Notatka z 3. spotkania roboczego

w ramach realizacji zadania WP 2.1
projektu SGI

1. **Data i miejsce spotkania:**
   - 2 lipca 2003 r. , Centrum Informatyczne TASK, Politechnika Gdańska, ul. G. Narutowicza 11/12, Gdańsk

2. **Uczestnicy:**
   - Ewa Politowska (TASK)
   - Bartosz Pliszka (TASK)
   - Rafał Tylman (TASK)
   - Michał Wróbel (TASK)
   - Maciej Brzeźniak (PCSS)

3. **Przebieg spotkania:**
   - Podsumowanie dotychczasowych prac:
     - raport z budowy instalacji testowej w PCSS i TASK
     - prezentacja działania systemu NetSolve w instalacji testowej
   - Dyskusja nad rozszerzeniem instalacji testowej
   - Dyskusja dotycząca oprogramowania, które ma być wcielone do systemu udostępniania.
   - Omówienie szczegółów mechanizmu adaptacji aplikacji użytkowników do korzystania z systemu udostępniania bibliotek (implementacji prekompilatora kodu źródłowego a podmiana biblioteki) i dyskusja.
   - Omówienie aplikacji testowych.

4. **Ustalenia**
   - **Instalacja testowa:**
     - Instalacja testowa obejmuje (1.07.2003) maszyny wymienione w tabeli 1. Schemat instalacji zawiera rysunek 1.
     - Instalacja testowa zostanie rozszerzona i obejmie dodatkowo maszyny wymienione w tabeli 2. Rozszerzenie instalacji testowej – lipiec 2003.
   - W pierwszym etapie prac, mechanizmy służące do adaptacji istniejących aplikacji użytkowników do korzystania z systemu udostępniania bibliotek matematycznych testowane będą dla bibliotek BLAS i LAPACK. W dalszym etapie mechanizmy zostaną sprawdzone dla innych bibliotek naukowych używanych w centrach obliczeniowych w Polsce.
   - **Pod uwagę brane są dwa warianty realizacji mechanizmu adaptacji:**
     a) prekompilator kodu źródłowego
     b) podmiana biblioteki matematycznej

     W toku dyskusji omówiono szczegóły oraz wady i zalety poszczególnych podejść. W pierwszym przypadku, w kodzie źródłowym aplikacji użytkownika wywołania lokalne funkcji matematycznych podmieniane są na wywołania zdalne. Użytkownik ponownie kompiluje otrzymany wskutek translacji kod i łączy go z biblioteką klienta systemu udostępniania (NetSolve) i ew. z lokalną biblioteką matematyczną (w celu

umożliwienia realizacji mechanizmu awaryjnego wywołania lokalnego).

Zalety rozwiązania:
- możliwość wyboru (przez translator i/lub użytkownika), które wywołania funkcji matematycznych podmieniać na zdalne;
- możliwość używania różnych metod zdalnego wywoływania funkcji przez bardziej świadomych użytkowników – synchroniczne, asynchroniczne, wywołania grupowe itp.;
- możliwość określania rozmiarów parametrów (tablic itp.) wywołania zdalnego funkcji (tam gdzie nie są one jawnie wyspecyfikowane przez inne parametry) na podstawie analizy zawartości kodu poprzedzającego wywołanie.

Wady:
- konieczność analizy składniowej kodu aplikacji przy podmianie wywołań (w celu zachowania jego poprawności składniowej po podmianie).

W drugim przypadku kod źródłowy aplikacji użytkownika nie podlegałby translacji. Podmianie podlegałby kod biblioteki matematycznej – zamiast z oryginalną, lokalną, użytkownik łączyłby swoja aplikację ze zmodyfikowaną biblioteką, zawierającą zamiast oryginalnych funkcji matematycznych „wrappery", które wywoływałyby zdalne funkcje przez system udostępniania lub lokalne funkcje biblioteki matematycznej (w zależności od dostępności systemu udostępniania).

Zalety:
- prostota i elegancja rozwiązania (użytkownik jedynie relinkuje swoją aplikacje z inną biblioteką),
- brak konieczności analizy składniowej kodu aplikacji.

Wady:
- brak możliwości selekcji funkcji, które mają być wołane lokalnie i zdalnie (wszystkie albo żadna), brak możliwości wprowadzenia narzędzia interaktywnego dla użytkownika,
- dostęp tylko do informacji zawartych w parametrach aktualnych wywołania funkcji (niemożność pozyskania informacji o rozmiarach tablic/ wektorów z kodu aplikacji poprzedzającego wywołanie funkcji).

W tej chwili PCSS i TASK uznaje pierwsze rozwiązanie za bardziej sensowne. Jednak kwestia wymaga dalszej analizy i rozważenia.

° PCSS i TASK przygotują aplikacje testowe pozwalające sprawdzić rozwijane mechanizmy przy użyciu bibliotek BLAS (TASK) i LAPACK (PCSS).

5. Termin następnego spotkania

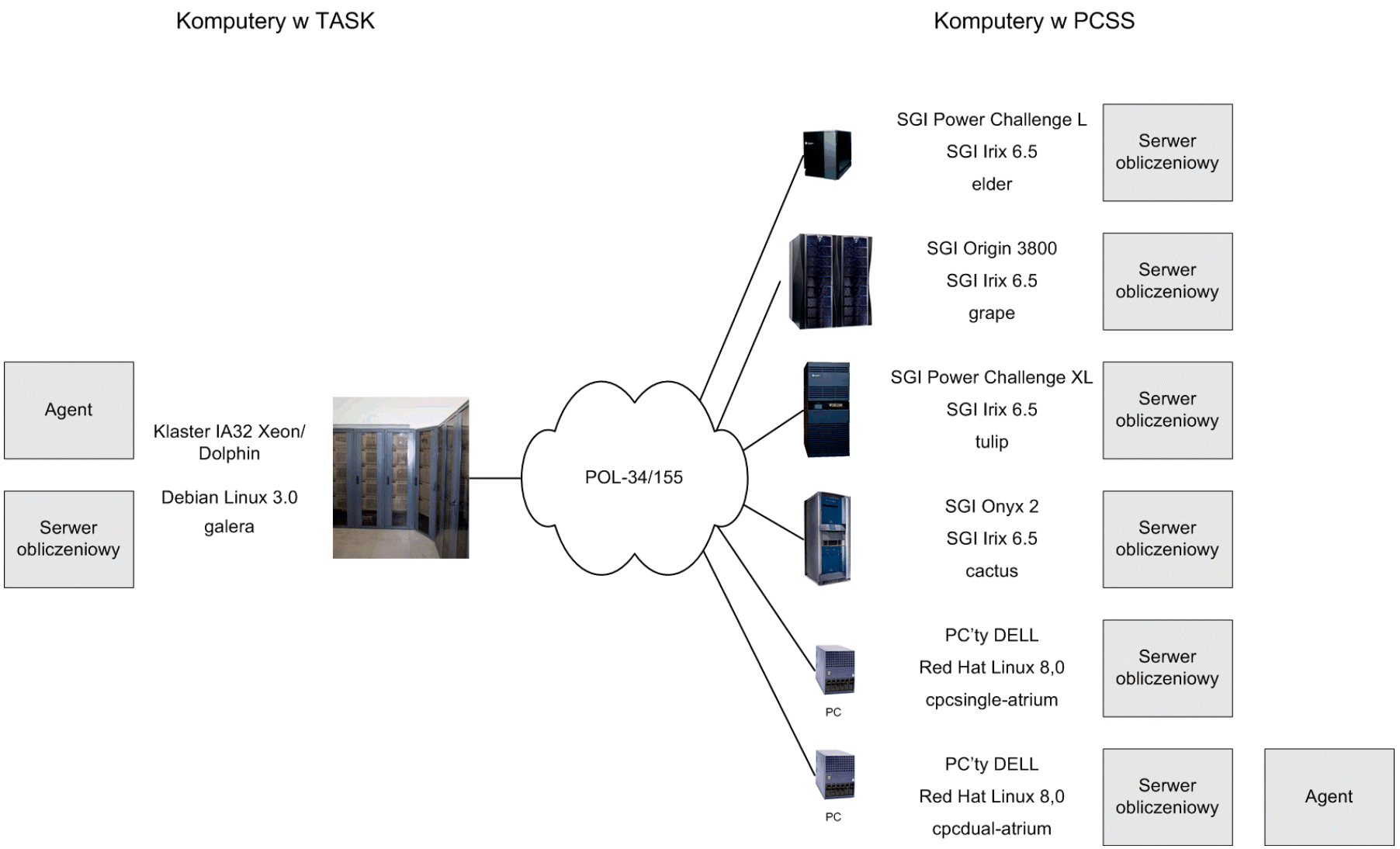
° koniec lipca 2003 r.

Ośrodek	Komputer	System operacyjny	nazwa domenowa maszyny	Liczba procesorów	Typ procesora	RAM
zainstalowane:						
PCSS	SGI Origin 3800	Irix 6.5	grape.man.poznan.pl	128	R12000, 400 MHz	80 GB
PCSS	SGI Onyx 2	Irix 6.5	cactus.man.poznan.pl	8	R10000, 185 MHz	6 GB
PCSS	SGI Power Challenge 10000 L	Irix 6.5	elder.man.poznan.pl	4	R10000, 195 MHz	384 MB
PCSS	SGI Power Challenge XL	Irix 6.5	tulip.man.poznan.pl	12	R8000, 90 MHz	1 GB
PCSS	Dell Power Edge 6300	Red Hat Linux 8.0	cpcsingle-atrium.man.poznan.pl	1	Intel Xeon 500 MHz	256 MB
PCSS	Dell Power Edge 6300	Red Hat Linux 8.0	cpcdual-atrium.man.poznan.pl	2	Intel Xeon 500 MHz	512 MB
TASK	Klaster IA32 Xeon/Dolphin	Debian Linux 3.0	galera.task.gda.pl.man.poznan.pl	128	Intel Xeon 700 MHz	16 GB
prace w toku:						
PCSS	Cray T3E	UNICOS/mk	lotus.man.poznan.pl	8	450 MHz	1 GB
PCSS	Cray SV1	UNICOS 10.0.0.7	croton.man.poznan.pl	8	SV1 300 MHz	16 GB
PCSS	Cray J916	UNICOS 8.04	pink.man.poznan.pl	16	C90	4 GB

Tabela 1. Lista komputerów w instalacji testowej

Ośrodek	Komputer	System operacyjny	nazwa domenowa maszyny	Liczba procesorów	Typ procesora	RAM
PCSS	SGI Challenge L	Irix 6.5	melisa.man.poznan.pl	4	MIPS R4400, 150 MHz	384 MB
PCSS	IBM/SP2	AIX 4.1.4	clover.man.poznan.pl	14+1 węzłów	POWER2	64MB/węzeł
TASK	SGI Origin 2000 + Onyx 2	Irix 6.5	fregata.task.gda.pl	8+16	8x MIPS R1000 195 MHz + 16x MIPS R12000 300 MHz	16 GB
TASK	SGI Octane	Irix	pancernik.task.gda.pl	2	MIPS R12000 400 MHz	512 MB

Tabela 2. Lista komputerów, o które rozszerzona zostania instalacja testowa



Rysunek 1. Schemat instalacji testowej (stan na 1 lipca 2003)