STABILNOST I PERFORMANSE GLITE WORKLOAD MANAGEMENT SYSTEM-A
STABILITY AND PERFORMANCE OF GLITE WORKLOAD MANAGEMENT SYSTEM

Neda Švraka, Antun Balaž, Aleksandar Belić, Aleksandar Bogojević, *SCL, Institut za fiziku, Beograd, Srbija*

Sadržaj - *Upotrebom računarskih resursa u okviru Grid infrastrukture rukovodi WMS, Workload Management System, čija je uloga da prihvati korisničke poslove, upari ih sa odgovarajućim računarskim resursima u okviru infrastrukture i distribura ih na trenutno dostupne resurse, prati poslove tokom njihovog izvršavanja i obezbjedi različite korisničke interfejse, koji omogućuju ispitivanje statusa u kom se posao nalazi i preuzimanje rezultata izvršenih poslova. Prikazani su i ukratko analizirani rezultati mjerenja stabilnosti i performansi WMS-a u okviru gLite middleware-a. Takođe, ukazano je na moguće probleme koji se mogu javiti u toku rada trenutne verzije ovog servisa.*

Abstract - *The production usage of computational Grids is governed by WMS, workload management systems, that accept jobs submitted by end users, match-make and distribute them to available clusters, monitor jobs while they are executed, and provide multiple interfaces to users for querying the status of jobs and retrieving their outputs. Here we present stability and performance measurement results of the gLite Grid middleware WMS. We also identify possible problems and bottlenecks in the current release of this service.*

1. UVOD U GRID PARADIGMU U RAČUNARSTVU

Pri izvođenju mnogih naučnih i inženjerskih eksperimenata generiše se velika količina podataka. Obrada ovih podataka zahtjeva upotrebu značajnih računarskih resursa, resursa za skladištenje podataka, kao i angažovanje osoblja na njihovom upravljanju i održavanju. Naučnici I inženjeri se suočavaju sa problemima za čije je riješavanje potrebna velika procesorska moć, koje možemo grubo podijeliti na: zadatke obrade velike količine distribuiranih podataka, zadatke intenzivnih numeričkih analiza, poslove koji zahtjevaju simultani rad grupe međusobno udaljenih istraživača, koji pristupaju istom resursu istovremeno. Treba napomenuti da se u okviru problema u praksi često preklapaju različite vrste navedenih zadataka, na primjer intenzivna računarska analiza velike količine distribuiranih podataka. Često jedan računar, klaster računara ili super-računar posebne namjene, nije dovoljan za riješavanje savremenih istraživačkih i razvojnih problema.

Da bi se opisani problemi prevazišli uvodi se pojam *middleware-a*, softverskog sloja koji omogućava povezivanje distribuiranih računarskih i skladištnih resursa, kao i njihovu interoperabilnost, pružajući korisniku jedinstven pristup svim resursima, čak i u uslovima heterogenosti sistema sastavljenog od različitih hardverskih komponenti, operativnih sistema i komunikacijskih protokola. Na primjer, sistem može koristiti različite *batch* sisteme na pojedinačnim klasterima, ili različite vrste elemenata za skladištenje podataka zasnovanih na sistemu magnetnih traka ili pak generičkih računara sa više disk jedinica. Naravno, ovaj sloj je izgrađen na postojećoj mrežnoj infrastrukturi, koja je od suštinskog značaja za ispravno funkcionisanje *Grid* mreža.

Na neki način ovakav pristup je sličan *World Wide Web-u (WWW)*, te se očekuje da *Grid* učini na polju dijeljenja rašunarskih resursa ono što je *WWW* učinio na polju razmjene i dijeljenja informacija. Ipak valja naglasiti da postoje suštinske razlike između *WWW-a* i *Grid* mreža. Osnovna ideja interneta je slobodna razmjena informacija, obično mehanizmom klijent server interakcije, dok se u *Grid* okruženju razmjenju resursi, koji su vrijedna svojina onih koji ih obezbjeđuju (vlasnika), te se njima raspolaže na način koji oni odrede. Pored toga, da bi se postiglo najefikasnije iskorištenje raspoloživih računarskih resursa, potrebno je razviti i primjeniti složene algoritme i interni informacioni sistem, kao i skup novih servisa koji omogućuju krajnjem korisniku jednostavnu upotrebu mreže.

Postoje različite vrste *Grid* mreža, čija je svrha različita, kao što su nacionalne *Grid* infrastrukture (sa ulogom da povežu resurse u okviru određene zemlje, npr. AEGIS [1] u Srbiji, ili *e-Science* program u Velikog Britaniji), projektne *Grid* mreže (koje finansiraju agencije za svoje potrebe), zatim one koje dobrovoljno formiraju pojedinci u cilju rješavanja važnih zajedničkih problema (npr. pronalaženje deficitarnih lijekova), potrošačke mreže ustanovljene od strane komercijalnih kompanija, itd.

Projektne *Grid* mreže su trenutno glavni izvor različitih distribucija *middleware-a*, od kojih su neke javno dostupne, što omogućava uključenje široke zajednice pojedinaca u mrežu i prilagođavanje iste njihovim potrebama. Kreiraju se radi zadovoljenja potreba različitih multiinstitucionalnih istraživačkih grupa i raznorodnih „virtuelnih grupa“, u okviru kratkoročnih i srednjeročnih projekata, tipa naučne saradnje ili inženjeringa. Takav projekat je i *World Wide LHC Computing Grid Project (WLCG)* [2], razvijen da pripremi računarsku infrastrukturu za simulaciju, obradu i analizu podataka u eksperimentima sprovedenim u velikom akceleratoru čestica *(Large Hadron Collider (LHC))*. *LHC* se konstruiše pod okriljem evropske laboratorije *European Laboratory for Particle Physics (CERN)* i biće najveći i najmoćniji akcelerator čestica ikad napravljen. Ovaj projekat dijeli veliki dio svoje infrastrukture i tijesno sarađuje sa drugim evropskim projektom, *Enabling Grids for E-Science (EGEE-II)* [3], čija je glavni cilj da obezbjedi cjelodnevni, neprekidni pristup naučnicima geografski distribuiranoj računarskoj *Grid* infrastrukturi. *SEE-GRID-2* [4] je regionalni

projekat s ciljem obezbjeđivanja *Grid* infrastrukture u regionu jugoistočne Evrope, uključenjem novih regionalnih zajednica i podsticanja razvoja novih *Grid* aplikacija.

2. UVOD U GRID MIDDLEWARE

Suštinu *Grid-a* čini softver koji omogućuje korisniku pristup računarima distribuiranim u okviru mreže. Koncepcijski, on se nalazi između dva tipa softvera, operativnog sistema, koji pokreće računar (npr. Linux) i aplikacijskog softvera, koji riješava konkretan korisnički problem (npr. korisnički program za vizuelizaciju), pa je za njegov naziv upotrebljen pojam *middleware* .

Uloga *middleware-a* je da organizuje i integriše distribuirane računarske resurse u okviru *Grid-a* u koherentnu strukturu. On treba da omogući pokretanje aplikacija na odgovarajućim računarima, ma gdje se oni nalazili, na efikasan i pouzdan način. Takođe, obezbjeđuje korisniku jedinstven pristup resursima, putem korisničkog interfejsa.

Danas postoje različite distribucije *middlleware-a*: *Globus*, *LCG*, *gLite*, *UNICORE* i drugi, od koji se najviše korist *gLite* [5, 6], naslijednik *LCG-2 middleware-a*, razvijenog u okviru *WCLG* i *EGEE* projekta.

EGEE-II projekat je usredsrijeđen na održavanje *gLite middleware-a* i upravljanje velikom računarskom infrastrukturom za dobrobit rasprostranjene i raznolike istraživačke zajednice. *gLite* skriva kompleksnost ovog okruženja od korisnika, predstavljajući korisniku koherentni virtuelni računarski centar sa svim dostupnim resursima.

Slijedi sažet opis osnovnih elemenata i dostupnih interfejsa koji omogućuju korisniku pokretanje poslova i upravljanje podacima.

Pristupna tačka *WLCG/EGEE-II/SEE-GRID-2 Grid-u* je korisnički interfejs, *User Interface (UI)*. To može biti bilo koja mašina na kojoj korisnik ima nalog i na kojoj je instaliran korisnički digitalni sertifikat. Preko korisničkog interfejsa moguće je ostvariti autentifikaciju, autorizaciju korisnika, što mu omogućava upotrebu *WLCG/EGEE/SEE-GRID-2* resursa i pristupiti funkcionalnostima koje pružaju slijedeći servisi: informacioni, za upravljanje poslovima i za upravljanje podacima.

Computing Element (CE) predstavlja skup računarskih resursa, lokalizovanih na sajtu, u obliku klastera računara ili računarske farme.

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 diskova i magnetnih traka. Većina *WLCG/EGEE/SEE-GRID-2* sajtova obezbjeđuje bar jedan *SE*. Podržani su različiti protokoli i interfejsi za pristup podacima.

Information Service (IS) obezbjeđuje informacije o *WLCG/EGEEGrid* resursima i njihovom statusu.

U *Grid* okruženju fajlovi mogu imati replike na različitim sajtovima. U idealnom slučaju, korisnik ne mora da zna gdje se fajl nalazi, već koristi logičko ime fajla koje će servis za upravljanje podacima *(Data Management)* upotrebiti da locira fajl i pristupi mu.

Workload Management System (WMS) [7] prihvata korisničke poslove, dodjeli im odgovarajući računarski resurs, prati njihov status i omogućuje preuzimanje rezultate završenih poslova.

Konačno, *Logging and Bookkeeping service (LB)* prati poslove kojima upravlja *WMS,* bilježeći događaje koje generišu različite komponente *WMS-a* i stanja u kojima se nalazi posao.

3. PRINCIP RADA WMS-a

Kao što je već rečeno, uloga *WMS-a* je da od korisnika prihvati zahtjev za izvršavanjem i upravljenjem izvršavanja posla i preduzme odgovarajuće akcije da taj zahtjev zadovolji. Složenost upravljanja aplikacijama i resursima je skrivena od korisnika, čija je komunikacija sa *WMS-om* 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 specifikacijski jezik, visokog nivoa *Job Description Language (JDL)* [8].

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, paralelna (MPI), 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.

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, što omogućava podnošenje i upravljanje zahtjevima programskim putem.

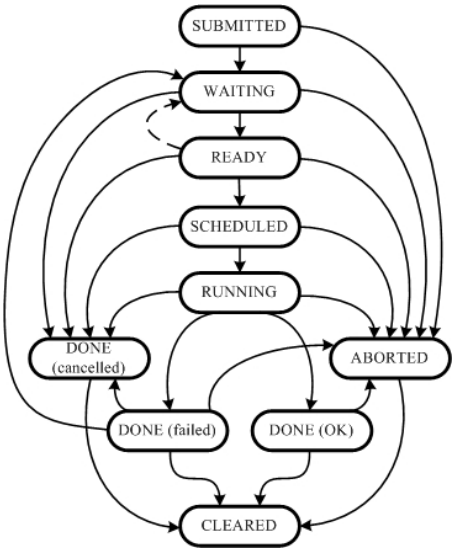
WMS-UI obezbjeđuje slijedeće operacije:

- pronalaženje liste resursa koji odgovaraju određenom poslu,
- podnošenje zahtjeva za izvršenjem posla na udaljenom *CE*-u,
- provjera statusa posla i otkaz posla,
- preuzimanje rezultata završenih poslova,
- potraživanje i prikaz informacija vezanih za istoriju izvršavanja posla,
- pregled statusa kontrolnih tačaka itd.

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, prikazanom na sl. 1.

Unutrašnja arhitektura *WMS-a* je data na sl. 2. Postoje dva načina za prihvatanje dolazećih zahtjeva. Jedan od njih je zasnovan na generičkom *daemon-u*, a drugi na interfejsu zasnovanom na *Web* servisima. Ova dva modula imaju ključnu ulogu u mjerenjima čiji su rezultati ovde prikazani.

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*.



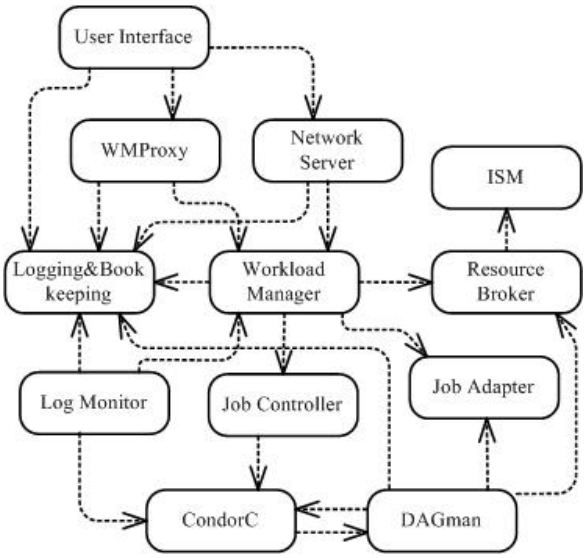
Sl.1. Konačni automat stanja Grid posla

Workload Manager Proxy (WMProxy) [9] 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-a* 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, koje vrše izbor resursa koji najviše odgovara zahtjevima posla *(Resource Broker)*, neposredno upravljaju izvršavanjem posla *(CondorC)*, pripremaju *CondorC* fajl i stvaraju odgovarajuće izvršno okruženje posla na radnom čvoru C*omputing Element-a (Job Adapter),* koordinirajući njihove akcije.

Logging and Bookkeeping (LB) servis 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.

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. Po podnošenju zahtjeva za izvršavanjem posla, korisnik može otkazati posao koristeći jedinstveni identifikator posla u bilo kom trenutku.



Sl. 2. Unutrašnja arhitektura WMS-a

4. MJERENJE PERFORMANSI WMS-a

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 zahtjeva značajne računarske resurse, te se stoga i koristi *Grid* mreža, razvijen je niz testova. Testovi podrazumijevaju podnošenje zahtjeva za izvršavanjem velikog broja poslova različitih zahtjeva, praćenje i analiziranje kritičnih događaja vezanih za iste.

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 *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*, zasnovan na *gLite 3.0 middleware*-u [10].

U prvom slučaju 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 zahtjev, u zavisnosti od načina podnošenja zahtjeva , koje može biti serijsko (niz poslova) u slučaju *Network Server-a* i *WMProxy-ja* ili paralelno (svi poslovi odjednom) u slučaju *WMProxy-ja* sa kolekcijama poslova. Takođe, interesantno je bilo vidjeti 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. Za izvođenje testova koriste se skript programi zasnovani na komandama linijskog interfejsa *(Command Line Interface - CLI)* koji je dio korisničkog interfejsa.

U okviru prvog mjerenja slali smo niz zahtjeva za izvršavanje poslova. Nizovi su sadržavali 100 i 200 jednostavnih poslova čije izvršavanje ne zahtijeva prenos dodatnih fajlova na *WMS* (ne postoji *Input Sandbox*). Nakon toga se postupak ponavlja za poslove čije izvršavanje zahtijeva prenos dodatnih fajlova na *WMS* (postoji *Input Sandbox*). Posmatrana ukupna veličina dodatnih fajlova u prvom slučaju iznosi 8 kB, a u drugom 5 MB.

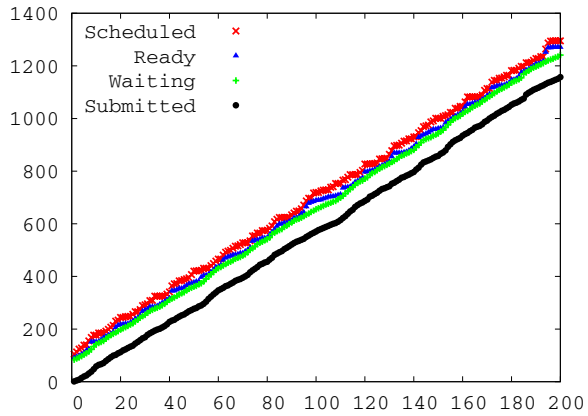
Takođe, istraživali smo šta se dešava ako se zahtjevi za izvršavanjem poslova paralelno predaju, upotrebom jedne komandne linije, što je novina koju unosi *WMProxy* servis. Mjerenja smo izveli kako za jednostavne poslove čije izvršavanje ne zahtijeva prenos dodatnih fajlova na *WMS* (ne postoji *Input Sandbox*), tako i za poslove čije izvršavanje zahtijeva 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 smo pratili slijedeća stanja:

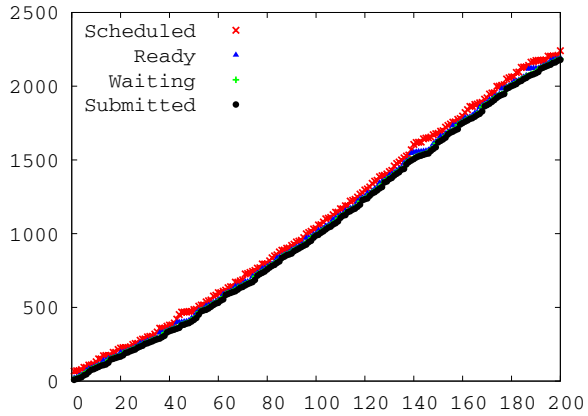
- *Submitted*: korisnik je predao posao *WMS-u*, ali još nije prebačen *NS-u* na obradu.
- *Waiting*: posao je prihvaćen od strane *NS-a* i čeka da ga obradi *WM* ili ga obrađuje neki od *WM-ovih* pomoćnih modula.
- *Ready*: *WM* ili neki od njegovih pomoćnih modula obrađuje posao, pronađen je odgovarajući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

Krive na slikama 3 i 4 predstavljaju tipičnu zavisnost vremena ulaska poslova u stanja *Submitted, Waiting, Ready* i *Scheduled* od broja poslova, u slučajevima podnošenja zahtjeva za izvršavanjem poslova posredstvom *Network Server-a* i *WMProxy-ja*, respektivno. Zahtjevi se podnose sekvencijalno. Može se uočiti da su krive koje odgovaraju trenucima ulaska poslova u stanje *Submitted* približno linearne, pri čemu kriva na slici 4 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 veći na slici 4. 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. Vremena prelazaka između susjednih stanja *Waiting*, *Ready* i *Scheduled* su približno jednaka u oba slučaju što se može vidjeti analizom usrednjenih vrijednosti datih u tabeli 1.

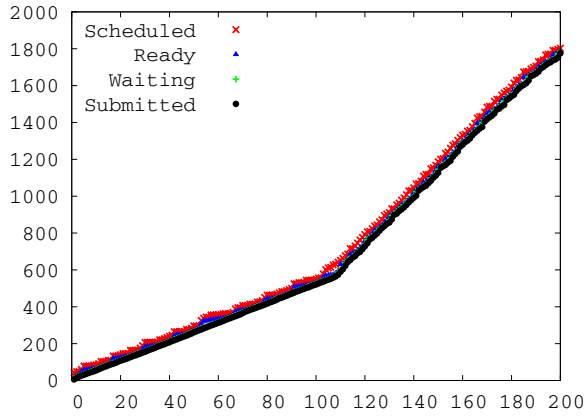
Prilikom izvođenja mjerenja uočena je nekonzistentnost u ponašanju *WMProxy* servisa, čiji primjer je dat na slici 5. Iako je prosječno vrijeme blisko očekivanom, krive umjesto jednog, imaju dva približno linearna segmenta. Međusobni odnosi krivih su održani za dati servis i način slanja zahtjeva.



Sl. 3. Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sekvencijalno na Network Server bez Input Sandbox-a.



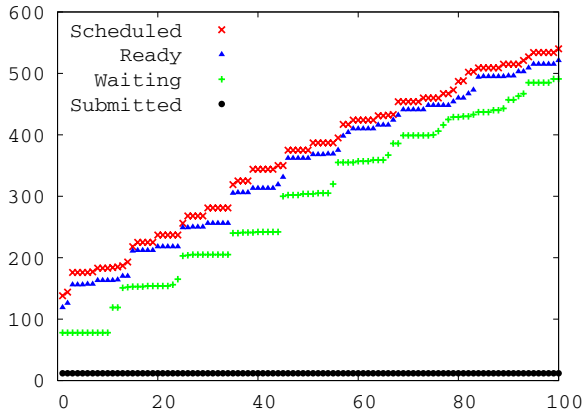
Sl .4. Zavisnost vremena (u sekundama) od broja poslova u slučaju slanja 200 jednostavnih poslova sekvencijalno na WMProxy sa Input Sandbox-om of 8 kB.



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

Tabela 1 daje prikaz srednjih vremena potrebnih da posao uđe u stanje *Submitted*, *Waiting*, *Ready* i *Scheduled*. Svaka od vrijednosti u tabeli je dobijena izračunavanjem srednje vrijednosti očitanih vremena za odgovarajuća stanja u po tri ponovljena eksperimenta za svaki posmatrani slučaj.

Krive na slici 6 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.



Sl .6. 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.

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 [11], 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 slici 5. Jedan od uzroka ovakvog ponašanja bi mogla biti promjena *gLite middleware-a*, sa *update-a* 04 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 [11].

Iz tabele 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 dedo 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 prohvatanja poslova usled većeg *Input Sandox-a.* Sa druge strane, *WMProxy* servis za kolekcije radi daleko bolje, uz očekivamo povećanje vremena prihvatanja po poslu između 1.1 i 1.2 puta.

Iz tabele 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.

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.

Ovaj rad je podržan od strane Ministarstva za nauku i zaštitu životne sredine Republike Srbije kroz projekat OI141035. 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).

LITERATURA

[1] http://aegis.phy.bg.ac.yu/
[2] http://lcg.web.cern.ch/LCG/
[3] http://www.eu-egee.org/
[4] http://www.see-grid.eu/
[5] http://glite.web.cern.ch/
[6] gLite 3.0 User Guide, https://edms.cern.ch/document/722398/1
[7] WMS Guide, https://edms.cern.ch/document/572489/1
[8] F. Pacini, Job Description Language (JDL) Attributes Specification, https://edms.cern.ch/document/590869/1/
[9] EGEE User's Guide, WMProxy Sevice, https://edms.cern.ch/document/674643/1
[10] https://gilda.ct.infn.it/UIPnP.html
[11] N. Švraka, A. Balaž, A. Belić, A. Bogojević, gLite Workload Performance Measurements, Zbornik radova konferencije INDEL 2006, str. 294-297.