UDC: 004.23:378 INFO M: str. 19-26

AUTOMATIZACIJA PROCESA RASPOREDJIVANJA ISPITA U OKVIRU INFORMACIONOG SISTEMA UNIVERZITETA METROPOLITAN (AUTOMATIZATION OF EXAM SCHEDULING PROCESS WITHIN THE METROPOLITAN UNIVERSITY INFORMATION SYSTEM)

Marko Manasijević, Svetlana Cvetanović, Slobodan Jovanović
Univerzitet Metropolitan, Fakultet informacionih tehnologija, Beograd, www.metropolitan.edu.rs
marko.manasijevic@gmail.com, svetlana.cvtanovic@metropolitan.ac.rs, slobodan.jovanovic@metropolitan.ac.rs

REZIME: Problem generisanja rasporeda polaganja ispita (*exam scheduling*) sastoji se od odredjivanja termina (dan i sat) i učionica za svaki ispit, tako da se ispiti rasporede u okviru specificiranog vremenskog intervala, i u oviru raspoloživog kapaciteta (broj mesta) učionica, i uz neka ograničenja, npr. da neki student ne može biti istovremeno rasporedjen na dva ispita. Ovaj rad opisuje automatizaciju procesa rasporedjivanja ispita kao i odgovarajući softver, u okviru Informacionog sistema Univerziteta Metropolitan,. Takodje, diskutuje se sam Informacioni sistem Univerziteta Metropolitan (ISUM), u okviru kojeg se integriše aplikacija za generisanje rasporeda polaganja ispita. Naime, razmatra se arhitektura ISUM-a, i realizacija poslovnog procesa "Prijavljivanje i rasporedjivanje ispita", kao i motivacija za unapređenje postojećeg IS-a Univerziteta Metropolitan.
KLJUČNE REČI: Poslovni proces, WfMS system, Arhitektura sistema
ABSTRACT: The problem of generating exam schedule consists of finding dates and times and classrooms for every exam, where all exams are withinspecified time interval, and within room capacities, also with respect to some constraints, eg. An student can not simultaneously be present on two exams. This paper presents the exam scheduling process automatization and corresponding software, within the Metropoltan University Information System (ISUM). Also, paper discusses the Metropoltan Univwersity Information System, which incorporates the exam scheduling application. The architecture of ISUM is analysed, and the bussiness process called „Exam application and scheduling". And finally, the motivation for the improvement of ISUM.
KEY WORDS: Business process, WfMS system, System architecture

1. UVOD

Problem generisanja rasporeda polaganja ispita koji se pravi pre svakog ispitnog roka, je problem koji postoji kod svih univerziteta. Problem generisanja rasporeda polaganja ispita sastoji se od odredjivanja termina (dan i sat) i učionica za svaki ispit, tako da se ispiti rasporede u okviru specificiranog vremenskog intervala, i u oviru raspoloživog kapaciteta (broj mesta) učionica, i uz neka ograničenja, npr. da neki student ne može biti istovremeno rasporedjen na dva ispita. Univerziteti nude veliki broj ispita studentima (veliki broj „izbornih predmeta"), i ovo daje studentima veliku fleksibilnost u izbru predmeta, ali ovo komplikuje problem generisanja rasporeda spita. Problem generisanja rasporeda polaganja ispita može biti posebno težak kod velikih univerziteta (npr. Univerzitet u Valenciji ima 65,000 studenata), ili kod univerziteta koji imaju limitiran prostor, ili limitiran vremenski interval za rasporedjivanje ispita. .Postoji čitav niz računarskih metoda i tehnika za rešavanje problema odredjivanja rasporeda polaganja ispita, npr. kombinatorni algoritmi, pretraživački algoritmi, genetski algoritmi, itd. I „ručna" metoda, gde odgovorna osoba za ispitni rok sastavlja raspored po određenim pravilima koja su manje više konstantna, ali bez upotrebe nekog specijalizovanog računarskog programa, već samo koristeći Excel tabelu.

U ovom radu se analizira automatizacija procesa za pravljenje rasporeda polaganja ispita kao i odgovarajući softver, implementiran u okviru Informacionog sistema Univerziteta Metropolitan. Ovaj proces je zamišljen da bude jedan od servisa integrisanog informacionog sistema Univerziteta Metropolitan. U radu se diskutuje arhitektura informacionog sistema, i detalji realizacije poslovnog procesa "Prijavljivanje i rasporedjivanje ispita", kao i sama softverska aplikacija za generisanje rasporeda polaganja ispita .

Sadržaj rada je sledeći:

Poglavlje 2: Diskutuje motivacija za unapređenje postojećeg IS-a Univerziteta Metropolitan.

Poglavlje 3: Prezentuje arhitekturu ISUM-a.

Poglavlje 4: Prikazuje realizaciju poslovnog procesa "Prijavljivanje i rasporedjivanje ispita".

Poglavlje 5: Sadrži opis aplikacije za generisanje rasporeda polaganja ispita.

Poglavlje 6: Zaključak.

2. MOTIVACIJA ZA UNAPREĐENJE POSTOJEĆEG IS-A UNIVERZITETA METROPOLITAN

U trenutku donošenja odluke o unapređenju postojećeg informacionog sistema Univerziteta Metropolitan, imala se u vidu uloga koju danas informacioni sistemi imaju u redizajniranju, upravljanju i u integraciji poslovnih procesa. U vreme velike tržišne konkurencije, od informacionih sistema u mnogome zavisi mogućnost bilo koje organizacije da ispuni svoju viziju i poboljša svoju konkurentsku poziciju na tržištu, pa se njihovom razvoju i implementaciji daje poseban značaj. U zadnje dve decenije se pored ERP sistema za podršku poslovnim procesima, koriste i WfMS (Workflow Management siste-

mi) koji se bez obzira na istu namenu, zasnivaju na potpuno različitim pristupima **[1].**

ERP sistemi se najčešće implementiraju kao gotova softverska rešenja a da bi se postiglo njihovo što bolje prilagođavanje potrebama organizacije, oni se korišćenjem različitih parametara konfigurišu na nivou aplikacije. Drugim rečima, u ERP sistemima ne postoji eksplicitno definisani *workflow* modeli kojim se specificiraju poslovni procesi organizacije, već su oni ugrađeni u samu aplikaciju kroz korišćenje različitih parametarskih tabela.

Kod WfMS se kreiraju *workflow* modeli poslovnih procesa, a zatim i instance tih modela koje predstavljaju nosioce stvarnih aktivnosti opisanih u modelu. Za vreme izvršenja workflow modela, njegove instance mogu pristupiti bazi podataka, različitim vrstama aplikacija, ili mogu ostvariti interakciju sa korisnicima. Poznavanje razlika i sličnosti između WfMS i ERP sistema je veoma važno jer obe tehnologije imaju glavnu ulogu u upravljanju poslovnim procesima kako unutar tako i između sadašnjih i budućih organizacija. Najnoviji trend u razvoju ERP rešenja od strane velikih proizvođača softvera jeste da se WfMS i ERP sistemi integrišu. Na primer, Oracle je svom ERP rešenju dodao WfMS tako da se svaka pojedinačna ERP aplikacija može integrisati u različite *workflow* modele i na taj način informacije automatski obraditi i rutirati s jedne strane, kao i podržati personalizovana poslovna pravila s druge. Opisani savremeni trendovi su jedna od motivacija i za unapređenje postojećeg informacionog sistema Univerziteta Metropolitan. Postojeći poslovni sistem Univerziteta treba posmatrati samo kao jednu od komponenti koje se mogu pozivati za vreme izvršenja *workflow* modela.

Drugi razlog za unapređenje postojećeg IS-a Univerziteta je potreba za korišćenjem nekog savremenog rešenja *e-learning* sistema koji se bazira na Web 2.0 tehnologijama a koji bi bio integrisan sa ostalim delovima sistema. Naime, Univerzitet Metropolitan je prvi u zemlji 2005. godine, dobio dozvolu za izvođenje nastave na daljinu - preko Interneta, i trenutno na Univerzitetu Metropolitan studira preko 500 internet studenata, iz svih krajeva sveta. Studentima su 24 časa dostupna multimedijalna predavanja i vežbe, literatura u digitalnom obliku i nastavni materijali za pripremu ispita, što omogućava uspešno studiranje svima koji rade i koji nisu u mogućnosti da svakodnevno fizički prisustvuju predavanjima. Trenutno se za podršku udaljenom studiranju koristi Oracle *e-learning* sistem, koji radi nezavisno od ostatka informacionog sistema. To dovodi do poteškoća u radu jer se zajednički podaci teško mogu održavati ažurnim u više od jednog sistema. Osim toga, verzija *e-learning* sistema koja se koristi nije realizovana korišćenjem novih Web 2.0 tehnologija, pa je funkcionalnost sistema dosta siromašna. Prema trenutnim potrebama Univerziteta, osim e-learning sistema, treba koristiti i neki od savremenih CMS (sistema za upravljanje sadržajem) u kojem bi se čuvali verzirani sadržaji tekstualnih dokumenata koji cirkulišu u sistemu.

3. ARHITEKTURA INFORMACIONOG SISTEMA UNIVERZITETA METROPOLITAN

Suočen sa prethodno opisanim motivima, menadžment Univerziteta Metropolitan je odlučio da unapredi svoj informacioni sistem. Cvetanović i ostali su predložili arhitekturu informacionog sistema koji korišćenjem Web servisa integriše četiri različita podsistema: sistem za upravljanje poslovanjem, e-learning sistem, sistem za upravljanje sadržajima (CMS), i *workflow* management sistem (WfMS) **[2]**. Međutim, imajući u vidu prednosti WfMS i njegove mogućnosti da integriše različite tipove podataka i aplikacija, ova odluka je promenjena, tako da se u sadašnjoj arhitekturi WfMS koristi kao jedna vrsta "middleware" platforme za integraciju i orkestraciju ostalih sistema i drugih softverskih rešenja **[3].**

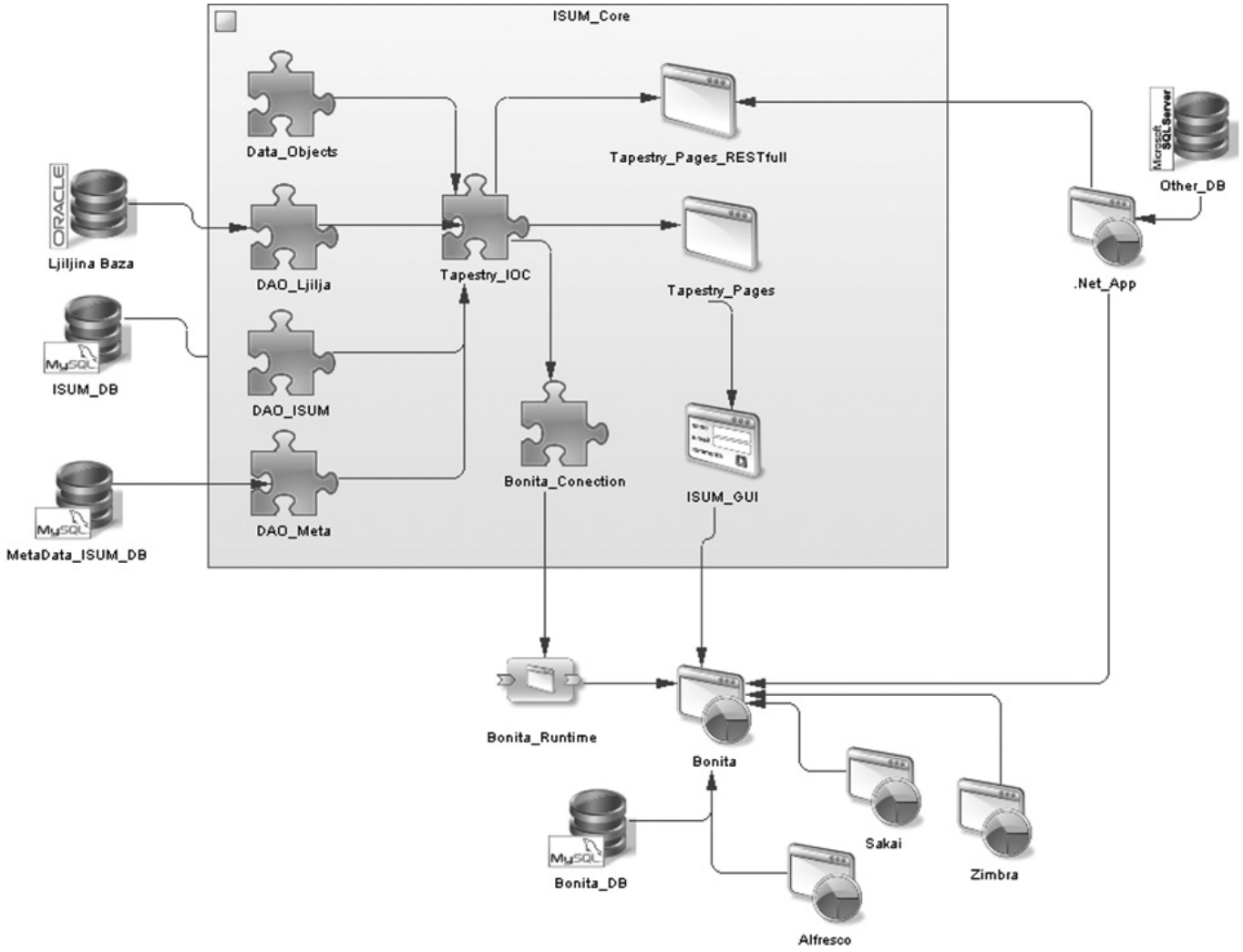
Sistem za podršku poslovanjem je prvenstveno fokusiran na poslovne procese od vitalnog značaja za poslovanje Univerziteta, kao što su: vođenje nastave, upis studenata, polaganje ispita, upravljanje ljudskim resursima, finansije itd. Ovaj sistem Univerzitet Metropolitan razvija interno, u okviru projekta razvoja informacionog sistema u kojem učestvuju studenti - sipendisti Univerziteta, kao i asistenti i profesori. Za implementaciju sistema se koristi softverska platforma Tapistry 5.

Kao *e-learning* sistem se koristi Sakai version 3.0., koji se bazira na Web 2.0 tehnologijama, i za krajnje korisnike omogućava veoma fleksibilno okruženje. Ova nova verzija Sakai se bazira na skupu osnovnih Sakai servisa - "kernel"-a, koji na najbolji način koristi *open source* tehnologije (npe. Jackrabbit i Shindig), i omogućava razvoj specifičnih resursa namenjenih akademskoj kolaboraciji. Sakai 3.0 posedije nove mogućnosti u odnosu na predhodne verzije, kao što su socijalne mreže i autorizacija fleksibilnog sadržaja, što današnji korisnici *e-learnig* sistema očekuju od Web aplikacija.

Workflow management sistem je implementiran korišćenjem *open source* rešenja Bonita. Bonita je izabrana kao najbolje rešenje između nekoliko *open source* softvera, kao što su JBoss, Joget, YWAL, Italio. Kriterijumi na osnovu kojih je vršena analiza postojećih rešenja su bili: razpoloživost dokumentacije, korisnički interfejs, kompatibilnost sa CMS i transakcionim sistemima (ERP), standardi itd. Bonita *workflow* system je lak za korišćenje jer ima mogućnost povezivanja za često korišćene baze podataka, a ako za to ne postoje odgovarajući konektori, oni se mogu lako kreirati korišćenjem postojećih modula u Bonita Studio.

Sistem za upravljanje sadržajem se bazira na *open source* rešenju Alfresko. Njegova modularna arhitektura koristi najnovije Java tehnologije. Alfresko podržava deljenje i čuvanje dokumenata, *workflow*, kolaboraciju, i služi kao platforma za publikovanje dokumenata.

Arhitekture sistema koja može da podrži ovako modularizovan sistem mora da pre svega bude fleksibilna, tj. ima mogućnost integracije nabrojanih sistema sa WfMS, ali i laku integraciju raznih drugih softverskih modula nastalih kroz stu-



Slika 1: *Modularna arhitektura informacionog sistema Univerziteta Metropolitan*

dentske projeke ili radove. Takođe, potrebno je da obezbedi servise za druge projekte, pa čak i druge platforme. Pri tom, arihtektura sistema treba da omogući lako kodiranje po savremenim principima i pravilima, pri čemu treba izbeći bilo kakvu redudantnost koda.

Kao *framework* za razvoj poslovnog dela informacionog sistema je korišen Tapestry 5. Tapestry je *framework* baziran na komponentama, što znači da će se svaka funkcionalnost poslovnog dela sistema kreira kao posebna komponenta. Važna prednost Tapestry-a je njegova otvorenost ka integraciji sa drugim *framework*-ovima i modulima. Već je povezan za Hibernate-om i JSON-om, a relativno lako se može dodati podrška i za bilo šta drugo pa čak i sam Spring framework. Na slici 1 je prikazana modularna arhitektura informacionog sistema Univerziteta Metropolitan koja zadovoljava predhodno definisane zahteve.

Osnovni (*Core*) deo arhitekture pokriva poslovni deo informacionog sistema, u koji mogu da se integrišu i drugi softverski moduli izrađeni u Tapestry-u ili drugim tehnologijama. Komunikacija između Tapestry osnovnog i drugih modula se izvodi pomoću RestFull servisa, baziranih na JSON-u koji predstavlja standardno rešenje, mada se mogu koristiti i klasični SOA servisi ili neki drugi principi. Veza između Bonite i Tapestry-a se realizuje preko Bonita Runtime API-a za Javu, koji obezbeđuje da se iz Jave mogu gađati direktno servisi Bonite.

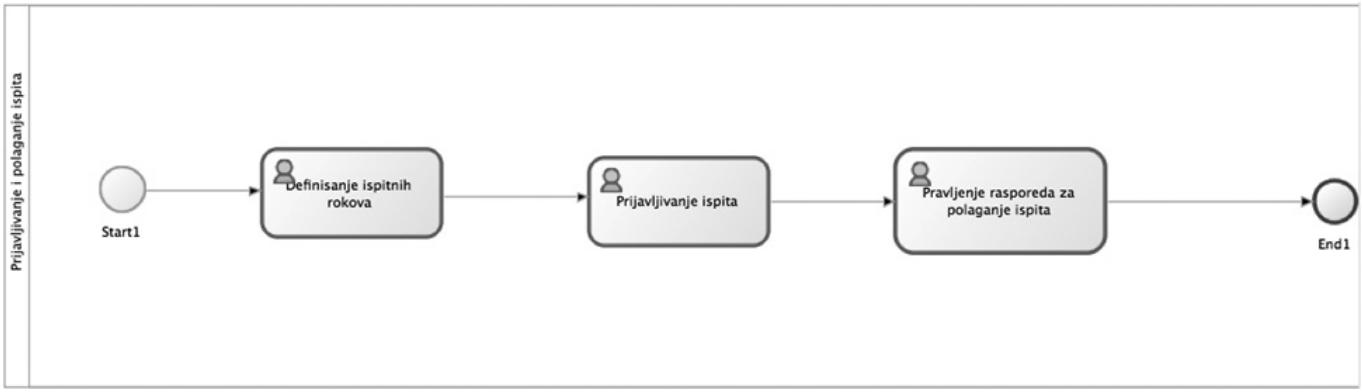
Na ovaj način je ostvarena fleksibilnost sistema, što podrazumeva da se GUI kreiran u Tapestry-u lako integriše sa Bonitom, da se iz poslovnog dela informacionog sistema lako dobiju potrebne informacije putem web servisa, da se putem Bonita Runitime API-a upravlja poslovnim procesima kao i da se kroz integraciju sa Bonitom ostvaruje jednostavna komunikacije sa nezavisnim projektima.

4. REALIZACIJA POSLOVNOG PROCESA "PRIJAVLJIVANJE I RASPOREDJIVANJE ISPITA"

Kao primer, na slici 2. je prikazan workflow modela za poslovni process "Prijavljivanje i rasporedjivanje ispita".

Model se sastoji od sledećih aktivnosti:

Definisanje ispitnih rokova: ima za cilj da se u poslovni deo informacionog sistema unesu podaci o predviđenim rokovima za polaganje ispita u toku jedne školske godine, kao što



Slika 2. *Workflow model poslovnog procesa "Prijavljivanje i rasporedjivanje ispita"*

su dozvoljeni periodi za prijavljivanje i polaganje ispita kao i raspoloživost učionica u kojima se ispiti održavaju.

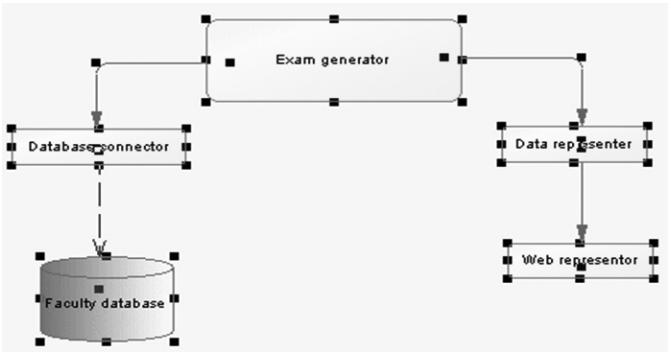
Prijavljivanje ispita: Studenti ispite prijavljuju online u za to predviđenom periodu, unoseći oznake predmeta i ispitni rok u kojem žele da polažu. Da bi prijave bile prihvaćene, potrebno je da student izmiri svoje finansijske obaveze vezane za polaganje ispita, da ima položene sve ispite koji predstavljaju preduslov za prijavljene ispite, kao i da su za pripremu ispita korišćeni važeći nastavni materijali. Prihvaćene prijave idu na dalju obradu a odbačene se ne pamte u sistemu.

Pravljenje rasporeda za polaganje ispita: aktivnost u okviru koje se na osnovu prihvaćenih prijava studenata za jedan ispitni rok, angažovanja nastavnika na predmetima koji se polažu, dužine trajanja roka i podataka o raspoloživim učionicama za polaganje ispita pravi raspored ispita. Njime se definiše termin polaganja ispita (datum i vreme), učionica u kojoj se ispit polaže kao i dežurni nastavnici na ispitu.

Aplikacija za generisanje rasporeda za polaganje ispita je relaizovana nezavisno od ostatka poslovnog dela informacionog sistema . Aplikacija se može izvršavati na svakom hardveru koji podržava Java 6 VM. Ona nema svoj korisnički interfejs za prihvatanje i prezentaciju rezultata, već je on realizovan u okviru WfMS, koji služi da ovu aplikaciju integriše sa ostatkom poslovnog dela informacionog sistema. Aplikacija ima dva spoljašnja interfejsa: prvi je prema ulaznim podacima, a drugi je za prikaz izlaza. Pod ulaznim podacima podrazumevaju se svi podaci potrebni za neometan rad aplikacije kao što su:

- Lista studentata (Student: ime, prezime, broj indeksa, lista prijavljenih ispita).
- Lista ispita (Ispit: naziv, oznaka, godina studija na kojoj se ispit nalazi)
- Lista asistenata (Asistent: ime, prezime, indentifikacioni broj)
- Period održavanja ispita
- Vremena u kojima se održavaju ispiti
- Lista učionica (oznaka učionice, broj mesta u učionici)

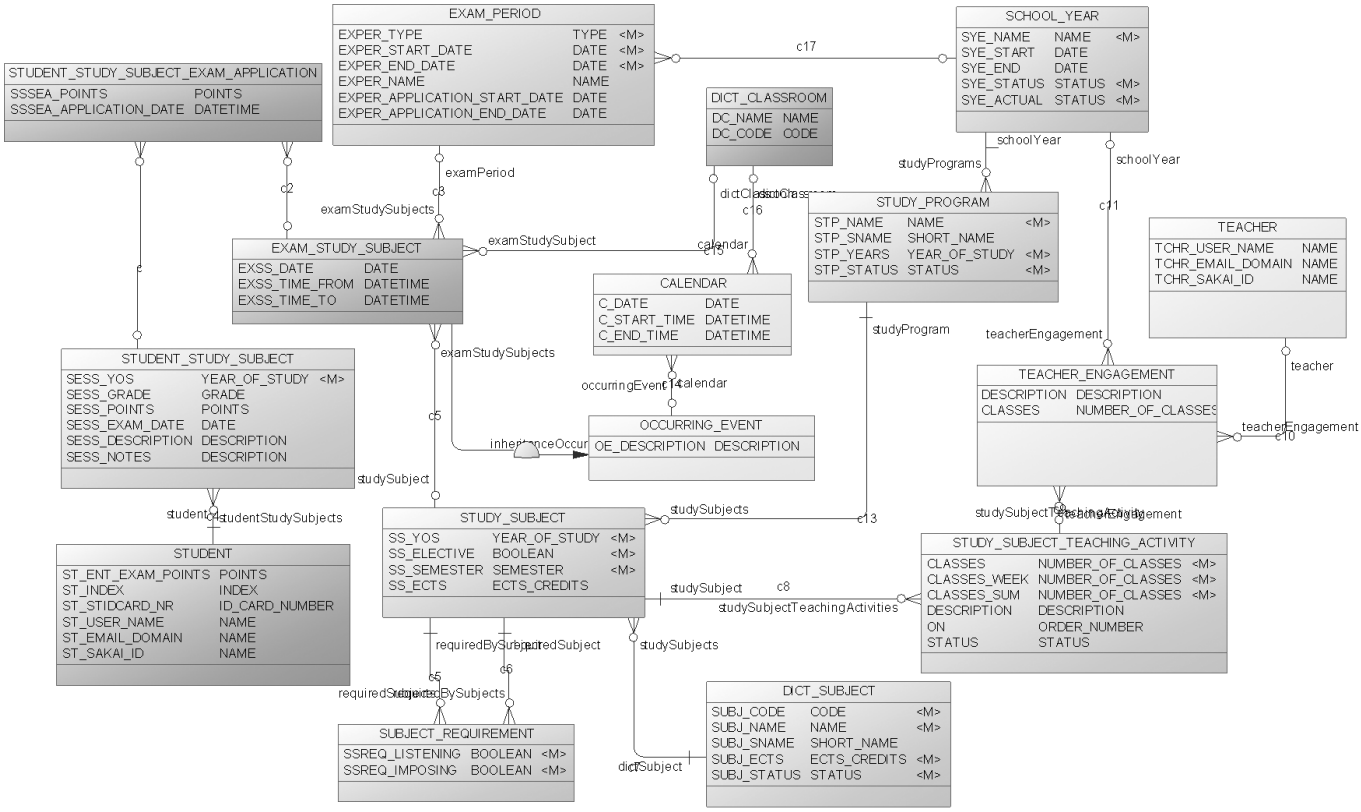
Spoljašnji interfejsi algoritma su prikazan na slici 3**.** Na slici 4 je prikazan deo konceptualnog modela baze podataka koji je predviđen za čuvanje podataka vezanih za prijavljivanje, polaganje ispita i napravljenog rasporeda polaganja.



Slika 3: *Spoljašnji interfejsi algoritma za generisanje rasporeda polaganja ispita*

Pri tome:

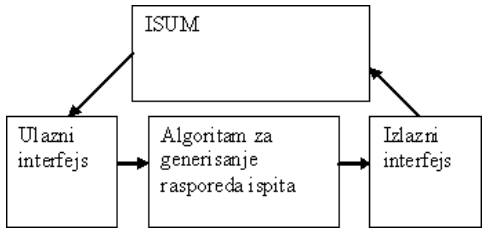
- Lista studentata (Student: ime, prezime, broj indeksa, lista prijavljenih ispita): dobija se na osnovu entiteta STUDENT, PREDMETI STUDENATA (STUDENT_STUDY_SUBJECT), PRIJAVE ISPITA (STUDENT_STUDY_SUBJECT_EXAM_APPLICATION)
- Lista ispita (Ispit: naziv, oznaka, godina studija na kojoj se ispit nalazi): dobija se na osnovu entiteta PREDMETI (STUDY_SUBJECT) koji su definisani važećim nNASTAVNIM PLANOM (STUDY_PROGRAM)
- Lista asistenata (Asistent: ime, prezime, indentifikacioni broj): dobija se na osnovu entiteta NASTAVNIK (TEACHER), ANGAŽOVANJE (TEACHER_ENGAGEMENT) i AKTIVNOSTI NASTAVNIKA NA PREDMETIMA (STUDY_SUBJECT_TEACHING_ACTIVITY)
- Period održavanja ispita: dobija se na osnovu entiteta ISPITNI KOJI SE POLAŽU (EXAM_STUDY_SUBJECT)
- Vremena u kojima se održavaju ispiti: zadaju se kao ulazni parametar u aplikaciju
- Lista učionica (oznaka učionice, broj mesta u učionici): odgovara entitetu UČIONICE.



Slika 4: *Konceptualni deo baze podataka za prijavljivanje i ispita i čuvanje rasporeda polaganja*

5. APLIKACIJA ZA GENERISANJE RASPOREDA ZA POLAGANJE ISPITA

Aplikacija za generisanje rasporeda polaganja ispita razvijena je u Java programskom jeziku, i kao takva podržana je od strane bilo kog operativnog okruženja koje sadrži Java 6 virtualnu mašinu. Aplikacija se sastoji od algoritma za generisanje rasporeda ispita, i od dva spoljašnja interfejsa. Prvi interfejs je za ulazne podatke koji su smešteni u odgovarajućoj bazi podataka u okviru ISUM-a (pogledati sliku 5), a drugi interfejs je za prikaz izlaza, bilo u obliku fajla, ili u obliku prikaza na Metropolitan Web-sajtu (koji je takodje deo ISUM-a). Vreme izvršavanja algoritma je oko sekundu (ili par sekundi), a ovo vreme zavisi od dimenzija ulaznih podataka (broj studenata, broj ispita, broj učionca, itd.). Očekivani ulazni podaci su: lista studenata, lista ispita, lista asistenata, datumi i vremena održavanja ispita, i lista učionica.



Slika 5: *Aplikacija za generisanje rasporeda polaganja ispita*

Problem rasporedjivanja polaganja ispita je jedan kombinatorni matematički problem, gde je potrebno u nekoliko različitih dana i sati (npr. 3 dana, u 9.00h, 12.00h, i 15.00h), i u nizu različitih učionica (gde učionice imaju različite kapacitete, npr. od 20 do 50 mesta), rasporediti odredjen broj studenata (npr. 1000 studenata), koji su prijavili po nekoliko različitih ispita, i gde u svakoj učionici postoji asistent koji nadgleda ispit (pogledati sliku 6). Postoji čitav niz metoda i tehnika za rešavanje problema odredjivanja rasporeda polaganja ispita, npr. kombinatorni algoritmi, pretraživački algoritmi, genetski algoritmi, itd. **[4-6,].** Na Internetu se mogu naći besplatne aplikacije za generisanje rasporeda ispita baziranih na ovakvim algoritmima [6]. Na Univerzitetu Metropolitan se trenutno koristi sopstvena aplikacija za generisanje rasporeda ispita (razvijena u Javi), ali po potrebi može se instalisati i integrisati u ISUM i neka od ovih besplatnih aplikacija (aplikacije otvorenog koda) sa Interneta.

Pri rešavanju problema generisanje rasporeda ispita potrebno je definisati nekoliko entiteta, i to su sledeći entiteti**:** Studenti, Asistenti, Ispiti,Termini, i Učionice. Entitet Studenti sadrži identifikacione podatke o studentima (ime, prezime, broj indeksa) i listu prijavljenih ispita za svakog studenta (npr. IT160, SE201, itd.). Entitet Asistenti obuhvata identifikacione podatke o asistentima (ime, prezime, identifikacioni broj) i listu ispita na kojima će svaki asistent da dežura. Entitet Učionce reprezentuje učionice gde se održavaju ispiti, i svaka učionica ima svoju oznaku i kapacitet broja mesta za studente (koliki broj studenata može da stane u učionicu). Svaki „ispit“ u entitetu Ispiti opisuje se identifikacionom šifrom ispita (npr. IT110, SE211, itd.), i listom studenata koji su prijavili da

dotični ispit hoće da polažu, kao i pridruženim asistentom za svaki ispit (asistent koji nadgleda ispit). Entitet Termini obuhvata sva moguća vremena polaganja npr. 9h, 12h, 15h, i to za svaki dan ispitnog roka, npr. 3 dana, što u ovom konkretnom primeru čini 9 različitih mogućih termina. Rezultat rada algoritma je entitet Raspored, tj. jedna lista, gde se za svaki ispit navodi dan i vreme i učionica gde će se polagati dotični ispit (npr. ispit IT110 se polaže na dan 1 u 15h u učionici 5) Dakle, cilj je da se odredi raspored polaganja ispita, gde se u vidu tabele, za svaki ispit odredi termin i učionica. Npr. tabela:

Ispit	**Dan**	**Vreme**	**Učionica**
IT110	Dan 1,	15h,	U1
SE211	Dan 2,	12h,	U2
CS323	Dan 3,	9h,	U3
CS322	Dan 1,	9h,	U4

Itd.....

Kod kreiranja entiteta Raspored, potrebno je voditi računa o sledećim ograničenjima (tj. mogućim „konfliktima"):

- da li neki ispit može da se smesti u neku učionicu, tj. da li je kapacitet učionice manji od broja studenata koji polažu ispit (učionica u „konfliktu")
- jedan student ne može biti na dva različita ispita u isto vreme (student u "konfliktu")
- jedan asistent ne može biti u dve različite učionice u isto vreme (asistent u „konfliktu").

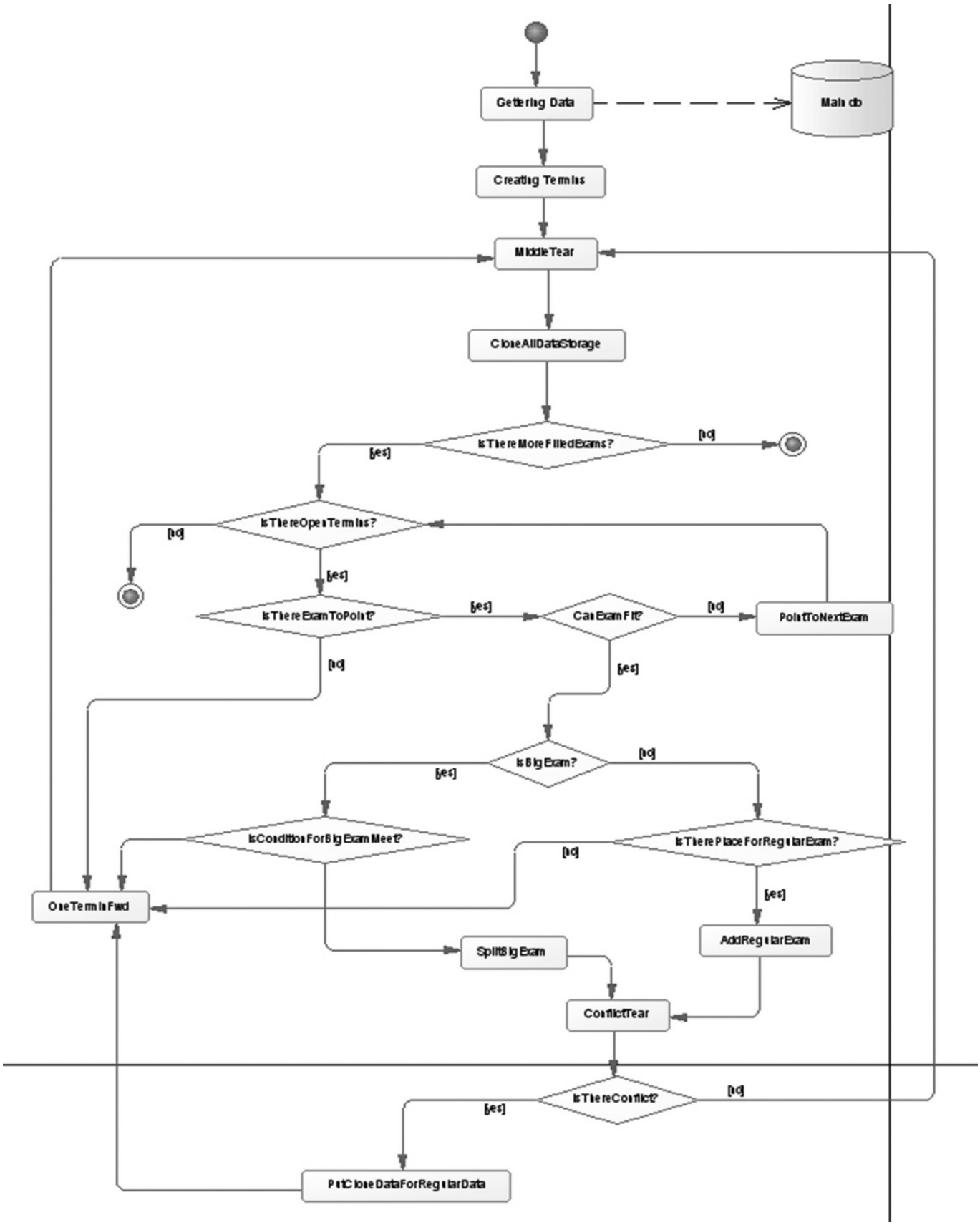
A izlazni podaci se prezentuju licu koje je odgovorno za kreiranje rasporeda polaganja ispita, u cilju provere ili dalje obrade.

Na Univerzitetu Metropolitan implementiran je (u Javi) i koristi se algoritam baziran na sopstvenoj heuristici, a koja je rezultat dosadašnjeg iskustva u generisanju rasporeda ispita. Algoritam generiše raspored polaganja ispita vodeći računa o ograničenju broja mesta u svakoj učionici, pri čemu omogućuje polaganje i više ispita u jednoj učionici ukoliko ima prostora. Algoritam koristi ulazne podatke smeštene u bazi podataka koja je deo ISUM-a. Algoritam se sastoji od sledećih koraka:

1. Prvo se učitaju ulazni podaci.
2. Pa se generiše lista ispita, i lista učionica, i svi termini, kao i lista studenata i lista asistenata (tj. odgovarajući entiteti u algoritmu).
3. I onda se pokušava jedan po jedan ispit sa liste ispita, da se smesti u nekoj učionici sa liste učionica i sa liste termina, pri čemu se proverava jedna po jedna učionica, i jedan po jedan termin. Ovde se omogućuje polaganje i više ispita u jednoj učionici ukoliko ima prostora, tako da se učionica popunjava sa ispitima dokle god ima mesta u učionici, pa se tek onda prelazi na sledeću učionicu.
4. Zatim, proverava se narušenost ograničenja tj. postojanje "konlikata", i u slučaju postojana „konflikta", preskače se predložena kombinacija, i ide se na sledeću kombinaciju (tj. sledeću učioncu ili termin). Ako ne postoji „konfllikt", ispit se smesti u dotičnu učionicu i termin.
5. A na kraju, ako su svi ispiti smešteni, bez "konflikata", onda se štampa raspored ispita. U slučaju da neki ispiti ne mogu da se smeste ni u jednu učionicu ni u jednom terminu bez "konflikta", štampa se pored postignutog rasporeda ispita, i lista neuspelih rasporeda ispita.

Dakle, algoritam pokušava jedan po jedan ispit sa liste ispita (E1, E2, ..), da smesti u nekoj učionici sa liste učionica (U1, U2, ..) i sa liste termina (T1, T2, ...), pri čemu se proverava jedna po jedna učionica, i jedan po jedan termin. Algoritam se može ilustrovati grafički, kao na slici 8. Gde se vidi da se ispiti E1, E2, E3... smeštaju u učionice U1,U2,U3, ... i termine T1,T2,T3,.. Pri tome, neki ispit, npr. E1, predstavla u stvari listu studenata koji su prijavili dotični ispit, plus asistent koji dežura na ispitu. Algoritam je kombinatoran, jer ispituje različite kombinacije, jednu po jednu kombinaciju rasporedjivanja liste ispita u listu učionica i listu termina. Razne kombinacije se generišu pomoću „petljastih" logičkih instrukcija ***while(){}.*** Pri proveri validnosti neke kombinacije tj. postojana „konflikta", koriste se logičke instrukcije „grananja" *if()then{}else{}*.

Na slici 6 dat je detaljni dizajn algoritma (u obliku *flowchart*), uradjen pomoću PowerDesigner softvera**.** Pre nego što počne sa radom algoritam mora da se "napuni" podacima iz baze podataka. Sledeći korak je kreiranje svih neophodnih entiteta(**Creating Termins**). Posle kreiranja liste ispita i učionica i termina, počinje izvršenje glavnog logičkog dela, opisanog na dijagramu 6 kao **MiddleTear**. U stvari, **MiddleTear** prestavlja početak **while** petlje, koja se prolazi pri svakoj iteraciji („smeštanju" nekog ispita u neku učionicu i neki termin). Nakon toga „klonira" se **AllDataStorage** objekat (**Clone AllDataStorage**) koji sadrži sve podatke potrebne za generisanje rasporeda polaganja ispita, pomoću koga se algoritam u slučaju detektovanja „konflikta" vraća u prethodno stanje. Algoritam nakon toga ulazi u niz uslova („flagova") od kojih zavisi njegova realizacija. Proverava se da li su svi ispiti raspoređeni u učionice i termine (**IsThereMoreFilledExams, IsThereOpenTermins).** U slucaju da jesu, dolazi do izlaza iz petlje i ispisivanja najboljeg generisanja. U slučaju da postoji barem jedan ispit koji nije rasporedjen, proverava se da li ispit koji se "delegira" može da se smesti u odredjenu učionicu i termin (**CanExemFit**.). U slučaju da ne može, proverava se sledeci ispit da li može biti smešten u posmatranoj učionici u posmatranom terminu, i tako redom (**PointToNextExam**). U slučaju da ne postoji ispit koji može biti "delegiran" u posmatranoj učionici, indeks učionice tj. termina se pomera za jedan (**OneTermin Forward**). U slučaju da postoji ispit koji može biti "delegiran", proverava se da li je "veliki" (**Is BigExam**) ili "regularan" ispit, i nakon toga sledi njegovo smeštanje u čionicu i termin. Dakle, postoji mogućnost da se „veliki ispiti" (ispiti koji se ne mogu smestiti ni u najvećoj učionici) smeste u dve učionice (**SplittingExam**). U slučaju da nije doslo do „konflikta", počinje sledeća iteracija sa indeksom ispita pomerenim za jedan. U slučaju da ima „konflikta", postavlja se „klonirani" **AllDataStorage** objekat (**PutCloneData ForRegularData),** i nastavlja se pokušaj rasporedjivanja sledećeg ispita. Algoritam sadrži unutrasnju proveru na „konflikte" (**ConflictTear**), koja vrši proveru nakon svake iteracije



Slika 6: *Detaljni dizajn algoritma (flow-chart)*

(**Is ThereConflict?**), i pokusava da dođe do rezultata bez „konflikta“, tako što u slučaju „konflikta“ u bilo kom objektu, vraća sistem u prethodno stanje, i pokusava rasporedjivanje sledećeg ispita, sve dok se ne nađe ispit koji odgovara, ili da se niti jedan ispit ne može ubaciti u posmatranu učionicu, a da ne dođe do „konflikta“. Tek u tom slučaju indeks učionice se pomera za jedan. U slučaju da na kraju rada algoritma postoje nerazrešeni “konflikti”, rezultat je ispisivanje poruke koja upozorava korisnika da postoje nerazrešeni „konflikti”.

Algoritam je testiran na velikom broju primera, počev od jednostavnijih, pa do veoma složenih**.** Variran je broj dana trajanja ispitnog roka , i broj studenata, a i broj ispita, kao i broj učionica. Npr. sledeći ulazni podaci: Broj studenata: 400; Trajanje ispitnog roka: 3 dana; Vreme odrzavanja ispita: 09:00, 12:00, 15:00; Broj asistenata: 5; Broj ispita: 10 (30 do 50 studenata po ispitu); Broj učionica: 3 (30 do 50 studenata po učionici). Ili npr.: Broj studenata: 800; Trajanje ispitnog roka: 3 dana; Vreme odrzavanja ispita: 09:00, 12:00, 15:00; Broj asistenata: 20; Broj ispita: 50 (10 do 30 studenata po ispitu); Broj učionica: 10 (10 do 30 studenata po učionici). U svim slučajevima algoritam je vrlo brzo radio, vreme izvršavanje je bilo manje od par sekundi. Za svaki primer korišćenja algoritma, program štampa izlazne podatke o ispitima, i to raspored polaganja ispita za svaki ispit (dan, vreme, učionica), kao i ispitnu listu za svaki ispit, a ispitna lista obuhvata podatke o asistentu koji dežura kao i listu studenata za dotični spit.

6. ZAKLJUČAK

U radu je analizirana automatizacije procesa izvođenja ispitnog roka na Univerzitetu Metropolitan. Pri tome, razmatra se arhitektura informacionog sistema Univerziteta Metropolitan, i realizacija poslovnog procesa “Prijavljivanje i rasporedjivanje ispita“, kao i motivacija za unapređenje postojećeg informacionog sistema. Aplikacija za generisanje rasporeda polaganja ispita treba da pomogne u ovom procesu, kroz efikasno automatizovano respoređivanje ispitnih termina. Automatizovani proces “Prijavljivanje i rasporedjivanje ispita“ je zamišljen da bude jedan od servisa integrisanog informacionog sistema Univerziteta Metropolitan.

ZAHVALNICA:

Ovaj rad podržan je od strane Ministarstva za nauku i obrazovanje (Projekat III44006).

LITERATURA

[1] J. Cardoso, R. P. Bostrom and A. Sheth, Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications, Information Technology and Management Journal. Special issue on Workflow and E-Business (Kluwer Academic Publishers), 5(3-4): pgs. 319-338, 2004.

[2] S. Cvetanović, M. Opačić, D. Beljić, Z. Polić, Softverska arhitektura za realizaciju učenja na daljinu u IS Univerziteta Metropolitan, Zbornik radova Elektronsko učenje na putu ka društvu znanja, pgs. 43-48, Univerzitet Metropolitan, Beograd, 2010.

[3] S. Cvetanović, M.Raspopović, S.Mirković, Integration OF SAKAI based E-learning and BUSINESSS MANAGEMENT SYSTEMS USING Workflow management systems, The Second International Conference on e-Learning (eLearning-2011)

[4] E.Cheng, R.Kleinberg, S.Kruk, W.Lindsey, D.Steffy, A strictly combinatorial approach to a university exam scheduling problem, http://citeseerx.ist.psu.edu/

[5] R.Alvarez-Valdes, E.Crespo, J.Tamarit, A tabu search algorithm to schedule university examinations, www.idescat.cat/sort/questo/

[6] A.Schaerf, A survey of automated timetabling, http://citeseerx.ist.psu.edu/

Marko Manasijević, Inž., Inženjer informacionih tehnologija, rodjen je 1983. godine, i završio je Fakultet Informacionih tehnologija, na Metropoltan Univerzitetu u Beogradu. Iskusan je programer, radio je u nekoliko inostranih softverskih firmi, na razvoju niza složenih softverskih aplikacija koristeći Javu, PHP, itd. Specijalsta je za Javu, C/C++, HTML, PHP, JavaScript, za baze podataka. Ima veliko iskustvo sa raznim operativnim sistemima i Web serverima.

Svetlana Cvetanović, Doc. Dr Inž., predaje na Univerzitetu Metropolitan, Fakultetu za informacione tehnologije, od 2007.g. Završila je Elektronski fakultet u Nišu, gde je i magistrirala. Doktorske studije završila je na Fakultetu za menadžment, na Univerzitetu u Novom Sadu. Godinama je radila kao projektant informacionih sistema i učestvovala u značajnim projektima u zemlji, od Narodne banke do državnih institucija. Kao docent na Fakultetu informacionih tehnologija radi od 2007. godine, a njena uža oblast ekspertize su baze podataka i njihova napredna primena. Autor je velikog broja stručnih radova i učesnik na velikom broj inostranih konferencija.

Slobodan Jovanović, Prof. Dr. Inž., predaje na Univerzitetu Metropolitan, Fakultetu za informacione tehnologije, od 2008.g. Bavi se razvojem Web aplikacija, i „pametnim“ električnim mrežama (*smart electric grids*), i veštačkom inteligencijom.. Ima veliki boj objavljenih nučnih radova u vodećim internacionalnim časopisima. Predavao je na *Strathclyde University, Glasgow, Scotland (UK),* u periodu 1993.-2008. U Institutu N.Tesla radio je od 1977. do 1990., a na Queen’s University of Belfast od 1990. do 1993. g. Učestvovao je u nizu naučnih i razvojnih, domaćih i inostranih projekata. Na Elektrotehničkom fakultetu u Beogradu (Beogradski univerzitet) diplomirao je 1977. i doktorirao 1984. g.