# Modelowanie rozproszonych systemów typu desktop grid

Autoreferat pracy doktorskiej

Jakub Jurkiewicz

2 maja 2012

## 1 Wstęp

Od dawna prowadzone są badania na temat obliczeń rozproszonych. Jeden z rodzajów systemów realizujących takie obliczenia to systemy gridowe. Są to rozproszone systemy komputerowe, w których łatwość korzystania z dostępnych zasobów jest taka, jak korzystania z sieci (ang. *grid*) energetycznej. Użytkownik ma w nich, przy użyciu jednego interfejsu, dostęp do heterogenicznych systemów. Każdy z systemów może być zarządzany przez różnych administratorów i różne organizacje. Interfejs użytkownika powinien ukrywać przed nim różnice między systemami, pozwalając mu zajmować się wyłącznie przygotowaniem danych odpowiednich do zadania.

Szczególnym rodzajem systemów gridowych są systemy typu desktop grid i gridy społecznościowe. Do obliczeń wykorzystują one komputery osobiste, nieużywane w danej chwili przez właścicieli lub użytkowników. W pozostałych systemach gridowych (dla odróżnienia nazywanych gridami usługowymi) jednostki obliczeniowe przeznacza się wyłącznie na potrzeby gridu. W gridach społecznościowych komputery należą do prywatnych użytkowników. Są one połączone wolną siecią, co powoduje, że obliczenia muszą być od siebie niezależne, a komunikacja powinna być ograniczona do minimum. Gridy społecznościowe nie umożliwiają dokładnego specyfikowania węzłów, na których dane mogą być przetwarzane. Nie ma możliwości zlecania obliczeń wyłącznie komputerom należącym do konkretnej organizacji.

W odróżnieniu od gridów społecznościowych systemy typu desktop grid bazują na komputerach należących do pewnych organizacji. Najczęściej stoją one w ośrodkach naukowych, więc są połączone szybszą siecią niż w przypadku gridów społecznościowych. Dzięki temu obliczenia mogą być bardziej drobnoziarniste. Możliwe jest zlecanie obliczeń, które powinny być wykonane nie po kilku dniach czy tygodnia, ale po paru godzinach. Systemy typu desktop grid powinny umożliwić definiowanie organizacji, na komputerach której mogą być wykonane

obliczenia. Jest to potrzebne w trakcie przetwarzania danych wrażliwych – np. liczenia statystyk dla danych medycznych.

### 1.1 Cel pracy

Badania opisane w pracy doktorskiej dotyczą modelowania wydajności systemów typu desktop grid. Do realizacji tego zadania został wykorzystany symulator. Celem było stworzenia symulatora, który ma takie właściwości jak rzeczywisty system. Musi on uwzględniać specyficzne cechy systemu typu desktop grid, czyli włączenia i wyłączenia węzłów oraz zmienną przepustowość sieci.

W trakcie przeglądu systemów typu desktop grid stwierdzono brak łatwej ich integracji z gridami usługowymi. Istniejące systemy typu desktop grid mają niewystarczający system zapewniania bezpieczeństwa. Celem pracy było stworzenie systemu typu desktop grid opartego na oprogramowaniu gridu usługowego – UNICORE. Stworzenie tego systemu pozwala na pozytywne odpowiedzenie na dwa pytania:

- Czy system stworzony na bazie istniejącego oprogramowania warstwy pośredniej, używanego w gridach usługowych, może być wystarczająco wydajny do pracy w systemie typu desktop grid?
- Czy istnieje możliwość stworzenia wydajnego systemu typu desktop grid z akceptowalnym poziomem bezpieczeństwa?

Wyniki wydajności stworzonego systemu porównano z wynikami pracy symulatora, aby potwierdzić poprawność symulacji.

## 2 Symulacje systemu typu desktop grid

Badania prowadzone w kilku ośrodkach na świecie doprowadziły do stworzenia kilku dobrze działających symulatorów gridów usługowych. Przykładami takich symulatorów są GridSim Network Weather Service[1] i GridSim [2]. Nie uwzględniają one problemów związanych z dużą zawodnością elementów obliczeniowych. Jest to spowodowane specyfiką elementów obliczeniowych w gridach usługowych. Elementy te to dedykowane komputery bardzo rzadko celowo wyłączane, w przeciwieństwie do elementów obliczeniowych systemów typu desktop grid, które są bardzo często wyłączane. Ze względu na duże oddalenie od siebie różnych elementów obliczeniowych systemów typu desktop grid, kluczowe staje się symulowanie wydajności sieci łączącej te elementy. Tworzono w tym celu symulatory i emulatory systemów typu desktop grid i gridów społecznościowych takie jak SimBOINC [3], SimBA [4] i EmBOINC [5]. Są one jednak za mało zaawansowane i nie pozwalają na połączenie symulacji zmiennej przepustowości sieci z krótkotrwałymi obliczeniami. Konieczne było stworzenie nowego symulatora, który miałby wszystkie potrzebne właściwości.

Stworzony symulator jest oparty na przetwarzaniu zdarzeń. Zdarzenia to początek i koniec obliczeń, włączenie i wyłączenie węzła itd. Jednym ze zdarzeń jest zmiana przepustowości sieci.

## 2.1 Symulacja sieci

Większość badań, dotyczących wyłącznie przepustowości sieci, przeprowadzano dla przebiegów krótkoterminowych. Przebiegi te dały się dobrze symulować przy użyciu szeregów czasowych. Opisy tych doświadczeń zostały przedstawione w pracach [17] i [18]. Przedstawione w nich symulacje obejmowały czasy od kilku do kilkunastu godzin, podczas gdy dla potrzeb systemu typu desktop grid potrzebne symulacje powinny obejmować kilka do kilkunastu dni. Rozdzielczość 1 s, użyta we wspomnianych pracach, jest zbyt duża. Wystarczająca jest rozdzielczość 360 s (6 minut). Na podstawie danych z pomiarów sieci Internet, znaleziono model SARIMA[19]: $(1,0,1) \times (0,1,1)_{240} \times (0,1,1)_{1680}$, który bardzo dobrze pasuje do rzeczywistych danych. Okresy 1680 i 240 odpowiadają dobowym i tygodniowym cyklom, gdy przyjęta rozdzielczość wynosi 6 minut.

## 2.2 Symulacja zawodności elementów obliczeniowych

W symulacjach założono, że węzły obliczeniowe są charakteryzowane przez średni czas pomiędzy awariami (MTBF). MTBF stanowił wartość oczekiwaną rozkładu wykładniczego, z którego były losowane czasy pomiędzy awariami węzła obliczeniowego. MTBFy różnych komputerów losowano z rozkładu stanowiącego sumę dwóch rozkładów normalnych, różniących się jedynie wartością średnią. Za awarię uważane jest również wyłączenie oprogramowania na węźle obliczeniowym przez użytkownika, który potrzebuje komputera. Wiadomo, że rozkład tych wyłączeń nie jest równomierny w ciągu doby. Właściwy rozkład można będzie zdefiniować dopiero po stworzeniu systemu typu desktop grid, wdrożeniu i rozpoczęciu monitorowania.

# 3 Wyniki przykładowych symulacji

Zadaniem przykładowych symulacji było odpowiedzenie na trzy pytania:

1. Na ile dobrze symulator oddaje zachowanie bardzo prostego systemu z nieskończoną przepustowością sieci?
2. W jaki sposób optymalizować politykę przydziału zasobów dla systemu, w którym węzły są zawodne, a przepustowość sieci ulega zmianom?
3. Jaki element wpływa najbardziej na wydajność systemu typu desktop grid?

## 3.1 Testy symulatora dla bardzo prostego systemu

W trakcie testów symulatora dla prostego systemu stwierdzono, że symulator działa zgodnie z oczekiwaniami. Symulacje zajmowały się czasem wykonania jednego zadania składającego się z różnej liczby podzadań o stałej wielkości. Testowano system z różnymi średnimi czasami pomiędzy awariami węzłów. Symulator zgodnie z oczekiwaniami wskazał, że czas wykonania zlecenia nie zależy

liniowo od liczby zadań, tylko jest to funkcja schodkowa. Zbliża się ona do liniowej wraz ze zmniejszaniem średniego czasu pomiędzy awariami węzłów. Przy zaniedbaniu wydajności sieci okazało się, że polityki polegające na liczeniu jednego podzadania na wielu węzłach dają bardzo dobre rezultaty.

### 3.2 Optymalizacja pracy systemu typu desktop grid

Symulator służył do znalezienie optymalnej polityki przydziału zasobów dla systemu typu desktop grid. Testowano system, w którym przepustowość sieci była ograniczona i dodatkowo podlegała wahaniom. Sprawdzano systemy z różną ilością węzłów. Okazało się, że przy prostych politykach przydziału zasobów, powyżej pewnej ilości węzłów, wydajność systemu maleje zamiast rosnąć. Jest to spowodowane ograniczoną przepustowością sieci. Wyprowadzono wzór na ograniczenie maksymalnej liczby węzłów, które mogą jednocześnie liczyć w systemie:

$$t_{je} + \frac{D}{b}\left(n_{tot} - \frac{\overline{t_{je}}\, b}{D}\right) < \overline{MTBF} \tag{1}$$

gdzie $D$ to liczba danych koniecznych do przesłania (w obie strony) w trakcie realizacji podzadania, $b$ to przepustowość dzielonego pasma (pasma elementu, przez który podłączony jest zarządca), $t_{je}$ to część czasu wykonania podzadania przeznaczona na obliczenia na węźle, $n_{tot}$ to liczba zadań przydzielonych w danej chwili do węzłów, a $\overline{MTBF}$ to średni czas pomiędzy awariami węzłów. Stosowanie tego wzoru jest ograniczone wyłącznie do sytuacji, gdy pasmo elementu sieci, którym podłączony jest zarządca, zostało już w pełni wypełnione. Zastosowanie powyższego wzoru umożliwiło stworzenie polityki, która nie pogarsza wydajności systemu wraz ze wzrostem liczby węzłów.

### 3.3 Przyczyny obniżenia wydajności w systemie typu desktop grid

W symulacjach poszukiwano parametru, którego zmiana ma największy wpływ na wydajność systemu. Badano następujące parametry: liczba węzłów obliczeniowych, szybkość podłączenia zarządcy i szybkość podłączenia węzłów.

W badanym systemie okazało się, że największy wpływ na wydajność ma, zgodnie z oczekiwaniami, szybkość podłączenia zarządcy.

## 4 Stworzenie systemu typu desktop grid

Tworzenie systemu gridowego wymaga integracji różnych systemów i różnych rodzajów oprogramowania. Do tego celu wykorzystuje się oprogramowanie warstwy pośredniej tzw. middleware. Oprogramowanie to musi umożliwiać łatwe uwierzytelnianie i autoryzację użytkownika w systemie, unifikację dostępu do zasobów itd. Najpopularniejsze takie oprogramowanie dla gridów usługowych to Globus Toolkit (GTK) [6], UNICORE [7] i gLite[8]. Istnieje już gotowe oprogramowanie warstwy pośredniej dla systemów typu desktop grid. Jako przykłady

takiego oprogramowania można wymienić BOINC [9], Condor [10] i Sztaki [11]. Obecnie podejmowane są próby integracji gridów usługowych z systemami typu desktop grid. Zostały one opisane w pracach [12, 13, 14]. Rozwiązania te są mało elastyczne. Utrzymywanie takiego systemu w działaniu wymaga śledzenia na bieżąco zmian w oprogramowaniu obu łączonych systemów. Jednym z celów pracy doktorskiej jest wykazanie, że można wykorzystać oprogramowanie warstwy pośredniej, stworzone na potrzeby gridu usługowego, dla potrzeb systemu typu desktop grid. Pierwszym etapem tworzenia takiego systemu typu desktop grid jest wybranie właściwego oprogramowania warstwy pośredniej.

## 4.1 Porównanie oprogramowania gridów usługowych

GTK dynamicznie się rozwija w wielu ośrodkach. Istnieje wiele wersji tego oprogramowania. Rozwiązanie to ma swoje wady i zalety. Do niewątpliwych zalet należy duża ilość funkcji. Wielość wersji powoduje, w przypadku tworzenia własnego gridu, konieczność stworzenia własnej linii rozwojowej GTK, dopasowanie jej do własnych potrzeb, a także utrzymywanie jej (Nordugrid, Swegrid). W odróżnieniu od GTK UNICORE utrzymywany jest przez kilka ściśle współpracujących ośrodków. Dzięki temu wersja stabilna systemu jest w pełni działająca i można pozostawić jej utrzymanie twórcom. Dość nowym oprogramowaniem jest gLite – oprogramowanie dużo „lżejsze" niż GTK, podobnie jak UNICORE oparte na technologii serwisów webowych. GTK oraz gLite wykorzystują do delegacji uprawnień certyfikaty proxy [15], natomiast UNICORE korzysta z bezpośredniej delegacji zaufania [16].

### 4.1.1 Bezpośrednia delegacja zaufania

Jednym z ważnych elementów systemu bezpieczeństwa UNICORE jest bezpośrednia delegacja zaufania (Explicit Trust Delegation). Dzięki niej dowolny użytkownik może upoważnić innego użytkownika do wykonania ściśle zdefiniowanych czynności, z uprawnieniami upoważniającego. Umożliwia to uruchamianie przez zarządcę podzadań na maszynach docelowych „na konto" użytkownika, który zlecił zarządcy wykonanie całego zadania. Bezpośrednia delegacja zaufania jest rozwiązaniem znacznie bezpieczniejszym niż certyfikaty proxy.

## 4.2 Budowa systemu UNICORE

Ze względu na zaawansowany, oparty na bezpośredniej delegacji zaufania, system bezpieczeństwa oraz wysoką stabilność, wybrano system UNICORE do stworzenia systemu typu desktop grid. System UNICORE jest zbudowany z następujących modułów:

- Gateway – jest elementem każdej lokalizacji (site) systemu gridowego, który musi być dostępny z sieci zewnętrznej. Sprawdza on tożsamość osoby łączącej się, zapewniając bezpieczeństwo pozostałym modułom. Cały ruch sieciowy danej lokalizacji przechodzi przez gateway;

- UNICORE/X – jest to serwer będący głównym elementem każdej instalacji systemu UNICORE. Umożliwia on komunikację przy użyciu protokołów z rodziny serwisów webowych. Dostarcza on dwa rodzaje usług. Jeden z nich to usługi zgodne ze standardami z rodziny OGSA. Drugi to atomowe serwisy systemu UNICORE. Istnieje możliwość uruchomiania zadań lokalnie na serwerze, na którym działa UNICORE/X. Ta możliwość została wykorzystana w trakcie budowy systemu typu desktop grid;
- XUUDB – baza użytkowników i kluczy;
- Rejestr – przechowuje informację o usługach;
- System zarządzania złożonymi zadaniami;
- IDB – baza tworzonych zadań;
- UVOS – system zarządzania wirtualnymi organizacjami;
- CIS – system informacji o stanie gridu;
- TSI – interfejs systemu docelowego;
- Oprogramowanie klienckie: klient graficzny, klient tekstowy lub biblioteka umożliwiająca twórcom aplikacji łączenie się z gridem.

Ponieważ wersja 6 systemu UNICORE opiera się na technologii serwisów webowych, nie jest ona zależna od użytych wersji oprogramowania, systemów itd. Można zakładać, że używane protokoły nie będą zasadniczo zmieniane pomiędzy różnymi wersjami systemu.

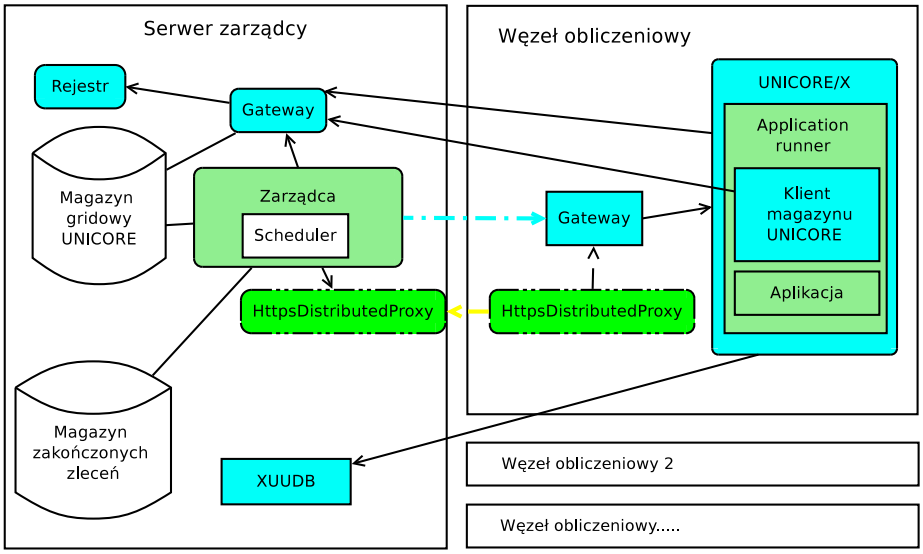
## 4.3 Architektura stworzonego systemu typu desktop grid

Stworzony system typu desktop grid składa się z zarządcy, węzłów obliczeniowych i oprogramowania klienckiego. Oprogramowanie klienckie podłączano na dwa sposoby: jako serwlet Javy umożliwiający dostęp do systemu przy użyciu przeglądarki WWW lub jako graficzną aplikację. Możliwe jest także stworzenie klienta, który udostępnia system typu desktop grid jako serwis UNICORE.

Komunikacja pomiędzy zarządcą a węzłami odbywa się przy użyciu serwisów UNICORE. Węzły mogą łączyć się z siecią, jednak może nie istnieć możliwość wykonywania połączeń z sieci do węzłów. Aby zarządca mógł łączyć się z węzłami, stworzono rozproszone proxy https.

## 4.4 Wykorzystanie modułów UNICORE do budowy systemu typu desktop grid

System typu desktop grid składa się z serwera rozdzielającego zadania (zarządcy) i węzłów obliczeniowych. Wykorzystanie modułów UNICORE w elementach systemu typu desktop gridu przedstawiono na rysunku 1.




Rysunek 1: Architektura systemu desktop grid

### 4.5 Bezpieczeństwo stworzonego sytemu typu desktop grid

Bezpieczeństwo w stworzonym systemie jest oparte na systemie bezpieczeństwa UNICORE i wykorzystuje certyfikaty X509. Certyfikatami identyfikują się użytkownik, zarządca i węzły. Zdarza się, że więcej niż jeden węzeł identyfikuje się tym samym certyfikatem. Ponieważ zadanie w trakcie wykonania nie przechowuje danych na węźle, musi ono także korzystać z certyfikatu do pobierania danych i zapisania wyników. Dla zwiększenia bezpieczeństwa każde zadanie otrzymuje oddzielny klucz, umożliwiający wykonanie wyłącznie tych czynności. Dzięki temu bardzo trudne jest celowe niszczenie lub zaburzanie wyników więcej niż jednego zadania. Zastosowanie bezpośredniej delegacji zaufania umożliwia przekazanie węzłom wiarygodnej informacji o zlecającym obliczenie. Umożliwia to węzłom przypisanie wyższych priorytetów zadaniom pochodzącym z ich grupy roboczej – wirtualnej organizacji.

Możliwość identyfikacji węzłów pozwala na zastrzeżenie miejsca, gdzie zlecenie ma być wykonane, np. tylko w ramach instytucji, do której należy zlecający. Wymaga to jednak zapewnienia, że klucze tych węzłów nie zostaną upublicznione.

## 5 Eksperymenty wydajnościowe

Ze względu na aspekt losowości nie jest możliwe dokładne przewidzenie czasu obliczeń systemu opartego na zawodnych komputerach połączonych siecią, której przepustowość podlega wahaniom. Z tego powodu badanie nie dotyczyło zgodności uzyskanych w trakcie symulacji czasów obliczeń z czasami obliczeń

na rzeczywistym systemie, lecz zgodności trendów opisujących zachowanie systemu, zaobserwowanych przy użyciu symulacji. Jedna z reguł sformułowanych w trakcie symulacji brzmi:
*Połączenie węzła centralnego z siecią może stanowić wąskie gardło systemów typu desktop grid, których celem jest prowadzenie dobrze zrównoleglonych obliczeń i zwracanie wyników w czasie kilkunastu lub kilkudziesięciu minut.*

W trakcie badań okazało się, że zasada ta odnosi się także do testowej instancji stworzonego systemu. Istotnie, powyżej pewnej liczby węzłów, to sieć limituje szybsze wykonanie obliczeń. W trakcie przeprowadzenia symulacji o parametrach dokładnie odpowiadających rzeczywistemu systemowi okazało się, że symulator bardzo dobrze przewiduje czas działania testowanego prostego systemu.

W trakcie testów stwierdzono dostateczną wydajność stworzonego systemu typu desktop grid.

# 6 Podsumowanie

Stworzony na potrzeby pracy model systemu ma właściwości takie jak rzeczywisty system. Prawidłowo symuluje on awarie. Szczególnie interesujące okazały się właściwości wynikające z ograniczonej, zmiennej przepustowości sieci i wyłączeń węzłów.

Stworzony system typu desktop grid, opisany w pracy, bazuje na oprogramowaniu UNICORE. Badania pokazały, że jest on dostatecznie wydajny.

Stworzony system ma ten sam poziom bezpieczeństwa co system UNICORE. Bezpieczeństwo w systemie UNICORE jest jednym z najlepiej rozwiązanych wśród systemów gridowych, a jego poziom jest całkowicie akceptowalny. Zatem poziom bezpieczeństwa stworzonego systemu także jest całkowicie akceptowalny.

Dalsza praca nad rozwojem opisanego systemu powinna mieć na celu stworzenie wygodnego interfejsu użytkownika oraz wykorzystanie w pełni możliwości bezpieczeństwa systemu UNICORE. Wykorzystanie w jednym systemie kilku serwerów zarządzającymi oraz wielu magazynów danych powinno znacząco zwiększyć wydajność, umożliwiając podłączanie większej ilości węzłów.

Po wdrożeniu systemu do pracy z zadaniami zlecanymi przez naukowców warto przeprowadzić nowe symulacje mające na celu optymalizację pracy systemu oraz dokładniej dopasować parametry modelu do rzeczywistych systemów.

Wyniki opisane w pracy były publikowane w następujących monografiach, czasopismach i na konferencjach:

- Jakub Jurkiewicz, Krzysztof Nowiński, Piotr Bała, Prediction of the Jobs Execution on the Community Grid, *CGW'07 Proceedings*, Kraków, 2008, str. 299-305,
- Jakub Jurkiewicz, Krzysztof Nowiński, Piotr Bała, Prediction of the Jobs Execution on the Community Grid with added network latency, *Distributed and Parallel Systems In Focus: Desktop Grid Computing*, Springer, 2008, str. 43-48,

- Jakub Jurkiewicz, Krzysztof Nowiński, Piotr Bała, Numerical simulations of the resource utilization in the community grids, *Polish Journal of Enviromental Studies*, 17(3B), 2008, str. 1230-1485,
- Jakub Jurkiewicz, Krzysztof Nowiński, Piotr Bała, UNICORE 6 as a platform for desktop grid, *Computer Science and Information Technology*, IMCSIT 2008, International Multiconference on Computer Science and Information Technology, 2008
- Jakub Jurkiewicz, Piotr Bała, Building UNICORE Based Desktop Grid, *UNICORE Summit 2010*, Schriften des Forschungszentrums Jülich IAS Series, vol. 5, 2010, str. 11-18.

# Literatura


[1] R. Wolski, N. Spring, J. Hayes, The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing, Journal of Future Generation Computing Systems,Volume 15, Numbers 5-6, pp. 757-768, (October 1999).

[2] B. Rajkumar and M. Manzur, GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing, The Journal of Concurrency and Computation: Practice and Experience (CCPE), Volume 14, Issue 13-15, Wiley Press, Nov.-Dec., 2002.

[3] D. Kondo. *Scheduling task parallel applications for rapid turnaround on desktop grids.* PhD thesis, La Jolla, CA, USA, 2005.

[4] M. Taufer, A. Kerstens, T. Estrada, D. Flores, and P. J. Teller. SimBA: A Discrete Event Simulator for Performance Prediction of Volunteer Computing Projects. In *PADS '07: Proceedings of the 21st International Workshop on Principles of Advanced and Distributed Simulation*, pages 189–197, Washington, DC, USA, 2007. IEEE Computer Society.

[5] T. Estrada, M. Taufer, K. Reed, and D. P. Anderson. EmBOINC: An emulator for performance analysis of BOINC projects. In *IPDPS '09: Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing*, pages 1–8, Washington, DC, USA, 2009. IEEE Computer Society.

[6] I. Foster, *Globus Toolkit Version 4: Software for Service-Oriented Systems.*, IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2006.

[7] M. Riedel, D. Mallmann, *Standardization Processes of the UNICORE Grid System*, Proceedings of 1st Austrian Grid Symposium 2005, Schloss Hagenberg, Austria, 2005, Austrian Computer Society, pages 191-203.

[8] E. Laure, C. Gr, S. Fisher, A. Frohner, P. Kunszt, A. Krenek, O. Mulmo, F. Pacini, F. Prelz, J. White, M. Barroso, P. Buncic, R. Byrom, L. Cornwall, M. Craig, A. Di Meglio, A. Djaoui, F. Giacomini, J. Hahkala, F. Hemmer, S. Hicks, A. Edlund, A. Maraschini, R. Middleton, M. Sgaravatto, M. Steenbakkers, J. Walk, and A. Wilson. Programming the Grid with gLite. In *Computational Methods in Science and Technology*, page 2006, 2006.

[9] D. P. Anderson David, *BOINC: A System for Public-Resource Computing and Storage*, 5th IEEE/ACM International Workshop on Grid Computing. November 8, 2004, Pittsburgh, USA.

[10] Thain D., Tannenbaum T., Livny M., *Distributed Computing in Practice: The Condor Experience*, Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pages 323-356, February-April, 2005

[11] A. C. Marosi, G. Gombás, Z. Balaton, P. Kacsuk, and T. Kiss. SZTAKI Desktop Grid: Building a scalable, secure platform for Desktop Grid Computing. In *Making Grids Work Proceedings of the CoreGRID Workshop on Programming Models Grid and P2P System Architecture Grid Systems, Tools and Environments 12-13 June 2007*, Heraklion, Crete, Greece , 2007.

[12] O. Lodygensky, G. Fedak, V. Neri, M. Livny, and D. Thain. XtremWeb and Condor: Sharing Resources Between Internet Connected Condor Pool. In *In Proceedings of the IEEE International Symposium on Cluster Computing and the Grid (CCGRID'03) Workshop on Global Computing on Personal Devices*, 2003.

[13] Z. Balaton, Z. Farkas, G. Gombas, P. K. Kacsuk, R. Lovas, A. C. Marosi, A. Emmen, G. Terstyanszky, T. Kiss, I. Kelley, I. Taylor, and F. Araujo. EDGeS, the common boundary between service and desktop grids. *Parallel Processing Letters*, 18(3):433–445, September 2008.

[14] M. Keller, J. Kovacs, and Brinkmann A. Desktop Grids Opening up to UNICORE. In M. Romberg, P. Bala, R. Müller-Pfefferkorn, and D. Mallmann (Ed.), *UNICORE Summit 2011 Proceedings, 7–8 July 2011— Torun, Poland*, 2011.

[15] B. C. Neuman. Proxy-based authorization and accounting for distributed systems. In *Distributed Computing Systems, 1993., Proceedings the 13th International Conference on*, pages 283–291, may 1993.

[16] K. Benedyczak, P. Bala, S. van den Berghe, R. Menday, and B. Schuller. Key aspects of the UNICORE 6 security model. *Future Generation Computer Systems*, 27(2):195–201, 2011.

[17] N. Groschwitz, G. Polyzos (1994). A time series model of long-term traffic on the NSFNET backbone. In: Proceedings of the IEEE International Conference on Communications, ICC' 94.

[18] S. Basu, A. Mukherjee, S. Kilvansky (1996). Time series models for Internet traffic, Technical Report GIT-CC-95-27, Georgia Institute of Technology.

[19] G.E.P. Box, G.M. Jenkins, (1976). Time series analysis: Forecasting and control, 2nd ed., Holden Day, San Francisco.