UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Odsek za računarsku tehniku i informatiku

STABILNOST I PERFORMANSE GLITE WORKLOAD MANAGEMENT SYSTEM-A

Diplomski rad

Mentor:
prof. dr Zoran Jovanović

Kandidat:
Neda Švraka 200/94

Beograd, maj 2007.

Predstavljeni rezultati su dobijeni na AEGIS GRID e-infrastrukturi [1], čiji rad je delimično podržan od stane EC FP6 projekata EGEE-II (INFSO-RI-031688) i SEE-GRID-2 (INFSO-RI-031775). Rad je urađen u Laboratoriji za primenu računara u nauci Instituta za fiziku u Beogradu [2] i podržan je od strane Ministarstva za nauku i zaštitu životne sredine Republike Srbije kroz projekat osnovnih istraživanja OI141035. Autorka se najsrdačnije zahvaljuje mr Antunu Balažu na savjetima, sugestijama i pomoći.

Sadržaj

1 Uvod u Grid

Razvoj hardvera i softvera tokom posljednje decenije doveo je do značajnih poboljšanja perfomansi računarskih sistema i mreža, što je omogućilo konstruisanje računarskih sistema visokih performansi, niske cijene, popularno nazvanih klasteri. Oni služe za riješavanje problema u mnogim aplikacijskim domenima koji zahtjevaju značajnu upotrebu računarskih resursa [3]. Značaj korišćenja ovakvih sistema naročito je vidljiv u nauci, jer je omogućio naučnicima da uvedu više parametara u svoje simulacije i eksperimente nego do tada. Razmjena podataka i rezultata eksperimenata između saradnika širom svijeta danas je skoro trenutna zahvaljujući postojanju brzih mreža. U posljednje vrijeme se razvijaju programi čiji je cilj riješavanje različitih naučnih problema, koji zahtjevaju značajne računarske resurse, upotrebu različite opreme koja se ne mora geografski nalaziti na istom mjestu, a koji podstiču saradnju različith naučnih institucija. Takvi programi se zajednički imenuju pojmom *e-Science* [4], koji ukazuje na ključnu ulogu računarske infrastrukture u ostvarenju istraživačke saradnje.

Podaci koji se dobijaju i analiziraju u okviru *e-Science* programa su po svojoj prirodi velikog obima i distribuirani, te je najveći izazov u okviru ovakvog okruženja pronalazak efikasnog rješenja za pristup, distribuciju, obradu i smještanje ovih podataka. To je bio motiv za stvaranje računarske infrastrukture koja povezuje međusobno udaljene resurse, kako u softverskom tako i u hardverskom smislu. Primjeri ovakvih resursa su baze podataka, uređaji za skladištenje podataka, uređaji za obradu podataka povezani brzim vezama, a infrastruktura koja ih sve povezuje u jednu cjelinu se naziva ***Grid***. Ovdje se može napraviti analogija sa elektrodistributivnom mrežom koja obezbjeđuje postojan, sveprisutan, pouzdan, otvoren pristup električnoj energiji nezavisno od izvora, što se želi i postići sa računarskim resursima u okviru *Grid* infrastrukture.

Veliki broj projekata u svijetu je usmjeren ka razvoju i upotrebi *Grid* mreža koje se koriste u različite svrhe i na različitim nivoima, kako na polju nauke, tako i na polju industrije, te stoga postoji više definicija *Grida.* Tako Globus projekat definiše *Grid* kao „infrastrukturu koja omogućuje objedinjenu upotrebu u okviru saradnje korisnika udaljenih računara, mreža, baza podataka i naučnih instrumenata, koje posjeduju i kojima upravljaju različite organizacije“ [5]. Uopšteno govoreći, riječ je o sistemu koji pruža skalabilan, siguran i efikasan mehanizam za pronalaženje i koordinisan pristup distribuiranim resursima od strane različitih interesnih grupa korisnika. Ovo okruženje se razvija na tri temelja:

- Koordinisanje upotrebe računarskih resursa. Izbjegava se izgradnja *Grid* sistema sa centralizovanom kontrolom. Umjesto toga, mora se obezbjediti neophodna infrastruktura za koordinaciju korišćenja resursa, zasnovana na politici vlasnika i sporazumima o usluzi.
- Protokoli i okviri otvorenih standarda. Korišćenje tzv. otvorenih standarda obezbeđuje interoperabilnost i mogućnost integracije. Ovi standardi se moraju primjeniti za regulisanje načina pronalaženja, pristupa i koordinacije resursa.
- Pružanje odgovarajućeg kvaliteta usluga. Sistem treba da pruži adekvatan odgovor svim zahtjevima korisnika. Veličine kojima se određuje kvalitet usluge mogu biti npr. vrijeme odziva, ukupne performanse, sigurnost, skalabilnost resursa, itd.

U okviru *Grid* okruženja postoje različiti servisi koji obezbjeđuju sigurnost, informacije, uputstva za korišćenje odgovarajućih servisa, alokaciju resursa, agregaciju resursa, razvoj aplikacija, upravljanje izvršavanjem poslova. Softverski alati i servisi koji omogućuju povezivanje različitih računarskih resursa sa izvorima podataka u cilju omogućavanja distribuirane analize i saradnje zainteresovanih grupa se jednim imenom nazivaju *Grid*

middleware. Da bi krajnji korisnik doživio *Grid* kao jedinstvenu cjelinu i bez prepreka mogao da pristupi različitim dijelovima takvog jednog sistema, zadatak *middleware-a* je da poveže i učini interoperabilnom postojeću heterogenu infrastrukturu koja obuhvata različite vrste hardvera i softvera, zasnovane na različitim tehnologijama, a na kojoj je izgrađen *Grid.* Takođe, potrebno je obezbjediti upravljanje i pristup geografski raspodjeljenim resursima koji su pod administrativnom upravom različitih organizacija. Rješenje je nađeno u formiranju virtuelnih organizacija (*Virtual Organizations*) koje sačinjavaju postojeće organizacije, institucije ili individualni korisnici koje dijele resurse, sarađuju na zajedničkim projektima i koje povezuje neki zajednički interes. Virtuelne organizacije u dogovoru sa vlasnicima resursa definišu prava korišćenja resursa: kojim resursima smiju da pristupe, koje od njihovih mogućnosti su im na raspolaganju, kao i koji deo ukupnog vremena rada resursa, ili njegovog skladišnog kapaciteta grupa korisnika može da koristi za svoje poslove. Da bi pristupio resursima G*rid*-a, svaki korisnik mora biti član jedne od virtuelnih organizacija.

Može se reći da koncept na kom je zasnovan *Grid* sličan konceptu i ideji *World Wide Web-a (WWW)* i da se očekuje da *Grid* da doprinos na polju razmjene i dijeljenja računarskih resursa koji će biti sličan doprinosu *WWW*-a na polju razmjene i dijeljenja informacija. Ipak, postoje suštinske razlike među njima. *Web* je namijenjen razmjeni informacija, koje su svakom dostupne i svako može da ih učini raspoloživim drugim korisnicima. Uglavnom je riječ o klijent-server interakciji. S druge strane, *Grid* prevazilazi jednostavnu komunikaciju među računarima i ima za cilj da globalnu mrežu računara pretvori u ogroman računarski resurs. S obzirom na to da su resursi vrijedna svojina svojih vlasnika, njihova upotreba mora biti u skladu sa pravilima koje određuju vlasnici.

U svijetu vlada veliko ineteresovanje za mogućnosti koje otvara *Grid computing* tehnologija. Trenutno, pokrenuti *Grid* projekti u mogu se grubo podijeliti u dvije klase:

- razvoj infrastrukture koje uključuje instalaciju i održavanje hardvera, softvera i administrativnih mehanizama koji omogućuju korisnicima upotrebu resursa,
- istraživanje i razvoj *middleware-a,* što predstavlja rad na izgradnji interoperabilnog softvera i definisanju pravila u cilju ostvarivanja punog potencijala *Grid computing-a.*

Mnogi od ovih projekata su pokrenuti u okviru realizacije obimnih naučnih projekata koji uključuju generisanje i analizu velike količine podataka. Takav projekat je *World Wide LHC Computing Grid Project (WLCG)* [6], koji je kreiran da pripremi računarsku infrastrukturu za simulaciju, obradu i analizu podataka dobijenih u *Large Hadron Collider (LHC)* eksperimentu. *LHC* je konstruisan u Evropskoj laboratoriji za fiziku čestica (*CERN*) i danas je najveći akcelerator čestica u svijetu, koji treba da otpočne sa radom 2007. godine. *WLCG* projekat dijeli priličan dio svoje infrastrukture sa *Enabling Grids for E-sciencE (EGEE)* [7] projektom čiji je glavni cilj da obezbjedi naučnicima neprekidan pristup geografski distribuiranoj *Grid* infrastrukturi i razvije robustan *middleware.* U regionu jugoistočne Evrope je aktivan *SEE-GRID* [8] projekat koji podstiče razvoj *Grid* infrastrukture u našem regionu.

U okviru ovog rada razvijena je metodologija i niz testova koji omogućuju mjerenje performansi jednog od *Grid* servisa, *Workload Management System-a*. Nakon uvoda u osobine *Grid middleware-a* u poglavlju 2, *Workload Management System* će biti detaljnije predstavljen u poglavlju 3. Centralni dio rada je izložen u poglavlju 4, a prije svega način mjerenja vremena koje je potrebno da se prihvati korisnički zahtjev za izvršavanjem posla, naročito kada je u pitanju veliki broj poslova, pri čemu se mijenjaju različiti parametri. Dobijeni rezultati prikazuju odgovor sistema na zahtjeve korisnika, što može biti korisno kako onima koji razvijaju sistem (poboljšanje kvaliteta usluge), tako i onima koji žele da gridifikuju svoje aplikacije (izbor najefikasnijeg načina za podnošenje zahtjeva za izvršavanjem poslova). U prilozima su dati primjeri primjene *Grid* komandi, korišteni skripti, kao i kompletni rezultati u tabelarnom i grafičkom obliku.

2 Grid middleware

Grid infrastruktura je sastavljena od različitih uređaja povezanih računarskom mrežom, gdje svaki uređaj, pojedinačno, komunicira sa spoljnim svijetom ali i sa uređajima u okviru *Grid* mreže. Da bi ovakav sistem funkcionisao, potrebno je obezbjediti softver koji će omogućiti povezanost i interoperabilnost svih komponenti sistema, tako da one mogu da razmjenjuju i koriste informacije dobijene od drugih komponenti. Takav posrednički softver nalazi se iznad sloja operativnog sistema i komunikacijskih protokola, a ispod aplikativnog sloja i ima ulogu da:

- sakrije distribuciju aplikacije, tj činjenicu da se aplikacija obično sastoji od više povezanih dijelova koji rade na različitim lokacijama,
- sakrije heterogenost sistema sastavljenog od različitih hardverskih komponenti, operativnih sistema, i komunikacijskih protokola,
- obezbjedi uniforman, standardan interfejs (*interface*) visokog nivoa onima koji razvijaju aplikacije, tako da se one mogu lako kreirati, koristiti, podići na različitim platformama i biti ineteroperabilne sa drugim aplikacijama,
- obezbjedi skup servisa različite funkcionalnosti u svrhu poboljšanja komunikacije među aplikacijama.

Koncepcijski, on se nalazi između dva tipa softvera , operativnog sistema, koji pokreće računar i aplikacijskog softvera, koji riješava konkretan korisnički problem, te je za njegov naziv upotrebljen pojam *middleware* [9].

Dakle, *middleware* čini *Grid* funkcionalnim i omogućuje korisniku jedinstven pristup različitim resursima koji postoje u okviru *Grid* okruženja. Postoje različite distribucije *middleware-a*, od kojih su najznačajniji *Globus, LCG, gLite* i *UNICORE*. Danas je u širokoj upotrebi *gLite middleware* [10], nasljednik *LCG-2 middleware-a* razvijenog u okviru *WLCG* projekta [6]. On se razvija i održava u okviru *EGEE* projekta [7] koji dijeli značajan dio infrastrukture sa *WLCG* projektom, te možemo govoriti o *WLCG/EGEE* infrastrukturi. *gLite middleware* sakriva kompleksnost ovakvog okruženja od korisnika, predstavljajući krajnjem korisniku koherentni virtuelni računarski centar u kom se nalaze svi dostupni resursi. Pristup sevisima je ostvaren putem korisničkih interfejsa, kako u komandnoj liniji, tako i u grafičkom okruženju ili korišćenjem odgovarajućih aplikacija zasnovanih na različitim *API-jima* (*Application Programming Interface).*

2.1 *WLCG/EGEE* arhitektura

U ovom odjeljku dat je pregled *WLCG/EGEE middleware* arhitekture, odnosno osnovnih blokova koji je sačinjavaju i servisa koji omogućavaju korisniku da efikasno i u skladu sa svojim potrebama koristi *Grid* [11].

2.1.1 Sigurnost

Da bi korisnik mogao da koristi dijeljene resurse mora biti član virtuelne organizacije, što znači

da je upoznat i saglasan sa pravilima upotrebe resursa i drugim pravilima koje propisuje određena virtuelna organizacija i da je registrovao svoje lične podatke u registracionom servisu (*Registration Service*).

Grid Security Infrastructure (GSI) [12], koji je baziran na enkripciji javnim ključem, X.509 sertifikatu i *Secure Sockets Layer (SSL)* komunikacijskom protokolu, omogućuje sigurnu autentifikaciju i komunikaciju preko otvorene mreže.

Neophodno je, u svrhu međusobne autentifikacije, da korisnici i servisi posjeduju digitalne X.509 sertifikate dobijene od odgovarajućeg autoriteta, *Certification Authority (CA),* koji potvrđuje identet i pravo posjedovanja sertifikata. Ovaj sertifikat, čiji je privatni ključ zaštićen lozinkom, se koristi za generisanje privremenog sertifikata pod nazivom *proxy certificate*, koji se koristi za stvarnu autentifikaciju i pristup servisima, ne posjeduje lozinku i ima kratak vijek trajanja, da bi se izbjegla mogućnost krađe i zloupotrebe.

2.1.2 User Interface

User Interface (UI) predstavlja tačku pristupa *Grid-u* od strane korisnika. To može biti bilo koja mašina na kojoj korisnik ima nalog i na kojoj je instaliran korisnički sertifikat. Preko *UI* je moguće autorizovati i autentifikovati se, a zatim i pristupiti servisima koji omogućuju pristup informacijama o resursima, upravljaju poslovima ili podacima. Postojeći alati, u linijskom okruženju, koriste se za izvršavanje osnovnih operacija :

- izlistavanje resursa koji odgovaruju potrebama posla koji treba da se izvrši,
- podnošenje posla na izvršenje,
- otkaz posla,
- preuzimanje rezultata izvršenog posla,
- pregled informacija o statusu posla koji se izvršava,
- smještanje, kopiranje i brisanje fajlova sa *Grid-a*,
- pregled informacija o statusu različitih resursa u okviru *Grid* mreže.

Takođe, korišćenjem *API* funkcija biblioteka koji se nalaze na *UI* mogu se razvijati aplikacije.

2.1.3 Computing Element

Computing Element (CE) predstavlja skup računarskih resursa u obliku klastera ili računarske farme. Čine ga *Grid Gate (GG)*, generički interfejs ka klasteru, *Local Resource Management System (LRMS)*, koji se naziva i *batch* sistem, kao i sam klaster, kolekcija radnih čvorova *(Worker Nodes - WNs)* na kojima se izvršavaju poslovi.

GG je odgovoran za prihvatanje i raspoređivanje poslova na odgovarajuće radne čvorove uz pomoć *LRMS-a*. Na radnim čvorovima su instalirane iste komande i biblioteke kao na *UI-u*, isključujući komande vezane za upravljanje poslovima.

Svaki *CE* posjeduje bar jedan red (*queue*) za čekanje, u koji se smještaju poslovi koje je potrebno izvršiti. Na istom klasteru različiti redovi odgovaraju različitim *CE-ovima*, što se koristi za definisanje redova prema karakteristikama poslova, njihovom trajanju ili potrebnoj memoriji, ili pak prema pripadnosti odgovarajućoj virtuelnoj organizaciji.

2.1.4 Storage Element

Storage Element (SE) obezbjeđuje uniforman pristup resursima za skladištenje podataka. U upotrebi su različiti sistemi od jednostavnih disk servera do *Mass Storage Systems (MSS*) sistema, koji se sastoje od puno diskova ili čak i magnetnih traka.

Podržani su različiti protokoli i interfejsi za pristup podacima. Za prenos kompletnih fajlova koristi se *GSIFTP* protokol [13], koji pruža brz, efikasan i siguran prenos podataka sa ili na resurs za skladištenje podataka. Osim njega, za udaljeni pristup podacima koriste se još dva protokola, *RFIO* [14] i *gsidcap*, *GSI* sigurna verzija *dcap* protokola [15].

U rad nekih *SE-a* uključen je modul *middleware-a Storage Resource Manager (SRM)* [16], koji sakriva kompleksnost resursa, omogućujući transparentnu migraciju fajlova, rezervaciju prostora, dostupnost fajlova u određenom vremenskom periodu, itd. U pozadini ovog interfejsa definisana su pravila prenosa fajlova sa diskova na trake, prava korisnika da čita i piše i dr. U opticaju su različite verzije *SRM* interfejsa, koje pružaju različite mogućnosti. Ovaj standard trenutno postaje obavezan u okviru EGEE infrastrukture.

SE bazirani na korišćenju diskova u zavisnosti od implementacije podržavaju *SRM* interfejs (*Disk Pool Manager*) ili ne *(classic SE)*, dok oni bazirani na *MSS* sistemima to uvijek čine. U najskorijoj budućnosti (do kraja 2007. godine) *SE* sistemi koji ne podržavaju *SRM* interfejs biće napušteni.

2.1.5 Information Service

Information Service (IS) obezbjeđuje informacije o *WLCG/EGEE Grid* resursima i njihovom statusu. One su od ključnog značaja za funkcionisanje cjelokupnog *Grid-a*, jer omogućuju pronalaženje odgovarajućih resursa i praćenje njihovog korišćenja.

Većina publikovanih podataka u okviru *IS-a* je u skladu sa *GLUE* shemom, koja definiše uobičajeni konceptualni model podataka za praćenje i otkrivanje *Grid* resursa [17]. U upotrebi su dva tipa *IS-a*: G*lobus Monitoring and Discovery Service (MDS)* [18], koji se koristi za otkrivanje resursa i objavljivanje njihovog statusa, i *Relational Grid Monitoring Architecture (R-GMA)* [19], koji se koristi za praćenje resursa i publikovanje informacija na korisničkom nivou.

2.1.5.1 MDS

MDS koristi specijalizovanu bazu podataka optimizovanu za čitanje, pristupanje i pretraživanje informacija. Pristup podacima ne koristi nikakvu vrstu autentifikacije ili autorizacije, kako za čitanje, tako i za pisanje podataka.

Model je baziran na ulazima (*entries*), koji imaju jedan ili više atributa, određenog tipa i vrijednosti, a jedinstveno ih određuje ime, *Distinguished Name (DN)*. *DN* se formira od sekvence atributa i njihove vrijednosti.

Arhitektura *MDS-a* je piramidalnog tipa. Računarski resursi, resursi za skladištenje podataka i ostali servisi generišu relevantne informacije o sebi i objavljuju ih koristeći *Grid Resource*

Information Server (GRIS). Na nivou sajta drugi server, *Site Grid Index Information Server (Site GIIS)*, sakuplja informacije od lokalnih *GRIS-eva* i ponovo ih objavljuje. Na vrhu hijerarhije se nalazi *top-level BDII (Berkley Database Information Index)* koji ispituje sve *site GIIS-eve*, smještajući dobijene informacije u svoju bazu podataka. Dakle, *top-level BDII* sadrži sve dostupne informacije o cijelom *Grid-u.*

2.1.5.2 R-GMA

U okviru *R-GMA* informacionog servisa, baza podataka ima oblik relacione baze podataka, sa nekim razlikama. Na primjer, moguće je postojanje više redova, ulaza u bazu, sa istim primarnim ključem. Ovaj model je bolji od korišćenog u slučaju *MDS-a*, jer omogućuje naprednije upite i lakše modelovanje sheme, što ga čini prilagođenijem potrebama korisnika.

Arhitekturu čine tri glavne komponente:
- Proizvođači (*Producers),* koji obezbjeđuju informaciju, registruju se u registru i daju opis i strukturu informacije koju obezbjeđuju.
- Potrošači *(Consumers)*, koji traže informaciju, ispituju registar u potrazi za dostupnim informacijama i proizvođačima koji ih obezbjeđuju. Upoznat sa informacijom, potrošač može da kontaktira prozvođača u cilju dobijanja relevantnih podataka.
- Registar (*Registry*), koji posreduje u komunikaciji između prozvođača i potrošača.

Posmatrano iz ugla korisnika sve izgleda kao velika relaciona baza podataka i upiti se definišu kao da ona to i jeste, koristeći podskup komandi *SQL* jezika. *R-GMA* server je *Java web* aplikacija zasnovana na servletima dizajnirana da radi u servletskom kontejneru kakav je *Jakarta Tomcat*.

2.1.6 Data Management

U distribuiranom okruženju moguće je da postoji više kopija fajla, tzv. replika, na različitim fizičkim lokacijama. Idealno, korisnik ne mora da zna gdje se fajl nalazi. On koristi logičko ime fajla i uz pomoć *Data Management* servisa, koji upravlja podacima, pronalazi fajl i koristi ga.

Fajl se može imenovati na različite načine:
- *Grid Unique Identifier (GUID)* ga jedinstveno identifikuje i dodjeljuje se prilikom registracije,
- *Logical File Name (LFN)* je logičko ime fajla koje daje sam korisnik,
- *Storage Unified Resource Locator (SURL)* obezbjeđuje informacije o fizičkoj lokaciji fajla,
- *Transport URL (TURL)* nosi neophodne informacije za pristup fajlu.

GUID i *LFN* identifikuju fajl nezavisno od njegove lokacije, a *SURL* i *TURL* sadrže informacije gdje se replika nalazi i kako joj se može pristupiti.

Mapiranje između različitih naziva vrši se u okviru *File Catalog* servisa, dok su sami fajlovi smješteni na *Storage Element-ima.*

2.1.7 Workload Management

Svrha *Workload Management System-a (WMS)* [20] je da prihvati korisničke poslove, dodjeli im odgovarajući *Computing Element*, prati njihov status i vrati rezultate završenih poslova. Mašina na kojoj *WMS* radi se obično zove *Resource Broker (RB)*.

Za opis poslova koristi se *Job Description Language (JDL)* [21][22]. Opis sadrži nazive izvršnih i drugih potrebnih fajlova za izvršenje posla i neophodne karakteristike okruženja u kom će se posao izvršavati. Na osnovu toga se odabira najpogodniji *CE*.

Najnovija implementacija *WMS-a* omogućuje ne samo podnošenje zahtjeva za izvršavanje jednog, već čitave kolekcije poslova paralelno, pri čemu ti poslovi mogu biti u zavisnom odnosu.

Logging and Bookkeeping service (LB) [23] prati izvršavanje poslova kojim rukovodi *WMS* i bilježi njihov status i istoriju. Ovaj servis je obično kolociran sa *WMS*-om, ali to ne mora da bude slučaj.

2.1.7.1 Tok izvršavanja posla

Nakon dobijanja digitalnog sertifikata, učlanjenja u virtuelnu organizaciju i otvaranja naloga na *User Interface-u*, korisnik je spreman da koristi *Grid*. U izvršavanju korisničkog posla učestvuju različite komponente *Grid-a*, koje međusobno interaguju. Sl. 1 prikazuje šta se dešava sa poslom od trenutka podnošenja zahtjeva za izvršavanjem od strane korisnika.

Da bi mogao da podnese zahtjev za izvršavanjem posla potrebno je da se prijavi na *UI* i kreira *proxy* sertifikat, koji će ga autentifikovati u procesu izvršavanja posla.

Zahtjev za izvršavanje posla, podnešen na *UI-u*, proslijeđuje se *WMS-u*. U okviru fajla koji opisuje posao i uslove pod kojim ga je potrebno izvršiti, moguće je specificirati fajlove koje je potrebno prebaciti sa *UI-a* na radni čvor. Takav skup fajlova naziva se *Input Sandbox*. *LB* servis bilježi događaj, a posao dobija status *submitted* (b).

WMS treba da pronađe *CE* koji najbolje odgovara zahtjevima korisnika. Informacije o statusu računarskih i resursa za skladištenje dobijaju se od *BDII-ja*, a lokacija potrebnih ulaznih fajlova korišćenjem *File Catalogue-a. LB* servis bilježi događaj,a posao dobija status *waiting* (c).

Zatim se posao pripremi za predaju odgovarajućem *CE-u* dopunjavanjem fajla sa početnim zahtjevima svim potrebnim parametrima (*wrapper script*) koji omogućuju precizno i efikasno izvršavanje posla. *LB* servis bilježi događaj,a posao dobija status *ready* (d).

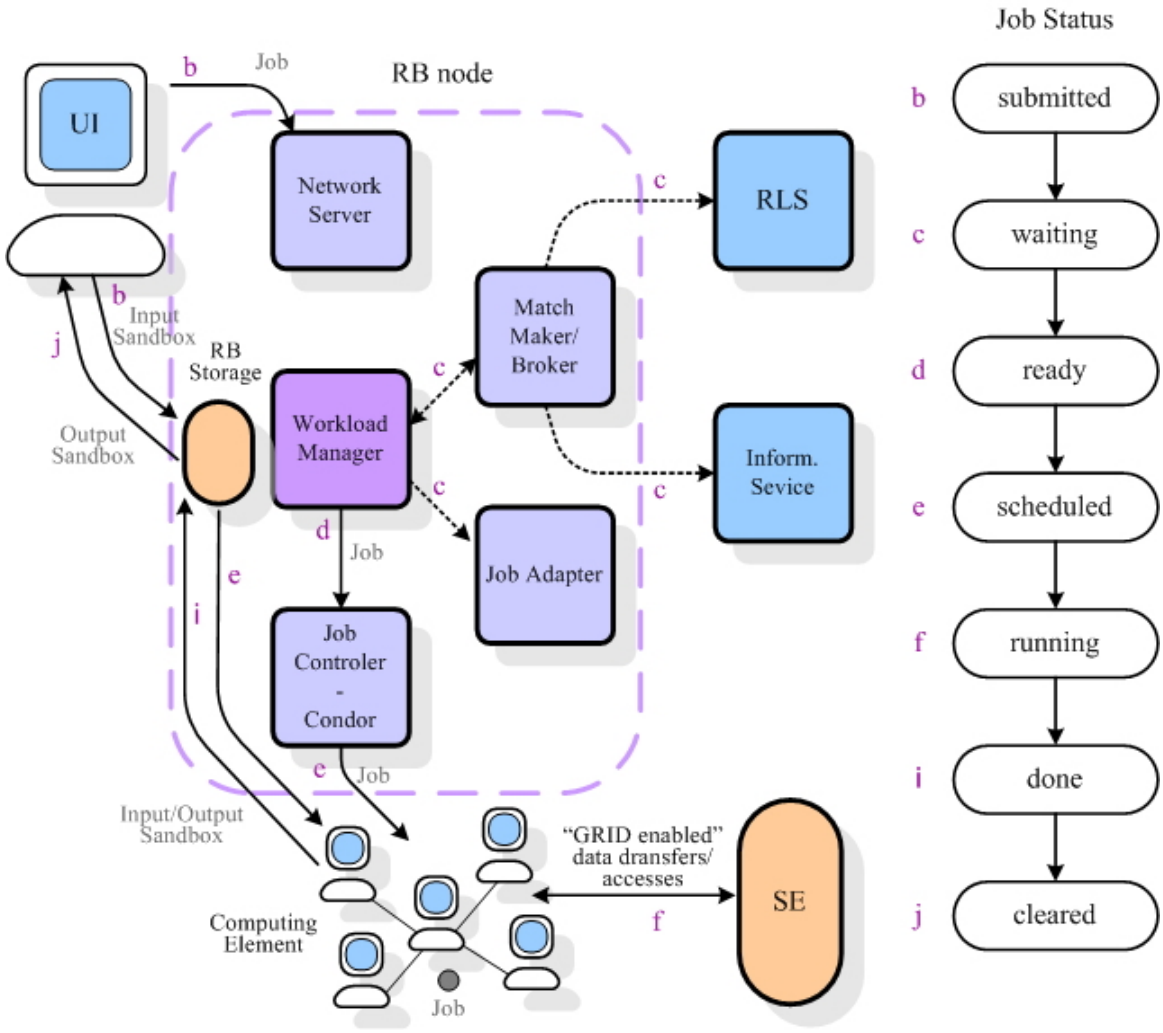
CE prima zahtjev za izvršenjem posla i šalje posao lokalnom *LRMS-u*. *LB* servis bilježi događaj, a posao dobija status *scheduled* (e).

LRMS upravlja izvršavanjem posla na dostupnom skupu radnih nodova. Korisnički fajlovi se kopiraju sa *WMS-a* na radni čvor, na kom se posao izvršava. *LB* servis bilježi događaj, a posao dobija status *running* (f).

Tokom izvršavanja posla, *Grid* fajlovima na *SE* se može direktno pristupati korišćenjem odgovarajućih protokola, sigurne verzije *RFIO* ili *gsidcap*, ili se oni prebacuju na radni čvor

alatima za upravljanjem podacima i koriste se lokalno. Fajlovi nastali u toku izvršavanja mogu se smjestiti na *SE* i registrovati u *File Catalogue-u* upotrebom alata za upravljane podacima i na taj način gridifikovati. Takođe, moguće je drugim korisnicima dozvoliti pristup istim.

Ako se posao izvrši bez greške, rezultati, koji nisu veliki fajlovi i označeni su od strane korisnika u okviru skupa fajlova, tzv. *Output Sandbox-a*, se prebacuju na *WMS*. *LB* servis bilježi događaj, a posao dobija status *done* (i).



Sl. 1: Tok izvršavanja posla u Grid sistemu.

Nakon toga, korisnik može da prebaci rezultate svog posla na *UI*. *LB* servis bilježi događaj, a posao dobija status *cleared* (j).

Ukoliko posao nije prihvaćen ili se ne može izvršiti na odabranom ili alternativnom *CE-u*, koji zadovoljava zahtjeve posla, biće označen statusom *aborted*.

Upotrebivši *UI* kao pristupnu tačku moguće je ispitivati status i istoriju posla, slanjem upita *LB* servisu, kao i status resursa, slanjem upita *BDII-u.*

3 Workload Management System (WMS)

Uloga *Workload Management System-a (WMS)* [20] je da od klijenta prihvati zahtjev za izvršavanjem i upravljanjem izvršenja posla i preduzme odgovarajuće akcije da taj zahtjev zadovolji. Složenost upravljanja aplikacijama i resursima je skrivena od korisnika. U Prilogu E dat je detaljan prikaz komponenti *gLite middleware-a* koje učestvuju u lancu radnji koje je potrebno izvršiti od trenutka kada korisnik podnese zahtjev za izvršavanjem posla do trenutka kada se posao izvršava. Korisnikova komunikacija sa *WMS-om* je ograničena na upotrebu odgovarajućih interfejsa, preko kojih upućuje zahtjev sa opisom karakteristika i uslova pod kojim se posao izvršava. Za definisanje zahtjeva se koristi korisnički orijentisan jezik visokog nivoa, *Job Description Language (JDL)* [21][22].

WMS je odgovoran za prevođenje apstraktnih potreba posla u skup postojećih *Grid* resursa kojima korisnik ima pravo da pristupi. Podržani su slijedeći tipovi zahtjeva:

- *Job*: jednostavna aplikacija, koja može biti *batch* (niz komandi koje se izvršavaju bez korisničke interakcije), interaktivna,zasnovana na proslijeđivanju poruka, sa kontrolnim tačkama, sastavljena od skupa nezavisnih poruka, parametrička,
- *DAG*: direktni aciklični graf zavisnih poslova,
- *Collection*: kolekcija ili skup nezavisnih poslova.

Osim podnošenja zahtjeva za izvršavanje posla, moguće je otkazati posao, preuzeti rezultate, pregledati fajlove vezane za posao i pratiti status posla.

Posao mora biti opisan u pogodnom *JDL* fajlu. Dozvoljeno je koristiti različite atribute za opis zahtjeva, a da to ne proizvede greške, ali je samo određen skup atributa podržan od strane *WMS* komponenti prilikom raspoređivanja i proslijeđivanja zahtjeva na izvršenje.

3.1 Arhitektura servisa

Korisnik pristupa različitim servisima za upravljanje poslom u okviru *WMS-a* posredstvom skupa klijentskih alata, koji nose zajedničko ime *WMS-UI (WMS User Interface)*. Pristup se ostvaruje u linijskom ili grafičkom okruženju, kao i upotrebom aplikacija zasnovanih na različitim ponuđenim *API* funkcijama, koje za podlogu imaju *C++* ili *Java* jezik.

WMS-UI obezbjeđuje slijedeće operacije:

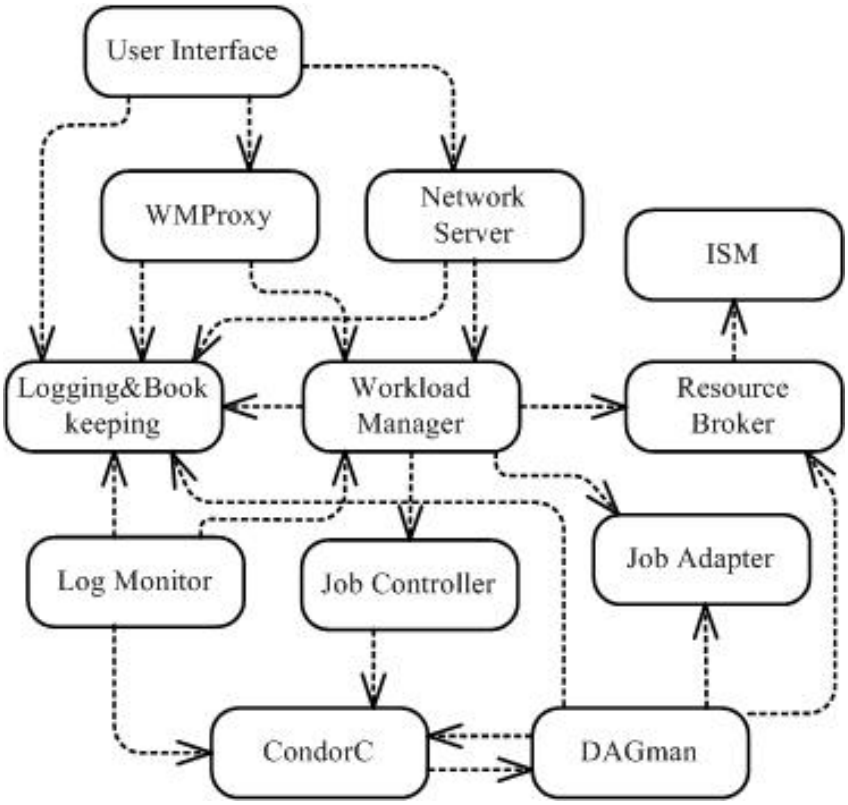
- pronalaženje liste resursa koji odgovaraju određenom poslu,
- podnošenje zahtjeva za izvršenjem posla na udaljenom *CE-u*,
- provjera statusa posla,
- otkaz posla,
- preuzimanje rezultata završenih poslova,
- potraživanje i prikaz informacija vezanih za istoriju izvršavanja posla,
- pregled statusa kontrolnih tačaka poslova koji imaju iste,
- pokretanje lokalnog osluškivača, kada su u pitanju interaktivni poslovi.

Nakon podnošenja putem *WMS-UI-a*, zahtjev prolazi kroz druge *WMS* komponente prije nego što se izvrši i pri tom mijenja stanja u kojima se nalazi, što se može prikazati konačnim automatom stanja.

Slijedeće komponente učestvuju u upravljanju poslom:

- *WMProxy/Network Server,*
- *Workload Manager,*
- *Resource Broker,*
- *Job Controller,*
- *CondorC/DAGMan,*
- *Logging and Bookkeeping,*
- *Log Monitor*.

Sl. 2 prikazuje unutrašnju arhitekturu *WMS-a*:



Sl. 2: Pregled arhitekture WMS-a.

Network Server (NS) je generički mrežni *daemon*, program koji se odvija u pozadini sa ciljem da prihvati korisničke zahtjeve i proslijedi ih dalje, ukoliko su valjani, *Workload Manager-u*. On obezbjeđuje podršku za funkcionalnost kontrole posla. Koncept je naslijeđen iz predhodno korišćenog *middleware-a LCG-2*.

Workload Manager Proxy (WMProxy) je servis koji obezbjeđuje pristup funkcionalnosti *WMS-a* putem interfejsa zasnovanog na *Web* servisima. Osim toga što je prirodna zamjena za *NS* pri prelasku na servisno orijentisanu arhitekturu *(SOA)*, upotreba *WMProxy-ja* unosi nove osobenosti sistema, kao što je podnošenje zahtjeva za paralenim izvršavanjem više poslova i podrška dijeljene i komprimovane skupove fajlova potrebnih za izvršavanje posla ili generisanih tokom izvršavanja posla za složene poslove.

Workload Manager (WM) je ključna komponenta *WMS-a*, koja po dobijanju valjanog zahtjeva

preduzima odgovarajuće akcije da bi ga zadovoljila. Ona djeluje u sadejstvu sa drugim komponentama, specifičnim za različite tipova zahtjeva.

Dva tipa zahtjeva su bitna za poslove koji vrše izračunavanja: zahtjev za izvršavanjem posla i otkaz posla. Konkretno, smisao zahtjeva za izvršavanjem posla je da prebaci odgovornost za posao na *WM*. Nakon toga, *WM* šalje posao na izvršavanje na odgovarajući *CE,* uzimajući u obzir specifikacije navedene u opisu posla. Odluka o odgovarajućem resursu proizilazi iz procesa uparivanja zahtjeva iz opisa posla i dostupnih resursa. Dostupnost resursa ne zavisi samo od stanja u kom se resurs nalazi, već i od uslova korišćenja istog koje postavlja onaj koji upravlja resursom i/ili virtuelna organizacija kojoj pripada korisnik.

Resource Broker (RB) ili ***Matchmaker*** je jedna od tzv. pomoćnih klasa koja pruža podršku *WM-u* u izboru resursa koji najviše odgovara zahtjevima posla.

WM može na dva različita načina da rasporedi posao. U prvom slučaju posao se uparuje sa nekim od slobodnih resursa koji odgovara zahtjevima posla i odmah se upućuje na izvršavanje. Riječ je o tzv. politici raspoređivanja bez zadrške *(eager scheduling policy)*. U drugom slučaju *WM* zadržava poslove dok se ne oslobodi odgovarajući resurs, koji odgovara zahtjevima poslova koje je potrebno izvršiti, a izvršava se posao čiji zahtjevi najviše odgovaraju karakteristikama resursa. Riječ je o raspoređivanju sa zadrškom *(lazy scheduling policy)*.

Ono što omogućuje fleksibilnu primjenu različitih politika raspoređivanja je mehanizam razdvajanja skupa informacija koji se tiču resursa i njegove primjene. ***Information Super Market (ISM)*** je komponenta koja implementira ovaj mehanizam i u osnovi se sastoji od skupa informacija o resursima koje modul za uparivanje može pročitati i čije je obnavljanje rezultat dostavljanja informacija ili aktivnog ispitivanja resursa, ili pak kombinacija jednog i drugog. Moguće je konfigurisati *ISM* tako da određene notifikacije pokrenu modul za uparivanje, što popravlja modularnost softvera i podržava implementaciju politike raspoređivanja sa zadrškom.

Task Queue (TQ) je komponenta koja pripada unutrašnjem dizajnu *WM*-a i omogućuje zadržavanje zahtjeva za izvršavanjem posla ukoliko nijedan slobodan resurs ne zadovoljava zahtjeve posla. U slučaju raspoređivanja bez zadrške pokušaj uparivanja se ponavlja periodično, dok se kod raspoređivanja sa zadrškom uparivanje vrši čim se pojavi slobodan resurs u *ISM-u*. Alternativno, takva situacija vodi trenutnom otkazu posla uslijed nedostatka odgovarajućeg resursa.

Job Adapter (JA) je odgovoran za završne akcije nad *JDL* fajlom koji opisuje posao i ono što je potrebno za njegovo izvršavanje, prije njegovog proslijeđivanja *CondorC* komponenti koja će izvršiti stvarnu predaju zahtjeva za izvršavanje posla. Osim za pripremanje *CondorC* fajla, ovaj modul je odgovoran za kreiranje skupa instrukcija (*job wrapper script*) koje će stvoriti odgovarajuće izvršno okruženje na *CE* radnom čvoru, što uključuje i prenos ulaznih i izlaznih fajlova vezanih za posao.

CondorC je modul odgovoran za konkretno upravljanje operacijama vezanim za posao koje zahtjeva *Workload Manager*.

Svrha ***DAGMan (DAG Manager)*** modula je da upravlja grafom poslova (*DAG*), razluči koji su čvorovi nezavisni i prati izvršavanje datih poslova. *CondorC* pokreće *DAGMan* kada je potrebno obraditi tip posla označen kao direktni aciklični graf, tj. *DAG*.

Log Monitor (LM) je odgovoran za praćenje *CondorC log* fajla, u kom se bilježi istorija aktivnosti *CondorC* modula, za presretanje interesantnih događaja vezanih za aktivni posao, tj.

događaja koji utiču na konačni automat stanja i pokretanje odgovarajućih akcija vezanih za to.

Proxy Renewal Service treba da osigura da tokom cjelokupnog trajanja izvršavanja posla, postoji valjan *proxy* u okviru *WMS-a*. On se u svom radu oslanja na *MyProxy* servis za obnavljanje ovlašćenja asociranih sa zahtjevom.

Logging and Bookkeeping (LB) obezbjeđuje podršku za nadgledanje posla, bilježeći informacije vezane za događaje generisane od strane različitih komponenti *WMS-a*. Koristeći ove informacije *LB* servis omogućava prikaz izvršavanja posla kroz konačni automat stanja.
Korisnik može da sazna u kom se stanju nalazi posao ispitujući *LB* servis, korišćenjem odgovarajuće naredbe obezbjeđene u okviru *WMS-UI* alata. Pored toga, pruža mu se mogućnost dobijanja obavještenja o naročitom stanju posla, npr. o završetku istog, korišćenjem odgovarajuće infrastrukture.

3.2 Interakcija sa drugim servisima

Sve komponente *WMS-a* interaguju sa *LB*-om proslijeđujući mu informacije o poslovima kojima upravljaju u tom trenutku i potražujući informacije o njima kada su im potrebne. Osim toga u fazi uparivanja *RB* komunicira sa katalozima datoteka u okviru servisa za upravljanje podataka *(Data Mangement Catalogues)* koristeći odgovarajući interfejs, *StorageIndex* interfejs, u cilju otkrivanja lokacije postojećih imena fajlova u katalozima, a navedenim u ulaznom skupu u okviru *JDL* fajla, ne bi li se posao izvršio na radnom čvoru koji se nalazi blizu korišćenih fajlova.

WMS, takođe, indirektno komunicira sa *Virtual Organisation Membership Service (VOMS)* servisom, jer iščitava podatke o virtuelnoj organizaciji, grupi i mogućnostima iz korisničkih ovlašćenja dobijenih od *VOMS-a*, a koji se koriste za korisničku autorizaciju i autentifikaciju. *Proxy Renewal* komponenta *WMS-a* kontaktira *MyProxy* servis u cilju automatskog obnavljanja ovlašćenja dugoživećih poslova.

Na kraju treba naglasiti interakciju sa *CE-om* pri predaji posla, preko *CondorC* modula, ili pri dobijanju sinhronih/asinhronih potvrda o karakteristikama i statusu resursa. Takođe, postoje konfiguracije gdje postoji komunikacija sa *BDII-em* da bi se dobila informacija o svim dostupnim resursima određene virtuelne organizacije.

3.3 Sigurnost

Da bi se osigurao rad u sigurnom okruženju, sve interakcije između *WMS* komponenti, a naročito onih koje koriste mrežu se obostrano autentifikuju. U zavisnosti od specifične interakcije entitet se autentifikuje koristeći sopstvena ovlašćenja ili delegira korisnička ili koristi oba. *WMS-UI* koristi *proxy* korisnički sertifikat da bi se smanjio rizik zloupotrebe originalnog sertifikata.

Podaci o korisniku ili servisu i njihov javni ključ su uključeni u X.509 sertifikat potpisan od strane *Certification Authority (CA)*, koji garantuje povezanost javnog ključa i njegovog vlasnika.

3.4 Stari i novi pristup: *Network Server* i *WMProxy*

WMProxy [24] je nova komponenta za pristup *gLite Workload Management System-u*, koja je razvijena ne samo iz potrebe za primjenu nove metodologije dizajna, *Service Oriented Architecture (SOA)*, već da popravi performanse postojećih sličnih komponenti, da efikasnije upravlja velikim brojem zahtjeva za izvršavanjem posla, kontrolom posla i da obezbjedi dodatne karakteristike.

WMProxy je implementiran kao *Web* servis. Ono što čini *Web* uspješnom tehnologijom je jednostavnost i rasprostranjenost. *Web* servis omogućava da se iskoriste prednosti *Web-a*, ne samo za razmjenu informacija, već da se ponude usluge većoj zajednici mogućih korisnika.

Tehnologija *Web* servisa omogućuje interoperabilnost *Grid* servisa koji čine *gLite middleware*, a obezbjeđuju ih i/ili upravljaju njima različite organizacije i lakše usaglašavanje sa novim standardima kao što su ***Open Grid Services Architecture (OGSA)*** **i** ***Web Services Resource Framework (WSRF).***

Naslijeđeni koncept je upotreba generičkog mrežnog *daemon-a*, *Network Server-a*, programa koji se odvija neprekidno u pozadini, koji prihvata pristigle zahtjeve i ukoliko su valjani proslijeđuje ih *Workload Manager-u*. Trenutno je moguće pristupiti *WMS-u* preko jedne ili druge komponente, upotrebom odgovarajućeg skupa naredbi [20][25].

3.4.1 Nove karakteristike

WMProxy uvodi neke novine u cilju poboljšanja performansi predaje poslova na izvršavanje i efikasnog upravljanja veoma velikim brojem poslova ili zahtjeva. Najznačajnije od njih su vezane za:

- paralelnu predaju više poslova *(Bulk Submission),*
- skupove fajlova vezane za posao *(Job's Sandboxes)*,
- asinhroni start posla *(Asynchronous Job Start)*,
- pregledanje fajlova vezanih za posao (*Job's Files Perusal*).

3.4.1.1 Paralelna predaja više poslova

Paralelna predaja više poslova korišćenjem *WMProxy-ja* je zasnovana na novim tipovima zahtjeva: kolekcijama nezavisnih poslova *(Collections)* i parametričkim poslovima *(Parametric Jobs)*, a oslanja se na podršku koju objezbjeđuje *WMS* za direktne aciklične grafove *(DAGs)*.

Nakon podnošenja jednog od takvih zahtjeva, upotrebom jedne komandne linije, *WMProxy* servis koristi *JDL* biblioteku da bi provjerio valjanost zahtjeva, a potom ga pretvara direktno aciklični graf poslova, po mogućstvu bez zavisnosti, koji može da obradi *WMS.*

Ovakav pristup omogućuje predaju velikog broja poslova kroz jedan zahtjev, spriječava opterećenje servisa uslijed višestrukih konekcija, vezanih za autentifikaciju i autorizaciju, omogućava dijeljene skupova fajlova smanjujući na taj način količinu podataka koju je potrebno prenijeti i omogućuje jednostavnu kontrolu cijelog skupa poslova odjednom.

3.4.1.2 Manipulisanje skupovima fajlova vezanim za posao

Pri podnošenju složenih zahtjeva, kao što su direktni aciklični grafovi, kolekcije ili parametrički poslovi, unaprijeđena *JDL* biblioteka omogućuje *WMProxy-u* da prepozna one fajlove koji su zajednički ulaznim skupovima fajlova različitih potposlova datog zahtjeva, da ih tačno lociraju i preurede opise poslova tako da fajlovi budu dostupni u toku izvršavanja svim poslovima kojima su potrebni. Ovo, zajedno sa mogućnošću *WMProxy-ja* da upravlja poslovima, i jednostavim i složenim, čiji su skupovi fajlova prenešeni na *WMS* čvor u obliku komprimovanih arhiva fajlova, smanjuje broj poziva dostupnim servisima za prenos fajlova i smanjuje količinu podataka koje treba prenijeti, što unaprijeđuje performanse predaje zahtjeva za izvršavanjem poslova.

3.4.1.3 Asinhroni start posla

Riječ je o konfigurabilnom operativnom modu *WMProxy* servisa, koji ima za cilj da vrati kontrolu klijentu odmah po završetku registracije posla i njegove predaje *Grid* okruženju na dalje upravljenje. Sve akcije usmjerene ka dovršenju zahtjeva, a za čije izvršenje je potrebno vrijeme odvijaju se u pozadini. Te akcije uključuju registracije potposlova, manipulacije nad njihovim *JDL* fajlovima i kreiranje rezervisanog prostora za potposlove i fajl sistemu. Sve se bilježi u okviru *LB* servisa, pa ukoliko se neka akcija izvrši neuspješno, korisnik to može da uoči i eventualno pokuša da pokrene posao ponovo bez gubljenja prethodno uspješnih procesa.

Osim što korisnik stiče utisak da je servis mnogo brži, ovakav pristup je omogućio razdvajanje modula koji obrađuju zahtjev od onih koji su zaduženi za komunikaciju sa korisnikom.

3.4.1.4 Pregledanje fajlova vezanih za posao

Korišćenje *WMProxy* servisa omogućava pregledanje sadržaja fajlova vezanih za posao u toku izvršavanja posla. Ovo je implementirano kao proces koji se odvija paralelno sa poslom na udaljenom resursu, koji šalje dijelove odabranih fajlova na *WMS* čvor ili odabranu adresu *(URI)*.

Odabir fajla može biti specificiran, promijenjen, uklonjen od strane korisnika upotrebom različitih operacija. Ova funkcionalnost može biti dozvoljena ili zabranjena u bilo kom trenutku predaje posla. *WMProxy* prikuplja i raspoređuje primljene dijelove fajla tako da se mogu proslijediti krisniku na uvid.

Pravilna upotreba ove funkcionalnosti omogućuje korisniku da prati razvoj i ponašanje posla, što utiče na njegovu odluku o mogućem zaustavljanju posla i oslobađanju resursa. Takođe, moguće je brže uočavanje i ispravljanje grešaka i pokretanje različitih testova.

3.4.2 Pregled komandi za pristupanje *WMS* servisu

Za ostvarenje interakcije sa *WMS* servisom korisniku su dostupna dva skupa naredbi u linijskom korisničkom okruženju, u zavisnosti da li se interakcija odvija preko *Network Server-a* ili *WMProxy-ja.*

3.4.2.1 Komande za pristup putem Network Server-a

glite-job-list-match

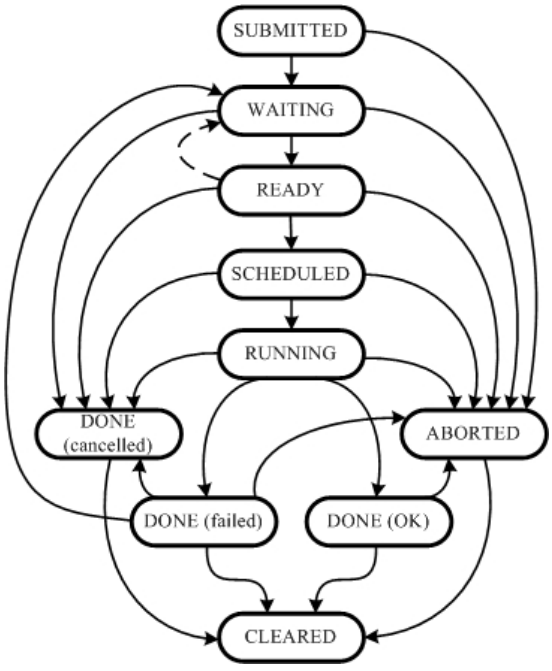
Rezultat izvršenja ove komande je lista identifikatora resursa koje je korisnik ovlašten da koristi, a koji zadovoljavaju zahtjeve posla iskazane u *JDL* fajlu. Moguće ju je izvršiti samo za jednostavne poslove ili pojedinačne čvorove, takođe jednostavne poslove, u okviru *DAG-a*.

glite-job-submit

Ovom komandom se predaje zahtjev za izvršavanjem posla. Potreban ulaz čini *JDL* fajl sa opisom posla, a kao odgovor se dobija identifikator posla.

glite-job-status

Rezultat komande je prikaz statusa posla koji je predat na izvršavanje korišćenjem *glite-job-submit* komande. *LB (Logging and Bookkeeping service)* servis obezbjeđuje traženu informaciju. Kada je u pitanju *DAG* daje se informacija o statusu cjelokupnog posla, kao i o statusu pojedinačnih čvorova. Takođe, moguće je tražiti informaciju o statusu pojedinačnog čvora, koristeći njegov sopstveni identifikator.



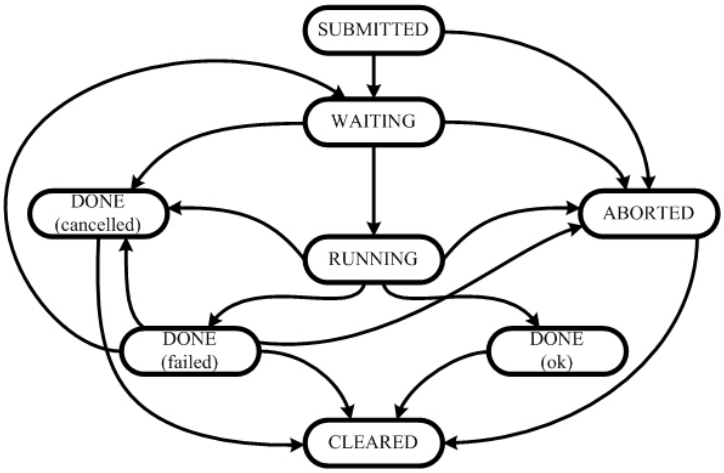
Sl. 3: Konačni automat stanja posla.

Status posla je moguće prikazati konačnim automatom stanja (Sl. 3) kroz koje posao prolazi:

- *submitted*: korisnik je predao posao *UI-u*, ali još nije prebačen *NS-u* na obradu.
- *waiting*: posao je prihvaćen od strane *NS-a* i čeka da ga obradi *WM* ili ga obrađuje neki od *WM-ovih* pomoćnih modula, npr. *WM* je zauzet, još uvijek nije pronađen odgovarajući *CE*, posao čeka na alokaciju resursa, itd.
- *ready*: *WM* ili neki od njegovih pomoćnih modula obrađuje posao, pronađen je odgovarajući *CE*, ali još uvijek nije prebačen u odgovarajući red na *CE* pomoću *Job*

Controller-a i *CondorC-a*. Ovo stanje ne postoji za *DAG* (na Sl. 4), jer ne podliježe uparivanju sa odgovarajućim resursom, već se prosljeđuje direktno *DAGMan-u.*

- *scheduled*: posao čeka u redu na *CE*. Stanje ne postoji za *DAG*, jer se zapravo njegov čvor šalje na *CE.*
- *running*: posao se izvršava. Kada je u pitanju *DAG*, riječ je o tome da je *DAGMan* počeo da ga obrađuje.
- *done*: posao se završio, uspješno ili ne.
- *aborted*: *WMS* je otkazao posao zbog predugog čekanja na neki modul, isteka kotisničkih ovlaštenja, itd.
- *canceled*: posao je uspješno otkazan na korisnički zahtjev.
- *cleared*: izlazni skup fajlova *(output sandbox)* je preuzeo korisnik ili je uklonjen nakon isteka dozvoljenog vremena za skladištenje.



Sl. 4: Konačni automat stanja DAG-a.

glite-job-output

Komanda služi za preuzimanje rezultata završenih poslova čiji opisni *JDL* fajl sadrži definisan izlazni skup fajlova (*Output Sandbox*), a koji se privremeno smještaju na *RB* mašinu. Ulazni argument je identifikator posla dobijen nakon *glite-job-submit* komande. Pošto *DAG* ne posjeduje sopstveni skup izlaznih fajlova, preuzimaju se rezultati svih potposlova.

glite-job-cancel

Korisnik može uptrebom ove komande da otkaže prehodno predat posao na izvršavanje. Potrebno je potvrditi ovaj zahtjev, koji *NS* prosljeđuje *WM-u* na izvršenje. Ne može se otkazati pojedinačni čvor *DAG-a*, već se to mora učiniti sa cjelokupnim *DAG-om.*

glite-job-logging-info

Rezultat komande je prikaz relevantnih događaja vezanih za posao, a zabilježenih u *LB-u* od strane *WMS* komponenti koje obrađuju zahtjev za izvršenjem posla. U slučaju *DAG-a* prikazuju se događaji vezani za njega, a ne za pojedinačne čvorove.

glite-job-attach

Komanda dodaje osluškivač na predhodno predat interaktivni posao, što će usmjeriti tokove posla u komandnu liniju ili određeni grafički prozor.

glite-job-get-chkpt

Komanda vraća stanje kontrolne tačke posla sa kontrolnim tačkama, koje će biti sačuvano u dokumentu na *WMS-UI* mašini, a može se koristiti kasnije za ponovno pokretanje istog ili nekog drugog posla od datog stanja, a ne od početka.

3.4.2.2 Komande za pristup WMS-u preko WMProxy-ja

glite-wms-delegate-proxy

Ova komanda omogućuje delegiranje korisničkog *proxy-a WMProxy* servisu, što omogućuje prenošenje korisničkih prava i privilegija na servise koji obavljaju radnje u ime korisnika interagujući sa drugim servisima.

Korisnik može sam da odredi identifikator delegacije ili da ga komanda automatski generiše.

Delegacija *proxy-a* je potrebna za slijedeće naredbe: *glite-wms-job-submit* i *glite-wms-job-list-match*.

glite-wms-job-submit

Upotrebom ove komande moguće je podnijeti na izvršavanje različite vrste poslova, jednostavne ili složene tipa direktnih acikličnih grafova ili kolekcija nezavisnih poslova. Podnošenje se vrši korišćenjem prethodno delegiranog *proxy-a*, što je preporučen način, ili delegiranjem novog *proxy-a* pri svakom novom podnošenju zahtjeva.

Opis i jednostavnih i složenih poslova nalazi se u *JDL* fajlu koji se predaje kao ulazni argument. Osim na ovaj način kolekcija nezavisnih poslova može se predati specificiranjem putanje do direktorijuma u kom se nalaze pojedinačni *JDL* fajlovi poslova koji čine kolekciju. Naredba će potom generisati *JDL* fajl kolekcije i predati ga.

Važno je primjetiti da se fajlovi koje dijele različiti poslovi, a nalaze se u skupovima fajlova vezanim za određeni posao, prebacuju samo jednom na *WMS* čvor.

Uspješno izvršena komanda vraća identifikator posla, koji se upotrebljava za dalje praćenje posla.

glite-wms-job-list-match

Rezultat izvršenja ove komande je lista *CE-ova* koje je korisnik ovlašten da koristi, a koji zadovoljavaju zahtjeve posla iskazane u *JDL* fajlu. Komanda ne podržava složene zahtjeve, kao što su direktni aciklični grafovi i kolekcije nezavisnih poslova.

glite-wms-job-cancel

Komanda omogućava otkaz prethodno podnešenog zahtjeva za izvršavanjem posla. Ulazni argument je identifikator posla ili lista identifikatora više poslova. Nemoguć je otkaz potposlova

DAG-a, jer bi to narušilo zavisnosti koje postoje među potposlovima.

glite-wms-job-output

Upotrebom ove komande je moguće preuzeti rezultate završenih poslova. Koji se fajlovi preuzimaju definiše se u izlaznom skupu fajlova pri opisu posla. Kada su u pitanju složeni poslovi, preuzimaju se rezultati svih potposlova koji se smještaju u zasebne direktorijume u okviru direktorijuma koji se odnosi na složeni posao u cjelini.

glite-wms-job-perusal

Komanda omogućava pregledanja fajlova posla predanog na izvršavanje identifikovanog identifikatorom posla. To je moguće samo ako je omogućena podrška za tako nešto prilikom podnošenja/registrovanja posla preko odgovarajućeg *JDL* atributa, *PerusalFileEnable.*

Moguće je dozvoliti pregledanje jednog ili više fajlova, preuzimanje dijelova tako specificiranih fajlova ili zabraniti pregledanje svih fajlova vezanih za posao.

Ova funkcionalnost je ostvariva nad jednostavnim poslovima ili čvorovima složenih poslova, ali ne i nad složenim poslovima u cjelini.

Komande *glite-wms-job-status* i *glite-wms-job-logging-info* imaju isti smisao kao i komande *glite-job-status* i *glite-job-logging-info* koje se koriste za pristup *WMS-u* preko *Network Server-a.*

Navedeni skup je u stvari prošireni naslijeđeni skup naredbi korištenih za pristup *WMS-u* preko *Network Server-a*, a koji pruža podršku novouvedenim funkcionalnostima.

4 Mjerenje perfomansi WMS-a

Jedan od temelja razvoja *Grid* sistema je pružanje odgovarajućeg kvaliteta usluga. Sistem treba da pruži adekvatan odgovor svim zahtjevima korisnika. Razvijanjem različitih testnih okruženja sa promijenljivim ulaznim parametrima moguće je pratiti ponašanje sistema i njegovih komponenti i dobiti povratnu informaciju o veličinama koje određuju kvalitet usluge: vrijeme odziva, ukupne performanse, sigurnost, skalabilnost resursa i dr. U ovom radu pažnja je usmjerena na dio sistema koji je zadužen da od klijenta prihvati zahtjev za izvršavanjem i upravljanjem izvršenja posla i preduzme odgovarajuće akcije da taj zahtjev zadovolji, *Workload Management System (WMS)* [20]. Rezultati prikazani ovdje su objavljeni i u okviru rada [26].

U cilju sagledavanja performansi *Workload Management System-a*, naročito pri slanju zahtjeva za izvršavanjem velikog broja poslova, što je tipičan slučaj za aplikacije čije izvršavanje zahtijeva značajne računarske resurse, te se stoga i koristi *Grid* mreža, razvijen je niz testova. Testovi podrazumijevaju podnošenje zahtjeva za izvršavanjem velikog broja poslova pod različitim uslovima, a zatím praćenje i analiziranje kritičnih događaja vezanih za iste.

4.1 Metodologija mjerenja performansi WMS-a

Kao što je navedeno u poglavlju 3, korisnik može na dva različita načina da preda zahtjeve za izvršavanjem poslova, sekvencijalno ili paralelno, korišćenjem usluga jednog od dva servisa, *Network Server-a* ili *WMProxy-ja*. Neke vrste poslova, u cilju svog izvršenja, zahtijevaju prenos određenog broja fajlova na *WMS* prilikom podnošenja zahtjeva za izvršavanjem posla. Da bi korisnik mogao da izabere najefikasniji način za podnošenje zahjeva neophodno je utvrditi kako promjena parametara kao što su izbor servisa, način podnošenja zahtjeva, broj poslova i veličina *Input Sandbox-a*, utiču na vrijeme koje je potrebno da se zahtjev prihvati od strane *WMS-a*, obradi i proslijedi izabranom *Computing Elementu*. U realnom okruženju na veličine koje se žele izmjeriti pored performansi samog *WMS-a* utiču performanse operativnog sistema, mrežne komunikacije drugih procesa koji mogu da postoje na *WMS* i *UI*, drugih poslova koji mogu da rade u tom trenutku, performanse hardvera kao što je disk, prebacivanje podataka iz primarne u sekundarnu memoriju i slično.

Za sagledavanje performansi samog *WMS-a* stvoreno je testno okruženje u kom je moguće izolovati druge uticaje na mjerena vremena. Korišćene komponente su izdvojene iz produkcionog okruženja u svrhu sprovođenja željenih mjerenja. Postoji samo jedan korisnik koji podnosi zahtjeve za izvršavanjem poslova. Zahtjevi se podnose na neopterećen *WMS* (*WMS* je završio obradu svih prethodnih zahtjeva). Ovakav način rada omogućuje poređenje različitih implementacija komponenti *middleware-a* i neophodan je za optimalizaciju rada komponenti *WMS-a.*

Rezultati dobijeni ovom metodologijom se razlikuju od rezultata koji bi se dobili u realnoj situaciji kada u sistemu postoje drugi korisnici koji koriste isti *WMS* ili *UI,* stoga je naredni korak mjerenje performansi produkcionih sistema, pošto se na taj način dobijaju podaci koji su relevantniji za korisnika.

Testno okruženje obuhvata korisnički interfejs *(User Interface (UI))* i *Workload Management System*, koji su povezani preko visoko kvalitetnog *3Com* gigabitnog mrežnog *switch-a*. Tehničke karakteristike korisničkog interfejsa su:

- laptop sa procesorom Pentium M na 1.8 GHz,
- 512 MB RAM memorije,
- 100 Mbps mrežna kartica,

dok je za dedikovani *WMS* korišćena mašina slijedećih karakteristika:

- dvoprocesorski *Intel Xeon* na 2.8 GHz sa omogućenim *hyperthreading-om,*
- 2 GB RAM memorije,
- 1 Gbps mrežna kartica.

Na *WMS* čvoru je instaliran *gLite middleware*, verzija 3.0.2, *update* 13, a na korisničkom interfejsu *GILDA User Interface Plug & Play* [27], zasnovan na *gLite 3.0 middleware*-u, koji omogućuje da se lični računar (*PC*) na kom je instaliran *Linux* operativni sistem koristi kao pristupna tačka *Grid* servisima.

U prvom eksperimentu podnošenje zahtjeva za izvršavanjem poslova je išlo preko *Network Server-a*, a u drugom preko *WMProxy-ja*. U oba slučaja informacije vezane za status posla su dobijane od *Logging and Bookkeeping* servisa. Posmatrana je promjena prosječnog vremena, koje je potrebno *WMS-u* da prihvati, obradi i proslijedi zahtjev *Computing Element-u,* u zavisnosti od načina podnošenja zahtjeva. Zahtjevi se mogu prosljeđivati sekvencijalno (niz poslova) u slučaju *Network Server-a* i *WMProxy-ja* ili paralelno (svi poslovi odjednom) u slučaju *WMProxy-ja* u obliku kolekcija poslova. Takođe, kao što je objašnjeno gore, od interesa je bilo ispitati kako će promjena veličine *Input Sandbox-a*, tj. ukupna veličina fajlova vezanih za posao koje je potrebno prenijeti na *WMS* tokom podnošenja zahtjeva, uticati na prosječno vrijeme podnošenja zahtjeva. Pored toga praćen je uticaj promjene broja zahtjeva na posmatrane vremenske veličine. Za izvođenje testova koriste se skript programi, navedeni u Prilogu B, zasnovani na komandama linijskog interfejsa *(Command Line Interface - CLI)* koji je dio korisničkog interfejsa. U Prilogu A dati su primjeri sekvenci osnovnih koraka koje je potrebno izvršiti da bi se predao zahtjev za izvršavanje posla i poslije toga pratio tok izvršavanja posla korišćenjem usluga *Network Server-a* [20] ili *WMProxy* servisa [24][25].

Za svaki od eksperimenata izvedeno je po dva tipa mjerenja. U okviru prvog tipa mjerenja šalje se sekvencijalno niz zahtjeva za izvršavanje poslova. Nizovi su sadržali 100 i 200 jednostavnih poslova čije izvršavanje ne zahtijeva prenos dodatnih fajlova na *WMS* (ne postoji *Input Sandbox*). Drugi tip mjerenja podrazumijeva poslove čije izvršavanje uključuje prenos dodatnih fajlova na *WMS* (postoji *Input Sandbox*). Posmatrana ukupna veličina dodatnih fajlova u prvom slučaju iznosi 8 kB, a u drugom 5 MB.

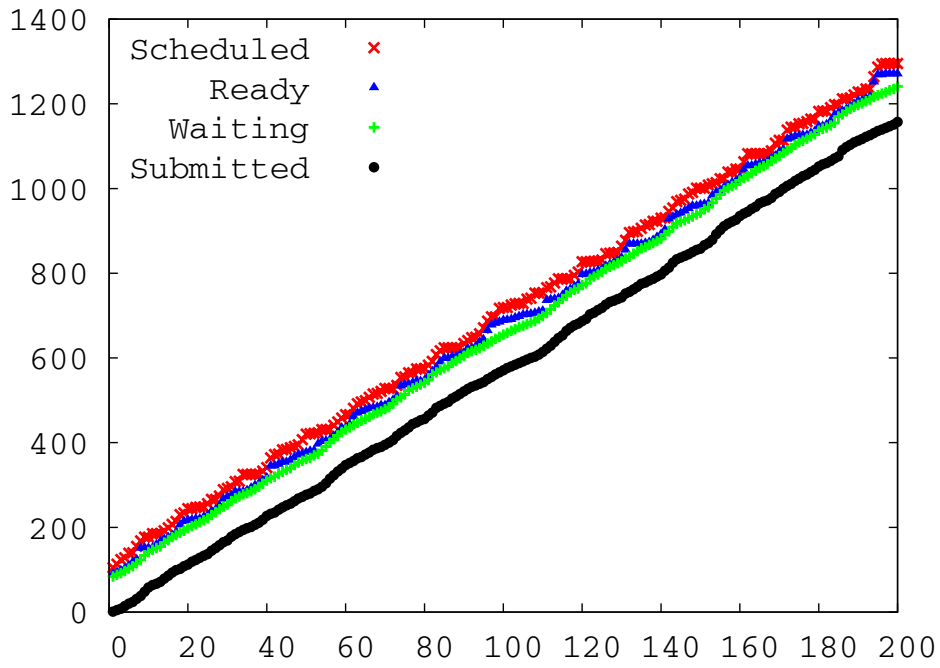
Takođe, istraživano je šta se dešava ako se zahtjevi za izvršavanjem poslova paralelno predaju, upotrebom jedne komandne linije, što je novina koju unosi *WMProxy* servis. Mjerenja su izvedena kako za jednostavne poslove čije izvršavanje ne zahtijeva prenos dodatnih fajlova na *WMS* (ne postoji *Input Sandbox*), tako i za poslove čije izvršavanje uključuje prenos dodatnih fajlova na *WMS* (postoji *Input Sandbox*). Posmatrana ukupna veličina dodatnih fajlova u prvom slučaju iznosi 8 kB, a u drugom 5 MB. Broj poslova u kolekciji iznosi 100 i 200.

U svim slučajevima su praćena slijedeća stanja:

- *Submitted*: korisnik je predao posao *WMS-u*, ali još nije prebačen *NS-u* na obradu.
- *Waiting*: posao je prihvaćen od strane *NS-a* i čeka da ga obradi *WM* ili ga obrađuje neki od *WM-ovih* pomoćnih modula.
- *Ready*: *WM* ili neki od njegovih pomoćnih modula obrađuje posao, pronađen je odgovarajući *CE*, ali još uvijek nije prebačen u odgovarajući red na *CE* pomoću *Job Controller-a* i *CondorC-a*.
- *Scheduled*: posao čeka u redu na *CE.*

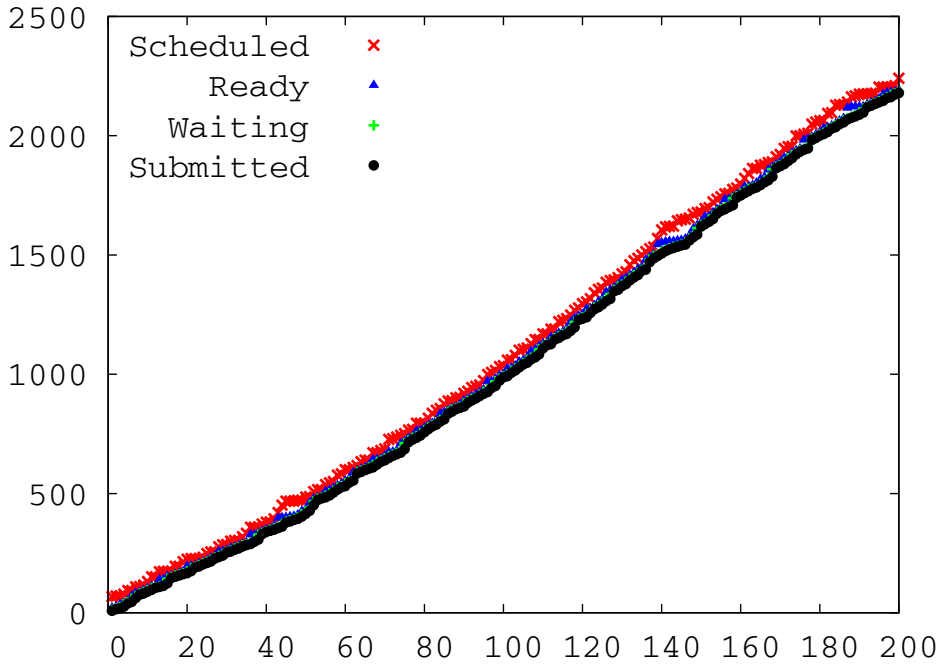
4.2 Rezultati mjerenja

U Prilogu C dat je prikaz grafika svih sprovedenih mjerenja. Krive na i *Sl. 6* predstavljaju tipičnu zavisnost vremena ulaska poslova u stanja *Submitted, Waiting, Ready* i *Scheduled* od broja poslova, u slučajevima sekvencijalnog podnošenja zahtjeva za izvršavanjem poslova posredstvom *Network Server-a* i *WMProxy-ja*, respektivno. Referentni početni trenutak od kog se mjeri vrijeme je trenutak u kom započinje predaja zahtjeva za izvršavanjem poslova. Može se

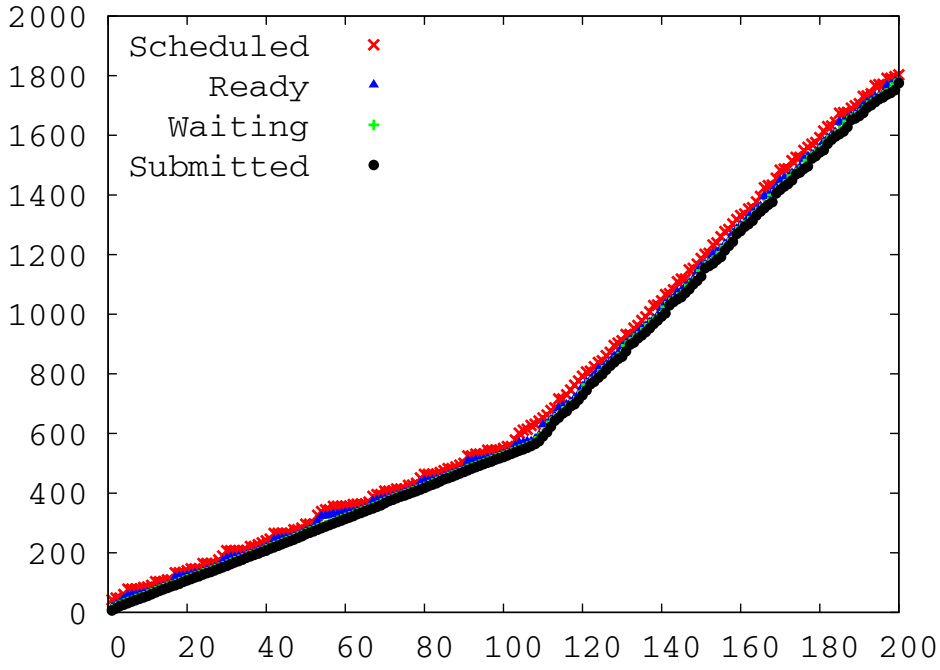


Sl. 5: Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sekvencijalno na Network Server bez Input Sandbox-a.

uočiti da su krive koje odgovaraju trenucima ulaska poslova u stanje *Submitted* približno linearne, pri čemu kriva na *Sl. 6* pokazuje nešto veća odstupanja od očekivane prave linije. To ukazuje na to da se zahtjevi prihvataju u većoj ili manjoj mjeri ravnomjenom brzinom. Takođe nije primjećena pojava zasićenja, što navodi na zaključak da i jedan i drugi servis mogu da prihvate veći broj poslova. Ovo je verifikovano i za veći broj poslova (500). Oblik krive vremena ulaska poslova u stanje *Waiting* prati oblik krive koja odgovara stanju *Submitted*, s tim da je pomjeraj po ordinati manji na *Sl. 6*. Drugim riječima, vrijeme zadržavanja posla u stanju *Submitted* je približno konstantno i manje je u slučaju *WMProxy* servisa. Odgovarajuće krive za stanja *Ready* i *Scheduled* takođe prate oblik krive za stanje *Submitted*, uz nešto izraženija odstupanja.



Sl. 6: Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sekvencijalno na WMProxy sa Input Sandbox-om of 8 kB.



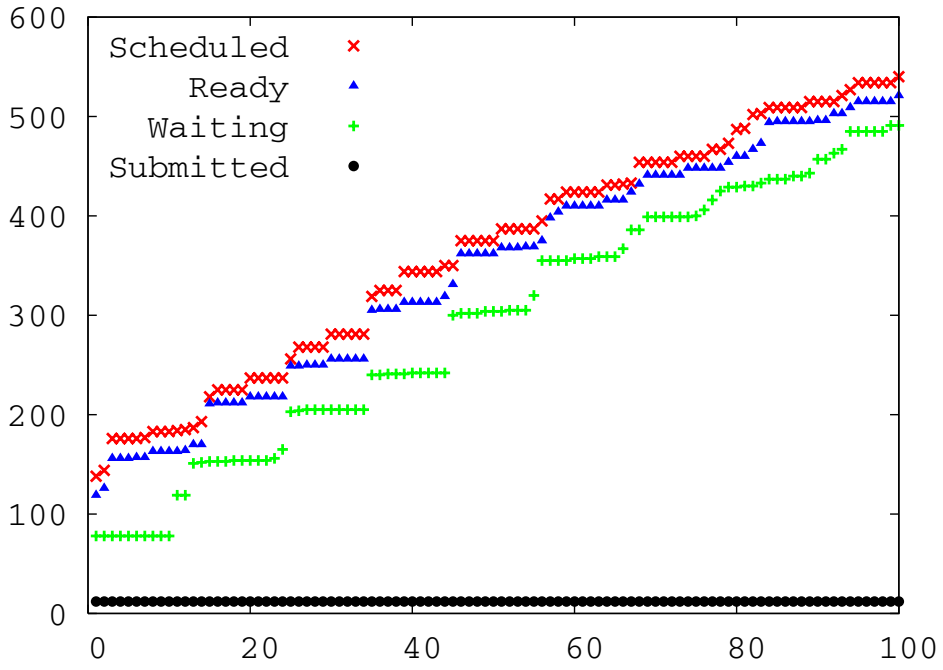
Sl. 7: Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sekvencijalno na WMProxy sa Input Sandbox-om of 5 MB.

Prilikom izvođenja mjerenja uočena je nekonzistentnost (nestabilnost) u ponašanju *WMProxy* servisa, čiji primjer je dat na *Sl. 7*. Iako je prosječno vrijeme blisko očekivanom, krive umjesto jednog, imaju dva približno linearna segmenta. Međusobni odnosi krivih su održani za dati servis

i način slanja zahtjeva.

U Prilogu D tabelarno su prikazana srednja vremena potrebna da posao pređe iz jednog stanja u drugo. Vrijednosti su dobijene posmatranjem razlike pripadajućih vremena odgovarajućih stanja u tri ponovljena eksperimenta za svaki posmatrani slučaj. Upoređivanjem srednjih vrijednosti može se uočiti da se u slučaju podnošenja poslova posredstvom *Network Server-a* posao najduže zadržava u stanju *Submitted* (približno dvije trećine vremena koje je potrebno da posao pređe iz stanja *Submitted* u stanje *Scheduled*), *Waiting* (približno desetinu) i *Ready* (približno petinu). U slučaju *WMProxy-ja* zadržavanje posla u stanju *Submitted* je najkraće (oko desetine vremena potrebnog da posao pređe iz stanja *Submitted* u stanje *Scheduled*), u stanju *Waiting* je manje od trećine tog vremena, a u stanju Ready je najduže (oko dvije trećine). U *Tab. 1* je dat pregled rezultata merenja koji su dobijeni za srednja vremena stanja poslova.

Krive na *Sl. 8* predstavljaju tipičnu zavisnost vremena ulaska poslova u stanja *Submitted, Waiting, Ready* i *Scheduled* od broja poslova, u slučaju paralelnog podnošenja zahtjeva za izvršavanjem poslova (kolekcija) posredstvom *WMProxy-ja*. Zahtjevi za izvršavanje poslova se prihvataju istovremeno (stanje *Submitted*), dok se prelasci u naredna stanja odvijaju u grupama, na šta ukazuje stepenast oblik krivih koje im odgovaraju. Iako su rezultati mjerenja obrađeni na isti način kao za prethodna dva slučaja, detaljno upoređivanje vrijednosti iz tabela ne može dati smislen zaključak, zbog različitih oblika krivih. Ipak, može se reći da se kod ovog načina podnošenja poslova posao najduže zadržava u stanju *Submitted,* što se uostalom može uočiti pogledom na krive na *Sl. 8*.



Sl. 8: Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova paralelno (kao kolekcija) na WMProxy sa Input Sandbox-om of 5 MB.

Tabela *Tab. 1* sadrži prikaz srednjeg vremena koje potrebno da posao uđe u određeno stanje (*Submitted, Waiting, Ready, Scheduled*), za svaki posmatrani slučaj. Pri izračunavanju su praćeni slijedeći parametri: trenutak kada je započet proces predaje poslova, trenutak kada je posljednji posao ušao u određeno stanje, broj poslova. Svaka od vrijednosti u tabeli je dobijena

izračunavanjem srednje vrijednosti očitanih vremena za odgovarajuća stanja u po tri ponovljena mjerenja za svaki posmatrani slučaj. Ukupno trajanje jednog mjerenja se kretalo od desetak minuta u slučaju 100 poslova bez *Input Sandbox-a*, pa do preko trideset minuta za 200 poslova sa većim *Input Sandbox-om.*

broj poslova	**100**				**200**			
status	**Sub**	**Wait**	**Ready**	**Sched**	**Sub**	**Wait**	**Ready**	**Sched**
NSs 0 kB [s/pos]	5.6	6.4	6.6	6.9	5.5	5.9	6.0	6.2
NSs 8 kB [s/pos]	6.5	7.4	7.6	7.8	6.1	6.6	6.6	6.7
NSs 5 MB [s/pos]	7.5	8.4	8.6	8.8	7.7	8.2	8.2	8.3
WMPs 0 kB [s/pos]	10.3	10.3	10.5	10.8	9.6	9.7	9.7	9.9
WMPs 8 kB [s/pos]	10.5	10.6	10.8	11.0	10.1	10.2	10.3	10.6
WMPs 5 MB [s/pos]	8.2	8.3	8.4	8.6	9.4	9.4	9.5	9.6
WMPk 0 kB [s/pos]	0.09	4.3	4.6	4.8	0.06	4.3	4.5	4.6
WMPk 8 kB [s/pos]	0.11	4.4	4.7	4.8	0.07	4.6	4.8	4.9
WMPk 5 MB [s/pos]	0.12	5.0	5.4	5.6	0.08	4.9	5.1	5.2

Tab. 1: Prikaz srednjih vremena potrebnih da posao uđe u stanje Submitted, Waiting, Ready i Scheduled. Zahtjevi su podnošeni serijski posredstvom Network Server-a (NSs) i WMProxy-ja (WMPs) ili paralelno posredstvom WMProxy-ja (WMPk), uz veličinu Input Sandbox-a od 0kB, 8kB, 5MB.

Na osnovu dobijenih grafika i rezultata sumiranih u *Tab. 1* može se izvesti nekoliko zaključaka o performansama *Workload Management System-a*. U slučaju sekvencijalnog podnošenja zahtjeva za izvršavanjem poslova, *Network Server* pokazuje bolje rezultate od *WMProxy-ja*, uz brzinu prihvatanja poslova veću 1.1 do 1.8 puta. Ovo je u suprotnosti sa rezultatima prezentovanim u radu [28], gdje je *WMProxy* 1.5 do 2 puta brže prihvatao poslove od *Network Server-a.* Tom prilikom nije uočena nestabilnost u radu *WMProxy* servisa koja se javlja u toku novih mjerenja vezanih, a čiji se jedan primjer može vidjeti na *Sl. 7*. Jedan od uzroka ovakvog ponašanja bi mogla biti promjena *gLite middleware-a*, sa *update-a* 4 na *update* 13, što je potrebno dodatno istražiti. Drugi uzrok bi mogla biti hardverska konfiguracija mašine na kojoj je instaliran *WMS*, jer je primjećeno da je za stabilan rad ovog servisa potrebno obezbjediti bar 2 GB RAM memorije.

Izvedena mjerenja pokazuju da paralelno slanje zahtjeva za izvršavanjem poslova najefikasnije (kolekcije nezavisnih poslova), što je saglasno sa rezultatima u radu [28].

Iz *Tab. 1:* vidimo da sa povećanjem ukupne veličine fajlova koje je potrebno prenijeti na *WMS* prilikom podnošenja zahtjeva na *Network Server* za izvršavanjem poslova povećava prosječno vrijeme prihvatanje zahtjeva. Ovo povećanje se kreće između 1.1 i 1.2 puta. Za *WMProxy* situacija je obratna, pa je vreme prihvatanja poslova sa većim *Input Sanbox-om* manje i do 1.1 puta. Ovo ukazuje da kod *WMProxy-ja* vreme za prenos fajlova čini zanemarljivi dio ukupnog vremena prihvatanja posla, i da problemi koji su identifikovani kod ovog servisa nakon najnovijih *update-a* usporavaju njegov rad mnogo više nego što bi bilo očekivano povećanje vremena prihvatanja poslova usled većeg *Input Sandox-a.* Sa druge strane, *WMProxy* servis za kolekcije radi daleko bolje, uz očekivano povećanje vremena prihvatanja po poslu između 1.1 i 1.2 puta.

Iz *Tab. 1:* vidimo i da se povećanjem broja poslova sa 100 na 200 ne povećavaju srednja vrijemene prihvatanja zahtjeva za posao, što znači da nijedan od servisa nije pokazao zasićenje performansi usled velikog broja poslova.

5 Zaključak

Cilj ovog rada je bio da se razvije i implementira adekvatna metodologija mjerenja koja omogućuje proučavanje performansi jednog od *Grid* servisa, *Workload Management System-a*. S korisničke tačke gledišta bitni parametri koji određuju kvalitet usluge ovog servisa su vremena koja su potrebna da se zahtjev prihvati od strane *WMS-a*, obradi i proslijedi izabranom *Computing Elementu*. Posmatrano je kako promjene načina podnošenja zahtjeva za izvršavanjem poslova, broja poslova, veličine fajlova koje je potrebno prenijeti na *WMS* tokom podnošenja zahtjeva utiču na ta vremena.

U prvom delu rada dat je uvod u koncepte *Grida* (poglavlje 1) i *Grid middleware-a* (poglavlje 2). Kroz presjek arhitekture *middleware-a* prikazana je uloga pojedinih njegovih komponenti u zadovoljenju korisničkih potreba, a potom je detaljnije u poglavlju 3 predstavljen *WMS* servis na koji je usmjerena pažnja ovog rada. Poseban osvrt je dat na uporedne karakteristike *Network Servis-a* i *WMProxy-ja*, komponenti *WMS-a* sa istom ulogom (prihvatanje zahtjeva korisnika), ali sa različitom koncepcijom. Navedena su dva skupa komandi za pristup *WMS-u* u zavisnosti od toga koji se od ova dva servisa koristi. U prilogu A su dati primjeri primjene komandi.

Centralni dio rada je izložen u poglavlju 4 gdje su prikazani metodologija i rezultati mjerenja performansi *Workload Management System-a*. Dobijeni rezultati ukazuju na to da:

- *Network Server* daje bolje rezultate od *WMProxy-ja* kada je u pitanju sekvencijalno podnošenje zahtjeva.
- Paralelno podnošenje zahtjeva (putem *WMProxy-ja*) je efikasnije od sekvencijalnog, bilo putem *NS-a* ili putem *WMProxy-ja.*
- *WMS* je u stanju da prihvati veliki broj poslova (ne dolazi do zasićenja).
- Povećanje broja poslova ne utiče značajno na prosječno vrijeme prihvatanja zahtjeva za izvršavanja posla, bez obzira na način podnošenja zahtjeva.
- Povećanje *Input Sandbox-a* se odražava na prosječno vrijeme prihvatanja zahtjeva za izvršavanja posla, bez obzira na način podnošenja zahtjeva.
- U slučaju sekvencijalnog podnošenja zahtjeva za izvršavanjem poslova putem *WMProxy-ja WMS-u* je potrebno manje vremena da obradi zahtjev i proslijedi posao odgovarajućem *Computing Element-u* nego kada je u pitanju sekvencijalno podnošenje zahtjeva za izvršavanjem poslova putem *NS-a*.
- Iako se se kolekcija poslova najbrže prihvata, potrebno je najviše vremena da se svi poslovi nađu na odgovarajućim *CE.*
- *WMProxy* pokazuje nestabilnosti u radu.

Kompletni rezultati mogu se naći u prilozima C (grafici) i D (tabele), dok prilog B sadrži korištene skript fajlove za podnošenje zahtjeva i obradu rezutata mjerenja.

Prilog A: Primjeri primjene komandi

U okviru ovog poglavlja dati su primjeri sekvenci osnovnih koraka koje je potrebno izvršiti da bi se predao zahtjev za izvršavanje posla i poslije toga pratio tok izvršavanja posla korišćenjem usluga *Network Server-a* [20] ili *WMProxy* servisa [24][25].

U oba slučaja, prije korišćenja bilo koje od *WMS-UI* komandi neophodno je imati validan *proxy* sertifikat na *WMS-UI* mašini. On se kreira korišćenjem *voms-proxy-init* komande ili alternativno, komande *grid-proxy-init.* Neophodno je unijeti odgovarajuću lozinku. Ako nije drugačije naglašeno, upotrebom mogućih opcija komandi, životni vijek kreiranog sertifikata je dvanaest sati.

```
[neda@ce neda]$ voms-proxy-init -voms aegis

Your identity: /DC=ORG/DC=SEE-GRID/O=People/O=Institute of Physics Belgrade/CN=Neda Svraka
Enter GRID pass phrase:
Creating temporary proxy ..................................... Done
Contacting se.phy.bg.ac.yu:15001 [/DC=ORG/DC=SEE-GRID/O=Hosts/O=Institute of Physics
Belgrade/CN=host/se.phy.bg.ac.yu] "aegis" Done
Creating proxy ..................................... Done
Your proxy is valid until Thu Dec 21 04:48:37 2006
```

Informacija o postojanju valjanog *proxy* sertfikata se dobija upotrebom komande *voms-proxy-info*.

```
[neda@ce neda]$ voms-proxy-info

subject:/DC=ORG/DC=SEE-GRID/O=People/O=Institute of Physics Belgrade/CN=Neda
Svraka/CN=proxy
issuer    : /DC=ORG/DC=SEE-GRID/O=People/O=Institute of Physics Belgrade/CN=Neda Svraka
identity : /DC=ORG/DC=SEE-GRID/O=People/O=Institute of Physics Belgrade/CN=Neda Svraka
type      : proxy
strength : 512 bits
path      : /tmp/x509up_u40001
timeleft  : 11:57:04
```

Sada se može pristupiti upotrebi *WMS-UI* komandi.

Ključni element procesa pronalaženja resursa koji odgovaraju zahtjevima posla je *JDL* fajl [21][22] koji opisuje potrebne ulaze, generisane izlaze i zahtjeve koje treba da ispune resursi da bi se posao izvršio. U ovim primjerima je korišten jednostavan *JDL* fajl, *hello.jdl*. Potrebno je izvršiti komandu *hostname,* koja vraća naziv računara na kom se izvršava, na radnom čvoru i rezultat vratiti u generisanom izlaznom fajlu *message.txt*. Eventualne greške se ispisuju u generisanom fajlu *stderror*.

```
[
  Type = "Job";
  Executable = "/bin/hostname";
  Arguments = "";
  StdOutput = "message.txt";
  StdError = "stderror";
  OutputSandbox = {"message.txt","stderror"};
```

]

A.1 Predaja i kontrola posla korišćenjem *Network Server-a*

Prije nego što se preda zahtjev za izvršavanjem posla korisno je pogledati listu resursa koji odgovaraju zahtjevima posla. Ona se dobija kao rezultat izvršavanja *glite-job-list-match* komande, čiji je ulazni argument *JDL* fajl posla. Resursi su rangirani, a prvi je onaj sa najvišim rangom.

```
[neda@ce neda]$ glite-job-list-match hello.jdl

Selected Virtual Organisation name (from proxy certificate extension): aegis
Connecting to host g01.phy.bg.ac.yu, port 7772

**********************************************************************
                COMPUTING ELEMENT IDs LIST
 The following CE(s) matching your job requirements have been found:

            *CEId*
 ce.phy.bg.ac.yu:2119/jobmanager-pbs-aegis
 cluster1.csk.kg.ac.yu:2119/jobmanager-pbs-aegis
 g02.phy.bg.ac.yu:2119/blah-pbs-aegis
 grid01.elfak.ni.ac.yu:2119/blah-pbs-aegis
 grid01.rcub.bg.ac.yu:2119/jobmanager-pbs-aegis
 gw01.rogrid.pub.ro:2119/jobmanager-lcgpbs-aegis
 rti29.etf.bg.ac.yu:2119/jobmanager-pbs-aegis
 seegrid2.fie.upt.al:2119/jobmanager-pbs-aegis
 grid01.elfak.ni.ac.yu:2119/jobmanager-pbs-aegis
**********************************************************************
```

Nakon toga, korisnik može da preda zahtjev za izvršavanjem posla, ako je zadovoljan izborom *Computing Element-a,*što čini naredbom *glite-job-submit.*

```
[neda@ce neda]$ glite-job-submit -o jid hello.jdl

Selected Virtual Organisation name (from proxy certificate extension): aegis
Connecting to host g01.phy.bg.ac.yu, port 7772
Logging to host g01.phy.bg.ac.yu, port 9002

================================glite-job-submit Success=========================
 The job has been successfully submitted to the Network Server.
 Use glite-job-status command to check job current status. Your job identifier is:

 - https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A

 The job identifier has been saved in the following file:
 /home/neda/jid
```

Ukoliko je zahtjev uspješno predat komanda vraća jedinstveni identifikator posla,

https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A

koji se poslije koristi pri praćenju izvršavanja posla i preuzimanju rezultata posla. Njega svakom poslu dodjeljuje *WMS* jedinstvenog i jasnog određenja posla u *Grid* sistemu. Čine ga dio koji predstavlja *URL LB* servera koji bilježi informacije o poslu (https://g01.phy.bg.ac.yu:9000) i dio

koji ga jedinstveno određuje,a generiše ga *WMS-UI* (TOPdK-V6gh28DP8gHPpR5A).

Ovdje je osnovna komanda dopunjena opcijom *-o* koja omogućuje pamćenje jedinstvenog identifikatora posla u željenom fajlu. U ovom slučaju je to *jid* fajl koji se nalazi direktorijumu */home/neda*.

Status posla se može provjeravati slanjem upita *LB* servisu, korišćenjem naredbe *glite-job-status,*čiji je ulazni argument identifikator posla. U ovom slučaju vrijednost identifikatora posla se uzima iz fajla u kom je zapamćen, upotrebom opcije *-i*, a fajl je *jid* u direktorijumu */home/neda.* Na ovaj način je moguće jednostavno pratiti status više poslova biranjem jedne od ponuđenih opcija.

```
[neda@ce neda]$ glite-job-status -i jid

---------------------------------------------------------------
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit
---------------------------------------------------------------

Choose one or more jobId(s) in the list – [1-2]all:1

*************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
Current Status:     Running
Status Reason:      unavailable
Destination:        rti29.etf.bg.ac.yu:2119/jobmanager-pbs-aegis
Submitted:          Wed Dec 20 16:58:13 2006 CET
*************************************************************
```

Stanja koja označavaju regularno izvršavanje posla su: *Submitted, Waiting, Ready, Scheduled, Running* i *Done*. Pojava neregularnosti u toku izvršavanja obično se završava stanjem *Aborted*.

Kada korisnik ustanovi da je posao uspješno završen, tj. dobio je status *Done* i izlazni fajlovi su prebačeni na *WMS* čvor, može preuzeti te fajlove komandom *glite-job-output* i prebaciti ih na mašinu na kojoj je pokrenut korisnički interfejs *(UI).*

```
[neda@ce neda]$ glite-job-output -i jid

---------------------------------------------------------------
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit
---------------------------------------------------------------

Choose one or more jobId(s) in the list – [1-2]all:1

Retrieving files from host: g01.phy.bg.ac.yu ( for https://g01.phy.bg.ac.yu:9000/TOPdK-
V6gh28DP8gHPpR5A )

*************************************************************
```

JOB GET OUTPUT OUTCOME
Output sandbox files for the job:
1. https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
have been successfully retrieved and stored in the directory:
/tmp/neda_TOPdK-V6gh28DP8gHPpR5A
****

Fajlovi su smješteni u direktorijumu *neda_TOPdK-V6gh28DP8gHPpR5A* čije je ime spoj korisničkog imena u datom operativnom sistemu i dijela identifikatora posla koji ga jedinstveno određuje. Direktorijum je smješten u */tmp*, jer je tako definisano u konfiguracijskom fajlu. Upotreba opcije *–dir* omogućava korisniku da smjesti rezultate na neku drugu lokaciju.

Ukoliko se primjete neke nepravilnosti u izvršavanju posla, kao što je stanje *Aborted* ili posao nikada ne uđe u stanje *Done*, moguće provjeriti događaje vezane za posao upotrebom komande *glite-job-logging-info*, čiji je ulazni argument jedinstveni identifikator posla.

[neda@ce neda]$ glite-job-logging-info https://g01.phy.bg.ac.yu:9000/TOPdK-6gh28DP8gHPpR5A

ili

[neda@ce neda]$ glite-job-logging-info -i jid

---
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit

Choose one or more jobId(s) in the list – [1-2]all:1

Takođe, bitno je pomenuti naredbu kojom korisnik otkazuje započeti posao *glite-job-cancel.*

[neda@ce neda]$ glite-job-cancel https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A

ili

[neda@ce neda]$ glite-job-cancel -i jid

---
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit

Choose one or more jobId(s) in the list – [1-2]all:1

A.2 Predaja i kontrola posla korišćenjem *WMProxy-ja*

Sekvenca izvršavanja operacija u cilju podnošenja zahtjeva za izvršavanjem posla i praćenje

njegovog izvršavanja je slična onoj koja se koristi kada je posrednik *Network Server*. Pored postojećih uvedene su nove operacije i opcije koje treba da podrže nove mogućnosti koje pruža *WMProxy* servis. Stoga će u ovom poglavlju naglasak biti na novinama, a neće se dodatno objašnjavati opcije viđene u prethodnom poglavlju.

Svakom poslu koji se predaje *WMProxy-u* dodjeljuje ovlašćenja korisnika koji ga predaje, a koja se koriste kada operacije treba da zadovolje siguronosni protokol. Postoje dva načina da se obavi delegacija *proxy-a,* traženjem automatske delegacije prilikom opracije predaje zahtjeva za izvršavanje ili eksplicitnom delegacijom na koju će se oslanjati operacije predaje zahtjeva za izvršavanje posla. Preporučuje se druga varijanta, naročito za višestruke predaje zahtjeva, jer proces delegacije *proxy-a* zahtjeva vrijeme.

```
[neda@ce neda]$ glite-wms-job-delegate-proxy -d dID

Connecting to the service https://wms.phy.bg.ac.yu:7443/glite_wms_wmproxy_server


================== glite-wms-job-delegate-proxy Success ==================

Your proxy has been successfully delegated to the WMProxy:
https://wms.phy.bg.ac.yu:7443/glite_wms_wmproxy_server

with the delegation identifier: dID
```

Identifikator delegacije se zatím koristi prilikom pregleda liste odgovarajućih resursa i predaje zahtjeva za izvršavanje posla, upotrebom komandi *glite-wms-job-list-match* i *glite-wms-job-submit* i opcije *-d* uz koju je navedena vrijednost identifikatora delegacije.

```
[neda@ce neda]$ glite-wms-job-list-match -d dID hello.jdl

Connecting to the service https://wms.phy.bg.ac.yu:7443/glite_wms_wmproxy_server

================================================================================

                COMPUTING ELEMENT IDs LIST
 The following CE(s) matching your job requirements have been found:

       *CEId*
 - ce.phy.bg.ac.yu:2119/jobmanager-pbs-aegis
 - cluster1.csk.kg.ac.yu:2119/jobmanager-pbs-aegis
 - g02.phy.bg.ac.yu:2119/blah-pbs-aegis
 - grid01.elfak.ni.ac.yu:2119/blah-pbs-aegis
 - grid01.rcub.bg.ac.yu:2119/jobmanager-pbs-aegis
 - gw01.rogrid.pub.ro:2119/jobmanager-lcgpbs-aegis
 - rti29.etf.bg.ac.yu:2119/jobmanager-pbs-aegis
 - seegrid2.fie.upt.al:2119/jobmanager-pbs-aegis
 - grid01.elfak.ni.ac.yu:2119/jobmanager-pbs-aegis
```

```
[neda@ce neda]$ glite-wms-job-submit -d dID -o jid hello.jdl

Connecting to the service https://wms.phy.bg.ac.yu:7443/glite_wms_wmproxy_server
```

```
====================== glite-wms-job-submit Success ======================

The job has been successfully submitted to the WMProxy
Your job identifier is:

https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q

The job identifier has been saved in the following file:
/home/neda/jid
```

Status posla se provjerava upotrebom naredbe *glite-wms-job-status*, a važi sve što je navedeno u prethodnom poglavlju za naredbu *glite-job-status*.

```
[neda@ce neda]$ glite-wms-job-status -i jid

---------------------------------------------------------------
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit
---------------------------------------------------------------

Choose one or more jobId(s) in the list - [1-2]all:2

*************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
Current Status:     Done (Success)
Exit code:          0
Status Reason:      Job terminated successfully
Destination:        rti29.etf.bg.ac.yu:2119/jobmanager-pbs-aegis
Submitted:          Wed Dec 20 17:35:47 2006 CET
*************************************************************
```

Nakon što se posao završi i dobije status *Done* korisnik može da preuzme rezultate posla upotrebom komande *glite-wms-job-output.* Ovdje je iskorištena opcija *–dir* koja omogućava korisniku da sam odredi lokaciju za smještanje preuzetih izlaznih fajlova.

```
[neda@ce neda]$ glite-wms-job-output --dir /home/neda/izlaz -i jid
---------------------------------------------------------------
1 : https://g01.phy.bg.ac.yu:9000/TOPdK-V6gh28DP8gHPpR5A
2 : https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
a : all
q : quit
---------------------------------------------------------------

Choose one or more jobId(s) in the list - [1-2]all (use , as separator or - for a range): 2

Connecting to the service https://147.91.84.25:7443/glite_wms_wmproxy_server

================================================================================

                JOB GET OUTPUT OUTCOME
```

```
Output sandbox files for the job:
https://g01.phy.bg.ac.yu:9000/w-sbZQIvNl1tQLVVwgqW_Q
have been successfully retrieved and stored in the directory:
/home/neda/izlaz
```

Rezultat izlistavanje sadržaja direktorijuma */home/neda/izlaz* su preuzeti izlazni fajlovi:

```
[neda@ce neda]$ ll izlaz
total 4
-rw-rw-r--   1 neda    neda         19 Dec 20 18:04 message.txt
-rw-rw-r--   1 neda    neda          0 Dec 20 18:04 stderror
```

Za provjeru događaja vezanih za izvršavanje posla koristi se naredba *glite-wms-job-logging-info*, čiji je ulazni argument jedinstveni identifikator posla, a za otkaz posla koji se izvršava *glite-wms-job-cancel,* sa istim ulaznim argumentom.

Treba napomenuti da se predaja zahtjeva za izvršavanje složenog posla vrši na isti način kao i kod jednostavnog posla, ako je složeni posao opisan jedinstvenim *JDL* fajlom, što je obično i slučaj. Izuzetak se može napraviti kod kolekcije nezavisnih poslova kad se može iskoristiti opcija *–collection* komande *glite-wms-job-submit* uz koju je navedena putanja ka direktorijumu u kom se nalaze *JDL* fajlovi pojedinačnih poslova.

Postupak kontrole statusa posla je isti kao kod jednostavnog posla, pri čemu je moguće kontrolisati cjelokupan posao ili pojedinčne potposlove koristeći kao ulazni argument jedinstvene identifikatore posla u globalu ili pojedinačnih potposlova.

Moguće je preuzeti rezultate složenih poslova upotrebom *glite-wms-job-output* komande i globalnog jedinstvenog identifikatora posla, pri čemu će rezultati potposlova biti razvrstani u posebne poddirektorijume.

Prilog B: Korišteni skript fajlovi

Operativni sistem *Linux* sadrži veliku kolekciju standardnih programa. Jedan od njih je *shell*. *Shell* je program koji pruža konzistentno i lako za upotrebu okruženje za izvršavanje programa u *Linux-u* i odgovoran je za proslijeđivanje korisničkih komandi *kernel-u* na izvršavanje. Njegova zgodna osobina je da omogućuje pisanje skript fajlova. Pojam skript *(script)* fajlova označava fajlove koji sadrže niz komandi koje korisnik želi da pokrene. Pošto se svaki skript nalazi u fajlu, koji ima svoje ime, moguće je kombinovati postojeće programe da bi se stvorio novi zadužen za riješavanje konkretnog problema.

U okviru mjerenja performansi *WMS-a* kreirani su skript fajlovi koji omogućuju sekvencijalno ili paralelno slanje velikog broja zahtjeva za izvršavanje poslova, kao i oni koji prikupljaju informacije o statusu posla. Oni uključuju naredbe koje se koriste za komunikaciju sa *WMS-om* u linijskom korisničkom okruženju. Osim toga, korišteni su pomoćni skriptovi za dodatnu obradu podataka.

Skript fajl za sekvencijalnu predaju niza zahtjeva za izavršavanje poslova preko *Network Server-a*:

```
#!/bin/bash

# Sekvencijalno submit-ovanje niza poslova. Koristi se Network Server.
# Upotreba komande: proslijedjuje se radni direktorijum i broj poslova koji
# se submit-uju.

if [[ $# == 0 || $# > 3 ]]
then
    echo "Usage: $0 <working directory> <number of jobs> <configuration file>"
    exit
fi

# radni direktorijum
WORK=$1

# broj poslova
N=$2

# konfiguracijski fajl
CF=$3

# jdl file se nalazi u radnom direktorijumu
JDL=`ls -l $WORK | grep jdl | awk '{print $8}'`

# ako se specificira konfiguracijski fajl
# odgovarajuci CE-ovi
 glite-job-list-match --config-vo $CF $WORK/$JDL > $WORK/match.txt

 for ((i=0;i<N;i++))
   do
    echo "Submitting job $(($i+1))" >>$WORK/submit.txt
    date +%s >> $WORK/submit.txt
    glite-job-submit --config-vo $CF -o $WORK/jid $WORK/$JDL >> $WORK/submit.txt 2>&1
    date +%s >> $WORK/submit.txt
   done
```

Skript fajl za sekvencijalnu predaju niza zahtjeva za izavršavanje poslova preko *WMProxy-ja*:

```
#!/bin/bash

# Sekvencijalno submit-ovanje niza poslova. Koristi se WMProxy.
# Upotreba komande:proslijedjuju se identifikator delegacije, radni
# direktorijum i broj poslova koji se submit-uje.

if [[ $# == 0 || $# > 4 ]]
then
    echo "Usage: $0 <delegation Id> <working directory> <number of jobs> <configuration file>"
    exit
fi

# delegation Id
DID=$1

# radni direktorijum
WORK=$2

# broj poslova
N=$3

# konfiguracijski fajl
CF=$4

# jdl file se nalazi u radnom direktorijumu
JDL=`ls -l $WORK | grep jdl | awk '{print $8}'`

# ako se specificira konfiguracijski fajl

# odgovarajuci CE-ovi
 glite-wms-job-list-match -d $DID -c $CF --rank $WORK/$JDL > $WORK/match.txt

 for ((i=0;i<N;i++))
    do
      echo "Submitting job $(($i+1))" >> $WORK/submit.txt
      date +%s >> $WORK/submit.txt
      glite-wms-job-submit -d $DID -c $CF -o $WORK/jid $WORK/$JDL >> $WORK/submit.txt 2>&1
      date +%s >> $WORK/submit.txt
    done
```

Skript fajl za paralelnu predaju niza zahtjeva za izavršavanje poslova preko *WMProxy-ja*:

```
#!/bin/bash

# Submit-ovanje kolekcija. Koristi se WMProxy.
# Upotreba komande:proslijedjuju se identifikator delegacije, radni
# direktorijum, direktorijum u kom se nalaze jdl fajlovi,konfiguracijski
# fajl

if [[ $# == 0 || $# > 4 ]]
then
    echo "Usage: $0 <delegation Id> <working directory> <collection directory><configuration file>"
    exit
```

```
fi

# delegation Id
DID=$1

# radni direktorijum
WORK=$2

# direktorijum u kom se nalaze jdl fajlovi
COLL=$3

# konfiguracijski fajl
CF=$4

echo "Submitting collection" >> $WORK/submit.txt
date +%s >> $WORK/submit.txt
glite-wms-job-submit -d $DID -c $CF -o $WORK/jid --collection $COLL >> $WORK/submit.txt 2>&1
date +%s >> $WORK/submit.txt
```

Skript fajl za prikupljanje informacija o statusu poslova preko *Network Server-a*:

```
#!/bin/bash

# upotreba komande

if [[ $# == 0 || $# > 2 ]]
then
  echo "Usage : $0 <working directory> <job id file>"
  exit
fi

# radni direktorijum
WORK=$1

# fajl koji sadrzi job id-e
FILE=$2

grep -v Submitted $WORK/$FILE > $WORK/jid-test

FILET=$WORK/jid-test

# ispitivanje statusa poslova

for i in $(cat $FILET)
  do
    j=${i##*\/}
    glite-job-status -v 3 "$i" > $WORK/status_$j
   egrep 'Current Status | Submitted | Waiting | Ready | Scheduled |Running | Done | Cleared | Aborted |
         Cancelled | Unknown' $WORK/status_"$j" > $WORK/time-status_"$j"
#    rm -f $WORK/status_"$j"
  done
```

Skript fajl za prikupljanje informacija o statusu poslova preko *WMProxy-ja*:

```
#!/bin/bash

# upotreba komande
```

```
if [[ $# == 0 || $# > 2 ]]
then
  echo "Usage : $0 <working directory> <job id file>"
  exit
fi

# radni direktorijum
WORK=$1

# fajl koji sadrzi job id-e
FILE=$2

grep -v Submitted $WORK/$FILE > $WORK/jid-test

FILET=$WORK/jid-test

# ispitivanje statusa poslova

for i in $(cat $FILET)
  do
    j=${i##*\/}
    glite-wms-job-status -v 3 "$i" > $WORK/status_$j
    egrep 'Current Status | Submitted | Waiting | Ready | Scheduled |Running | Done | Cleared | Aborted |
         Cancelled | Unknown' $WORK/status_"$j" > $WORK/time-status_"$j"
#    rm -f $WORK/status_"$j"
  done
```

Da bi se dobile informacije o statusu kolekcije poslova, dovoljna je jedna komandna linija oblika:

```
glite-wms-job-status -v 3 -i hello_coll0_100/jid > hello_coll0_100/status.txt
```

Slijede skript fajlovi koji daju informaciju o trenutku ulaska posla u određeno stanje:
Submitted

```
#!/bin/bash

# Izvlaci vrijeme submit-ovanja iz time-status* fajlova.
# upotreba komande
if [[ $# == 0 || $# > 1 ]]
then
  echo "Usage : $0 <working directory>"
  exit
fi

# Radni direktorijum, koji se proslijedjuje kao ulazni argument skripte.
WORK=$1

# Vrijeme zapocinjanja submit-ovanja niza poslova.
START=`head -n 2 $WORK/submit.txt | grep -v Submitting`

# Izvlaci vrijeme submit-ovanja iz time-status* fajlova u obliku: mjesec dan hh:mm:ss
# i smjesta u fajl sub_time u radnom direktorijumu.
grep 'Submitted' $WORK/time-status* | grep -v 'Current Status' | awk '{print $5, $6, $7}' | sort >
$WORK/sub_time
```

```
# Formira fajl sub_sec koji sadrzi vremena submit-ovanja izrazena u sec od 01.01.1970.
date -f $WORK/sub_time +%s >> $WORK/sub_sec

# Izracunava razliku vremena submit-ovanja odredjenog posla i trenutka kad je
# zapoceo proces submit-ovanja niza poslova.
for i in $(cat $WORK/sub_sec)
  do
   echo $(($i-$START)) >> $WORK/diff_time
  done

# Ubacuje redne brojeve poslova u prvu kolonu fajla job_time.
cat -n $WORK/diff_time > $WORK/job_time.dat
```

Waiting

```
#!/bin/bash

# Izvlaci vrijeme Waiting iz time-status* fajlova.
# upotreba komande
if [[ $# == 0 || $# > 1 ]]
then
  echo "Usage : $0 <working directory>"
  exit
fi

# Radni direktorijum, koji se proslijedjuje kao ulazni argument skripte.
WORK=$1

# Vrijeme zapocinjanja submit-ovanja niza poslova.
START=`head -n 2 $WORK/submit.txt | grep -v Submitting`

# Izvlaci vrijeme Waiting iz time-status* fajlova u obliku: mjesec dan hh:mm:ss
# i smjesta u fajl wait_time u radnom direktorijumu.
grep 'Waiting' $WORK/time-status* | grep -v 'Current Status'| grep -v '\-\-\-' | awk '{print $5, $6, $7}' |
sort > $WORK/wait_time

# Formira fajl wait_sec koji sadrzi vremena submit-ovanja izrazena u sec od 01.01.1970.
date -f $WORK/wait_time +%s >> $WORK/wait_sec

# Izracunava razliku vremena Waiting odredjenog posla i trenutka kad je
# zapoceo proces submit-ovanja niza poslova.
for i in $(cat $WORK/wait_sec)
  do
   echo $(($i-$START)) >> $WORK/wdiff_time
  done
```

Ready

```
#!/bin/bash

# Izvlaci vrijeme Ready iz time-status* fajlova.
# upotreba komande
if [[ $# == 0 || $# > 1 ]]
then
  echo "Usage : $0 <working directory>"
  exit
fi
```

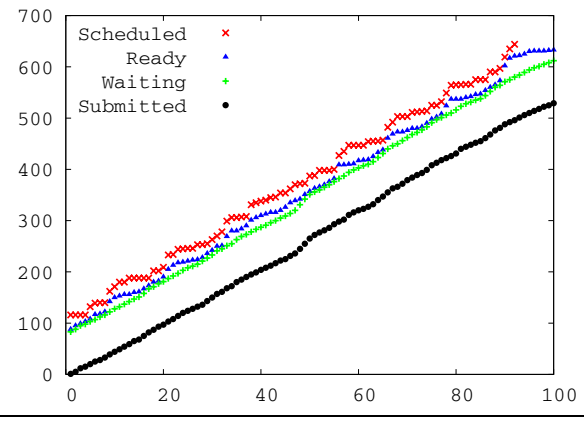
```
# Radni direktorijum, koji se proslijedjuje kao ulazni argument skripte.
WORK=$1

# Vrijeme zapocinjanja submit-ovanja niza poslova.
START=`head -n 2 $WORK/submit.txt | grep -v Submitting`

# Izvlaci vrijeme Ready iz time-status* fajlova u obliku: mjesec dan  hh:mm:ss
# i smjesta u fajl wait_time u radnom direktorijumu.
grep 'Ready' $WORK/time-status* | grep -v 'Current Status'| grep -v '\-\-\-' | awk '{print $5, $6, $7}' | sort
> $WORK/ready_time

# Formira fajl wait_sec koji sadrzi vremena submit-ovanja izrazena u sec od 01.01.1970.
date -f $WORK/ready_time +%s >> $WORK/ready_sec

# Izracunava razliku vremena Waiting odredjenog posla i trenutka kad je
# zapoceo proces submit-ovanja niza poslova.
for i in $(cat $WORK/ready_sec)
  do
   echo $(($i-$START)) >> $WORK/rdiff_time
  done
```

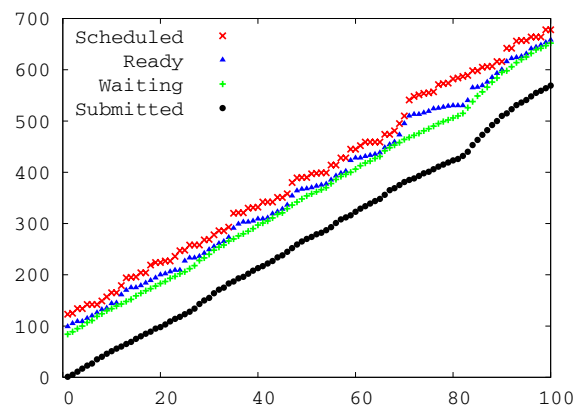
Scheduled

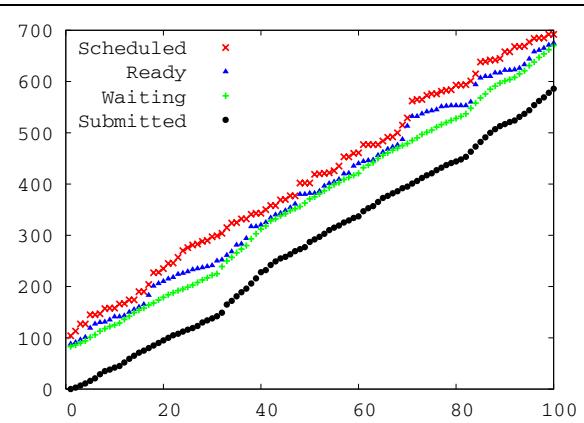
```
#!/bin/bash

# Izvlaci vrijeme Scheduled iz time-status* fajlova.
# upotreba komande
if [[ $# == 0 || $# > 1 ]]
then
  echo "Usage : $0 <working directory>"
  exit
fi

# Radni direktorijum, koji se proslijedjuje kao ulazni argument skripte.
WORK=$1

# Vrijeme zapocinjanja submit-ovanja niza poslova.
START=`head -n 2 $WORK/submit.txt | grep -v Submitting`

# Izvlaci vrijeme Scheduled iz time-status* fajlova u obliku: mjesec dan hh:mm:ss
# i smjesta u fajl sched_time u radnom direktorijumu.
grep 'Scheduled' $WORK/time-status* | grep -v 'Current Status'| grep -v '\-\-\-' | awk '{print $5, $6, $7}' |
sort > $WORK/sched_time

# Formira fajl sched_sec koji sadrzi vremena submit-ovanja izrazena u sec od 01.01.1970.
date -f $WORK/sched_time +%s >> $WORK/sched_sec

# Izracunava razliku vremena Scheduled odredjenog posla i trenutka kad je
# zapoceo proces submit-ovanja niza poslova.
for i in $(cat $WORK/sched_sec)
  do
   echo $(($i-$START)) >> $WORK/sdiff_time
  done
```

Prilog C: Grafički prikaz rezultata mjerenja

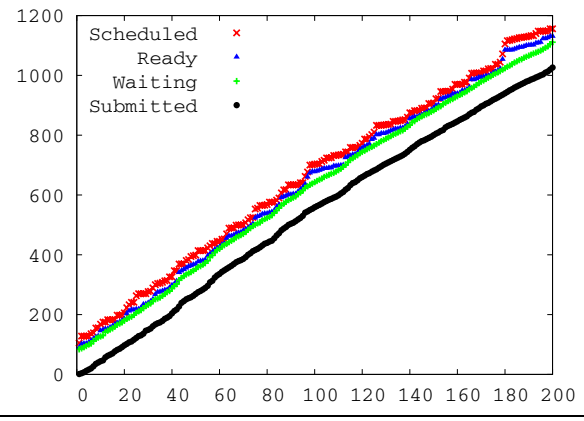
Rezultati svih sprovedenih mjerenja su grafički prikazani. Grafici prikazuju vremenske krive ulazaka poslova u određeno stanje (*Submitted, Ready, Waiting, Scheduled)* za različite načine podnošenja zahtjeva (serijski ili paralelno), pri čemu se mijenja broj zahtjeva (100 ili 200), veličina *Input Sandbox*-a (nema IS-a, IS ~ 8kB, IS ~ 5MB) i servis koji se koristi (*Network Server* ili *WMProxy).* Referentni početni trenutak mjerenja je trenutak kada je započeo ciklus podnošenja zahtjeva za izvršavanjem poslova.

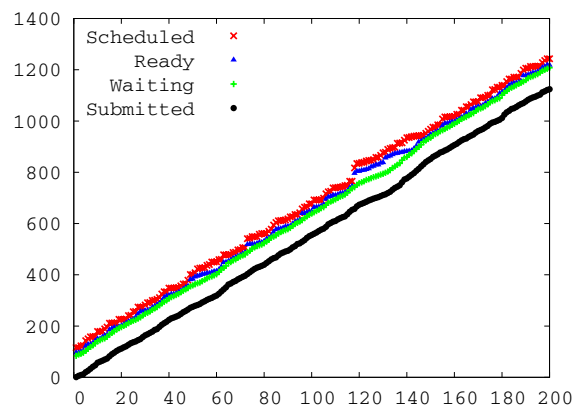


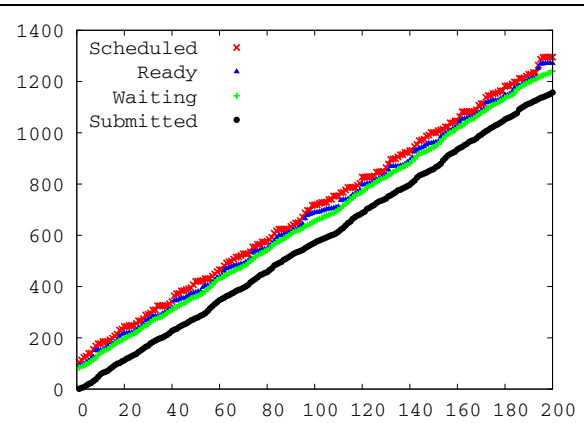




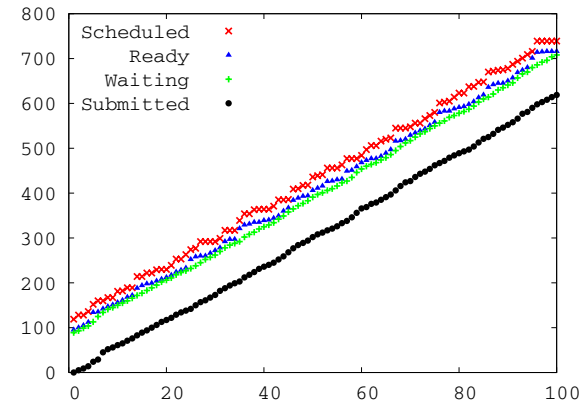
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova bez Input Sandbox-a, sekvencijalno, na Network Server.

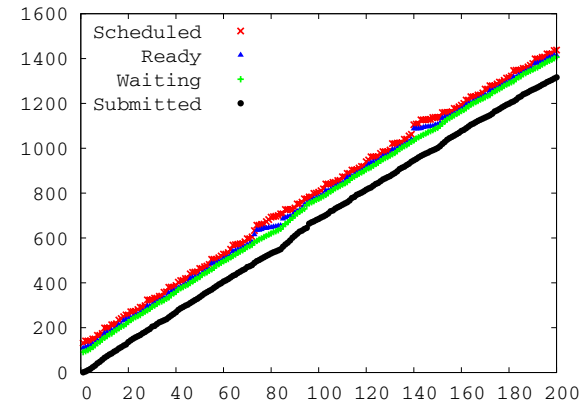


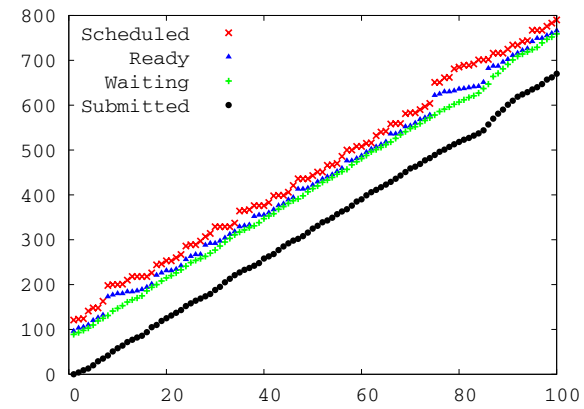


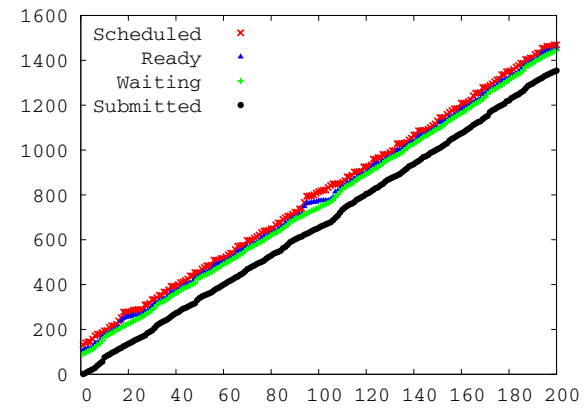


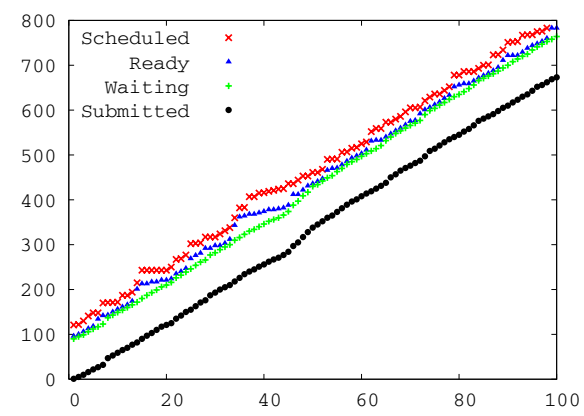
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova bez Input Sandbox-a, sekvencijalno, na Network Server.

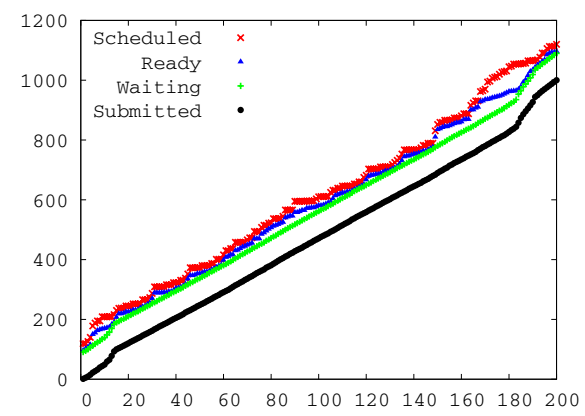






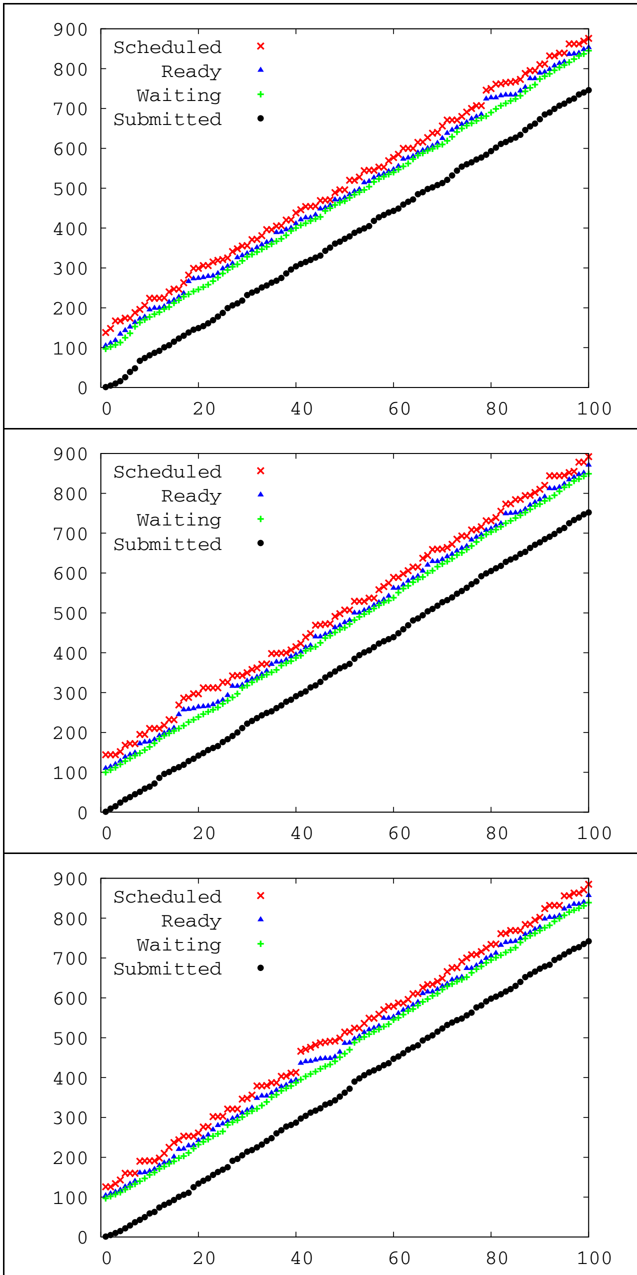




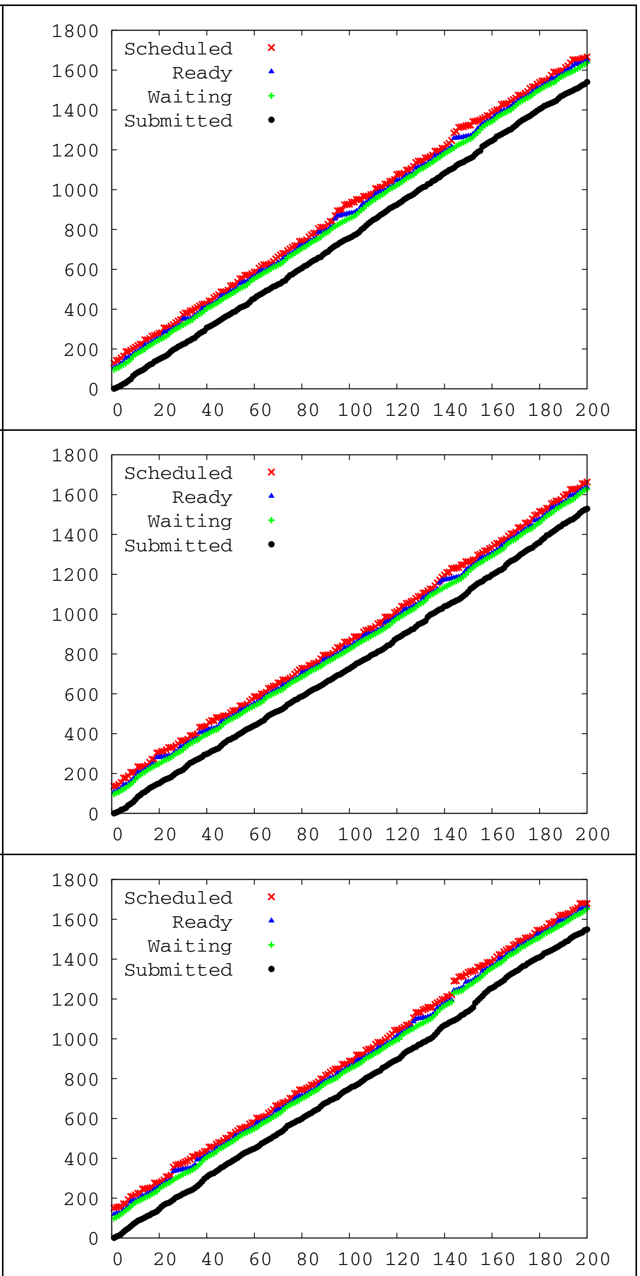


Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova sa malim Input Sandbox-om (8 kB), sekvencijalno, na Network Server.

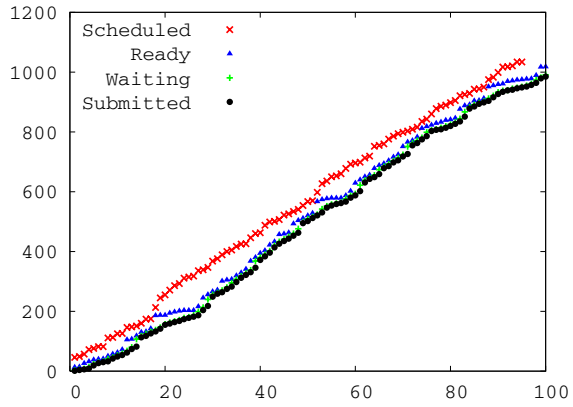
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sa malim Input Sandbox-om (8 kB), sekvencijalno, na Network Server.

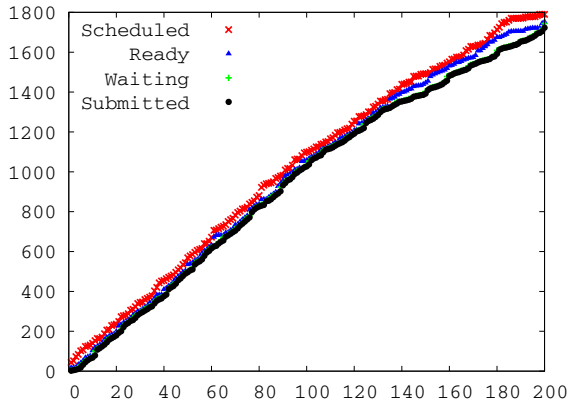


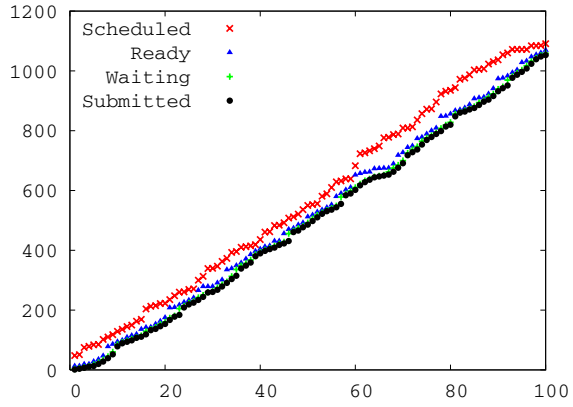
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova sa velikim Input Sandbox-om (5 MB), sekvencijalno, na Network Server.

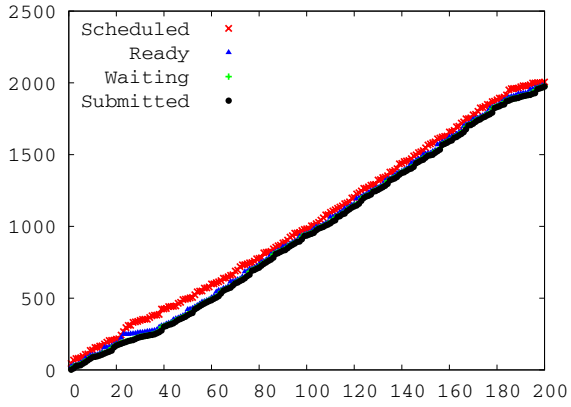


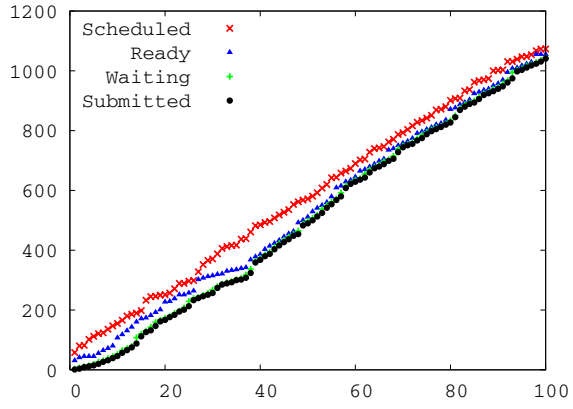
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sa velikim Input Sandbox-om (5 MB), sekvencijalno, na Network Server.

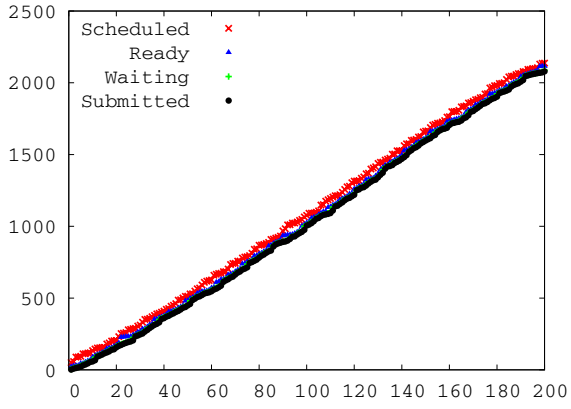






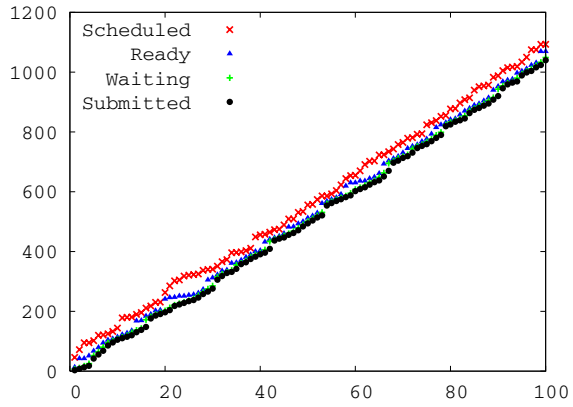


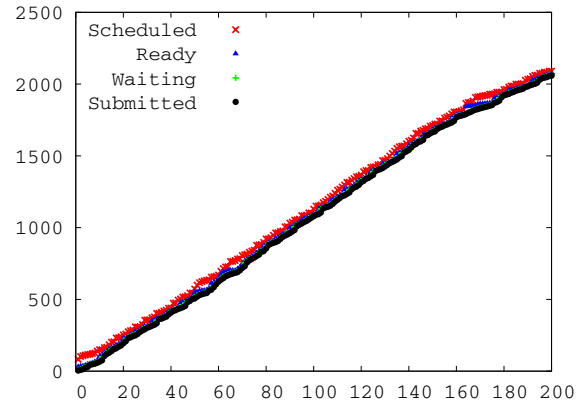


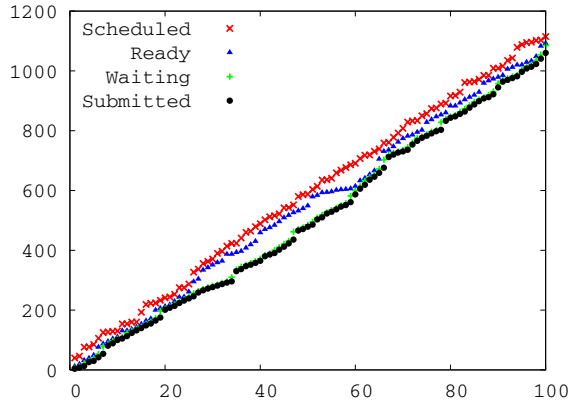


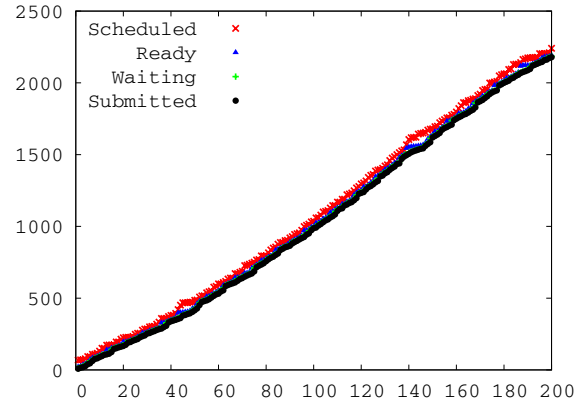
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova bez Input Sandbox-a, sekvencijalno, na WMProxy.

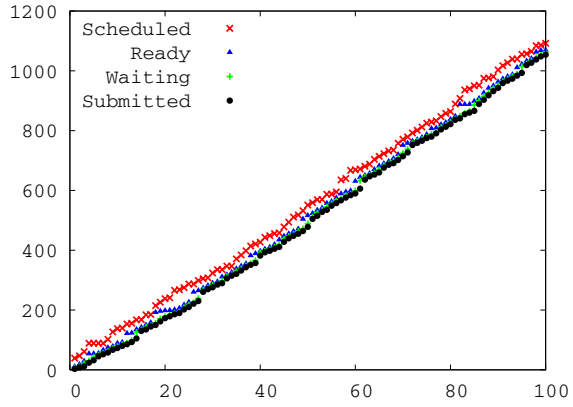
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova bez Input Sandbox-a, sekvencijalno, na WMProxy.

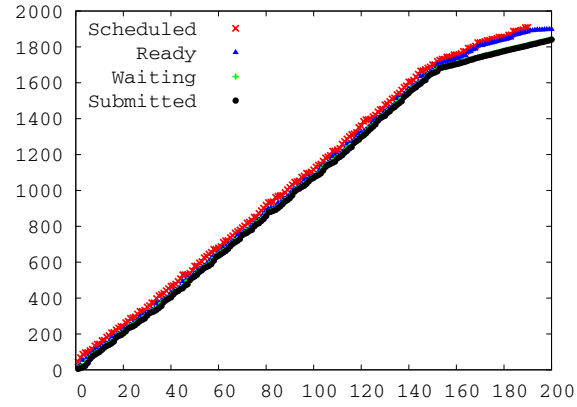






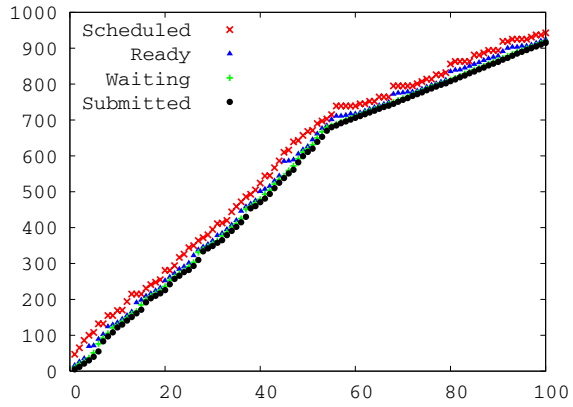


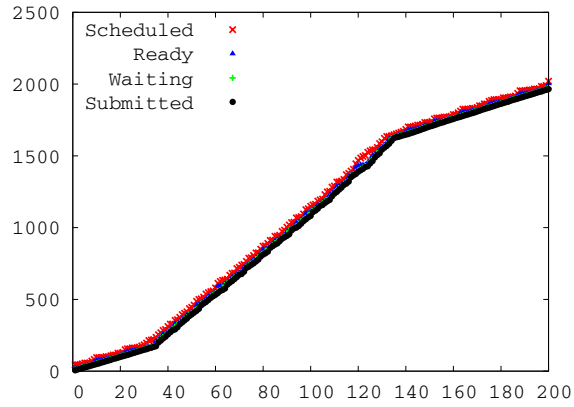


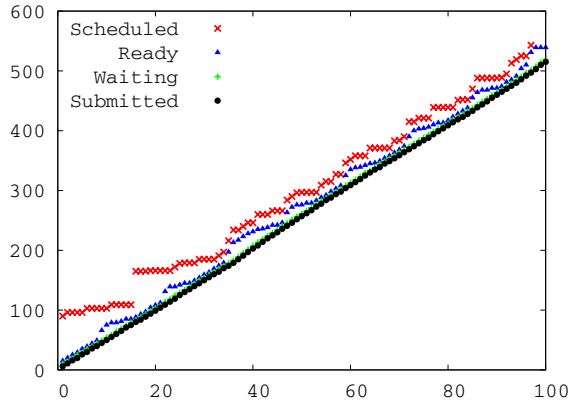


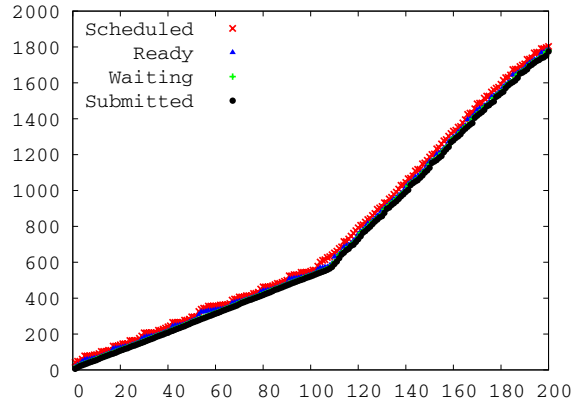
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova sa malim Input Sandbox-om (8 kB), sekvencijalno, na WMProxy.

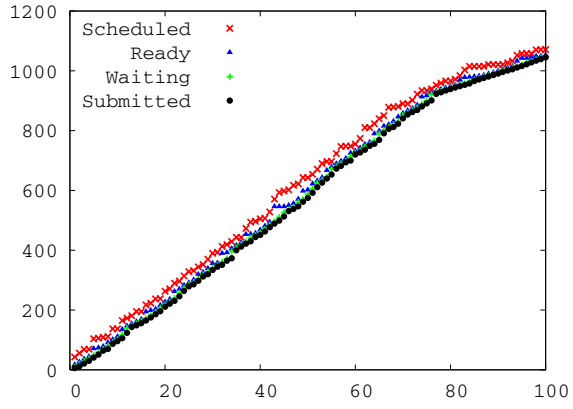
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sa malim Input Sandbox-om (8 kB), sekvencijalno, na WMProxy.

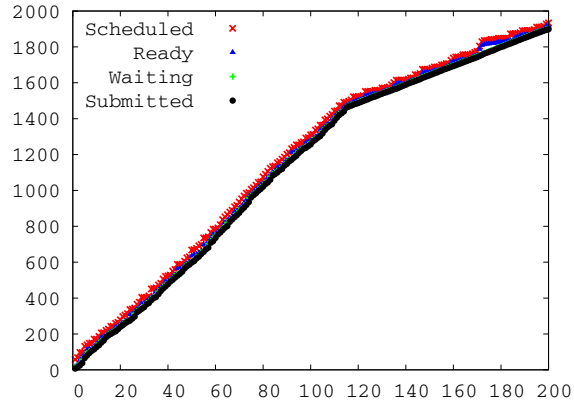






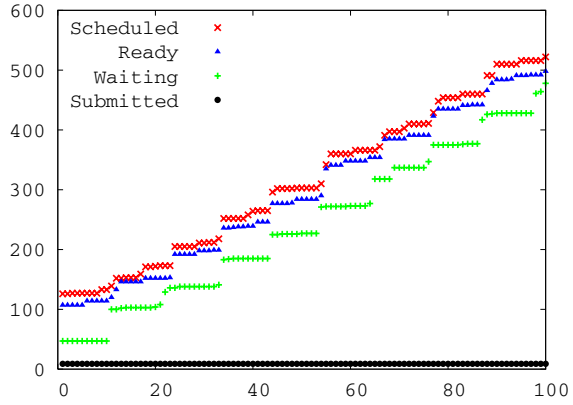


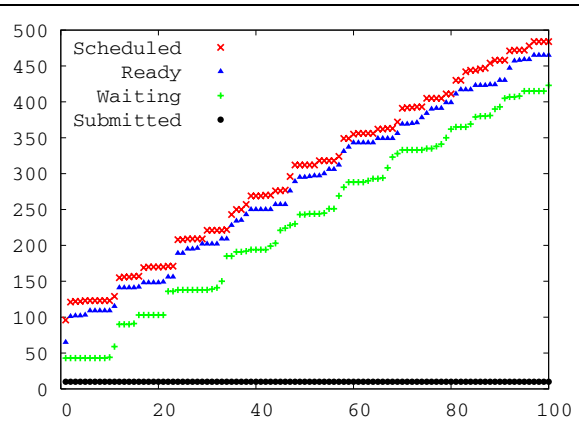


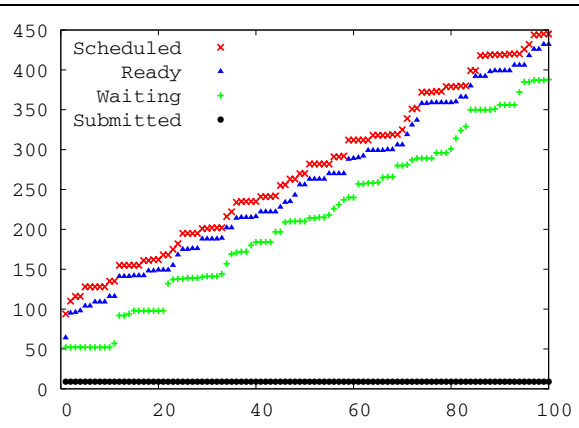


Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova sa velikim Input Sandbox-om (5 MB), sekvencijalno, na WMProxy.

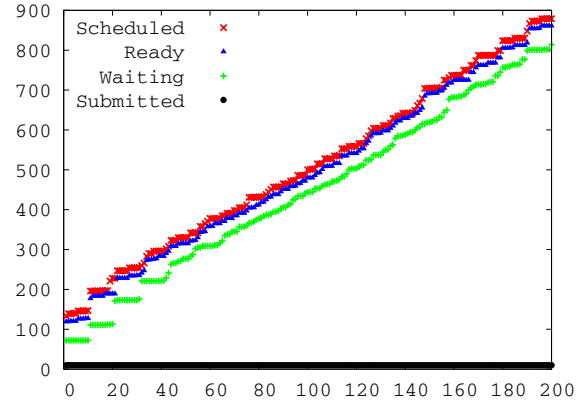
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sa velikim Input Sandbox-om (5 MB), sekvencijalno, na WMProxy.

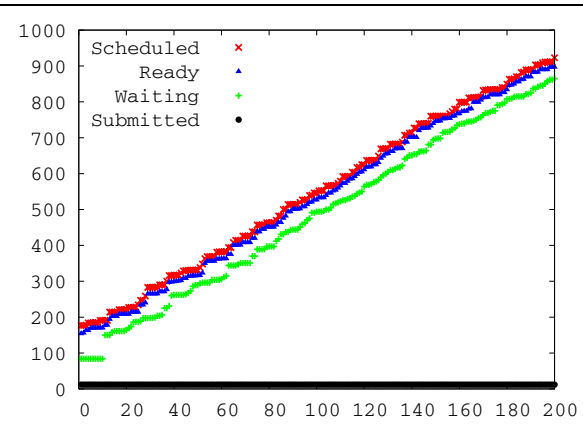


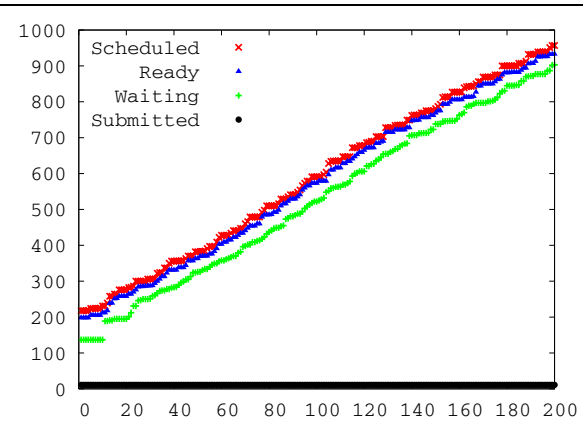




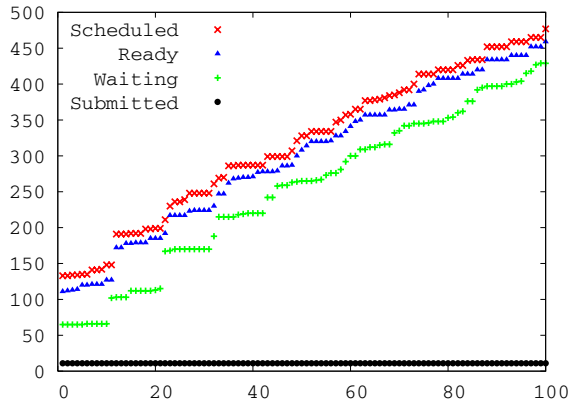
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova paralelno (kolekcija) na WMProxy bez Input Sandbox-a.

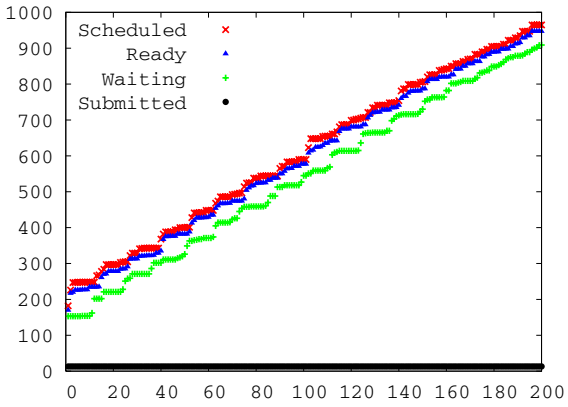


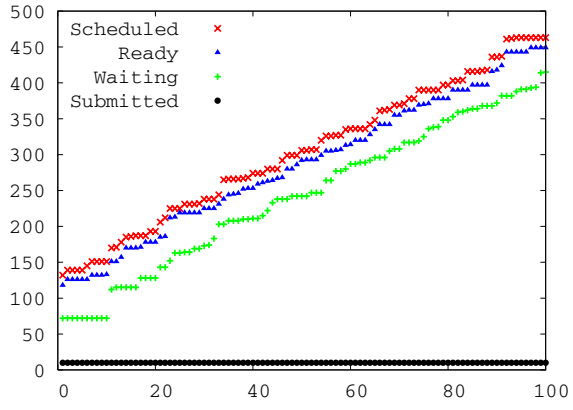


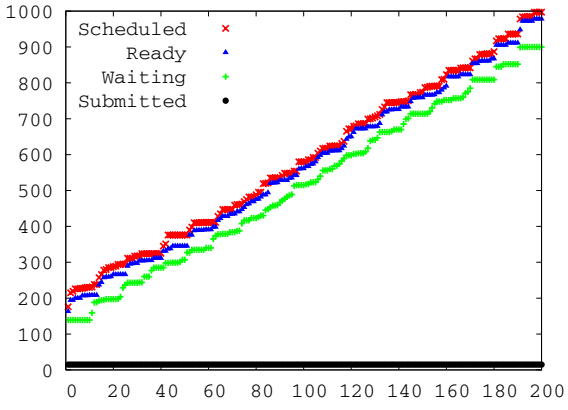


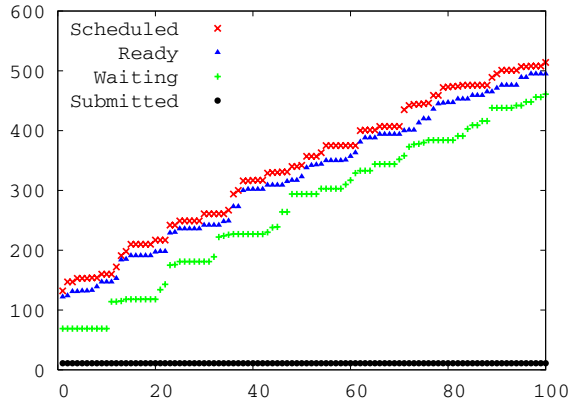
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova paralelno (kolekcija) na WMProxy bez Input Sandbox-a.

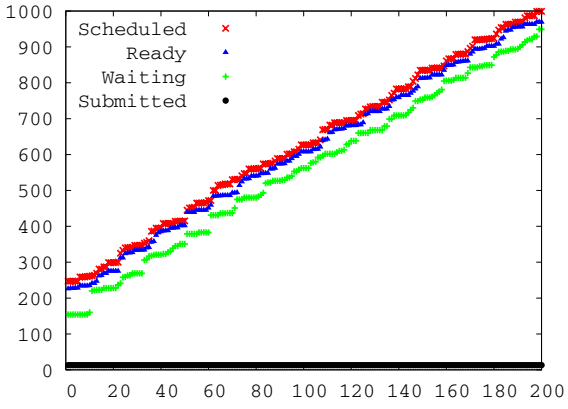






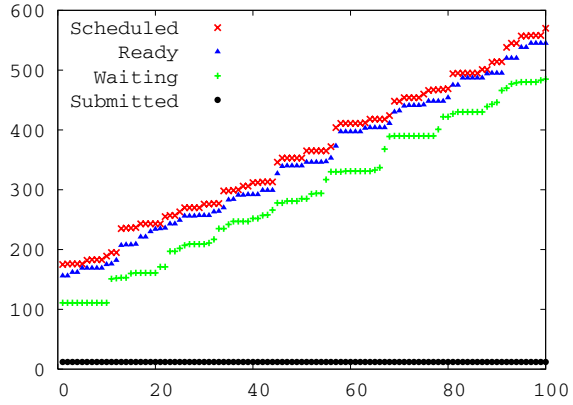


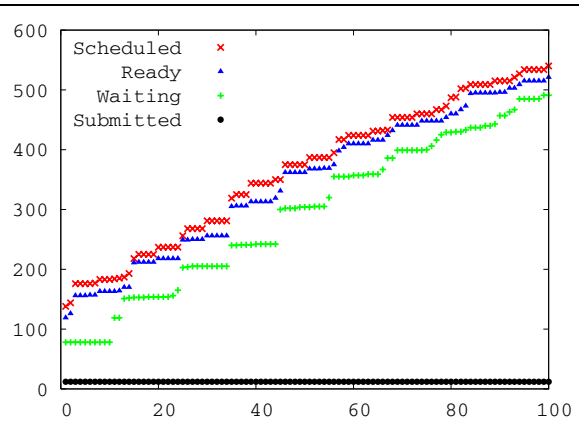


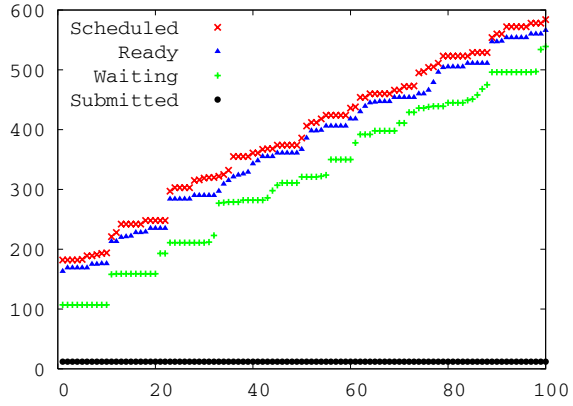


Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova paralelno (kolekcija) na WMProxy sa malim Input Sandbox-om (8 kB).

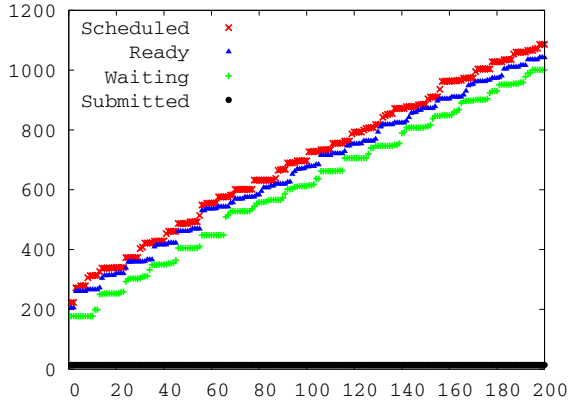
Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova paralelno (kolekcija) na WMProxy sa malim Input Sandbox-om (8 kB).

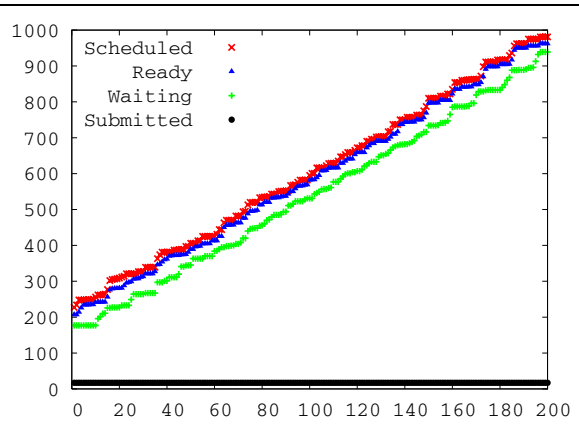


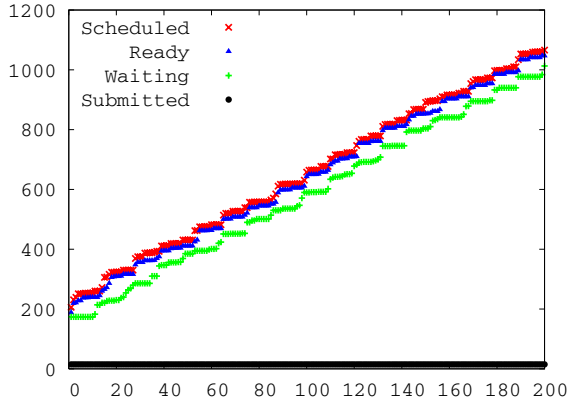




Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 100 jednostavnih poslova paralelno (kolekcija) na WMProxy sa velikim Input Sandbox-om (5 MB).







Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova paralelno (kolekcija) na WMProxy sa velikim Input Sandbox-om (5 MB).

Prilog D: Detaljni rezultati mjerenja

Slijedeće tabele predstavljaju prikaz srednjih vremena potrebnih da da posao pređe iz jednog stanja u drugo. Posmatrani su prelasci:

- iz stanja *Submitted* u stanje *Waiting* (kolona *SubWait*),
- iz stanja *Waiting* u stanje *Ready* (kolona *WaitReady*),
- iz stanja *Ready* u stanje *Scheduled* (kolona *ReadySched*),
- iz stanja *Submitted* u stanje *Ready* (kolona *SubReady*),
- iz stanja *Submitted* u stanje *Scheduled* (kolona *SubSched*).

Prikazane su srednje vrijednosti dobijene posmatranjem razlike pripadajućih vremena odgovarajućih stanja u po tri ponovljena eksperimenta za svaki posmatrani slučaj.

Slučaj sekvencijalnog slanja prostih poslova bez *Input Sandbox-a* na *Network Server*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
hello_ns0_100	83,53	15,03	25,34	98,56	122,58
hello_ns1_100	83,69	14,35	26,53	98,04	124,57
hello_ns2_100	84,12	17,41	29,99	101,53	131,52
avg	**83,78**	**15,60**	**27,29**	**99,38**	**126,22**
hello_ns0_200	83,74	16,82	26,05	100,56	126,61
hello_ns1_200	84,18	15,54	26,31	99,72	126,03
hello_ns2_200	84,44	14,79	27,22	99,23	126,46
avg	**84,12**	**15,72**	**26,53**	**99,84**	**126,37**

Slučaj sekvencijalnog slanja prostih poslova sa malim *Input Sandbox-om* reda veličine 8 kB na *Network Server*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
brolin_ns0_100	89,2	12,65	25,04	101,85	126,89
brolin_ns1_100	89,72	13,9	26,12	103,62	129,74
brolin_ns2_100	89,92	14,06	26,19	103,98	130,01
avg	**89,61**	**13,54**	**25,78**	**103,15**	**128,88**
brolin_ns0_200	90,02	12,58	23,95	102,60	126,56
brolin_ns1_200	90,50	12,06	25,21	102,55	127,76
brolin_ns2_200	90,34	22,74	26,28	113,08	139,36
avg	**90,29**	**15,79**	**25,15**	**106,08**	**131,23**

Slučaj sekvencijalnog slanja prostih poslova sa velikim *Input Sandbox-om* reda veličine 5 MB na *Network Server*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
mb_ns0_100	97,23	13,00	25,80	110,23	136,03

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
mb_ns1_100	97,37	12,83	26,95	110,20	137,15
mb_ns2_100	97,53	13,29	26,72	110,82	137,54
avg	**97,38**	**13,04**	**26,49**	**110,42**	**136,91**
mb_ns0_200	97,74	14,16	28,10	111,9	140,00
mb_ns1_200	98,31	13,28	27,99	111,60	139,59
mb_ns2_200	100,29	13,14	28,36	113,42	141,79
avg	**98,78**	**13,53**	**28,15**	**112,31**	**140,46**

Slučaj sekvencijalnog slanja prostih poslova bez *Input Sandbox-a* na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
hello_wmp3_100	6,48	19,43	60,07	25,91	85,85
hello_wmp4_100	6,47	16,82	55,65	23,29	78,94
hello_wmp5_100	6,33	24,78	49,53	31,11	80,64
avg	**6,43**	**20,34**	**55,08**	**26,77**	**81,81**
hello_wmp3_200	5,59	26,49	40,80	32,08	72,88
hello_wmp4_200	6,36	16,10	51,89	22,46	74,35
hello_wmp5_200	6,80	17,54	45,56	24,34	69,91
avg	**6,25**	**20,04**	**46,08**	**26,29**	**72,38**

Slučaj sekvencijalnog slanja prostih poslova sa malim *Input Sandbox-om* reda veličine 8 kB na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
brolin_wmp3_100	7,11	14,46	36,65	21,57	58,22
brolin_wmp4_100	7,64	36,32	38,21	43,96	82,17
brolin_wmp5_100	7,06	12,79	34,82	19,85	54,67
avg	**7,27**	**21,19**	**36,56**	**28,46**	**65,02**
brolin_wmp3_200	7,29	14,11	36,91	21,4	58,31
brolin_wmp4_200	7,57	13,98	38,44	21,55	59,99
brolin_wmp5_200	6,33	22,76	30,31	29,10	57,28
avg	**7,06**	**16,95**	**35,22**	**24,02**	**58,53**

Slučaj sekvencijalnog slanja prostih poslova sa velikim *Input Sandbox-om* reda veličine 5 MB na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
mb_wmp3_100	6,76	11,70	26,45	18,46	44,91
mb_wmp4_100	3,91	11,97	28,01	15,88	43,46
mb_wmp5_100	7,72	11,56	30,29	19,28	49,57

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
avg	**6,13**	**11,74**	**28,25**	**17,87**	**45,98**
mb_wmp3_200	7,05	11,84	26,34	18,89	45,22
mb_wmp4_200	6,66	11,77	25,41	18,42	43,84
mb_wmp5_200	6,92	13,00	25,74	19,92	45,66
avg	**6,88**	**12,20**	**25,83**	**19,08**	**44,91**

Slučaj paralelnog slanja prostih poslova (kolekcije) bez *Input Sandbox-a* na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
hello_coll0_100	232,89	54,34	17,53	287,23	304,76
hello_coll1_100	225,76	48,61	17,63	274,37	292,00
hello_coll2_100	205,02	42,86	18,00	247,88	265,88
avg	**221,22**	**48,60**	**17,72**	**269,83**	**287,55**
hello_coll0_200	436,48	46,55	16,92	483,02	499,94
hello_coll1_200	472,00	47,44	16,84	519,44	536,28
hello_coll2_200	514,29	46,145	17,98	560,44	578,42
avg	**474,26**	**46,71**	**17,25**	**520,97**	**538.21**

Slučaj paralelnog slanja prostih poslova (kolekcije) sa malim *Input Sandbox-om* reda veličine 8 kB na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
brolin_coll0_100	241,73	45,72	18,23	287,45	305,68
brolin_coll1_100	233,33	43,77	16,71	277,10	293,81
brolin_coll2_100	259,93	51,56	18,97	311,49	330,46
avg	**245,00**	**47,02**	**17,97**	**292,01**	**309,98**
brolin_coll0_200	526,93	49,70	17,38	576,64	594,02
brolin_coll1_200	502,18	50,14	19,10	552,32	571,43
brolin_coll2_200	544,90	49,28	20	594,18	614,18
avg	**524,67**	**49,71**	**18,83**	**574,38**	**593,21**

Slučaj paralelnog slanja prostih poslova (kolekcije) sa velikim *Input Sandbox-om* reda veličine 5 MB na *WMProxy*, gdje broj poslova iznosi 100 ili 200:

	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
mb_coll0_100	283,18	50,89	16,11	334,07	350,18
mb_coll1_100	282,68	51,44	18,45	334,12	352,57
mb_coll2_100	305,51	56,08	18,03	361,59	379,62
avg	**275,46**	**52,80**	**17,53**	**343.26**	**360,79**
mb_coll0_200	579,54	53,50	33,96	651,03	684,98

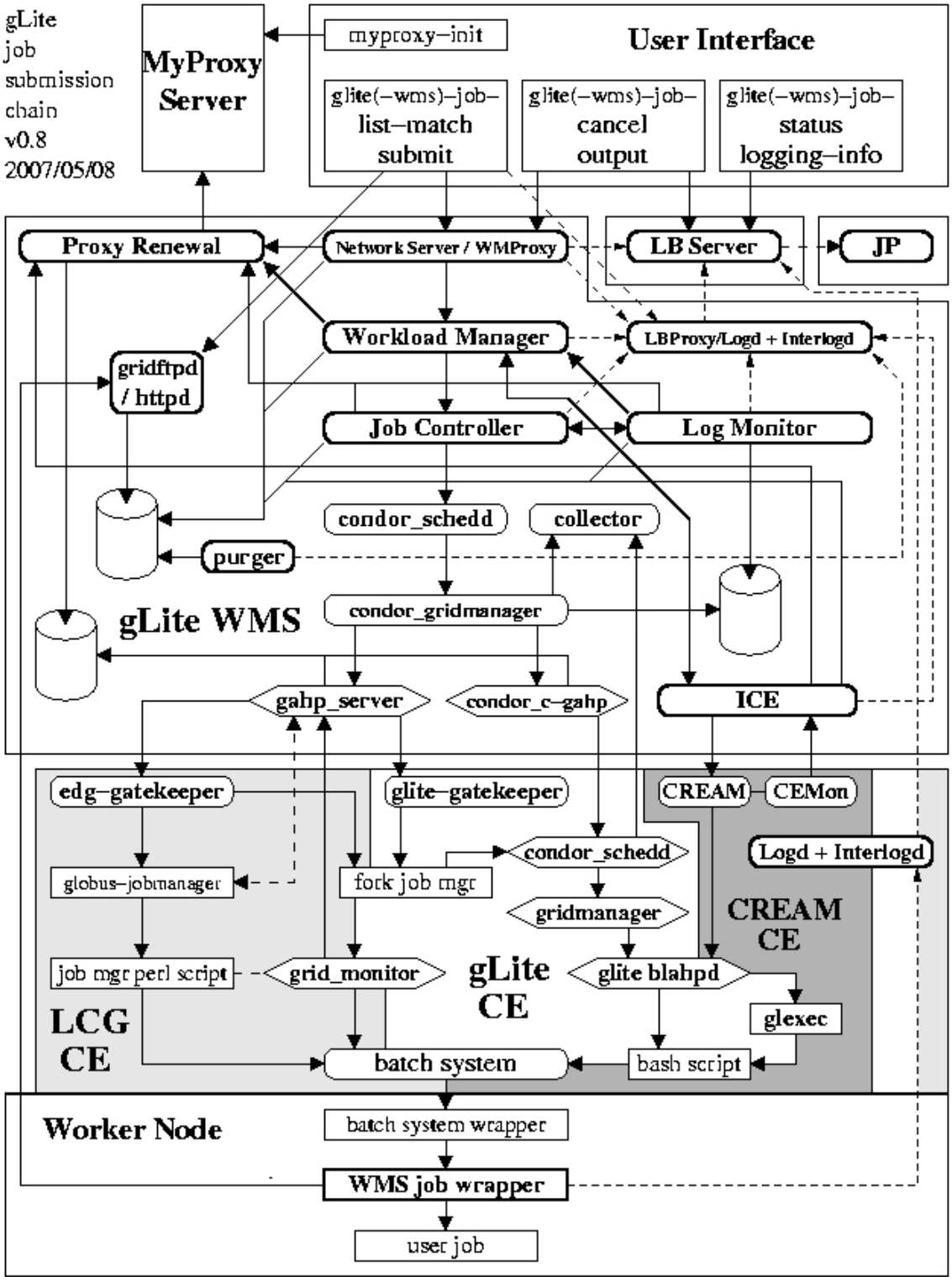
	SubWait	***WaitReady***	***ReadySched***	***SubReady***	***SubSched***
mb_coll1_200	521,73	47,84	16,94	569,57	586,52
mb_coll2_200	570,24	53,02	18,08	623,27	641,35
avg	**557,17**	**51,45**	**22,99**	**614,62**	**637,62**

Slijedeća tabela sadrži prikaz srednjeg vremena koje potrebno da posao uđe u određeno stanje (*Submitted, Waiting, Ready, Scheduled*), za svaki posmatrani slučaj. Pri izračunavanju su praćeni slijedeći parametri: trenutak kada je započet proces predaje poslova, trenutak kada je posljednji posao ušao u određeno stanje, broj poslova. Svaka od vrijednosti u tabeli je dobijena izračunavanjem srednje vrijednosti očitanih vremena za odgovarajuća stanja u po tri ponovljena eksperimenta za svaki posmatrani slučaj.

broj poslova	**100**				**200**			
status	**Sub**	**Wait**	**Ready**	**Sched**	**Sub**	**Wait**	**Ready**	**Sched**
NSs 0 kB [s/pos]	5.6	6.4	6.6	6.9	5.5	5.9	6.0	6.2
NSs 8 kB [s/pos]	6.5	7.4	7.6	7.8	6.1	6.6	6.6	6.7
NSs 5 MB [s/pos]	7.5	8.4	8.6	8.8	7.7	8.2	8.2	8.3
WMPs 0 kB [s/pos]	10.3	10.3	10.5	10.8	9.6	9.7	9.7	9.9
WMPs 8 kB [s/pos]	10.5	10.6	10.8	11.0	10.1	10.2	10.3	10.6
WMPs 5 MB [s/pos]	8.2	8.3	8.4	8.6	9.4	9.4	9.5	9.6
WMPk 0 kB [s/pos]	0.09	4.3	4.6	4.8	0.06	4.3	4.5	4.6
WMPk 8 kB [s/pos]	0.11	4.4	4.7	4.8	0.07	4.6	4.8	4.9
WMPk 5 MB [s/pos]	0.12	5.0	5.4	5.6	0.08	4.9	5.1	5.2

Prilog E: Obrada korisničkog zahtjeva za izvršavanjem posla

Sl. 9 daje detaljan prikaz komponenti *gLite middleware-a* koje učestvuju u lancu radnji koje je potrebno izvršiti od trenutka kada korisnik podnese zahtjev za izvršavanjem posla do trenutka kada se posao izvršava [29].



Sl. 9: Obrada korisničkog zahtjeva za izvršavanjem posla.

Komponente, komande i skript fajlovi su smješteni u poligone. Koriste se tri različita poligona za prikaz objekata:

- pravougaonik zaobljenih uglova: postoji samo jedna instanca tog objekta koja obrađuje sve poslove.
- pravougaonik: kreira se instanca po poslu.
- šestougao: kreira se instanca za svaku kombinaciju niza karaktera koji sadrži informacije o korisniku (*Distinguished Name, DN*) i niza karaktera koji specificira privilegije koje mu pripadaju(*Fully Qualified Attribute Name, FQAN*).

Strelice ukazuju na veze koje postoje među komponentama, ko pokreće šta, ko čita i odakle, ko i gdje piše (tok kontrole). Za ovu sliku zahvaljujemo se Maartenu Litmaath-u iz CERN-a [29].

Reference

[1] http://aegis.phy.bg.ac.yu/

[2] http://scl.phy.bg.ac.yu/

[3] R. Buyya and S. Venugopal, A Gentle Introduction to Grid Computing and Technologies http://www.gridbus.org/papers/GridIntro-CSI2005.pdf

[4] T. Hey and A. E. Trefethen, The UK e-Science Core Programme and the Grid, Journal of Future Generation Computer Systems (FGCS), vol. 18, no. 8, pp. 1017-1031, 2002.

[5] The Globus Project, http://www.globus.org

[6] *LHC Computing Grid Project*, http://lcg.web.cern.ch/LCG/

[7] EGEE project, http://www.eu-egee.org/

[8] SEE-GRID project, http://www.see-grid.eu/

[9] http://middleware.objectweb.org/

[10] gLite, http://glite.web.cern.ch/glite

[11] gLite 3.0 User Guide, https://edms.cern.ch/document/722398/1

[12] Overview of the Grid Security Infrastructure, https://www-unix.globus.org/security/overview.html

[13] GSIFTP Tools for Data Grid, http://www.globus.org/toolkit/docs/2.4/datagrid/deliverables/gsfiftp-tools.html

[14] RFIO: Remote File Input/Output, http://doc.in2p3.fr/doc/public/products/rfio/rfio.html

[15] dCache, http://www.dCache.org

[16] The Storage Resource Manager, http://sdm.lbl.gov/srm-wg/

[17] The GLUE schema http://infnforge.cnaf.infn.it/glueinfomodel/

[18] MDS 2.2 Features in the Globus Toolkit 2.2 Release, http://www.globus.org/toolkit/mds/#mds_gt2

[19] R-GMA: Relational Grid Monitoring Architecture, http://www.r-gma.org/index.html

[20] WMS Guide, https://edms.cern.ch/document/572489/1

[21] F. Pacini, Job Attributes Specification, https://edms.cern.ch/document/555796/1/

[22] F. Pacini, Job Description Language (JDL) Attributes Specification, https://edms.cern.ch/document/590869/1/

[23] LB Service User's Guide, https://edms.cern.ch/document/571273/

[24] http://trinity.datamat.it/projects/EGEE/wiki/wiki.php?n=Main.HomePage

[25] EGEE User's Guide, WMProxy Service,https://edms.cern.ch/document/674643/1

[26] N. Švraka, A. Balaž, A. Belić, A. Bogojević, Stabilnost i performanse gLite Workload Management System-a, Zbornik radova naučno-stručnog Simpozijuma INFOTEH-JAHORINA 2007

[27] https://gilda.ct.infn.it/UIPnP.html

[28] N. Švraka, A. Balaž, A. Belić, A. Bogojević, gLite Workload Performance Measurements, Zbornik radova konferencije INDEL 2006, str. 294-297.

[29] http://litmaath.home.cern.ch/litmaath/UI-WMS-CE-WN/