Implementação de um servidor replicado

TFD08 - Rui Guerra nº25268 Sandra Lessa nº26613 Tiago Almeida nº25868

Resumo

Cada vez mais a informática é baseada em sistemas distribuídos, dispersos geograficamente e assentes sobre vários níveis de fiabilidade. Com a crescente importância das aplicacações distribuídas surgem problemas como garantir que uma falha não prejudique o sistema.

A utilização do Appia possibilita a criação de um servidor replicado através de pilhas protoculares que fornecerão um detector de falhas e a esperada tolerância a faltas.

1 Introdução

Com o crescente desenvolvimento das novas tecnologias, os sistemas distribuídos tornaram-se importantes e essenciais em sistemas de larga escala. Um dos aspectos fundamentais é a tolerância a faltas que estes sistemas devem fornecer. No ambito da cadeira de Tolerância a Faltas Distribuídas teremos de implementar um servidor replicado num sistema distribuído e tolerante a faltas.

A plataforma utilizada, o Appia suporta a composição de protocolos, permitindo através de uma configuração eficiente, a criação de um sistema tolerante a faltas.

Uma das formas de concretizar este tipo de aplicações é utilizando chamadas a procedimentos remotos (RPC). Este modelo, tambem conhecido por cliente/servidor, funciona enviando um pedido pela rede a um servidor, que o executa e envia a resposta ao cliente. Este modelo tem problemas relacionados com a falha do servidor, se o servidor sofre algum problema, todo o sistema pode falhar. Assim com o recurso à replicação da informação, podemos obter um sistema tolerante a faltas pelo que este não será afectado pela perda de uma máquina ou de um serviço.

2 Arquitectura

O projecto proposto vai ser desenvolvido sobre o Appia[4], um sistema de comunicação baseado em micro-protocolos desenvolvido na Faculdade de Ciências. O sistema pode ser dividido em 2 partes: clientes e servidores. Os clientes comunicam com um dos servidores replicados usando comunicação ponto-a-ponto. Assim os servidores possuem uma camada especifica da aplicação(Server) e uma camada genérica responsavel por entregar o pedido a 1 réplica e tentar distribuir de modo equilibrado os pedidos pelas diferentes réplicas. Os clientes possuem duas camadas que implementam a parte genérica do sistema (RRPC) e a parte específica da aplicação cliente (Client). Estas camadas ficam em cima de uma pilha de protocolos que descreveremos mais abaixo.

2.1 Composição de protocolos

Uma das facilidades do Appia é a composição de protocolos. Com isto é possível definir com facilidade uma QoS e organizar de um modo estruturado a composição de protocolos .Uma das preocupações foi a utilização das camadas que dão suporte aos grupos e filiações. Como meio de controlar o correcto envio/recepção de mensagens por parte dos processos, foi adicionada a camada FIFO, pois vamos usar udp, e como se sabe este não garante a propriedade FIFO ao contrário do tcp. Como já existe suporte a grupos, e este garante a sincronizacão virtual, só teremos de garantir que as mensagens são enviadas e recebidas pela ordem FIFO. Assim todos os processos correctos recebem as mensagens na ordem correcta. Descrevemos em seguida a pilha de protocolos usada em comum por parte dos servidores e clientes:

- **Udp Simple** - Interface para o protocolo de transporte UDP

- **FIFO** - Fornece comunicação ponto-a-ponto fiável e entrega Multicast Fifo para SendableEvents
- **Bottom** - Interface entre as camadas de comunicação em grupo e as camadas ponto a ponto inferiores
- **Suspect** - Este protocolo implementa um detector de falhas que iremos utilizar.
- **Intra** - Protocolo responsável pela correcta manutenção das vistas
- **Stable** - Garante que todas as mensagens recebidas por qualquer membro vivo, são recebidas por todos os membros vivos.
- **Leave** - Remove um elemento do grupo
- **Sync** - Este protocolo garante que todos os participantes do grupo receberam todas as mensagens antes de uma mudança de vista

3 Modelo

Nesta secção descrevemos o modelo que pensamos concretizar para obter tolerância a faltas no nosso projecto. Existem vários modos de replicação, e nesta secção vamos descrever as 2 mais importantes. Vamos também descrever em seguida os componentes em que se baseia o nosso sistema.

3.1 Primitivas de comunicação

1. **Comunicação ponto-a-ponto** - Permite a comunicação entre clientes e servidores usando udp
2. **Comunicação em grupo** - Permite a comunicação entre os elementos do grupo usando Broadcast ou Multicast
3. **Sincronia Virtual -** A sincronia virtual fornece uma lista de processos que poderão estar correctos numa determinada altura da execução. Essa lista de processos é igual para todos os processos correctos, sendo utilizada na difusão atómica para se saber de quais processos esperar resposta. A uma lista de processos correctos numa determinada altura dá-se o nome de vista, podendo ser alterada ao longo da execução do algoritmo quando se detecta uma falha num processo ou quando um processo se deseja juntar ao grupo. Para a implementação iremos usar o suporte em que o APPIA fornece sincronia virtual.
4. **Difusão Atómica** - Esta camada do appia vai permitir que haja garantias que uma mensagem entregue a um dos membros do grupo então todos os membros correctos desse grupo a recebem, no caso toma-se como membros correctos os que aparecem na vista seguinte. O método de funcionamento deste protocolo passa pelo reenvio de mensagens até que todos os membros do grupo recebam a mensagem ou então se detecte a falha de um membro e se instale uma nova vista no sistema. A correcção deste protocolo resume-se às seguintes propriedades:

- Validade: Se um processo envia uma mensagem essa mesma mensagem é entregue aos restantes processos do grupo num tempo indeterminado mas finito.
- Unanimidade: Se uma mensagem é entregue a um processo correcto de um grupo então essa mesma mensagem é entregue aos restantes processos correctos do grupo.
- Esta camada vai permitir à aplicação uma abstracção do canal de comunicação por parte da aplicação devido ás fortes garantias que fornece sobre a comunicação.

3.2 Replicação Activa

Este modo de replicação consiste em ter várias réplicas do mesmo servidor a serem executadas em diferentes processos. Para assegurar a consistência do estado destas réplicas, cada um dos comandos é disseminado por difusão atómica para todas as máquinas. Isto assegura que todas as réplicas recebem os mesmos comandos na mesma ordem, o que implica que produzem os mesmos resultados. A computação tem de ser determinista.

3.3 Replicação Passiva

Neste modelo de replicação em vez de se realizar a operação em todas as réplicas, a operação é apenas re-

alizada numa réplica. Esta réplica é chamada a réplica primária e é a responsavel por realizar as operações e por transmitir o novo estado para as réplicas secundárias. As réplicas secundárias apenas terão a responsabilidade de alterar o estado quando recebem as mensagens do servidor primário. No caso de falha da réplica primária é eleita entre as réplicas secundárias uma para substituir a primária.

4 Detecção de falhas

No caso de um servidor falhar é importante que este seja retirado do grupo. O protocolo suspect implementado na camada SuspectLayer resolve este problema simulando um detector de falhas. Quando se dá uma falha de um processo este comunica à camada de suporte à aplicação a ocorrência e esta por sua vez trata dos pormenores da tolerância a faltas.

5 Reintegração de Servidores

Este é um problema importante, pois caso um servidor tenha uma falha e depois recupere, então a reintegração deste têm de ser contemplada. Assim à partida os serviços de comunicação em grupo concretizados pelo Appia facilitam a reintegração do servidor. Tendo em conta a aplicação em causa, quando uma servidor é reintegrado, vai originar um evento no grupo que vai obrigar junção das duas vistas, a do grupo actual e a do processo que quer reintegrar o grupo.

6 Conclusões

Com base na infraestrutura presente (rede do DI) e da natureza da aplicação, várias opções foram tomadas pelo grupo com vista a alcançar o objectivo proposto pela cadeira. Os aspectos mais considerados prendem-se com a plataforma usada na construção desta aplicação, nomeadamente o Appia. A escolha de protocolos foi decidida tomando em conta as características proprias do sistema a desenvolver.

Espera-se conseguir bons resultados, tanto no que toca a tolerância a faltas como em ganho de conhecimento pessoal de cada um dos elementos do grupo.

References

[1] Veríssimo, P. and Rodrigues, L. (2001).Distributed Systems for System Architects.Kluwer Academic Publishers

[2] Rachid Guerraoui and Luís Rodrigues. Abstractions for Distributed Programming.

[3] Pedro Vicente e João Martins. Arquitectura de um Sistema de Chamadas a Procedimentos Remotos Servidores Replicados.

[4] http://appia.di.fc.ul.pt