Keuze van communicatiemethode

(Gebruikersapplicatie <--> Agent)

Motivatie

Op een gegeven moment zullen de gebruikersapplicatie en de agent gegevens met elkaar kunnen uitwisselen. Performantie is heel belangrijk bij de profiler, en dus moet de communicatie voor zo weinig mogelijk overhead zorgen. Dit zal vooral in de tweede fase van het project belangrijk worden, wanneer er een momentopname van de JVM moet weergegeven worden. Hier zal constant een flow van gegevens van de agent naar de viewer bestaan.

Referentiesysteem

De tests werden uitgevoerd op een systeem met:

- 1.6 Ghz. mobile processor.
- 512 Mib RAM.
- Linux met ReiserFS als filesystem.

Oplossingen & tests

We beschouwen 3 mogelijkheden:

- Shared memory
- Sockets
- Een Pipe

Shared memory werd nog niet onderzocht. Deze optie blijft dus nog even open.

De de volgende testen wordt de tijd bepaald die nodig is om 4 Gib. data te versassen dmv. sockets of een pipe. Elke test wordt 3x doorlopen.

De tests van de sockets maken gebruik van een client en een serverprogramma. Ze zijn beide in java geschreven. De server wacht tot de client een verbinding maakt, en begint dan direct van de socket te lezen. De client maakt een verbinding via een opgegeven poort op “localhost” en verstuurt de 4 Gib. aan data.

TCP	***UDP***
108.969 ms.	56.618 ms.
111.297 ms.	56.435 ms.
111.317 ms.	56.418 ms.

In de test van de pipe zal een java programma een tweede programma opstarten, dat 4 Gib. aan data op de “standard out” zet. Deze stream wordt dan via de pipe door de client terug ingelezen. Er worden twee testen gedaan. In de eerste plaats zal de data-generator in

C geschreven zijn en van “stdio.h” gebruik maken. In het andere wordt C++ gebruikt en wordt er geschreven via “iostream”.

c	***c++***
87.744 ms.	998.315 ms.
87.622 ms.	998.315 ms.
87.778 ms.	998.336 ms.

Datagramsockets komen duidelijjk als winnaars uit de bus. Verder dienen ook nog de verschillende voor en nadelen in rekening worden gebracht.

Pipe:

Voordelen
- Zoizo aanwezig, als de gebruikersapplicatie de JVM start
- Sneller dan TCP
- Full Duplex

Nadelen
- De geteste applicatie en JVM kunnen ook op stdout schrijven. De gegevens worden dan grondig vervuild.
- Zou eigenlijk vrijgehouden moeten worden om het testprogramma aan te sturen, en de uitvoer weer te geven.

TCP Sockets:

Voordelen
- Geïsoleerde sessie tussen de Agent
- Makkelijk in gebruik
- Full Duplex
- Maar 1 poort nodig om nieuwe verbindingen te maken

Nadelen
- Trager dan Pipe en veel trager dan UDP

UDP Sockets:

Voordelen
- Enorm snel
- Op de localhost zullen de pakketten niet uit volgorde of verloren gaan (ook getest)

Nadelen
- Externe bronnen kunnen ook op een een UDP poort sturen. Identificatie van pakketten is noodzakelijk.
- One way traffic (per poort)

Conclusie

Aangezien de pipe wegvalt, zal de keuze tussen een UDP of TCP socket zijn. Als er veel data doorgegeven moet worden zal er een UDP socket nodig zijn. In het andere geval zal een TCP socket volstaan. Hier is ook iets makkelijker mee te werken.