Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie
Wydział Informatyki, Elektroniki i Telekomunikacji
Katedra Informatyki

Autoreferat rozprawy doktorskiej

Massively Self-Scalable Platform for Data Farming (Masywnie samoskalowalna platforma wspierająca eksperymenty typu "data farming")

mgr inż. Dariusz Król

Promotor: prof. dr hab. inż. Jacek Kitowski

Kraków, wrzesień 2013

1 Wprowadzenie

Postęp technologiczny w ostatnich latach, umożliwiający przeprowadzanie skomplikowanych symulacji komputerowych w krótkim czasie, doprowadził do powstania nowych sposobów prowadzenia badań naukowych silnie wykorzystujących dane. Przykładem tego typu podejścia jest tzw. "The Fourth Paradigm of Science" [1], w ramach którego, nowe odkrycia w nauce mają być osiągane poprzez analizę dużych ilości danych. W tym celu można stosować metody typu "data mining" [2] do analizy danych, np. zebranych z fizycznych eksperymentów. Podejściem komplementarnym, zdobywającym coraz większą popularność w ostatnim czasie jest metodyka "data farming" [3]. Ideą tej metodyki jest zdobywanie wiedzy nt. badanego zjawiska poprzez przeprowadzanie dużej liczby symulacji komputerowych, z których każda zostaje uruchomiona z innych wektorem wartości wejściowych. Wyniki zebrane ze wszystkich wykonanych symulacji są następnie analizowane w celu lepszego zrozumienia badanego zagadnienia jak również wykrycia anomalii w stosowanym modelu symulacyjnym.

Badania "data farming" organizowane są w tzw. eksperymenty typu "data farming", którymi nazywamy procesy badawcze wybranych zjawisk obejmujące:

1. określenie celów eksperymentu w postaci pytań na które poszukuje się odpowiedzi w danym eksperymencie,
2. przygotowanie scenariusza symulacyjnego, tj. określenie modelu symulacyjnego badanego zjawiska, środowiska w którym symulacje będą wykonywane oraz określenie typu wejścia i wyjścia z pojedynczej symulacji,
3. specyfikacja przestrzeni parametrycznej, tzn. dla każdego parametru wejściowego określa się wartości z jakimi symulacje w danych eksperymencie mają zostać wykonane,
4. wykonanie zbioru symulacji na podstawie określonej przestrzeni parametrycznej oraz zebranie wyników ze wszystkich symulacji,
5. eksploracja zebranych danych wyjściowych na podstawie której zdobywa się informacje niezbędne do odpowiedzi na pytania stawiane w danym eksperymencie.

Ze względu na możliwy duży rozmiar przestrzeni parametrycznej eksperymentów typu "data farming", niezbędne jest użycie środowisk umożliwiających prowadzenie dużej ilości skomplikowanych obliczeń w sposób równoległy, np. środowisk gridowych i chmur obliczeniowych. W takim przypadku, niezbędnym może się okazać konieczność wsparcia użytkownika przez zintegrowaną platformę wirtualną, która oferować będzie niezbędną funkcjonalność do realizacji wszystkich etapów przedstawionego procesu. Przez platformę wirtualną rozumiany jest zestaw zintegrowanych usług programowych dostarczających pożądanej funkcjonalności w sposób przyjazny dla użytkownika. Poza dostarczaną funkcjonalnością, taka platforma powinna być na tyle elastyczna aby zostać użyta zarówno w niewielkich eksperymentach, które wymagają kilku komputerów, jak i do wykonywania eksperymentów dużej skali, w których zasoby obliczeniowe pojedynczego ośrodka obliczeniowego nie są wystarczające. Co więcej, przejście od wykonywania małych do dużych eksperymentów powinno odbywać się bez potrzeby manualnej rekonfiguracji platformy, co ma istotne znaczenie dla użytkowników, którymi mogą być naukowcy nie posiadający specjalistycznej wiedzy nt. architektur komputerowych czy wykorzystywanych zasobów obliczeniowych. Oprogramowanie, które umożliwia tak opisaną adaptację do rozmiarów rozwiązywanego problemu, nazywa się samoskalowalnym. Zapewnienie samoskalowalności usług programowych stanowi istotny problem badawczy coraz częściej dyskutowany w literaturze, w szczególności w kontekście elastycznych infrastruktur takich jak chmury obliczeniowe.

2 Teza i cele rozprawy

W rozprawie postawiono i wykazano następującą tezę:

Platformy wspierające eksperymenty typu "data farming" wymagają użycia heterogenicznej infrastruktury obliczeniowej oraz wsparcia dla automatycznego skalowania komponentów oprogramowania w celu uzyskania wysokiej efektywności zarówno względem czasu przetwarzania jak i kosztów realizacji.

Na podstawie tak postawionej tezy, zdefiniowane zostały następujące cele pracy:

- zaprojektowanie i zaimplementowanie masywnie samoskalowalnej platformy wspierającej poszczególne fazy eksperymentu typu "data farming" z wykorzystaniem heterogenicznej infrastruktury obliczeniowej,
- opracowanie zestawu reguł na potrzeby samoskalowania elementów opracowanej platformy zapewniających wysoką wydajności oraz minimalizujących koszt działania platformy.

Przyjęta metodyka badań zakładała zdefiniowanie wymagań funkcjonalnych i niefunkcjonalnych względem tworzonej platformy na podstawie określonych celów, implementację prototypu platformy, a następnie weryfikacje ustalonych wymagań.

Wymagania funkcjonalne platformy zostały zweryfikowane poprzez użycie jej do przeprowadzania rzeczywistych eksperymentów typu "data farming" z zakresu badania strategii postępowania jednostek porządkowych w scenariuszach asymetrycznych.

Wymagania niefunkcjonalne, w tym masywna samoskalowalność, zostały zweryfikowane przy pomocy zestawu testów syntetycznych symulujących wykonywania eksperymentów o różnej wielkości przestrzeni parametrycznej z wykorzystaniem różnych konfiguracji zasobów sprzętowych i różnych zestawów reguł skalowania.

3 Stan wiedzy

Metodyka prowadzenia badań „data farming“ została początkowo zaproponowana do zdobywania wiedzy nt. zjawisk lub procesów dla których opis analityczny jest nietrywialny. Pierwsze zastosowania dotyczyły scenariuszy militarnych, m.in. weryfikacji istniejących strategii militarnśych. Dzięki procesowi opartemu o wykorzystanie symulacji komputerowych, możliwe było iteracyjne odkrywanie nowych faktów nt. badanego zjawiska oraz ulepszanie modelów symulacyjnych [4, 5, 6].

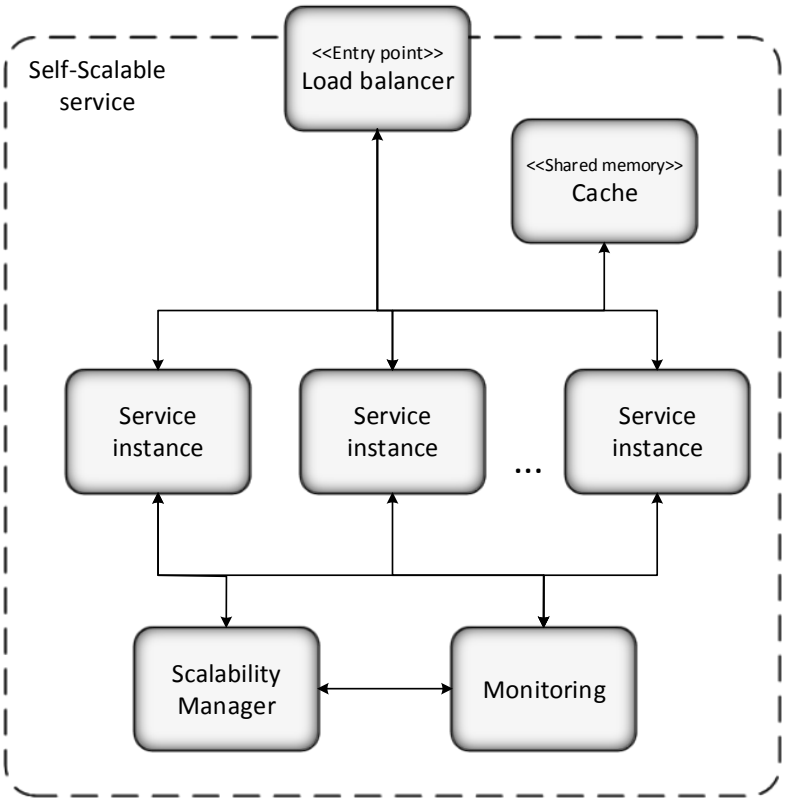
Istotnym aspektem prowadzenia takich eksperymentów stało się wykorzystanie infrastruktur obliczeniowych dużej skali do zarządzania uruchamianiem symulacji. W celu automatyzacji tego procesu, używa się dedykowanych narzędzi takich jak OldMcData [7], które umożliwia tworzenie plików konfiguracyjnych potrzebnych do uruchomienia symulacji z różnymi wektorami wartości wejściowych, nadzorowania wykonania zbioru symulacji przez zewnętrzny system wykonawczy i powiadomienia użytkownika o zakończeniu przetwarzania. OldMcData wykorzystuje system Condor [8] do uruchamiania zadań symulacyjnych na rozproszonych zasobach obliczeniowych. Użycie tego narzędzia ogranicza się do wsadowego przetwarzania kolejnych eksperymentów, pomiędzy którymi należy użyć zewnętrznego narzędzia do analizy danych wyjściowych. Dodatkowo, żadne metody samoskalowania części nadzorującej narzędzia do nie są dostarczane.

Problem skalowalności oprogramowania jest szeroko poruszany w literaturze światowej. W szczególności, dużą uwagę poświęca się procesowi budowy skalowalnych aplikacji sieciowych [9], baz danych relacyjnych [10] i nierelacyjnych [11]. Znacznie mniej uwagi poświęca się tworzeniu oprogramowania samoskalowalnego. W szczególności, brak jest tego typu platform wspierających eksperymenty typu "data farming". Przykładem samoskalowalnego rozwiązania komercyjnego wspierającego aplikacje biznesowe jest "eXtreme Appliaction Platform" (XAP) [12]. Celem platformy XAP jest zapewnienie skalowalności aplikacji, zarówno na poziomie usług jak i na poziomie dostępu do danych, poprzez wykorzystywanie niezależnych od siebie jednostek przetwarzających obejmujących cały stos aplikacyjny. XAP dostarcza wbudowanego modelu klastrowania, opartego o wysokowydajny, własnościowy podsystem komunikacji i składowania danych typu "in memory data grid". Skalowalność systemu zbudowanego w oparciu o XAP otrzymuje się poprzez uruchamianie niezależnych jednostek przetwarzających na różnych zasobach sprzętowych. Automatyzacja procesu skalowania jest zapewniana poprzez definiowanie kontraktów Service Level Agreement (SLA) obejmujących warunki, spełnienie których powoduje uruchomienie dodatkowych lub zatrzymanie już uruchomionych jednostek przetwarzania.

4 Samoskalowalne usługi i reguły skalowania

Na podstawie przeprowadzonej analizy istniejących rozwiązań jak również wymaganej funkcjonalności platformy wspierającej eksperymenty typu "data farming", zaproponowane zostały koncepcje dwóch elementów kluczowych do zapewnienia samoskalowalności takiej platformy:

- *Samoskalowane usługi* będące rozszerzeniem koncepcji usług występującej w architekturze zorientowanej na usługi (SOA), które grupują wiele instancji danej usługi wraz z dodatkowymi, dzielonymi, modułami i zarządza ich skalowaniem w sposób automatyczny. Głównym celem proponowanego typu usług jest zapewnienie samoskalowalności budowanej usługi w ustandaryzowany sposób już na etapie jej projektowania, poprzez dołączenie dodatkowych modułów umożliwiających jej monitorowania oraz skalowanie.
- *Reguły skalowania* stanowiące koncepcję uzupełniającą samoskalowalne usługi o możliwość



Rysunek 1: Struktura samoskalowalnej usługi.

reprezentacji reguł dot. skalowania usługi w ustandaryzowany sposób. Celem reguł skalowania jest umożliwienie elastycznego określenia sposobu w jaki samoskalowalna usługa ma być skalowana, w szczególności kiedy należy uruchamiać dodatkowe instancje usługi a kiedy je zatrzymywać w zależności od obciążenia używanych zasobów.

Struktura samoskalowalnej usługi została przedstawiona na Rysunku 1. W celu ujednolicenia sposobu dostępu do instancji usługi, wykorzystuje się moduł równoważący obciążenie (ang. load balancer), dzięki któremu zapewniona zostaje przezroczystość lokalizacji usługi niezależnie od faktycznej liczby uruchomionych instancji. Podsystem monitoringu gromadzi informacje dotyczące obciążenia poszczególnych instancji, które są wykorzystywane przez zarządcę skalowania (ang. scalability manager) do podejmowania decyzji dot. skalowania usługi. Pamięć podręczna (ang. cache) umożliwia przechowywanie rzadko zmieniających sięданych w formacie *klucz–wartość* w sposób zoptymalizowany do odczytu, które są dostępne dla wszystkich instancji usługi. Dodatkowo, instancje usługi mogą wykorzystywać pamięć podręczną do komunikacji zgodnie z modelem pamięci współdzielonej (ang. shared memory).

Reguła skalowania została zdefiniowana jako następująca krotka:

$$ScalingRule := < Metric, MeasurementType, Condition, Threshold, Action >$$

gdzie poszczególne elementy krotki oznaczają:

- $Metric$ wskazuje na mierzalny parametr usługi, np. czas odpowiedzi na dane zapytanie,

- $MeasurementType$ określa sposób pomiaru wskazanego parametru, np. pomiar prosty lub uśredniony ze wskazanego przedziału czasu,
- $Condition$ określa operator logiczny, który będzie używany do porównywania wartości parametru i wartości progowej określonej przez $Threshold$,
- $Threshold$ jest wartością progową wskazanego parametru, której przekroczenie powoduje uruchomienie akcji wskazanej przez $Action$,
- $Action$ jest identyfikatorem akcji skalowania, która określa sposób w jaki opisywana usługa ma być skalowana w górę lub w dół. Dana akcja będzie podjęta po spełnieniu warunku logicznego zdefiniowanego przez tę regułę.

Tak zdefiniowane reguły skalowania mogą być określane w stosunku do utworzonej samoskalowalnej usługi przez osoby nimi zarządzające, np. administratorów, a następnie przetwarzane w sposób automatyczny przez zarządcę skalowania danej samoskalowalnej usługi. Poprzez połączenie koncepcji reguł skalowania z samoskalowalnymi usługami możliwe zarządzania skalowaniem usług w elastyczny sposób, w szczególności określenie w jaki sposób i pod jakimi warunkami należy uruchamiać bądź zatrzymywać instancje usługi na zasobach obliczeniowych.

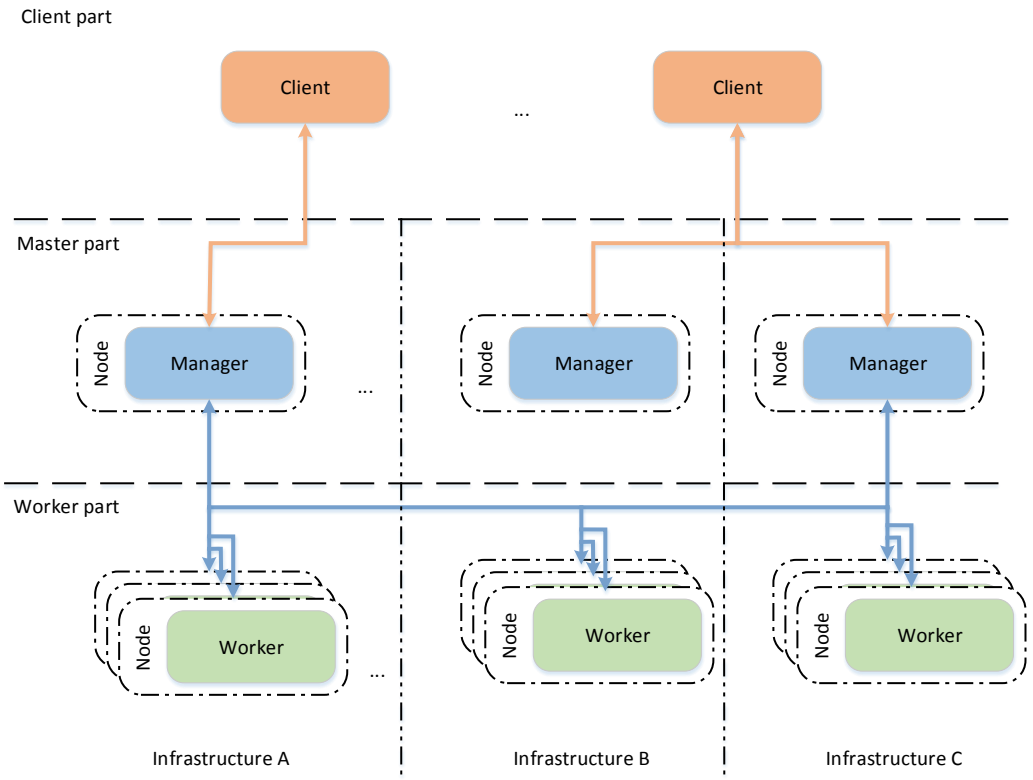
5 Masywanie samoskalowalna platforma dla eksperymentów typu "data farming"

W celu weryfikacji wprowadzonych koncepcji samoskalowalnych usług i reguł skalowania zaprojektowana i zaimplementowana została platforma o nazwie Scalarm [13]. Celem tej platformy jest wsparcie poszczególnych etapów eksperymentów typu "data farming" zaczynając od wyboru przestrzeni parametrycznej, poprzez wykonywanie symulacji, na eksploracji danych wynikowych kończąc. Dodatkowo, platforma umożliwia użytkownikom zarządzanie zasobami wykorzystywanymi do uruchamiania symulacji w ramach przeprowadzanego eksperymentu.

Architektura platformy Scalarm wykorzystuje wzorzec "master-worker", tzn. spośród usług wchodzących w skład platformy można wydzielić usługi zarządzające oraz usługi wykonawcze tak jak przedstawiono na Rysunku 2. Instancje poszczególnych usług mogą działać równolegle na różnych infrastrukturach obliczeniowych. Zasadniczą trudnością w skalowaniu tak zbudowanej platformy jest skalowanie jej części zarządzającej ze względu na konieczność utrzymania spójności stanu platformy, który może być zmieniany przez każdą z instancji usługi zarządzającej. Instancje usług wykonawczych są niezależne od siebie dzięki czemu warstwa wykonawcza skaluje się w sposób liniowy.

Scalarm składa się z zestawu usług współpracujących ze sobą tak jak zostało to przedstawione na Rysunku 3. Poza usługą zarządcy informacji (ang. Information manager), wszystkie usługi zaprojektowano wykorzystując wprowadzoną koncepcję usług samoskalowalnych. W skład platformy wchodzą następujące usługi:

- Zarządca eksperymentu (ang. experiment manager) obsługuje interakcję z użytkownikiem poprzez udostępniany graficzny interfejs użytkownika, oraz umożliwia tworzenie nowych i monitorowanie uruchomionych eksperymentów. Dodatkowo, zarządca eksperymentu decyduje o kolejności wykonania symulacji na podstawie przestrzeni parametrycznej danego eksperymentu.



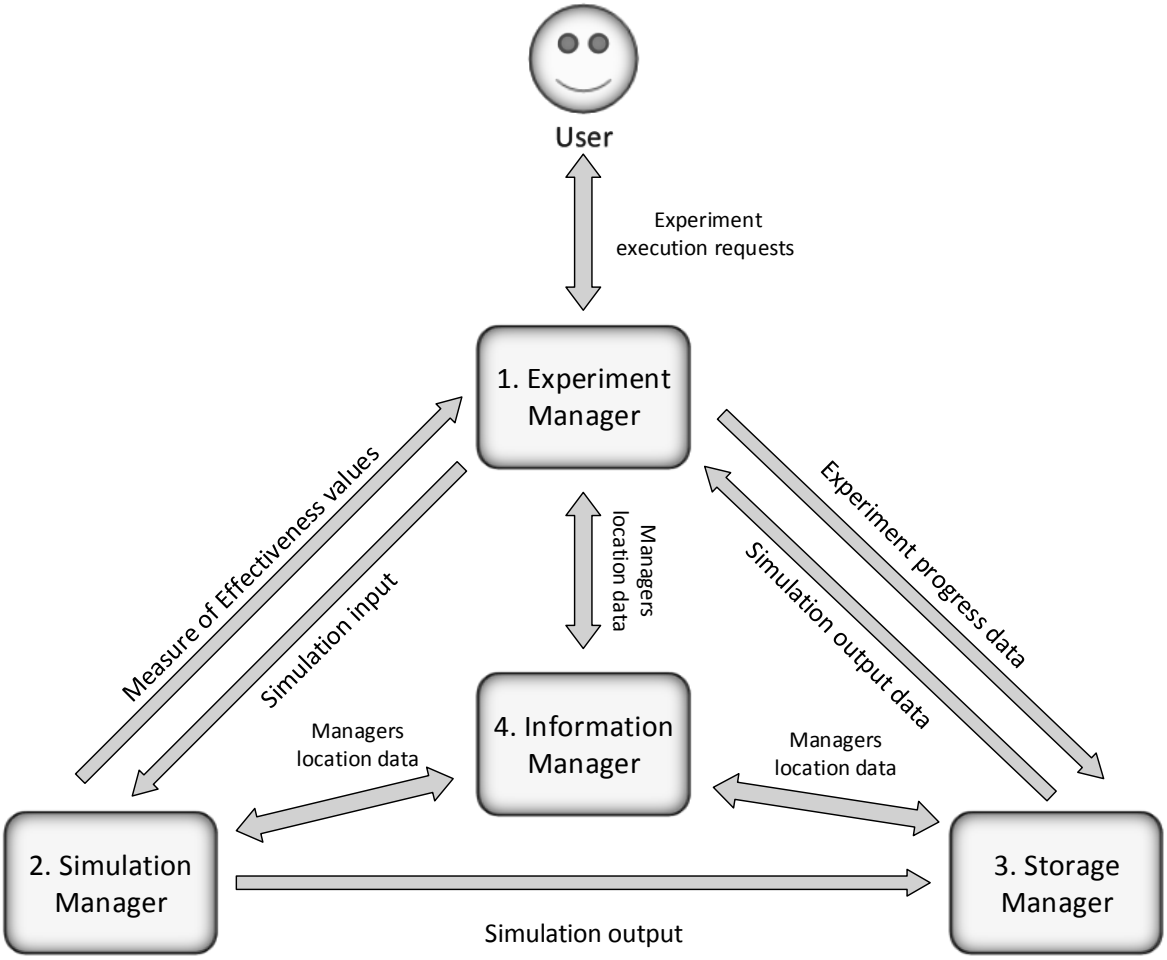
Rysunek 2: Struktura architektury platformy Scalarm.

- Zarządca symulacji (ang. simulation manager) jest implementacją koncepcji zadań pilotowych (ang. pilot jobs), tj. zajmuje się pozyskiwaniem zasobów obliczeniowych potrzebnych do uruchamiania symulacji w ramach eksperymentu. Po wykonaniu symulacji, zarządca symulacji pobiera kolejne wektory wejściowe (stanowiące punkty przestrzeni parametrycznej eksperyment), wykorzystując technikę "*pull*", za pomocą których wykonuje symulacje zamiast zwolnić pozyskane zasoby. Takie podejście zwiększa poziom wykorzystania zasobów obliczeniowych przez eliminację czasu potrzebnego na pozyskanie zasobów dla kolejnych symulacji.

- Zarządca pamięci masowej (ang. storage manager) dostarcza funkcjonalności związanej ze składowaniem wyników przeprowadzanych symulacji, zarówno w postaci strukturalnej jak i plików binarnych. Poprzez wykorzystanie modułu do równoważenia obciążenia, zarządca pamięci masowej umożliwia zunifikowany dostęp do fizycznie rozproszonych zasobów pamięci masowej.

- Zarządcy informacji (ang. information manager) stanowi rejestr usług platformy, implementujący wzorzec "Service locator". Przechowując informacje o lokalizacji poszczególnych usług, zarządca informacji zapewnia przezroczystość ich lokalizacji w stosunku do innych usług i klientów.

Istotną cechą prezentowanej platformy jest wsparcie heterogenicznej infrastruktury obliczeniowej w czasie wykonywania symulacji, umożliwiając przeprowadzanie eksperymentów, które wymagają mocy obliczeniowej przekraczającej możliwości pojedynczego ośrodka. W ramach prac zostało



Rysunek 3: Usługi składowe platformy Scalarm.

zaimplementowane wsparcie dla środowisk gridowych, na przykładzie infrastruktury PLGrid [14], publicznych chmur obliczeniowych, na przykładzie Amazon Elastic Compute Cloud (EC2) [15], oraz prywatnych zasobów, na przykładzie lokalnych klastrów obliczeniowych.

6 Weryfikacja działania platformy

Funkcjonalność platformy Scalarm została zweryfikowana w ramach projektu European Defence Agency (EDA) European Urban Simulation for Asymmetric Scenarios (EUSAS) [16]. Głównym celem projektu było wzbogacenie treningu służb porządkowych w scenariuszach rozgrywających się w środowiskach miejskich, gdy wrogie siły stosują niekonwencjonalne strategie i środki. W tego typu scenariuszach, służby porządkowe występują z reguły w niewielkich grupach mających kontakt z dużą liczbą cywili o różnych stopniu nastawienia, od neutralnego do wrogiego. Radzenie sobie z takimi sytuacjami wymaga specjalnego treningu i umiejętności.

Platforma Scalarm została wykorzystana do przeprowadzania eksperymentów typu "data farming", których celem była analiza zachowań i strategii żołnierzy w opisywanym typie scenariuszy. W ramach przeprowadzanych symulacji, cywile oraz żołnierze byli reprezentowani przez agentów opisywanych wieloma parametrami reprezentującymi m.in. cechy charakteru, np. zdolność do zachowań agresywnych, oraz stany emocjonalne, np. początkowy poziom agresji. Z uwagi na potencjalnie bardzo dużą przestrzeń parametryczną prowadzonych eksperymentów, zostały zaimplementowane następujące metody projektowania eksperymentów [17, 18]:

- plan kwadratów łacińskich, (ang. Near Orthogonal Latin Hypercubes – NOHL),
- plan kompletny (ang. full factorial),
- plan częściowy (ang. fractional factorial).

Do przeprowadzania symulacji i weryfikacji wsparcia heterogenicznej infrastruktury obliczeniowej, wykorzystano zasoby obliczeniowe pochodzące z trzech różnych typów infrastruktur:

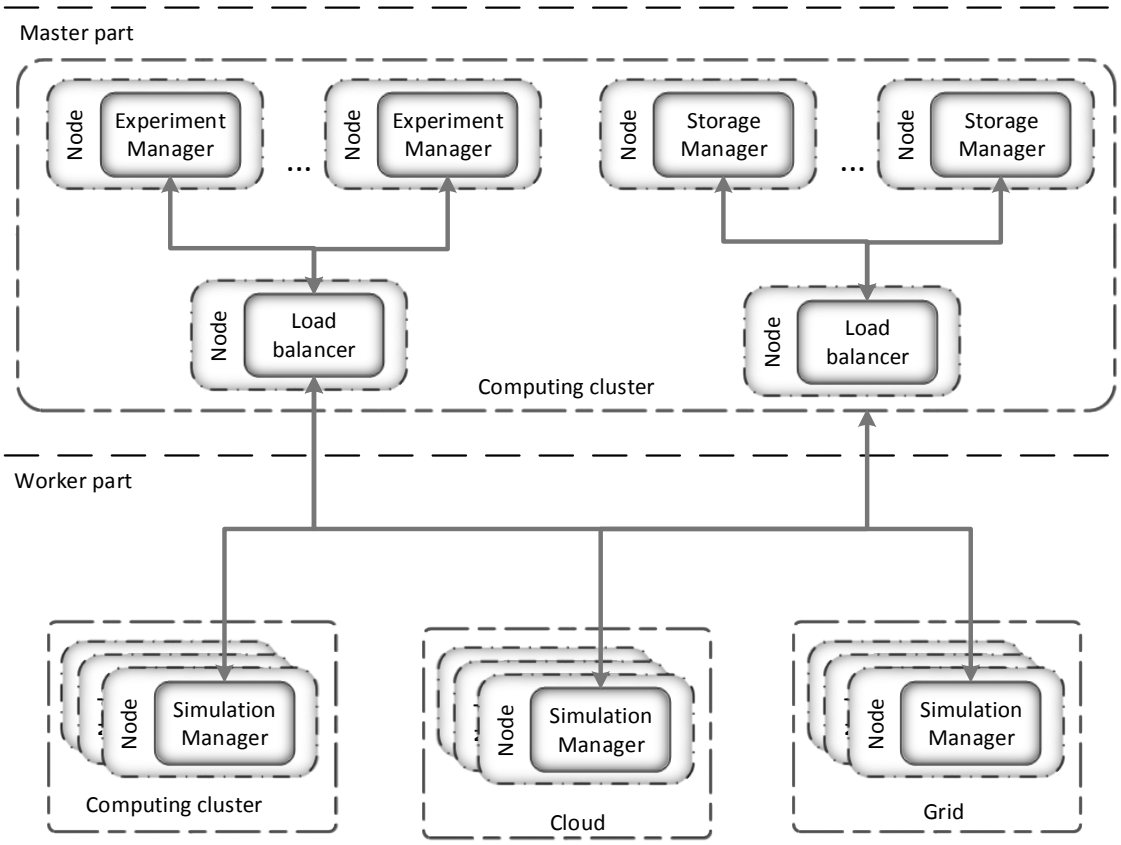
- 9 węzłów obliczeniowych pochodzących z lokalnego klastra,
- 50 zadań obliczeniowych uruchomionych w infrastrukturze gridowej PLGrid (w szczególności uruchomionych na klastrze "Zeus" ACK Cyfronet AGH),
- 50 maszyn wirtualnych typu High-CPU Extra Large uruchamianych w chmurze Amazon EC2.

W celu umożliwienia eksploracji rezultatów symulacji zaimplementowano następujące metody [19]:

- histogramy,
- drzewa regresji,
- wykresy punktowe.

Ostatnią ze zweryfikowanych funkcjonalności platformy była możliwość eksploracyjnego prowadzenia eksperymentów, tzn. możliwość rozszerzania przestrzeni parametrycznej prowadzonego eksperymentu w trakcie obliczeń na podstawie przeprowadzonej eksploracji uzyskanych wyników już zakończonych symulacji.

Do weryfikacji masywnej skalowalności i samoskalowalności platformy wykorzystane zostały testy syntetyczne, w ramach których uruchamiano testowe eksperymenty o różnej wielkości przestrzeni parametrycznej, wykorzystując do tego różne konfiguracje zasobów obliczeniowych odpowiadające ilości uruchomionych instancji usług. Na potrzeby badanie skalowalności platformy zostały użyte



Rysunek 4: Struktura środowiska dla testów masywnej samoskalowalności.

różne konfiguracje ilości instancji poszczególnych usług platformy jak przedstawiono to w Tabeli 1. Struktura środowiska testowego została przedstawiona na Rysunku 4.

Dla każdej z przedstawionych konfiguracji wykonano eksperymenty o następujących rozmiarach przestrzeni parametrycznych:

- 100 000,
- 200 000,
- 500 000,
- 1 000 000,
- 2 000 000,
- 5 000 000.

Każdy eksperyment był wykonywany dwukrotnie w celu zmniejsza wpływu losowych zaburzeń w używanym środowisku obliczeniowym. Głównym parametrem mierzonym w ramach przeprowadzonych testów był czas wykonania eksperymentu, tzn. czas wykonania wszystkich jego symulacji. Na

Tabela 1: Konfiguracje zasobów obliczeniowych użytych w trakcie testów.

Konfiguracja zasobów			
Experiment managers	Storage managers	Simulation managers	Etykieta
1	1	25	Configuration(1, 1)
2	2	50	Configuration(2, 2)
4	4	100	Configuration(4, 4)
8	8	200	Configuration(8, 8)

podstawie otrzymanych rezultatów obliczono szereg metryk określających skalowalność zbudowanej platformy, w tym podstawowej metryki przyspieszenia (ang. speedup) dla której uśrednione wyniki zostały zgromadzone w Tabeli 2.

Tabela 2: Uśredniona wartość przyspieszenia dla różnych wielkości eksperymentu w zależności od konfiguracji zasobów.

Wartości metryki przyspieszenia dla platformy Scalarm		
Etykieta Konfiguracja zasobów	Średnie przyspieszenie	Odchylenie standardowe
Configuration(2, 2)	1.78	0.29
Configuration(4, 4)	3.08	0.17
Configuration(8, 8)	5.80	1.04

W celu weryfikacji samoskalowalności platformy, wykonano testy polegające na przeprowadzaniu testowego eksperymentu o określonym rozmiarze przestrzeni parametrycznej przy zmieniającej się liczbie instancji zarządców symulacji w czasie. Tego typu testy zostały przeprowadzone dla następujących konfiguracji samoskalowalności platformy:

- brak określonych reguł skalowania,
- reguły skalowania określone dla zarządcy eksperymentu,
- reguły skalowania określone dla zarządcy eksperymentu i dla zarządcy pamięci masowej.

W przypadku konfiguracji z brakiem określonych reguł skalowania, cały czas działały 4 instancje zarządcy eksperymentu oraz 4 instancji zarządcy pamięci masowej. W przypadku konfiguracji z określonymi regułami skalowania, początkowa ilość instancji poszczególnych usług wynosiła 1 i była dynamicznie zmieniana w trakcie działania testu poprzez realizację reguł skalowania.

W czasie testów dokonano pomiaru wykonanych symulacji oraz ilości zużytych zasobów. Na podstawie ilości zużytych zasobów i koszcie zasobów obliczonych na podstawie oferty komercyjnej chmury Amazon EC2, został obliczony koszt wykonania każdego z testów oraz ilość testowych symulacji przypadających na 1$ - Tabela 3.

Tabela 3: Efektywność kosztowa dla różnych konfiguracji reguł skalowania.

Efektywność kosztowa platformy Scalarm			
Reguły skalowania	Ilość wykonanych symulacji	Koszt całkowity [$]	Ilość wykonanych symulacji dla $1
Brak	499 292	5.80	86 084
Określone dla zarządcy eksperymentu	375 951	4.12	91 250
Określone dla zarządcy eksperymentu i zarządcy pamięci masowej	454 059	4.76	95 390

7 Podsumowanie

Głównym celem pracy było zaprojektowanie i zrealizowanie masywnie samoskalowalnej platformy do przeprowadzania eksperymentów typu "data farming" z wykorzystaniem heterogenicznej infrastruktury obliczeniowej. W pracy autor zaproponował koncepcję samoskalowalnych usług oraz reguł skalowania, których wspólne użycie pozwoliło na implementację wymaganej platformy. Wykonane testy pozwoliły zweryfikować spełnienie wymagań zarówno funkcjonalnych jak i niefunkcjonalnych. Dodatkowo, na podstawie otrzymanych rezultatów potwierdzono prawdziwość tezy postawionej przez autora w rozprawie.

Wkład pracy badawczej autora obejmuje następujące elementy:

- koncepcja samoskalowalnych usług będących rozszerzeniem koncepcji usług SOA o wbudowane mechanizmy skalowania,
- koncepcja reguł skalowania umożliwiających określenie sposobu skalowania usług programowych w postaci reguł, które mogą być przetwarzane w sposób automatyczny,
- platforma Scalarm będąca kompletną platformą do przeprowadzania eksperymentów typu "data farming", zbudowana przy użyciu samoskalowalnych usług i wykorzystująca reguły skalowania.

Prowadzone prace badawcze stanowiły wkład do projektu EDA EUSAS A-0676-RT-GC JIP-FP Call 4, grantu Narodowego Centrum Nauki nr. 2012/05/N/ST6/03461 oraz projektu PLGrid Plus POIG.02.03.00-00-096/10. Wykonane testy zostały zrealizowane z wykorzystaniem infrastruktury ACK Cyfronet AGH oraz infrastruktury PL-Grid [20], w szczególności klastra "Zeus".

W trakcie prac, zindentyfikowane zostały interesujące możliwości dalszego rozwoju badań:

- możliwość automatycznego wykrywania efektywnych reguł skalowania na podstawie danych historycznych z podsystemu monitoringu,
- automatyczne dostosowywanie ilości instancji zarządcy symulacji na podstawie wymagań użytkownika,
- optymalizacja kosztu wykonywania symulacji w środowiskach chmur obliczeniowych,
- efektywniejszy dostęp do dużych zbiorów danych i uwzględnienie lokalizacji danych w czasie skalowania usług i wykonywania symulacji.

Literatura

[1] T. Hey, S. Tansley, and K. Tolle, eds., *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Redmond, Washington: Microsoft Research, 2009.

[2] X. Zhu and I. Davidson, *Knowledge Discovery and Data Mining: Challenges and Realities*. Hershey, PA, USA: IGI Publishing, 2007.

[3] D. Kallfass and T. Schlaak, "NATO MSG-088 case study results to demonstrate the benefit of using data farming for military decision support," in *Proceedings of the Winter Simulation Conference*, WSC '12, pp. 221:1–221:12, Winter Simulation Conference, 2012.

[4] G. E. Horne and K.-P. Schwierz, "Data farming around the world overview," in *Proceedings of the 40th Conference on Winter Simulation*, WSC '08, pp. 1442–1447, Winter Simulation Conference, 2008.

[5] G. E. Horne and T. E. Meyer, "Data farming: discovering surprise," in *Proceedings of the 36th conference on Winter simulation*, WSC '04, pp. 807–813, Winter Simulation Conference, 2004.

[6] T. Meyer and S. Johnson, *Visualization for Data Farming: A Survey of Methods*. Marine Corps Combat Development Command, United States Marine Corps Project Albert. Quantico, Virginia, 2001.

[7] S. Upton, "Users Guide: OldMcData, the Data Farmer, Version 1.1." `http://harvest.nps.edu/software.html`. Accessed: 21/03/2013.

[8] D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: The Condor Experience," *Concurrency - Practice and Experience*, vol. 17, no. 2-4, pp. 323–356, 2005.

[9] C. Henderson, *Building Scalable Web Sites: Building, Scaling, and Optimizing the Next Generation of Web Applications*. O'Reilly Media, Inc., 2006.

[10] A. Davies and H. Fisk, *MySQL Clustering*. MySQL Press, 2006.

[11] K. Chodorow, *Scaling MongoDB*. O'Reilly Media, Inc., 1st ed., 2011.

[12] "The GigaSpaces Runtime Environment website." `http://wiki.gigaspaces.com/wiki/display/XAP91/The+Runtime+Environment`. Accessed: 21/03/2013.

[13] D. Krol, M. Wrzeszcz, B. Kryza, L. Dutka, and J. Kitowski, "Massively Scalable Platform for Data Farming Supporting Heterogeneous Infrastructure," in *The Fourth International Conference on Cloud Computing, GRIDs, and Virtualization*, IARIA Cloud Computing 2013, (Valencia, Spain), pp. 144–149, 2013.

[14] PLGrid Plus project, "`http://www.plgrid.pl/en#section-1t`," last access 14 April, 2013.

[15] Amazon, "Amazon Elastic Compute Cloud." Online http://aws.amazon.com/ec2/, 2010.

[16] M. Kvassay, L. Hluchý, S. Dlugolinský, M. Laclavík, B. Schneider, H. Bracker, A. Tavčar, M. Gams, D. Król, M. Wrzeszcz, and J. Kitowski, "An integrated approach to mission analysis and mission rehearsal," in *Proceedings of the Winter Simulation Conference*, WSC '12, pp. 362:1–362:2, Winter Simulation Conference, 2012.

[17] A. Dean and D. Voss, *Design and Analysis of Experiments*. Springer Texts in Statistics, Spring-Verlag, 1999.

[18] J. Antony, *Design of Experiments for Engineers and Scientists*. Elsevier Science, 2003.

[19] B. Kryza, D. Krol, M. Wrzeszcz, L. Dutka, and J. Kitowski, "Interactive cloud data farming environment for military mission planning support," *Computer Science : rocznik Akademii Gorniczo-Hutniczej imienia Stanislawa Staszica w Krakowie*, vol. 13, no. 3, pp. 89–100, 2012.

[20] PL-Grid project, "`http://www.plgrid.pl/en`," last access 14 April, 2013.

A Informacje dodatkowe

A.1 Lista publikacji (wrzesień 2013)

1. T. Gubala, B. Balis, M. Malawski, M. Kasztelnik, P. Nowakowski, M. Assel, D. Harezlak, T. Bartynski, J. Kocot, E. Ciepiela, D. Krol, J. Wach, M. Pelczar, W. Funika, and B. Marian, "Virolab Virtual Laboratory," in Cracow'07 Grid Workshop : October, 2007, Krakow, Poland, pp. 35–40, 2007.

2. W. Funika, D. Harezlak, D. Krol, and M. Bubak, "Environment for Collaborative Development and Execution of Virtual Laboratory Applications," in Proceedings of the 8th International Conference on Computational Science, Part III, ICCS '08, (Berlin, Heidelberg), pp. 446–455, Springer-Verlag, 2008.

3. W. Funika, D. Harezlak, D. Krol, P. Pegiel, and M. Bubak, "User interfaces of the Virolab Virtual Laboratory," in Cracow'07 Grid Workshop : October, 2007, pp. 47–52, 2008.

4. M. Bubak, T. Gubala, M. Malawski, B. Balis, W. Funika, T. Bartynski, E. Ciepiela, D. Harezlak, M. Kasztelnik, J. Kocot, D. Krol, P. Nowakowski, M. Pelczar, J. Wach, M. Assel, and A. Tirado-Ramos, "Virtual Laboratory for Development and Execution of Biomedical Collaborative Applications," in Proceedings of the 2008 21st IEEE International Symposium on Computer-Based Medical Systems, CBMS '08, (Washington, DC, USA), pp. 373–378, IEEE Computer Society, 2008.

5. M. Bubak, T. Gubala, M. Malawski, B. Balis, W. Funika, T. Bartynski, E. Ciepiela, D. Harezlak, M. Kasztelnik, J. Kocot, D. Krol, P. Nowakowski, M. Pelczar, and M. Assel, "A platform for collaborative e-science applications," in Proceedings of 2nd ACC Cyfronet AGH users' conference, Zakopane, 2009, p. 36, ACC Cyfronet AGH, 2009.

6. J. Meizner, M. Malawski, E. Ciepiela, M. Kasztelnik, D. Harezlak, P. Nowakowski, D. Krol, T. Gubala, W. Funika, M. Bubak, T. Mikolajczyk, P. Plaszczak, K. Wilk, and M. Assel, "ViroLab Security and Virtual Organization Infrastructure," in Advanced Parallel Processing Technologies (Y. Dou, R. Gruber, and J. Joller, eds.), vol. 5737 of Lecture Notes in Computer Science, pp. 230–245, Springer Berlin Heidelberg, 2009.

7. D. Krol and W. Funika, "Semantic-based SLA-oriented performance monitoring in the ProActive environment," in Cracow'09 Grid Workshop : October, 2009, Krakow, Poland, pp. 151–157, 2010.

8. D. Krol, W. Funika, R. Slota, and J. Kitowski, "SLA-based data storage-oriented semi-automatic management of distributed applications applications," in KU KDM 2010 : Third ACC Cyfronet AGH user's conference : Zakopane March, 2010, p. 39, 2010.

9. D. Krol, W. Funika, R. Slota, and J. Kitowski, "SLA-Oriented Semi-Automatic Management of Data Storage and Applications in Distributed Environment," Computer Science, vol. 11, no. 1, 2010.

10. D. Krol, B. Kryza, K. Skalkowski, D. Nikolow, R. Slota, and J. Kitowski, "QoS provisioning for data-oriented applications in PL-Grid," in Cracow'10 Grid Workshop : October, 2010, Krakow, Poland, pp. 149–150, 2010.

11. D. Krol, R. Slota, and W. Funika, “Behaviour-inspired Data Management in the Cloud,” in Proc. of CLOUD COMPUTING 2010 The First International Conference on Cloud Computing, GRIDs, and Virtualization, IARIA CLOUD COMPUTING 2010, pp. 98–103, IARIA, 2010.

12. K. Skalkowski, J. Sendor, M. Pastuszko, B. Puzon, J. Fibinger, D. Krol, W. Funika, B. Kryza, R. Slota, and J. Kitowski, “SOA-based support for dynamic creation and monitoring of virtual organization,” in SOA infrastucture tools concepts and methods (e. a. S. Ambroszkiewicz, ed.), pp. 371–374, Poznan University of Economics Press, 2010.

13. W. Funika, P. Pegiel, P. Godowski, and D. Krol, “Semantic-oriented performance monitoring of distributed applications,” in KU KDM 2010 : third ACC Cyfronet AGH user’s conference : Zakopane March, 2010, pp. 33–34, 2010.

14. R. Slota, D. Krol, K. Skalkowski, B. Kryza, D. Nikolow, and J. Kitowski, "FiVO/QStorMan: toolkit for supporting data-oriented applications in PL-Grid", in KU KDM 2011 : fourth ACC Cyfronet AGH users’ conference : Zakopane, March, 2011, p. 68, ACK Cyfronet AGH, 2011.

15. S. Dlugolinsky, M. Kvassay, L. Hluchy, M. Wrzeszcz, D. Krol, and J. Kitowski, “Using parallelization for simulation of human behaviour,” in Proceedings of the 7th International Workshop on Grid Computing for Complex Problems, GCCP 2011, (Bratislava, Institute of Informatics SAS), pp. 258–265, 2011.

16. D. Krol, R. Slota, and W. Funika, “Behaviour-inspired Data Management in the Cloud,” International Journal on Advances in Intelligent Systems, vol. 4, no. 3 & 4, pp. 256–267, 2011.

17. D. Krol and J. Kitowski, “Distributed Storage Support in Private Clouds Based on Static Scheduling Algorithms,” in Proc. of CLOUD COMPUTING 2011 The Second International Conference on Cloud Computing, GRIDs, and Virtualization, IARIA CLOUD COMPUTING 2011, pp. 141–146, IARIA, 2011.

18. R. Slota, D. Krol, K. Skalkowski, M. Orzechowski, D. Nikolow, B. Kryza, M. Wrzeszcz, and J. Kitowski, "A Toolkit for Storage QoS Provisioning for Data-Intensive Applications", Computer Science, vol. 13, no. 1, 2012.

19. R. Slota, D. Nikolow, J. Kitowski, D. Krol, and B. Kryza, "FiVO/QStorMan Semantic Toolkit for Supporting Data-Intensive Applications in Distributed Environments", Computing and Informatics, vol. 31, no. 5, pp. 1003–1024, 2012, **current IF=0,254**.

20. K. Skalkowski, R. Slota, D. Krol, M. Orzechowski, B. Kryza, and J. Kitowski, "Towards scalable, semantic-based virtualized storage resources provisioning", in KU KDM 2012 : fifth ACC Cyfronet AGH user’s conference, pp. 76–77, 2012.

21. D. Krol, R. Slota, B. Kryza, D. Nikolow, W. Funika, and J. Kitowski, "Policy Driven Data Management in PL-Grid Virtual Organizations", in Remote Instrumentation for eScience and Related Aspects (F. Davoli, M. Lawenda, N. Meyer, R. Pugliese, J. Weglarz, and S. Zappatore, eds.), pp. 257–266, Springer New York, 2012.

22. D. Krol, A. Chrabaszcz, R. Slota, and J. Kitowski, "Evaluation of QStorMan dynamic storage provisioning strategies in PL-Grid", in Cracow’12 Grid Workshop : October, 2012, Krakow, Poland, pp. 81–82, 2012.

23. D. Krol, B. Kryza, M. Wrzeszcz, L. Dutka, and J. Kitowski, “Elastic Infrastructure for Interactive Data Farming Experiments,” Procedia Computer Science, vol. 9, no. 0, pp. 206 – 215, 2012. Proceedings of the International Conference on Computational Science, ICCS 2012.

24. D. Krol, M. Wrzeszcz, B. Kryza, L. Dutka, and J. Kitowski, “Scalarm: massively self-scalable platform for data farming,” in Cracow’12 Grid Workshop : October, 2012, Krakow, Poland (K. W. Marian Bubak, Michal Turala, ed.), pp. 53–54, Academic Computer Centre CYFRONET AGH, 2012.

25. B. Kryza, D. Krol, M. Wrzeszcz, L. Dutka, and J. Kitowski, “Interactive cloud data farming environment for military mission planning support,” Computer Science : rocznik Akademii Gorniczo-Hutniczej imienia Stanislawa Staszica w Krakowie, vol. 13, no. 3, pp. 89–100, 2012.

26. R. Slota, D. Krol, K. Skalkowski, B. Kryza, D. Nikolow, M. Orzechowski, and J. Kitowski, “A Toolkit for Storage QoS Provisioning for Data-Intensive Applications,” in Building a National Distributed e-Infrastructure PL-Grid (M. Bubak, T. Szepieniec, and K. Wiatr, eds.), vol. 7136 of Lecture Notes in Computer Science, pp. 157–170, Springer Berlin Heidelberg, 2012.

27. W. Funika, P. Godowski, P. Pegiel, and D. Krol, “Semantic-oriented performance monitoring of distributed applications,” Computing and Informatics, vol. 31, no. 2, pp. 427–446, 2012, **current IF=0,254**.

28. M. Kvassay, L. Hluchy, S. Dlugolinsky, M. Laclavik, B. Schneider, H. Bracker, A. Tavcar, M. Gams, D. Krol, M. Wrzeszcz, and J. Kitowski, “An integrated approach to mission analysis and mission rehearsal,” in Proceedings of the Winter Simulation Conference, WSC ’12, pp. 362:1–362:2, Winter Simulation Conference, 2012.

29. D. Krol, M. Wrzeszcz, B. Kryza, L. Dutka, R. Slota, and J. Kitowski, “Scalarm: scalable platform for data farming,” in KU KDM 2013 : Sixth ACC Cyfronet AGH user’s conference : Zakopane, February, 2013, p. 48, 2013.

30. K. Skalkowski, R. Slota, D. Krol, and J. Kitowski, “QoS-based storage resources provisioning for grid applications,” Future Generation Computer Systems, vol. 29, no. 3, pp. 713 – 727, 2013. Special Section: Recent Developments in High Performance Computing and Security, **current IF=1,864**.

31. D. Krol, M. Wrzeszcz, B. Kryza, L. Dutka, and J. Kitowski, “Massively Scalable Platform for Data Farming Supporting Heterogeneous Infrastructure,” in The Fourth International Conference on Cloud Computing, GRIDs, and Virtualization, IARIA Cloud Computing 2013, (Valencia, Spain), pp. 144–149, 2013, **Nagroda dla najlepszego artykułu na konferencji**.

A.2 Dane bibliometryczne (wrzesień 2013)

H-index (Web of science): 2
Cytowania (Web of science - całkowita liczba) : 11
Cytowania (Web of science - bez autocytowań) : 10

A.3 Inne

Recenzje artykułów na konferencjach i czasopismach:

- CLOUD COMPUTING 2011, The Second International Conference on Cloud Computing, GRIDs, and Virtualization, Wrzesień 25-30, 2011 - Rzym, Włochy
- CLOUD COMPUTING 2012, The Third International Conference on Cloud Computing, GRIDs, and Virtualization, Sierpień 22-27, 2012 - Nicea, Francja
- CLOUD COMPUTING 2013, The Fourth International Conference on Cloud Computing, GRIDs, and Virtualization, Maj 27 - Czerwiec 1, 2013 - Walencja, Hiszpania
- International Journal On Advances in Intelligent Systems, issn: 1942-2679,
- CS, Computer Science, Wydawnictwa AGH, Zeszyt Naukowy (kwartalnik)
- CAI Journal of Computing and Informatics, Slovak Academic Press Ltd., Bratislava. Czasopismo Listy Filadelfijskiej.