Tolerância a Faltas Distribuidas num Jogo Multi-Utilizador

TFD004
Hugo Branco 23900 Luís Silva 24389 Rui Martinho 24110

Abstract

Este artigo trata da problemática que envolve a tolerância a faltas distribuídas num tipo de aplicações em particular, isto é, aborda o tema de como garantir a continuidade de uma aplicação distribuída por vários utilizadores. Neste caso trata-se de um jogo e da eventualidade de poder haver uma falha na funcionalidade de um ou mais destes utilizadores.

Uma dúvida surge: será que é uma solução viável termos um servidor principal a cuidar dos diversos problemas? E será que existe alguma forma perfeita de concretizar este tipo de objectivo?

Em suma, propomos uma solução para os problemas que surgem na implementação destas aplicações e apontamos alguns caminhos de entre os que podem ser tomadas para os resolver e a respectiva justificação.

1. Introdução

O conceito de arquitectura distribuída de computação, com comunicação entre dois ou mais pontos, começou a ser desenvolvida entre 1987 e 1990.

A partir deste momento foram sendo desenvolvidos vários algoritmos e arquitecturas paralelas para tentar resolver todos os problemas que iam surgindo - problemas como a escalabilidade a diversos níveis, como o desempenho dos sistemas distribuídos ou como a eficiência em termos da comunicação em si. Além disto, temos que sublinhar ainda o facto dos investigadores terem investido muito da sua atenção na questão da interligação de sistemas computacionais heterogéneos, de modo a que se pudesse interagir para obter um resultado comum.

Com estes e outros problemas colmatados, por meio de soluções mais ou menos viáveis (um problema pode ser melhor ou pior solucionado...), abriu-se uma porta para que outro tipo de problema chegasse, nomeadamente, o das faltas que teimam em surgir nestes sistemas.

Neste ponto chegamos ao cerne do nosso projecto, que tenta abordar este problema e resolvê-lo da melhor forma que nos é possível.

O nosso projecto baseia-se na tentativa de encontrar uma das possíveis e melhores soluções para garantir o máximo de tolerância a faltas que possam surgir num tipo de aplicação em particular, neste caso um jogo. Neste jogo, temos um mínimo de 2 até um máximo de 4 jogadores, onde o objectivo é não deixar a bola tocar na tabela que é protegida pelo utilizador. Para isso o jogador usa uma plataforma que vai deslizando paralelamente ao longo da respectiva tabela. O nosso problema complica-se a partir do momento em que este jogo é utilizado em rede, não existindo um servidor central, e em que é necessário garantir a continuidade da aplicação, mesmo que um ou mais jogadores sejam desligados da rede ou hajam problemas na comunicação entre eles.

Para isto, tentámos conceber uma arquitectura de rede de modo a remediar, ou se for possível, prevenir o máximo de problemas que naturalmente, e infelizmente, são parte constante deste tipo de aplicações.

Resta-nos definir o formato deste artigo:

- **No ponto 2** temos uma breve interligação entre o nosso trabalho e outros realizados na mesma área;
- **No ponto 3** abordamos, de forma directa, qual o método e arquitectura que pretendemos conceber para solucionar o problema que nos foi proposto;
- **No ponto 4** resumimos tudo o que foi dito.

2. Trabalho Relacionado

Como já foi referido, não existe apenas um modo de abordagem desta problemática. Um deles é a utilização de um coordenador, eleito entre os utilizadores, e que vai monitorizando o funcionamento do sistema distribuído, de forma a detectar possíveis falhas e tentar garantir com que todos os intervenientes recebem a informação necessária e no tempo devido. A par desta temos a partilha de responsabilidade por todos os utilizadores e não por um único, numa tentativa de garantir o bom comportamento do sistema.

O que propomos é uma arquitectura baseado nesta última ideia.

3. Parte Técnica

O jogo começa com um utilizador e espera pela ligação de outros jogadores. Quando outro jogador se liga a qualquer um dos que já se encontram no jogo, este recebe uma lista com a informação sobre os portos e endereços de todos os participantes, para que cada utilizador ganhe autonomia na comunicação e não necessite de um servidor que lhe forneça esta informação. Para implementar este sistema utilizamos estrutura de software em camadas, que correm em várias máquinas chamada APPIA, que nos fornecem a possibilidade de programação e configuração da rede.

A partir do momento em que existam dois utilizadores ligados, o jogo pode iniciar e é feito um sorteio para decidir quem começa. A partir daqui desenrola-se um ciclo de operações que constitui o âmago da aplicação. Este ciclo passa por várias etapas, das quais destacamos as seguintes:

- a sincronização é realizada quando a bola toca na plataforma ou na tabela; o responsável por informar todos os outros que a bola tocou e onde tocou é o utilizador que protege a tabela onde a bola bate;

- para saber se a plataforma não saltou em vez de deslizar, se o jogador perdeu ou se lançou a bola na direcção que indicou, simplesmente aceitamos o que ele nos diz, portanto assumimos que os jogadores não são maliciosos nesse sentido;

- para saber se houve alguma falha na ligação de um determinado jogador, para que o jogo possa continuar, é enviado periodicamente, de todos para todos, a posição da plataforma e, caso um jogador não receba a posição do leme de um dos outros, pergunta-se se ele continua ligado e, se o próprio não responder, assume-se que não;

- caso alguns não recebam a mensagem que indica que o jogador em questão, no ponto anterior, está ligado, então temos duas opções: sincronização imediata dos ”ligados”, ou sincronização dos ”ligados” após uma votação que dita se está ”ligado” ou não;

- quando a bola bate na tabela, o jogador responsável[1] pela tabela ou plataforma indica a posição da bola para existir sincronização a este nível;

- o jogo só termina quando for desligado por todos, pois podem entrar sempre jogadores, até um máximo de quatro.

[1] Este responsável é o que joga do lado respectivo ou então o jogador activo imediatamente anterior

4. Conclusão

Neste artigo descrevemos a arquitectura que pretendemos concretizar e que servirá para colmatar várias lacunas em diversas aplicações distribuídas. Para isto, tentamos reduzir ao máximo a quantidade de mensagens enviadas pela rede através de uma única sincronização periódica da bola e não toda e qualquer posição em que a bola se encontra. Além disto, como todos têm forma independente de se comunicarem entre si, garantimos autonomia a cada um dos participantes e continuidade da aplicação caso uma ou mais ligações falhem.

References

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