnosu, o uporabniku, ki je zagnal delovni tok in še marsikaj. Gre torej za res uporabno zbirko informacij o delovnemu toku. Pri nekaterih nalogah je pomembno, da hranimo lastnosti o nalogi pred in po izvedbi te naloge. V ta namen lahko uporabljamo razred *SPWorkflowTaskProperties*. Vsi delovni tokovi se začnejo z inicializacijo lastnosti delovnega toka in z nalogo oz. aktivnostjo *onWorkflowActivated*. S pomočjo te aktivnosti se naredi korelacija med žetonom in lastnostjo *workflowId*. Omenjena aktivnost mora biti nujno prva v delovnem toku in tudi če se pojavi večkrat, jo SharePoint ignorira, v kolikor korelacija uspe. V delovni tok lahko nato vstavljamo poljubne aktivnosti oz. naloge, kakor tudi pišemo lastno izvorno kodo. Prav tako lahko vstavljamo vejitve `if-else` ter paralelno izvajanje.

Pri razvijanju SharePoint delovnih tokov je potrebno vedeti, da v kolikor ne razvijamo vse s pomočjo programske kode, celotni delovni tok temelji na t.i. nalogah. Pred vsako aktivnostjo, kjer mora uporabnik pregledati ali oddati zahtevane podatke, naredimo novo nalogo za akterja. Ta v seznamu nalog (angl. Task List), katerega smo navedli v začetku pri ustvarjanju projekta za delovni tok, doda novo nalogo za vse določene uporabnike. Tem se nato ob prijavi v portal v seznamu prikažejo naloge, ki jih morajo opraviti. V seznamu ob imenu naloge se prav tako pojavi status naloge, prioriteta in rok za zaključek naloge ter koliko odstotkov naloge je že zaključenih. S klikom na nalogo se uporabniku odpre splošen obrazec za obdelavo ali poseben obrazec za točno določeno nalogo. Po uspešno zaključeni nalogi, se proces premakne naprej po delovnem toku. Ustvarjanje naloge znotraj SharePoint delovnega toka je mogoče s kontrolnikom `createTask`. Ta pokliče metodo, v kateri s programsko kodo naredimo novo nalogo za zaposlenega. Nastavljanje lastnosti znotraj naloge je mogoče s predhodno inicializacijo razreda *SPWorkflowTaskProperties*, katerega nato povežemo z nalogo. Primer dodeljevanja lastnosti naloge znotraj delovnega toka prikazuje programska koda 7. V primeru smo dodelili akterja, datum zapadlosti naloge in naslov naloge.

```
public SPWorkflowTaskProperties createTaskDirektor_TaskProperties = new
Microsoft.SharePoint.Workflow.SPWorkflowTaskProperties();

private void createTaskDirektor_MethodInvoking(object sender, EventArgs e)
{
            createTaskDirektor_TaskProperties.AssignedTo = izbraniDirektor;
            createTaskDirektor_TaskProperties.DueDate = new
DateTime(DateTime.Now.Year, DateTime.Now.Month,
DateTime.Now.AddDays(1).Day);
            createTaskDirekt_TaskProperties.Title = "Pregled in oddaja
pripomb direktorja";
}
```

Programska koda 7: Primer dodeljevanja lastnosti naloge.

Zaradi zaporedne izgradnje SharePoint delovnega toka, je ob zahtevnih delovnih tokovih preglednost slabša. Prav tako pri izgradnji ni mišljeno sočasno oz. paralelno izvajanje več kot dveh aktivnosti. V kolikor proces zahteva sočasno izvajanje več kot dveh akterjev, je potrebno improvizirati s pomočjo paralelnih vejitev in pogojev, kot je to razvidno iz slike 12.



Slika 12: Del razvitega SharePoint delovnega toka.

SharePoint delovni tok tako načeloma ni namenjen izgradnji zahtevnih delovnih tokov, temveč za bolj enostavne, kot je na primer poročanje o procesu, potrjevanje dokumentov ipd. V kolikor so želje večje od možnosti izgradnje s pomočjo SharePoint delovnega toka, je

potrebno poseči po drugih, tujih orodjih. V naslednjem razdelku je opisana uporaba orodja za izgradnjo bolj zahtevnejših delovnih tokov.

3.1.4. Izdelava delovnega toka s pomočjo orodja K2

K2 predstavlja ime za orodja, namenjena izdelavi delovnih tokov in upravljanju s poslovnimi aplikacijami [14]. Vizualna razvojna orodja, kakršno so tudi orodja K2, omogočajo ljudem lažjo izdelavo delovnih tokov in aplikacij, ki avtomatizirajo potek procesa. Enostavnost popravljanja procesa omogoča, da je ob potrebnih spremembah v procesu, to možno narediti zelo enostavno. K2 orodja se na trgu pojavljajo že od leta 2000, tako da so bila razvita že za predhodne različice SharePoint portala.

Trenutno so na voljo tri različice K2 orodij [14]: *K2 Blackpearl*, *K2 Blackpoint* ter *K2 Connect*. K2 orodja so zgrajena na Microsoft tehnologiji in dopolnjujejo programe kot sta Visual Studio in SharePoint. Osnovna ideja K2 orodij je, da ima uporabnik možnost grafičnega oblikovanja delovnih tokov. Tako je izdelava delovnih tokov zelo podobna risanju diagramov poteka v programu Visio oz. drugih podobnih okoljih. S tem se razbremeni končnega uporabnika, saj se mu ni potrebno učiti novih veščin.

Medtem, ko je *K2 Connect* samo orodje, ki se uporablja za povezavo SAP informacij z aplikacijo in delovnimi tokovi narejenimi v K2 Blackpearl, sta K2 Blackpearl in K2 Blackpoint dejansko namenjena izgradnji delovnih tokov in procesno usmerjenih aplikacij.

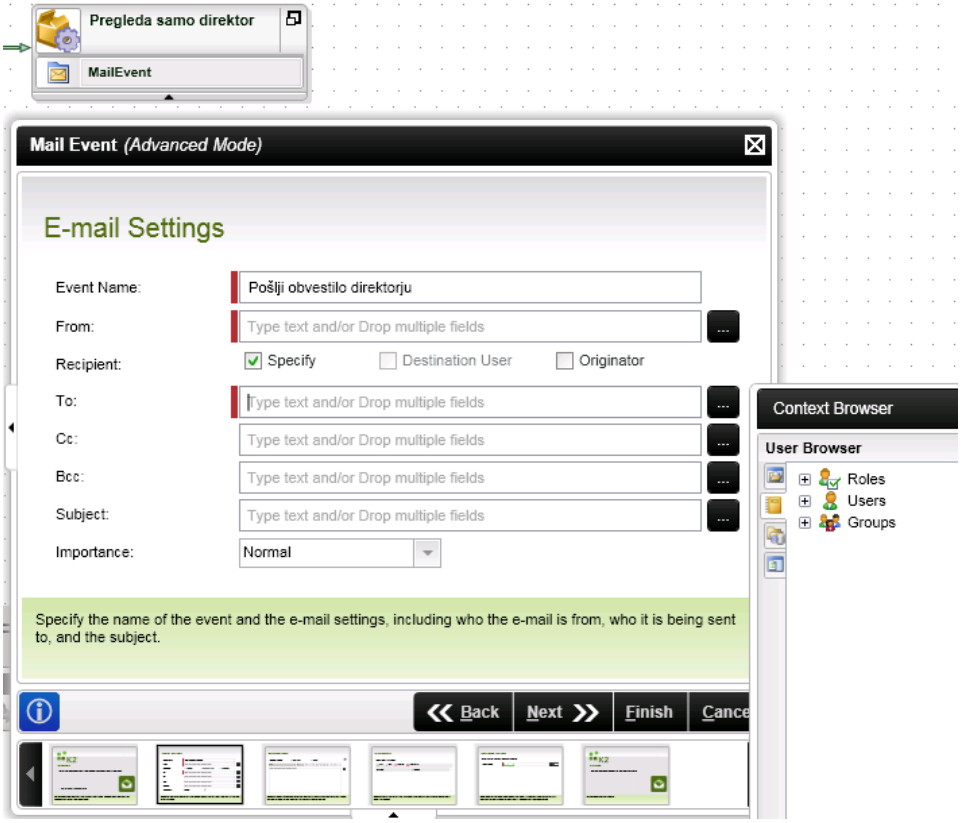
K2 Blackpoint je namenjen SharePoint uporabnikom in skrbnikom portala. Je grafično orodje in omogoča enostavno grajenje delovnih tokov brez pisanja programske kode. Ravno zaradi tega ni primeren za gradnje zelo kompleksnih delovnih tokov. Namenjeno je hitri izgradnji enostavnih delovnih tokov in procesno usmerjenih aplikacij. Do delovnega okolja dostopamo preko spletnega brskalnika. Programska rešitev je izdelana s pomočjo tehnologije Silverlight [19] in okoljem .NET.

K2 Blackpearl je trenutno najbolj razvito orodje, ki omogoča izgradnjo zahtevnejših delovnih tokov in procesno usmerjenih aplikacij. Prav zaradi večletne tradicije, je to orodje zelo močno in omogoča izgradnjo tudi najbolj kompleksnih delovnih tokov. Prav tako ponuja grafično okolje za hitrejše delo, omogoča pa tudi vključevanje programske kode znotraj procesov. Programsko okolje za izgradnjo delovnih procesov so v tej različici preselili kar znotraj orodja Visual Studio. Vseeno pa so nekatera orodja, predvsem administracija procesov, dostopna preko spletnega brskalnika.

Zaradi omejitev pri uporabi K2 Blackpoint, se za pisanje zahtevnejših delovnih tokov, kjer je v ozadju potrebna lastna programska koda, uporablja orodje K2 Blackpearl, s pomočjo katerega je spodaj prikazana izdelava skoraj ekvivalentnega delovnega toka, kot v dejansko

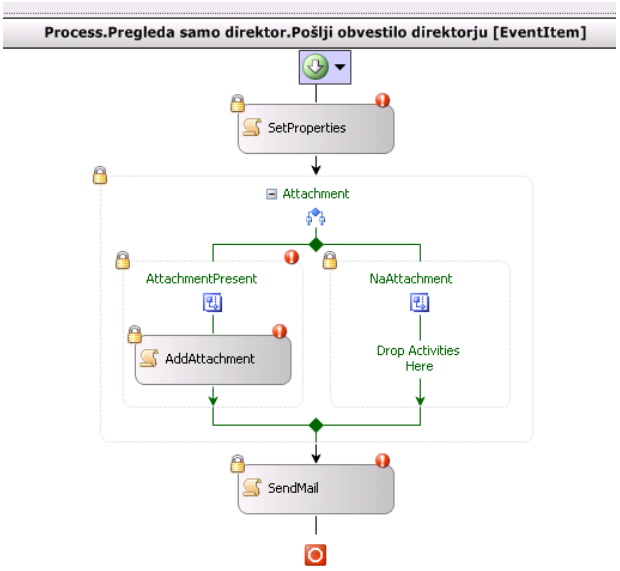
izdelani končni aplikaciji za podporo potrjevanja nastanka pogodbe in izvedi le-te s pomočjo Windows Workflow Foundation.

Za razliko od razvite programske rešitve, ki je v celoti spisana s programsko kodo in ne vsebuje dejanskega delovnega toka, pri orodju K2 poteka razvoj delovnega toka preko grafičnega vmesnika. Prav tako se K2 orodje razlikuje od Windows Workflow Foundation po terminologiji, kajti tukaj delamo z aktivnostmi in dogodki. S pomočjo Visual Studio Workflow Designer je prav tako možno aktivnosti oz. dogodke v proces dodajati s pomočjo t.i. metode "povleci in spusti" (angl. "drag and drop"). Ob vsakem dodajanju novih dogodkov se pojavi čarovnik za namestitev tega dogodka, kot je to prikazano na sliki 13. S tem se pohitri sam proces izdelave.



Slika 13: Čarovnik pri dodajanju dogodkov - K2 BlackPearl.

V kolikor želi razvijalec ročno popraviti programsko kodo v dogodku, je potrebno izbrati želeni dogodek in s pomočjo desnega klika dobimo možnost vpogleda v kodo `View Code - Event Item`. Pred vpogledom v programsko kodo, se najprej odpre delovni tok dogodka.



Slika 14: Vpogled v programsko kodo dogodka v K2 BlackPearl.

Iz zgornje slike 14 je razvidno, da K2 preoblikuje lastne dogodke, kot je npr. dogodek za pošiljanje pošte, v sekvenčni oz. zaporedni delovni tok, kot ga uporablja SharePoint in ga upravlja Windows Workflow Foundation (opisano v razdelku 3.1.3). Ta podatek je zelo pomemben, kajti iz tega je razvidno, da se celoten delovni tok, ki ga izdelamo s pomočjo K2 grafičnega okolja, pretvori v obliko delovnega toka kakršno uporablja in poganja Workflow Foundation. Na takšen način orodje K2 preoblikuje vsak dogodek, ki ga vstavimo v K2 delovni tok in ponastavimo s pomočjo čarovnika. Kot lahko vidimo, K2 ne uporablja kontrol `Task`, kot je to poznano iz SharePoint delovnih tokov, temveč le t.i. kontrolo `Code`, v katerih je vsa logika sprogramirana. Tako se generira osnovni zaporedni delovni tok za vsak dogodek, skupaj s programsko kodo, v kateri so podatki o dogodku, ki smo jih nastavili preko čarovnika. Iz tega je razvidno, da je še tako kompliciran delovni tok, ki ga naredimo s pomočjo K2 grafičnega orodja, mogoče preurediti oz. narediti z sekvenčnim delovnim tokom, s pomočjo Windows Workflow Foundation, kar dokazuje, da je WWF osnova za izgradnjo delovnih tokov.

Sedaj, ko imamo odprt delovni tok dogodka, je dodajanje lastne programske kode v K2 dogodke enako, kot pri že opisanem sekvenčnem delovnem toku WWF. To naredimo tako, da na primerno mesto dodamo t.i. kontrolo `Code`, nanjo dvokliknemo ali z desnim klikom izberemo `View Code` in nato dopišemo programsko kodo (glej programsko kodo 8).

```
private void codeActivity_ExecuteCode(object sender, EventArgs e) {
      programski stavki za izvedbo...
}
```

Programska koda 8: Metoda za izvedbo lastne programske kode znotraj procesa.

Prednost pri uporabi K2 orodja je, da lahko naredimo podatkovna polja, v katerih bomo hranili informacije skozi izvajanje procesa. Do informacij znotraj delovnega toka preko programske kode dostopamo s pomočjo instance razreda K2, ki je izpeljana s pomočjo razreda `EventItemContext`.

```
public class EventItemContext_[generiranID] {
        private SourceCode.KO.ClientEventContext _context;
        private SourceCode.Framework.Data.ResolverManager _resolverManager;
        private ExtenderConfig _config;
        public
EventItemContext_[GUID_Dogodka](SourceCode.KO.ClientEventContext context,
SourceCode.Framework.Data.ResolverManager resolverManager) {
            this._context = context;
            this._resolverManager = resolverManager;
            _config = new ExtenderConfig(this);
        }

        ...

        public SourceCode.KO.StringTable StringTable {
            get { return this._context.StringTable; }
        }

        public SourceCode.KO.ProcessInstance ProcessInstance {
            get { return this._context.ProcessInstance; }
        }

        public ExtenderConfig Configuration {
            get { return _config; }
        }
        public SourceCode.KO.ActivityInstanceDestination
ActivityInstanceDestination {
            get {
                return this._context.ActivityInstanceDestination;
            }
        }
        ...
}
public Project_[GUID_Projekta].EventItemContext_[GUID_Dogodka] K2
        {
            get { return _k2; }
            set { _k2 = value; }
          }
```

Programska koda 9: Avtomatično implementirane metode znotraj razreda `EventItemContext`.

Programska koda 9 se generirana avtomatično za vsak dodani dogodek znotraj delovnega toka. Metode, ki so implementirane znotraj kontekstnega razreda `EventItemContext`, nam omogočijo manipulacijo nad vsemi lastnostmi dogodka, prav tako pridobivanje podatkov iz podatkovnih polj v okviru celotnega delovnega toka (t.i. `Data Fields`, `String Table` ipd.).

S pomočjo teh metod lahko znotraj programske kode preskakujemo iz ene aktivnosti v drugo, nastavljamo končne uporabnike za določeno aktivnost, ročno začnemo in končamo določeno aktivnost in še marsikaj.

Podatkovna polja lahko prav tako nastavimo na nivoju dogodka oz. aktivnosti in do njih dostopamo preko *`K2.ProcessInstance.ActivityInstances[]`*, kakor tudi na nivoju celotnega procesa.

Primer uporabe razreda `EventItemContext` in klic podatkovnega polja s pomočjo programske kode prikazuje spodnja programska koda 10.

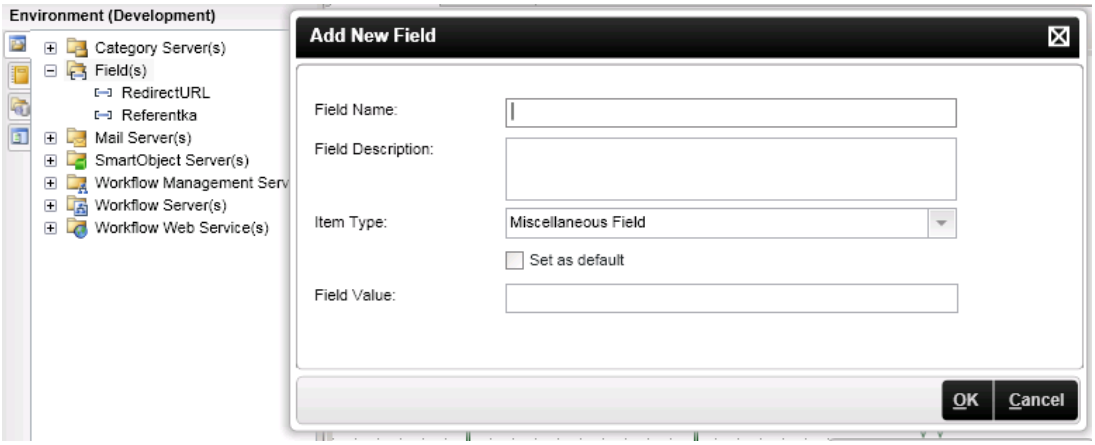
```
string strIme = K2.ProcessInstance.DataFields["ImePriimek"].ToString();

string sqlConnectionString = K2.StringTable["SqlConnStr"].ToString();
```

Programska koda 10: Pridobivanje podatkov iz `DataFields` in `StringTable`

Pri podatkovnih poljih je potrebno prav tako omeniti razliko med t.i. *`Data Fields`* in *`String Table`*. Prvi predstavljajo podatkovno polje, ki ga kot že omenjeno, lahko nastavimo na ravni aktivnosti ali na ravni procesa. Vrednosti lahko v vsakem trenutku preberemo ali spreminjamo. Polja so lahko različnih tipov: nizi, število, datum in čas, itd. Vsaka instanca procesa ima svojo tabelo podatkovnih polj (`Data Fields`) in se uporablja na primer za vpisovanje imen, informacij o uporabniku, različne številčne vrednosti, datumi zahtevka itd.

Tabela nizov znakov (*`String Table`*) je namenjen podobni uporabi kot *`web.config`* pri .NET aplikacijah. Predstavlja tabelo podatkovnih polj tipa `string` (niz znakov), v katerega dodamo ime polja ter vrednost polja v obliki niza znakov. Vrednosti so statične in za vse instance procesa enake. Uporablja se za vpisovanje vrednosti, ki se ne bodo spreminjale skozi proces, torej povezovalni niz (angl. `connection string`) za dostop do podatkovne baze, imena strežnikov, ime SharePoint strani in seznama ipd. Slika 15 prikazuje dodajanje polj v tabelo nizov znakov preko čarovnika.

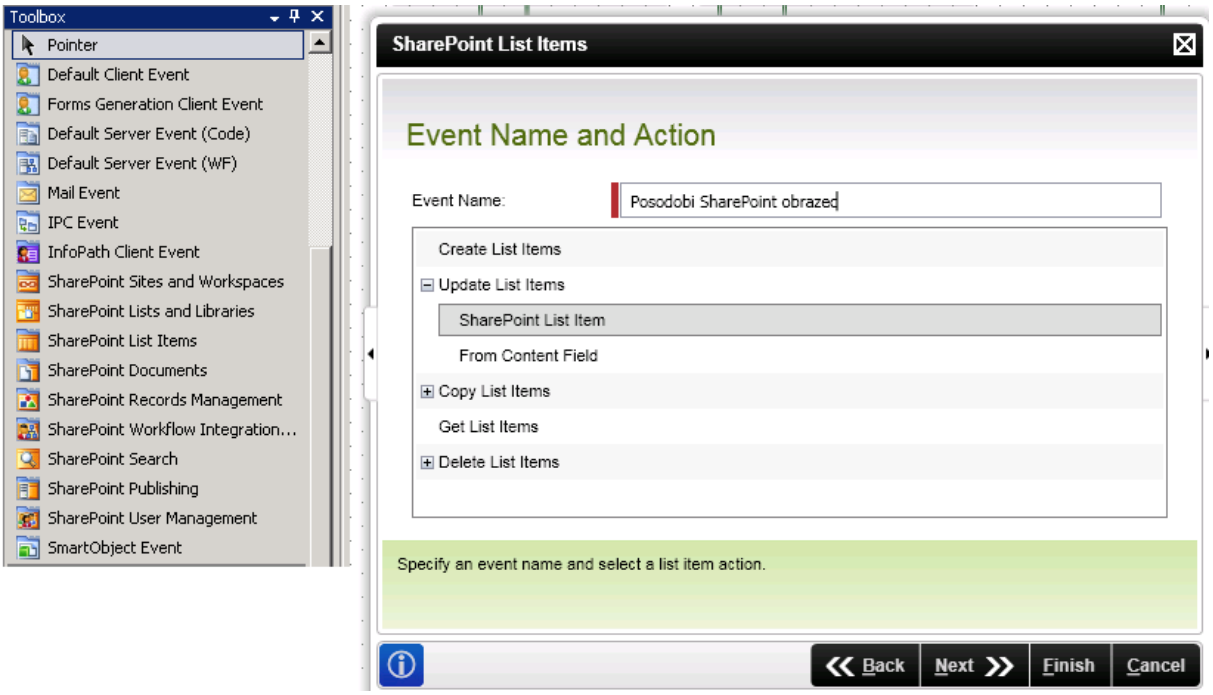
Potrebno je omeniti, da med razvojem procesa znotraj orodja K2 Blackpearl dodajamo nize v razdelek *Environment*, ki se kasneje, ko naredimo izvoz na produkcijski strežnik, kjer bo aplikacija gostovala, preslikajo v omenjeni *StringTable*. V kolikor spreminjamo vrednosti podatkovnih polj znotraj Environment spremenljivk se ta vrednost ne prenese avtomatično na produkcijo, zato je potrebno vsakič, ko spremenimo vrednosti narediti še izvoz.



Slika 15: Dodajanje polj v tabelo nizov znakov.

Interakcija SharePoint portala preko K2 delovnega toka je mogoča z vpeljavo dogodkov oz. aktivnosti. K2 vsebuje celo paleto dogodkov, ki omogočajo izvajanje vseh administrativnih nalog, kot smo jih vajeni ročno opravljati na portalu SharePoint ali programersko preko objektnega modela ali SharePoint spletnih storitev. K2 prav tako omogoča sodelovanji z drugimi Microsoft orodji, med drugimi sem spada tudi InfoPath. Ta je primeren predvsem za izgradnjo obrazcev, katere lahko povežemo z delovnim tokom.

Na sliki 16 je prikazan čarovnik za povezovanje K2 orodja s SharePoint seznamom. Kot vidimo je preko čarovnika omogočeno posodabljanje, kopiranje, brisanje in ustvarjanje vnosov v seznamu.



Slika 16: Interakcija med SharePoint portalom in K2 orodjem.

V kolikor imamo lasten ASP.NET obrazec, lahko le-tega prav tako povežemo s K2 delovnim tokom. V obrazcu imamo običajno nekaj vnosnih polj, katere uporabniki izpolnijo, nato pa jim sledi gumb za oddajo vnosa. Poleg vse poslovne logike, ki jo zahteva poslovni proces in se zgodi ob kliku na omenjeni gumb, je potrebno vpisati tudi programsko kodo za interakcijo s K2 procesom. Programska koda 11 prikazuje lastno metodo za ustvarjanje nove instance K2 procesa.

```
public void CreateProcessInstance(Collection<CollectionItem>
dataCollection) {
      try {
            //Branje nastavitev
            string K2Server = ConfigurationManager.AppSettings["K2Server"];
            string K2ProcessName =
ConfigurationManager.AppSettings["ProcessName"];
            //Povezava na K2 server
            var K2Connection = new SourceCode.K2ROM.Connection();
            K2Connection.Open(K2Server);
            //Ustvarimo novo instanco K2 procesa
            var process =
K2Connection.CreateProcessInstance(K2ProcessName);
            //Vnesemo podatke iz zbirke v K2 proces
            if ((dataCollection != null)) {
                  foreach (var i in dataCollection)
```

```
                        process.DataFields[i.Name].Value = i.Value;
                    }
                    //Zaženemo K2 proces
                K2Connection.StartProcessInstance(process, true);
        }
        catch (Exception ex) { ... }
        finally {
                    K2Connection.Close();
                }
        }
}
```

Programska koda 11: Ustvarjanje nove instance K2 procesa.

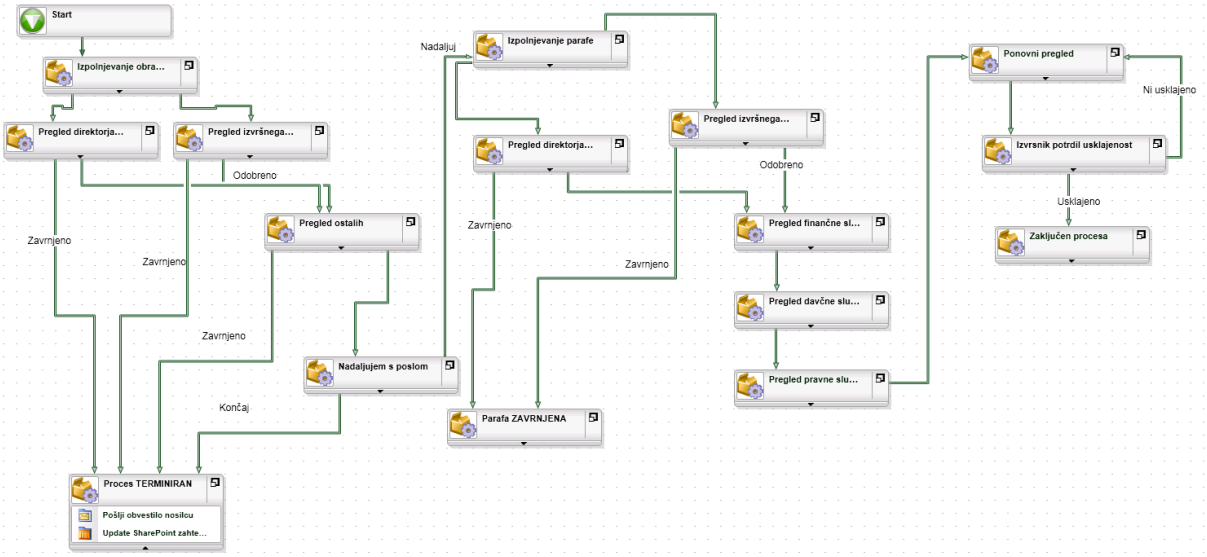
Metoda `CreateProcessInstance(Collection<T>)` omogoča ustvarjanje nove instance znotraj K2 procesa. Najprej povezavo s pomočjo razreda *`K2ROM.Connection`* ustvarimo s klicem metode *`Open(String)`*. Nato znotraj K2 procesa, ki smo ga izvozili že ob izdelavi delovnega toka s pomočjo orodja K2 BlackPearl naredimo novo instanco. To za nas opravi klic metode *`CreateProcessInstance(String)`*, ki pa je metoda znotraj razreda *`K2ROM.Connection`*. V kolikor do tukaj ni prišlo do nobenih napak, je sedaj mogoče iz naše zbirke prepisati shranjene podatke iz obrazca v novo instanco procesa. Pri načrtovanju zbirke, je potrebno paziti, da imamo poimenovanje stolpcev enako kot je poimenovano v K2 podatkovni tabeli. Za konec instanco procesa še zaženemo in s tem dejansko proces poženemo. Nato K2 strežni na podlagi narejenega delovnega toka in vpisanih podatkov, primerno usmeri potek procesa.

Pri dogodkih, kjer moramo podatke v K2 procesu osvežiti, je programska koda podobna kot pri zgornjem primeru, le da ne uporabimo klica metode *`CreateProcessInstance(Collection<T>)`*, ampak pokličemo metodo *`OpenWorklistItem(String)`*, kateri podamo številko instance K2 procesa. Metoda vrne instanco delovnega toka s pomočjo razreda *`Worklist`*. V njej so zapisani vsi podatki o procesu. Popravljanje oz. osveževanje podatkov v procesu je nato mogoče na isti način kot v zgornjem primeru. Na koncu je potrebno instanco delovnega razreda zapreti s klicem metode *`worklistitem.Finish()`*.

K2 strežnik skrbi za izvajanje vseh instanc delovnih tokov. Po uspešnem zagonu instance, K2 na podlagi podatkov in pogojev proces usmerja na pravo pot. Znotraj delovnega toka imamo lahko aktivnosti, kjer morajo svojo nalogo narediti tudi drugi akterji iz podjetja. Ob prihodu na takšno aktivnost K2 strežnik postavi instanco procesa v čakanje, vse dokler zahtevani uporabnik ne odpre obrazca in ga obdela oz. sprosti. Prav tako lahko na vsako aktivnost nastavimo t.i. pravilo stopnjevanja (angl. `escalate rule`), ki se obnaša kot nekakšen

časovnik. Tako lahko v primeru, da aktivnost predolgo čaka na uporabnikovo dejanje, točno določenemu uporabniku začnemo pošiljati opomnike. Datum in ura kakor tudi število ponovitev je prosto nastavljiva. V kolikor željene funkcionalnosti ne moremo doseči s čarovnikom, pa to lahko izdelamo s pomočjo programske kode. Prav tako lahko preskočimo aktivnost in gremo na naslednjo.

Vsaka črta v procesu med aktivnostmi predstavlja možno pot. Pot je možna samo v smeri puščice ali s preskoki, ki so bolj izjema kot pravilo. Vsaki črti je možno nastaviti pogoje, katerim mora instanca procesa ustrezati, v kolikor želi pot nadaljevati po tej črti. V nasprotnem primeru pot po tej črti ni mogoča. Tako se lahko hitro zgodi, da če nismo pravilno načrtovali, da proces obstane v kakšni aktivnosti, saj noben od pogojev ne ustreza in proces ne more nadaljevati poti.

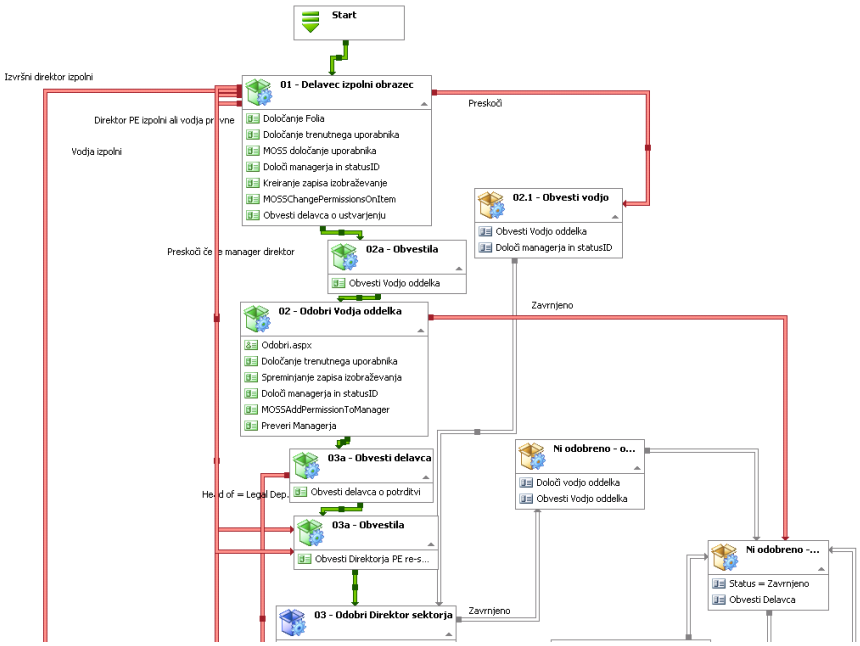


Slika 17: Izdelan proces z orodjem K2 BlackPearl.

Zgornja slika 17 prikazuje zahtevnejši proces, ki je do neke mere enakovreden končni rešitvi, ki smo jo izdelali v podjetju, vendar s pomočjo programske kode. Kot vidimo, je proces v grafičnem okolju veliko bolj preglednejši in nam omogoča hitrejše popravke. V kolikor bi bil proces še večji, se lahko lažje in hitreje pomikamo z drsniki skozi cel proces, povečujemo in pomanjšamo velikost narisanega procesa. Vsaka aktivnost znotraj procesa vsebuje dogodke, ki jih prikažemo s klikom na puščico znotraj okenca aktivnosti. Tako dobimo pregled nad vsemi dogodki in z desnim klikom nanje lahko tudi hitro pregledujemo programsko kodo ali čarovnika.

Izvoz delovnega toka naredi programer s klikom na gumb `Deploy`. Pojavi se čarovnik za izvoz delovnega toka, ki nam ponudi izvoz v razvojno ali produkcijsko okolje. S tem se omogoči razvijalcem, da imajo ločene podatkovne tabele za razvojno in produkcijsko okolje, ter šele ko je uspešno zaključeno testno obdobje, zadevo preselijo v produkcijsko okolje.

Pri prenosu procesa v produkcijsko okolje, se znotraj K2 strežnika naredi mesto, ki predstavlja naš novi proces. K2 strežnik vse podatke, tako `StringTable`, `DataTable`, kakor tudi celoten proces shranjuje v tabele znotraj podatkovne baze. Vsaka instanca bo tako imela svoj zapis v tabeli. Informacije o zagnanih procesih je mogoče dobiti preko aplikacije `K2 Workspaces`. Ta prebira podatke iz podatkovne baze in jih prikazuje znotraj spletnega brskalnika, v uporabniku prijaznejši obliki. Tako so prikazane informacije o vseh zagnanih procesih, njihovih podatkovnih tabelah, uspešno prepotovani poti skozi delovni tok, kje se trenutno proces nahaja in še marsikaj.



Slika 18: Prikaz stanja delovnega toka znotraj K2 Workspaces.

Slika 18 prikazuje potek procesa skozi delovni tok zahtevnejšega procesa, izdelanega v orodju K2.2003 (predhodna različica K2 BlackPearl). Rdeča barva pomeni, da proces ni ustrezal pogojem in tako ni šel po tej poti. Zelena barva pomeni, da je proces šel čez vse z zeleno barvo pobarvane aktivnosti in naloge. Prav tako zelene črte pomenijo, da je zadovoljil pogojem in je lahko nadaljeval pot. Na koncu imamo še modro pobarvano aktivnost, ki označuje mesto, kjer se proces trenutno nahaja.

Skrbniška orodja K2 strežnika prav tako vključujejo orodje za pregled vseh napak in odpravo le-teh. Pri novejši različici K2 strežnika je aplikacija dostopna preko spletnega brskalnika, tako kot že omenjena K2 Workspaces, medtem ko je pri starejših različicah dostop do teh informacij dostopen še preko Windows aplikacije, podobne sistemski aplikaciji `EventViewer`.

4. ZAKLJUČEK

V diplomskem delu smo si ogledali Microsoft SharePoint Server, ki je trenutno eden najbolj uporabljenih sistemov za upravljanje z dokumenti in vsebino. SharePoint strežnik v Sloveniji uporabljajo predvsem večja podjetja, medtem ko po svetu omenjeno tehnologijo uporabljajo tudi manjša podjetja ali zasebniki. Kot zanimivost lahko omenim, da imajo podjetja kot so Škoda Auto, Kempinski Hotels, AMD in še marsikateri drugi, svojo spletno stran postavljeno s pomočjo SharePoint strežnika. Večina teh uporablja SharePoint strežnik tudi za svoje intranet portale za zaposlene.

V diplomskem delu smo si tako pogledali strukturo SharePoint strani kakor tudi strežniško arhitekturo in topologijo. Bistvo diplomske naloge je predstavljala prenova poslovnih procesov s pomočjo izgradnje delovnih tokov. Vse skupaj smo združili v SharePoint okolje. Pogledali smo dva različna načina izgradnje delovnega toka in eno simulacijo delovnega toka s pomočjo programske kode. Vse tri načine smo podrobno razložili, bistvene dele pa tudi demonstrirali s pomočjo programske kode.

Simulacija delovnega toka s pomočjo programske kode predstavlja oteženo delo v primeru dodajanja nove funkcionalnosti. Tako je potrebno ob dodajanju nove funkcionalnosti preoblikovati nekatere dele ali kar celotno programsko kodo. Vseeno pa tak način dela predstavlja nekatere prednosti. Podjetju ni potrebno plačevati dodatnih licenc za specifično programsko opremo za izgradnjo delovnih tokov. Prav tako ni potrebno razvijalcem zgubljati časa z učenjem delovanja novih programskih orodij.

Windows Workflow Foundation ni isto kot SharePoint delovni tokovi. Windows Workflow Foundation predstavlja osnovo za SharePoint delovne tokove. SharePoint delovni tokovi so namenjeni izgradnji lažji delovnih tokov, kjer akterjem v procesu dodeljujemo naloge, ki jih morajo opraviti skozi proces. Prednost je tako hitra izdelava lažjih delovnih tokov in polna združljivost s SharePoint portalom.

Izgradnja delovnih tokov s pomočjo orodja K2 omogoča hitro in enostavno delo. Največja slabost je plačevanje licenc in čas izgubljen za učenje rokovanja z orodjem, vendar se naložba povrne, v kolikor podjetje orodje veliko uporablja in z njim služi. Orodje K2 omogoča izdelavo tudi zelo zahtevnih delovnih tokov in je za gradnjo le-teh vodilno na svetovnem tržišču. Prav tako omogoča polno združljivost s SharePoint portalom.

4.1. Uporaba in možnosti za izboljšave

Pomembno je omeniti, da so aplikacijo, ki smo jo naredili in je v tem diplomskem delu opisana kot simulacija delovanja delovnega toka, zaposleni hitro sprejeli medse in jo takoj začeli uporabljati. Aplikacija dnevno pridela okoli pet novih vnosov v SharePoint seznam, in tako bi lahko rekli, da se dnevno zažene tudi toliko novih instanc delovnega toka. Tako smo uresničili večletne želje nosilcev poslov, ki so morali po nepotrebnem leta in leta po podjetju prenašati fizične dokumente, kljub temu, da je tehnologija omogočala izgradnjo takšne rešitve že pred leti.

Vsekakor pa je želeno, da v kratkem preselimo aplikacijo na novejši interni SharePoint strežnik 2010. Kmalu za tem bo najverjetneje prav tako sledila prenova vseh obrazcev z uporabo novejših tehnologij, kot so ogrodje .NET 4.0, Silverlight ipd. V kolikor bo zahtevano, bo sledila še najpomembnejša izboljšava - izgradnja dejanskega delovnega toka v orodju K2. Trenutno ta prenova še ni potrebna, saj aplikacija zelo dobro simulira izvajanje. Proces se najverjetneje ne bo več spreminjal, zato s tem ne bo nobenih težav.

4.2. Sklep

Pri izdelavi diplomskega dela smo pridobili veliko administrativnega znanja pri uporabi SharePoint portala. Kljub temu, da je priporočljivo, da spremembe na SharePoint portalu dela za to namenjeni skrbnik, je vseeno razvijalec največkrat tisti, ki dostopa do strukture SharePoint strani in ostalih lastnosti portala, zaradi česar bo znanje zelo koristilo tudi v prihodnji karieri. Pri razvoju aplikacije smo prav tako pridobili veliko izkušenj o rokovanju s SharePoint portalom preko programske kode, s pomočjo SharePoint objektnega modela in spletnih storitev. Moramo omeniti, da je SharePoint portal za razvijalce zelo odprt in je preprosto preveč stvari, ki bi jih bilo potrebno še opisati, da bi bilo to diplomsko delo popolno. Tako je tudi v primeru izgradnje SharePoint delovnega toka. Potrebno bi bilo bolj natančneje razložiti izgradnjo zaporednega delovnega toka, medtem ko smo se delovnega toka stanja stroja samo dotaknili.

KAZALO SLIK

KAZALO TABEL

VIRI IN LITERATURA

[1] T. Pattison, D. Larson, Inside *Microsoft Windows SharePoint Services 3.0*, Redmond: Microsoft Press, 2007, pogl. 1-4, 6

[2] T. Pattison, A. Connell, S. Hillier, D. Mann, *Inside Microsoft SharePoint 2010*, Redmond: O'Reilly, 2011, pogl. 1

[3] W. Leon, W. Tynes, S. Cathey, *Microsoft SharePoint Server 2007 Bible*, Indianapolis: John Wiley & Sons, 2007, pogl. 3, 4

[4] M. Talley, *Professional K2 blackpearl*, Indianapolis: Wiley Publishing, 2009, pogl. 9, pogl. 12

[5] Microsoft (2010). Novosti v strežniku Microsoft SharePoint Server 2010. Dostopno na: http://office.microsoft.com/sl-si/sharepoint-server-help/novosti-v-strezniku-microsoft-sharepoint-server-2010-HA010370058.aspx (junij 2011)

[6] Microsoft (2010). SharePoint Server 2010. Dostopno na: http://technet.microsoft.com/en-us/library/cc303422.aspx (junij 2011)

[7] Microsoft (2010). Technical Diagrams (SharePoint Server 2010). Dostopno na: http://technet.microsoft.com/en-us/library/cc263199.aspx (junij 2011)

[8] Microsoft (2010). About SharePoint Site Collections. Dostopno na: http://technet.microsoft.com/en-us/library/cc742548.aspx (junij 2011)

[9] Microsoft (2010). Using Central Administration (Office SharePoint Server). Dostopno na: http://technet.microsoft.com/en-us/library/cc263312(office.12).aspx#section2 (junij 2011)

[10] Microsoft (2006). Developer introduction to Workflows for Windows SharePoint Services 3.0 and SharePoint Server 2007. Dostopno na: http://msdn.microsoft.com/en-us/library/aa830816(v=office.12).aspx (junij 2011)

[11] Microsoft (2010). Uporaba upravljanja pravic do informacij za seznam ali knjižnico. Dostopno na: http://office.microsoft.com/sl-si/sharepoint-foundation-help/uporaba-upravljanja-pravic-do-informacij-za-seznam-ali-knjiznico-HA101790607.aspx (julij 2011)

[12] Microsoft (2006). Create a workflow. Dostopno na: http://office.microsoft.com/en-us/sharepoint-designer-help/create-a-workflow-HA010100591.aspx (junij 2011)

[13] David Chappell (2007). Introducing Windows Workflow Foundation. Dostopno na: http://msdn.microsoft.com/library/ee210343.aspx (junij 2011)

[14] K2 (2011). What is K2. Dostopno na: http://www.k2.com/whatisk2.aspx (julij 2011)

[15] Microsoft (2010). Introduction to Collaborative Application Markup Language (CAML). Dostopno na: http://msdn.microsoft.com/en-us/library/ms426449.aspx (junij 2011)

[16] Microsoft (2010). Query Schema. Dostopno na: http://msdn.microsoft.com/en-us/library/ms467521.aspx (junij 2011)

[17] Nintex (2010). Nintex Workflow 2010. Dostopno na: http://www.nintex.com/en-US/Products/Pages/NintexWorkflow2010.aspx (avgust 2011)

[18] DataPolis (2010). DataPolis Workbox 2010. Dostopno na: http://datapolis.com/Products/DatapolisWorkbox2010.aspx (avgust 2011)

[19] Microsoft (2011). What is Silverlight. Dostopno na: http://www.microsoft.com/silverlight/what-is-silverlight/ (avgust 2011)

[20] Mozilla (2011). Mozilla Firefox. Dostopno na: http://www.mozilla.org/sl/firefox/ (september 2011)

[21] Apple (2011). Apple Safari. Dostopno na: http://www.apple.com/safari/ (september 2011)

[22] Microsoft (2011). Microsoft Internet Explorer. Dostopno na: http://windows.microsoft.com/sl-SI/internet-explorer/products/ie/home (september 2011)

[23] Microsoft (2011). Online Services: SharePoint Online. Dostopno na: http://www.microsoft.com/en-us/office365/sharepoint-online.aspx (september 2011)

[24] Microsoft (2011). What is Office 365. Dostopno na: http://www.microsoft.com/en-us/office365/what-is-office365.aspx#fbid=6dzbT1w7GfO (september 2011)

[25] Microsoft (2011). Microsoft Office. Dostopno na: http://office.microsoft.com/sl-si/ (september 2011)

[26] Microsoft (2010). Microosft AJAX Overview. Dostopno na: http://msdn.microsoft.com/en-us/library/bb398874.aspx (september 2011)

[27] Microsoft (2010). Microsoft ASP.NET Overview. Dostopno na: http://msdn.microsoft.com/en-us/library/4w3ex9c2.aspx (september 2011)