Symulacja w Badaniach i Rozwoju
Vol. 1, No. 3/2010

Tadeusz NOWICKI, Ewa WRZOSEK
Wojskowa Akademia Techniczna, ul. Kaliskiego 2, 00-908 Warszawa
E-mail: tadeusz.nowicki@wat.edu.pl, wardak@o2.pl

Modelowanie, symulacja i analiza systemów klasy klient-serwer

1 Wprowadzenie

Praca związana jest z badaniem własności systemów informatycznych, których praca polega na założonej, specyficznej formie współpracy między pewnymi komputerami, zwanymi klientami, oraz innymi, zwanymi serwerami. System klient – serwer [1], [2] pozwala na lepsze wykorzystanie pracy użytkownika przy zminimalizowanych kosztach. Systemy tej klasy bazują na funkcjonowaniu pojedynczych komputerów dużej mocy zwanych serwerami i licznego zbioru stacji roboczych zwanych klientami. Zazwyczaj podstawowa decyzja o architekturze systemu klient-serwer dotyczy tego, czy podstawowe procesy obliczeniowe są po stronie serwera (cienki klient), czy po stronie stacji klienckich (gruby klient).

Bardzo ważna jest decyzja o odpowiednim doborze modelu klient – serwer. Zły wybór, niedostosowany do rzeczywistego zapotrzebowania, może okazać się bardziej kosztowny, niż zapewnienie tego, by każdy komputer posiadał każdy zasób w dowolnej wielkości na własność. System taki okazuje się wtedy niewydajny, a poszczególne komponenty są nadmiernie, bądź niedostatecznie obciążone.

Poszukuje się zatem rozwiązania optymalnego, to znaczy takiej konfiguracji systemu klient – serwer, aby jego rzeczywiste działanie było zgodne z oczekiwaniami, wydajne oraz jak najtańsze.

Powstaje pytanie: jak bez podejmowania decyzji o organizacji fizycznych eksperymentów zbadać własności, w tym efektywność, danego systemu klient – serwer? Najlepszym w tym przypadku rozwiązaniem będzie zaprojektowanie i zaimplementowanie symulatora systemów klient – serwer, który umożliwi zbadanie i ocenę wskaźników efektywności systemów. W niniejszej pracy szczegółowo opisano budowę i funkcjonalność symulatora systemów klient – serwer oraz wykonano serię eksperymentów symulacyjnych dla ilustracji uzyskanych wyników.

Wyjaśniono to, co rozumie się pod pojęciem klient oraz serwer [1], [2]. Przedstawiono architekturę badanych systemów oraz rozłożenie aplikacji klient – serwer pomiędzy stacjami klienckimi, a maszynami serwerów. Wymieniono wskaźniki efektywności, które będą badane przy pomocy programu symulacyjnego.

Początkowo sprecyzowano granice symulatora, tzn. wybrano klasę systemów klient – serwer, których działanie ma być symulowane. Kolejnym punktem jest przedstawienie biblioteki symulacyjnej [3], na podstawie której działa symulator. Wymieniono te elementy biblioteki, które posłużyły do implementacji podstawowych elementów systemu klient – serwer oraz wymieniono możliwości pobierania statystyk z wykonanej symulacji. W dalszej części zaprezentowano projekt symulatora wraz z częściową implementacją niektórych klas.

2 Systemy klient-serwer

Sformułowanie klient – serwer jest związane z relacją między dwoma procesami bądź systemami. W tej relacji procesy są zorganizowane w system w taki sposób, aby jedne świadczyły usługi drugim. Klientem (ang. *client*) nazywamy proces zlecający zadania do wykonania na serwerze, zaś serwerem (ang. *server*) jest proces świadczący usługi.

Procesy klientów i serwerów mogą być zlokalizowane na tych samych, bądź innych komputerach. Nie jest wymagana odrębność sprzętowa. W rzeczywistych systemach proces może nie mieć ściśle określonej roli. Niejednokrotnie bywa tak, że pełni funkcję zarówno serwera, jak i klienta. Proces serwera może być klientem serwera innej usługi. Mimo tego wygodnie jest założyć, że badany proces, w zależności od kontekstu, pełni tylko jedną z ról.

Interakcja pomiędzy klientem i serwerem znana jest jako zachowanie typu zamówienie-odpowiedź (ang. *request-reply behavior*). Klient po zamówieniu usługi oczekuje odpowiedzi. Serwer pobiera napływające od klientów zlecenia, realizuje je, po czym wysyła odpowiedzi do klientów.

Podstawowe elementy systemu klient – serwer to:

- klienci,
- serwery,
- sieć komputerowa,
- wspólne protokoły sieciowe,
- wspólne zasady komunikacji,
- oprogramowanie.

Warto scharakteryzować pokrótce elementy systemów klient-serwer. Ich zrozumienie pomoże w prezentacji kolejnych wyników pracy.

Klient

Stacją roboczą (ang. *work station*) nazywamy komputer bądź terminal, na którym uruchamiane jest oprogramowanie z interfejsem użytkownika. Mianem klienta określamy najczęściej oprogramowanie. Rolą klienta jest udostępnienie użytkownikowi interfejsu, nawiązanie połączenia komunikacyjnego z serwerem, generowanie zleceń, przetwarzanie i wyświetlanie odpowiedzi.

Cechy charakteryzujące stacje klienckie, to:

- aktywność – klient jest inicjatorem działań,
- wysyłanie żądań do serwera,
- oczekiwanie odpowiedzi od serwera.

Ze względu na wielkość udziału klienta w implementacji interfejsu użytkownika, logiki aplikacji oraz składowania danych rozróżniamy dwa typy klientów:

Cienki klient (ang. *thin client*).

W tym przypadku mówimy o systemie cienki klient – serwer (ang. *thin client / server*). Klient ma niewielki udział w działaniu całego systemu. Cała logika aplikacji jest

przeniesiona na serwer, który przechowuje ponadto wszystkie dane potrzebne do obsłużenia żądań klientów.

Gruby klient (ang. *fat client*).

Stacja kliencka jest maszyną w pełni wyposażoną w urządzenia peryferyjne, z kompletnym oprogramowaniem umożliwiającym uruchamianie aplikacji. W tym modelu serwer służy do przechowywania danych i udostępniania ich na żądania klientów.

Serwer

Serwerem nazywamy program bądź system oprogramowania świadczący usługi na rzecz innych programów. Jest to definicja niekompletna, ponieważ serwerem często określa się komputer, bądź inny sprzęt fizyczny, świadczący usługi. Istnieje zatem podział na serwery sprzętowe i serwery oprogramowania.

Do ogólnych cech serwera można zaliczyć:

- pasywność – serwer czeka na zainicjowanie połączenia z klientem,
- czekanie na żądania klientów – jest tylko realizatorem przysłanych zleceń,
- automatyczność – w momencie otrzymania żądania serwer przetwarza je i wysyła odpowiedź.

Typowe serwery stosowane w praktyce profesjonalnych zastosowań, to:

- serwery plików,
- serwery baz danych,
- serwery nazw,
- serwery aplikacji,
- serwery internetowe.

Oprogramowanie klient-serwer

Aplikacja klient – serwer jest oprogramowaniem uruchamianym na komputerze klienta, która odwołuje się do zdalnego serwera. W aplikacji klient – serwer można wyróżnić trzy poziomy:

- interfejs użytkownika,
- poziom przetwarzania,
- poziom danych.

3 Wskaźniki efektywności systemów klient-serwer

Przed podjęciem badań efektywnościowych systemów klient – serwer należy określić wskaźniki efektywności, jakimi kierować się należy przy ocenie systemów tej klasy. Dowolna metoda badania systemów polega na ocenie wartości tych wskaźników (uzyskanych na przykład metodą eksperymentów symulacyjnych). Do wskaźników efektywnościowych odpowiadających systemom klasy klient-serwer zaliczyć można:

- czas obsługi przez serwer – jest to czas potrzebny serwerowi na przetworzenie pojedynczego zamówienia,
- obciążenie serwera – jest to procentowo wyrażony iloraz czasu aktywności serwera do całego czasu symulacji,
- obciążenie klienta – jest to procentowo wyrażony iloraz czasu aktywności klienta do całego czasu symulacji,
- czas oczekiwania na wysyłanie pakietu zadania/odpowiedzi – jest to czas, jaki musi minąć, zanim zostanie wysłany pakiet,
- czas przebywania zadania/odpowiedzi w kolejce – czas oczekiwania w kolejce na pewien zasób, którym może być serwer, zarządca bądź klient,
- czas odpowiedzi – jest to czas, po jakim na wysłane przez klienta zlecenie przychodzi odpowiedź; włączamy w to czas przebywania w sieci komunikacyjnej, (w co wliczamy czasy przebywania w kolejkach oraz czas obróbki przez zarządcę) oraz czas obsługi przez serwer,
- długość kolejki – jest to liczba zleceń przebywających w kolejce; przez długość kolejek badamy obciążenie sieci, ponieważ to właśnie stany kolejek obrazują obciążenie sieci komputerowej,
- liczba niezrealizowanych zadań – jest to liczba żądań, które przekroczyły założony, maksymalny czas odpowiedzi.

Należy przyjąć, że powyższe definicje nie mają charakteru ścisłego w tym sensie, że dotyczą wielu wystąpień każdej z charakterystyk. Ponadto, liczne z wielkości mają charakter stochastyczny. W tym sensie powinno się raczej mówić o oczekiwanych czasach, oczekiwanych długościach kolejek, oczekiwanych liczbach zadań, itd. Oczywiście, podane wyżej charakterystyki są jedynie wybranymi wskaźnikami służącymi ocenie funkcjonowania systemów klient-serwer. Można by wymieniać ich więcej, jednak są one wystarczającą propozycją dla opracowania metody badań efektywnościowych systemów tej klasy. Inne wskaźniki wzbogaciłyby jedynie listę parametrów efektywnościowych, jednak nie zmieniłyby sposobu badań systemu.

4 Symulator systemów klient-serwer

Program symulacyjny, odwzorowujący funkcjonowanie systemu klient-serwer, obejmuje działanie układu składającego się z klientów i generowanych przez nich zadań, sieci komunikacyjnej, zarządcy oraz serwerów. Warto rozpocząć rozważania od określenia tego, jaką architekturę systemu będzie reprezentował skonstruowany do badań model. Z uwagi na próbę zbadania efektywności podstawowego systemu, symulator będzie reprezentował architekturę dwupoziomowa. Mamy zatem do czynienia z klientami korzystającymi z usług serwerów, one zaś realizują zamówienia samodzielnie, bez używania obcych zasobów.

Wcześniej wprowadzono pojęcie cienkiego i grubego klienta. Balansowanie pomiędzy tymi dwoma koncepcjami systemu klient – serwer uzyskuje się dzięki możliwości ustawienia parametrów czasu tworzenia nowego zadania przez klienta oraz czasu przetwarzania zadania na serwerze. Zatem, zmieniając proporcje tych wielkości, uzyskuje się odpowiednio:

- model cienki klient–serwer, gdy czas generacji zadania, w porównaniu do czasu obsługi, jest bardzo krótki,
- model gruby klient–serwer, gdy czas obsługi jest znacznie dłuższy od czasu tworzenia zlecenia.

W rzeczywistych systemach klient – serwer sieć komunikacyjna pozwala na przesłanie pewnej porcji informacji zwanej pakietem. Jest to podstawowa jednostka nośnika informacji. W symulatorze istnieje możliwość generacji wielkości zadania oraz ustawienia rozmiaru pakietu. Należy wspomnieć również o tym, że symulowane łącze ma swoją szybkość, z którą przesyłane są pakiety.

W programie symulacyjnym zaimplementowano działanie serwera sekwencyjnego. Jest to najprostsza metoda obsługi zleceń. Niestety, nie jest zbyt efektywna, jeśli chodzi o długość czasu oczekiwania zadania na obsługę. Żądania muszą czekać w kolejce, gdy serwer jest zajęty. Zdecydowano się zatem na symulację tego typu serwera, ponieważ nadal jest dość popularny w implementacjach systemów klasy klient-serwer. Przykładowo, bardzo popularny obecnie serwer WWW jest serwerem sekwencyjnym. Żądania napływające przez sieć są obsługiwane w kolejności przybycia, jedno po drugim.

Zarządca zaimplementowany w programie symulacyjnym obrazuje logikę działania sieci komunikacyjnej. Pakiety w rzeczywistości niosą w sobie informację dotyczącą miejsca przeznaczenia. W pracy posłużono się tzw. zarządcą, który nadzoruje ruch w sieci. Dba o to, by pakiety trafiały do adresatów. Pełni również funkcję decyzyjną. Przypisuje zadaniom odpowiednich wykonawców i rozsyła je zgodnie z algorytmem przydziału zadań do serwerów.

Modelowany system będzie składać się z następujących elementów:

- klientów i serwerów,
- zarządcy,
- kolejek (w tym uwzględniana jest sieć komunikacyjna),
- zadań i algorytmów rozdziału zadań między serwery,
- generatorów liczb pseudolosowych.

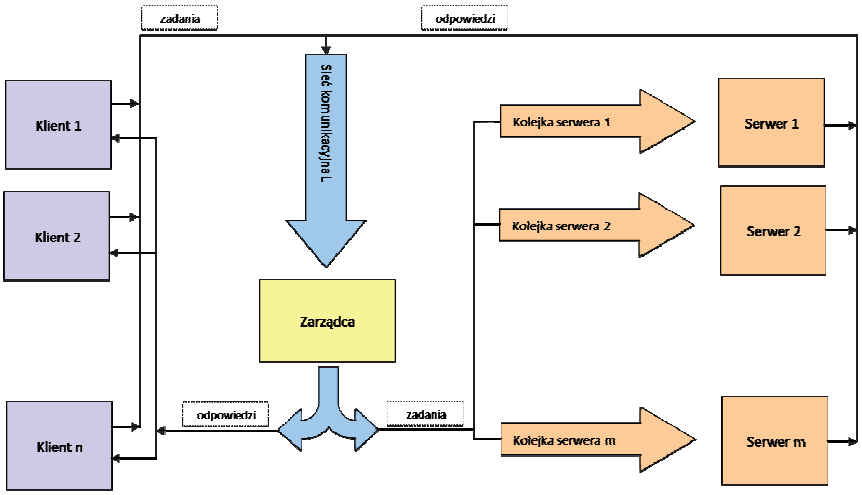
Zarządca (Manager) zajmuje się głównie rozdzielaniem zadań i odpowiedzi do kolejek odbiorców. Jego działanie, podobnie jak klienta, jest cykliczne. Zadania mogą być przydzielane do serwerów zgodnie z następującymi, popularnymi w implementacjach algorytmami przydziału zadań:

- algorytm „po kolei" - zadania są tu przydzielane do procesorów po kolei, na przykład zgodnie z umowną ich numeracją,
- algorytm „najkrótsza kolejka" – zadanie w tym algorytmie jest przydzielane do tego procesora, do którego w kolejce do realizacji czeka najmniejsza liczba zadań,
- algorytm „ważona najkrótsza kolejka" – zadanie w tym algorytmie jest przydzielane do tego procesora, do którego w kolejce do realizacji czeka najmniejsza liczba zadań, przy czym liczba zadań jest relatywnie

modyfikowana uwzględniająca moce obliczeniowe poszczególnych procesorów,

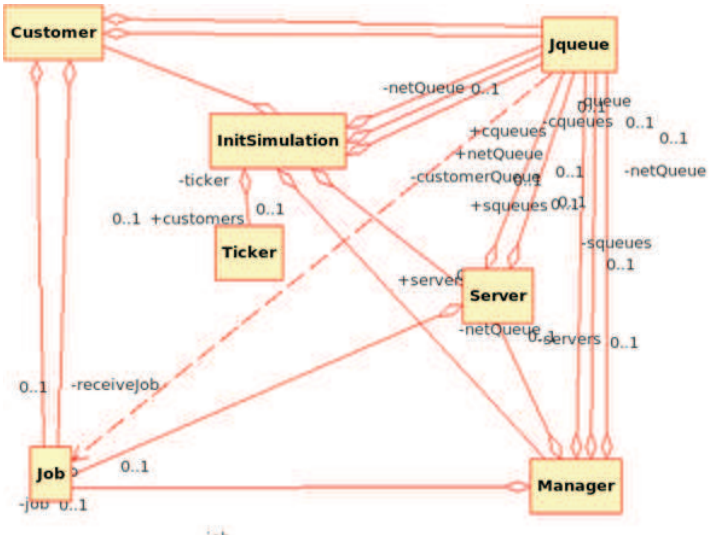
- algorytm „losowy" - zadania są w tym przypadku przydzielane są w sposób losowy (równomierny lub inny, w zależności na przykład od mocy procesorów),
- algorytm „zrównoważenie obciążenia serwerów" - zadania są w tym przypadku przydzielane do tych procesorów, które w ustalonym horyzoncie czasowym miały najmniejsze obciążenie.

Odpowiedzi serwerów zawierają w sobie informację, do jakiego klienta należą. Zgodnie z nią Zarządca umieszcza odpowiedzi w odpowiedniej kolejce. Wzajemny relacje pomiędzy elementami systemu przedstawia poniższy rysunek.



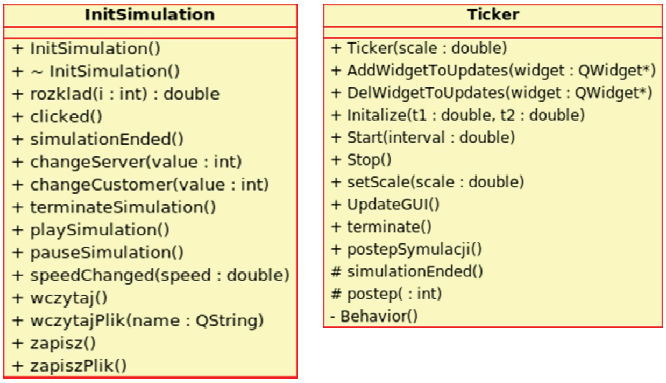
Rys.1. Schemat modelu systemu klient-serwer
Fig.1. Client-server system scheme

Projekt został stworzony za pomocą aplikacji Umbrello w języku UML [3]. Z uwagi na czytelność przedstawianych schematów diagram klas podzielono na części. W każdym schemacie umieszczone są tylko te informacje, które są istotne do zrozumienia danej funkcjonalności systemu (Rys.2).



Rys.2. Schemat klas w modelu systemu klient-serwer.
Fig.2.Client-server system class scheme

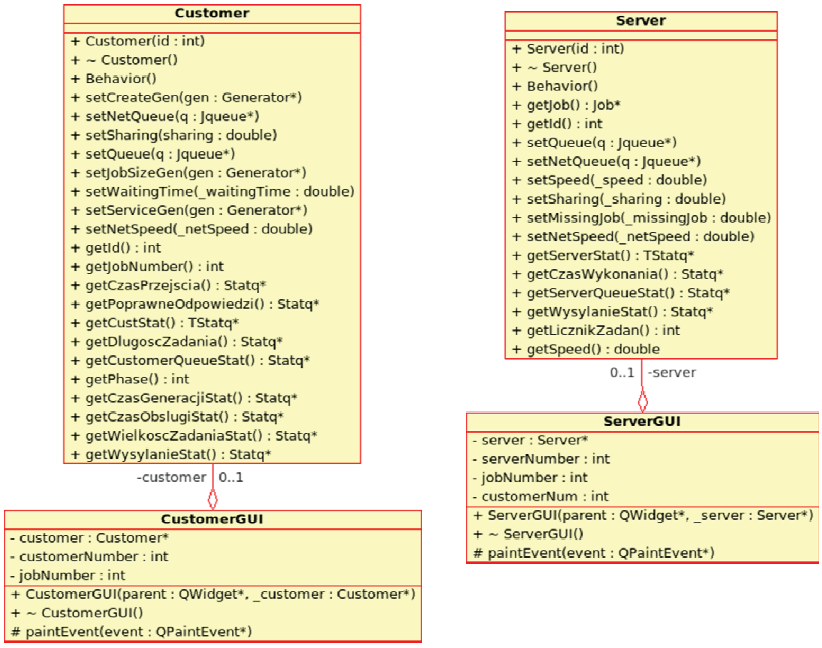
Poniżej przedstawione zostały skonstruowane klasy istotne z punktu widzenia samego procesu symulacji: klasa InitSimulaton i klasa Ticker.



Rys. 3. Klasa InitSimulation i klasa Ticker
Fig. 3. Init simulation class and Ticker class

Graficznym przedstawieniem działania klienta zajmuje się klasa CustomerGUI. Pobiera i wyświetla stan każdego klienta. Każdemu obiektowi klasy Customer przypisany jest jeden obiekt klasy CustomerGUI. Patrząc z kolei na listę metod klasy Server widzimy, że z wyjątkiem odziedziczonej metody Behavior zawiera tylko metody pobierające i

przekazujące informacje potrzebne w trakcie działania serwera. Zapewniają one poprawne działanie programu symulacyjnego. Obiekty klasy Server przez cały okres swojej aktywności pobierają zadania ze swoich kolejek, obsługują je i wysyłają gotowe odpowiedzi. Działanie serwera jest obrazowane dzięki graficznej klasie ServerGUI, która zajmuje się pobieraniem tych informacji, jakie trzeba wyświetlić w trakcie działania programu symulacyjnego (Rys.4).



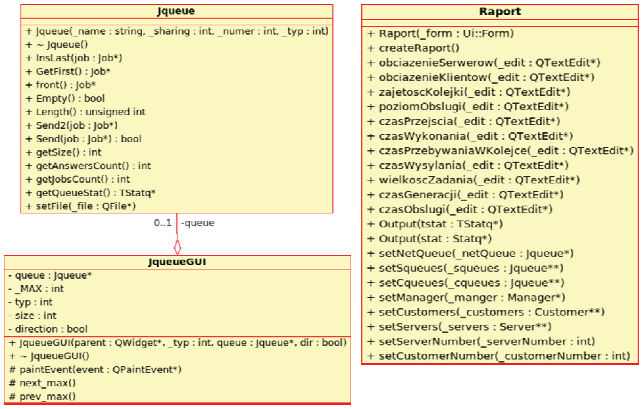
Rys. 4. Klasa Customer i klasa Serwer
Fig. 4.Customer class and Server class

Obiekty klasy Job służą jako nośniki informacji statystycznych i pomocniczych w przesyłaniu i kompletowaniu ich przez klienta, zarządcę oraz serwera (Rys.4). Zarządca (Manager) zajmuje się głównie rozdzielaniem zadań i odpowiedzi do kolejek odbiorców. Jego działanie, podobnie jak klienta, jest cykliczne. Zadania mogą być przydzielane do serwerów zgodnie z następującymi algorytmami przydziału zadań: ważona najkrótsza kolejka, losowy, zrównoważenie obciążenia serwerów, najkrótsza kolejka, po kolei. Odpowiedzi zawierają w sobie informację, do jakiego klienta należą. Zgodnie z nią Zarządca umieszcza odpowiedzi w odpowiedniej kolejce (Rys.5).

Rys. 5. Klasa Job i klasa Manager

Fig. 5. Job class and Manager class

Klasa Jqueue definiuje kolejki w systemie klient-serwer. Rozszerzono ją o metodę Send, która zajmuje się przesyłaniem pakietów. Każdemu obiektowi klasy Jqueue odpowiada obiekt klasy JqueueGUI, który zajmuje się wyświetlaniem stanu kolejki. Z kolei klasa Raport posiada informacje na temat wszystkich obiektów, z których ma pobierać statystyki. Zatem, w momencie tworzenia raportu metodą createRaport, można bezpośrednio odwołać się do obiektów i pobrać dane (Rys.6).

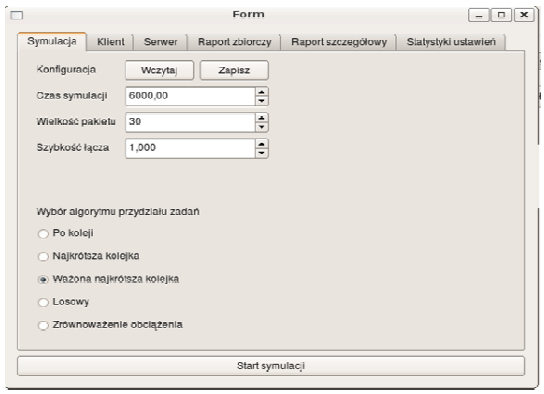


Rys. 6. Klasa Jqueue i klasa Raport

Fig. 6. Jqueue class and Raport class

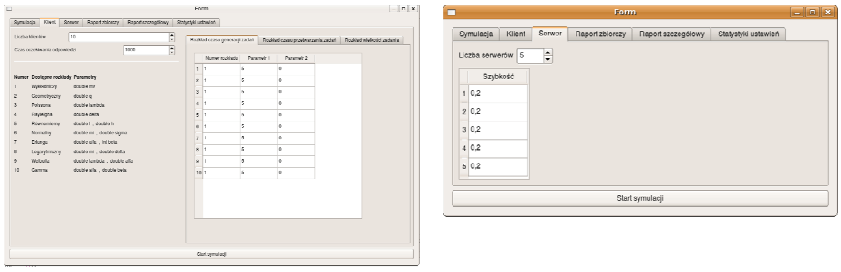
5 Implementacja symulatora systemu klient-serwer

Program symulacyjny stworzony został w oparciu o bibliotekę SIMLIB/C++, która jest projektem uczelnianym napisanym w języku C++ w Instytucie Informatyki i Inżynierii na Uniwersytecie Technicznym w Brnie. Prace nad nią trwają od 1990 roku i jest to projekt nadal rozwijany. Okno główne symulatora zostało przedstawione poniżej (Rys.7).



Rys. 7. Okno główne symulatora klient-serwer
Fig. 7. Main windows of client-server simulator

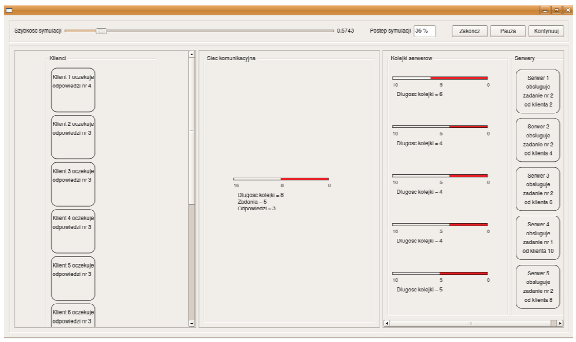
Można tu skonfigurować podstawowe charakterystyki eksperymentów symulacyjnych: czas symulacji, wielkość pakietu, szybkość łącza transmisji oraz wybrać można interesujący algorytm przydziału zadań. Poniżej przedstawiono możliwości w zakresie ustalania wartości parametrów klientów i serwerów systemu (Rys.8).



Rys. 8. Ustalanie wartości parametrów klientów i serwerów
Fig. 8. Client and server parameters configuration

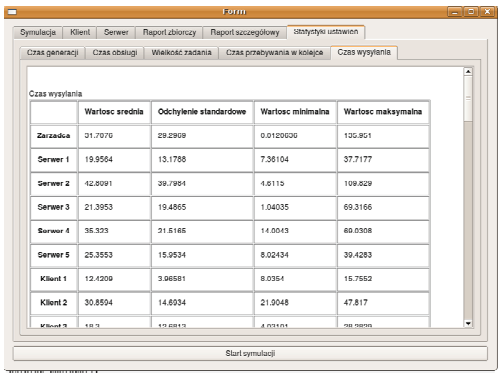
Symulację uruchamia się przyciskiem Start symulacji, który jest dostępny pod każdą z zakładek. W momencie naciśnięcia przycisku na ekranie pojawia się okno zobrazowania (animacji) procesu symulacji (Rys.9). Suwak prędkości symulacji służy

zwalnianiu lub przyspieszaniu symulacji. Z pierwszej opcji korzysta się, gdy ważny jest sam przebieg, a z drugiej, gdy interesujące są same wyniki symulacji. Wskaźnik postępu symulacji wyświetla to, ile procent tego procesu zostało już wykonane. Przycisk Zakończ powoduje natychmiastowe zakończenie symulacji i wygenerowanie raportów na podstawie zebranych danych. Przycisk Pauza chwilowo wstrzymuje przebieg symulacji. Przycisk Kontynuuj z kolei wznawia wcześniej zatrzymany przebieg symulacji.



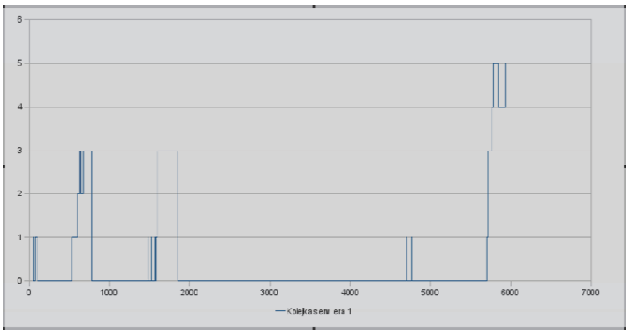
Rys. 9. Animacja procesu symulacji
Fig. 9. Simulation process animation

Raport szczegółowy składa się z oddzielnych zakładek, w których można odnaleźć następujące statystyki: zajętość kolejki, obciążenie serwerów, czas przejścia zadania przez system, czas wykonania zadania, poziom realizacji zadań, obciążenie klientów (Rys.10).

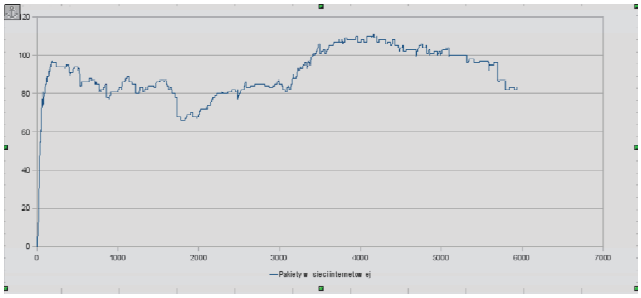


Rys. 10. Raporty z procesu symulacji
Fig. 10. Simulation process raporting

Podejrzeć też można różne charakterystyki dynamiczne, np. wykres zmian długości kolejki serwera w trakcie symulacji (Rys.11). Podobnie zarejestrować i zobrazować można wykres liczby pakietów zadań i odpowiedzi w sieci komputerowej (Rys. 12), jak też wiele innych, ważnych do oceny efektywności funkcjonowania systemów klasy klient-serwer, charakterystyk.



Rys. 11. Długości kolejek serwera
Fig. 11. Length of server queue



Rys. 12. Wykres liczby pakietów w sieci komputerowej
Fig. 12. Number of packages in computer network

6 Podsumowanie

Przedstawiono projekt i implementację symulatora programowego pozwalającego dla zadanych parametrów systemu klasy klient-serwer uzyskiwać oszacowania wartości wskaźników efektywności jego funkcjonowania. Pokazano też to, jak posługiwać się opracowanym symulatorem i możliwości analizy systemów klasy klient-serwer na podstawie wyników uzyskanych z eksperymentów symulacyjnych.

Uzyskane wyniki pozwalają na ocenę funkcjonowania algorytmów rozdziału zadań miedzy serwery, pozwalają również na ustalenie proporcji między grubym i cienkim klientem w sensie projektowym dla konkretnych zastosowań (aplikacji). Dzięki

odpowiednio skonstruowanym scenariuszom można również uzyskać ocenę wpływu wartości parametrów sieci komputerowej na pracę całego systemu.

Literatura

1. Tanenbaum A.,Van Steen M. *Systemy rozproszone Zasady i paradygmaty,* WNT, Warszawa 2006
2. Hall C. T*echniczne podstawy systemów klient – serwer,* WNT, Warszawa 1996
3. Strona biblioteki SIMLIB/C++ http://www.fit.vutbr.cz/~peringer/SIMLIB/,

Streszczenie

W pracy przedstawiono projekt i implementację symulatora programowego systemu klient-serwer. Omówiono charakterystyki takiego systemu. Zdefiniowano poszczególne elementy modelu systemu. Zaprezentowano możliwości symulatora w zakresie badań efektywności systemów klasy klient-serwer. Pokazano szczegółowe charakterystyki, jakie można uzyskać posługując się opracowanym symulatorem.

Modeling, simulation and analysis of client-server system

Summary

In this work project and implementation of software simulator of client-server system is presented. Selected characteristics of such system are discussed. Every elements of client-server model are defined. Possibilities of simulator from the system efficiency point of view are presented. Many characteristics which can be obtained during simulation experiments of client-server exploitation are shown.