Universidade Federal de Viçosa
Instituto de Ciência Exatas e Tecnológicas
CCF 355 - Sistemas Distribuídos e Paralelos

Trabalho - Parte 02

Gemmarium

Grupo:
Henrique de Souza Santana Matrícula: 3051
Pedro Cardoso de Carvalho Mundim Matrícula: 3877

Professora:
Thais Regina de Moura Braga Silva

8 de junho de 2022

1 Introdução

Nesta parte do projeto da disciplina, descrevemos a modelagem do sistema em relação à sua arquitetura e aos modelos fundamentais estudados. Além disso, já apresentamos o logotipo do sistema (Fig. 1), que deverá aparecer na interface gráfica.

Figura 1: Logotipo do sistema.

Retomando a ideia apresentada na primeira parte, o sistema que está sendo projetado se chama *Gemmarium* e permite a troca de pedras preciosas (gemas) virtuais entre os usuários. As aplicações dos usuários (Clientes) interagem com funcionalidades fornecidas por diferentes servidores do sistema: o Cofre realiza cadastro e autenticação, a Galeria permite a busca e descoberta entre os Clientes, e a Forja disponibiliza os dados das gemas a serem trocadas. A troca de gemas ocorre diretamente entre os Clientes.

2 Modelagem

2.1 Modelo Físico

Em relação ao modelo físico, inicialmente todos os processos estarão em máquinas numa mesma rede local, portanto se trata de um sistema distribuído **primitivo**. Porém, considerando a divisão de camadas físicas apresentada a seguir e sua pequena escala, seria possível tornar o sistema **adaptado para Internet**, desde que duas condições fossem atendidas:

- Os endereços dos processos servidores devem ser bem conhecidos para as entidades do sistema;
- Na camada par-a-par (P2P), o sistema deve ser capaz de estabelecer conexões mesmo quando NAT (Network Address Translation) estiver sendo usado nas redes locais dos processos clientes.

A segunda condição pode ser atendida através de técnicas de *NAT traversal* [Hu, 2005], como *hole punching* [Ford et al., 2005], *UPnP* [Boucadair et al., 2013] ou redirecionamento dinâmico de portas [Wu et al., 2006]. Essas são apenas opções disponíveis, mas que não serão necessariamente inclusas na implementação, devido aos prazos de entrega do projeto.

2.2 Modelo de Arquitetura

O paradigma de comunicação usado é de **comunicação direta**, e, a princípio, teremos **processos** como entidades arquitetônicas. Mais especificamente, na primeira entrega, o paradigma é o de **comunicação entre processos**, visto que será utilizada diretamente a implementação de soquetes.

Já na segunda entrega, o paradigma é de **invocação remota**, especificamente o de **chamada de procedimento remoto (RPC)**, no qual as entidades serão os *procedimentos* nos processos de computadores remotos, que poderão ser chamados como se fossem procedimentos no espaço de endereçamento local.

As entidades arquitetônicas na terceira entrega passam a ser vistas como **serviços Web**, e o paradigma usado se mantém como invocação remota, porém como **protocolos de requisição-resposta**.

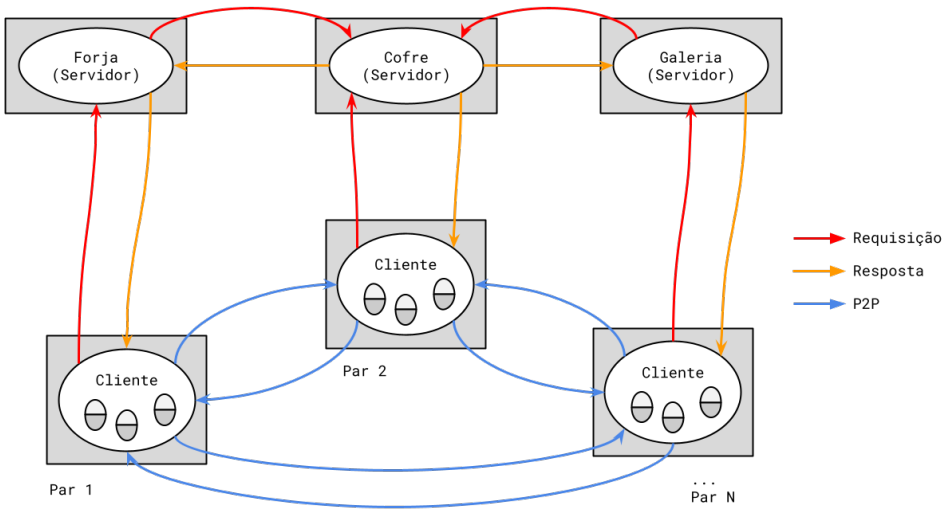

Figura 2: Arquitetura básica do sistema.

Em relação aos modelos básicos de arquitetura, o sistema será constituído de ambos os estudados, ou seja, será um modelo **híbrido**. Será **par-a-par (P2P)** em relação às comunicações feitas entre os clientes para a funcionalidade de troca. E será **cliente-servidor** no que diz respeito à comunicação entre as aplicações clientes e os servidores que disponibilizam os demais serviços do sistema. Além disso, quando necessário identificar um usuário, algum dos servidores (Galeria ou Forja) pode se comunicar com o servidor de autenticação (Cofre), e neste caso também há uma interação cliente-servidor, com um dos servidores sendo o cliente nessa comunicação em específico. A Figura 2 ilustra essas possibilidades. No servidor de autenticação (Cofre), os clientes podem fazer seu cadastro. No servidor de índice (Galeria), os clientes podem registrar os itens de troca que têm interesse, e quais estão dispostos a oferecer. No servidor de conteúdo (Forja), os clientes podem solicitar novos itens. Cada processo terá seu próprio armazenamento, para persistir os dados que lhe são relevantes.

As **camadas físicas** são complementares às camadas lógicas. Elas representam uma técnica para organizar a funcionalidade de determinada camada lógica e colocar tal funcionalidade em servidores apropriados.

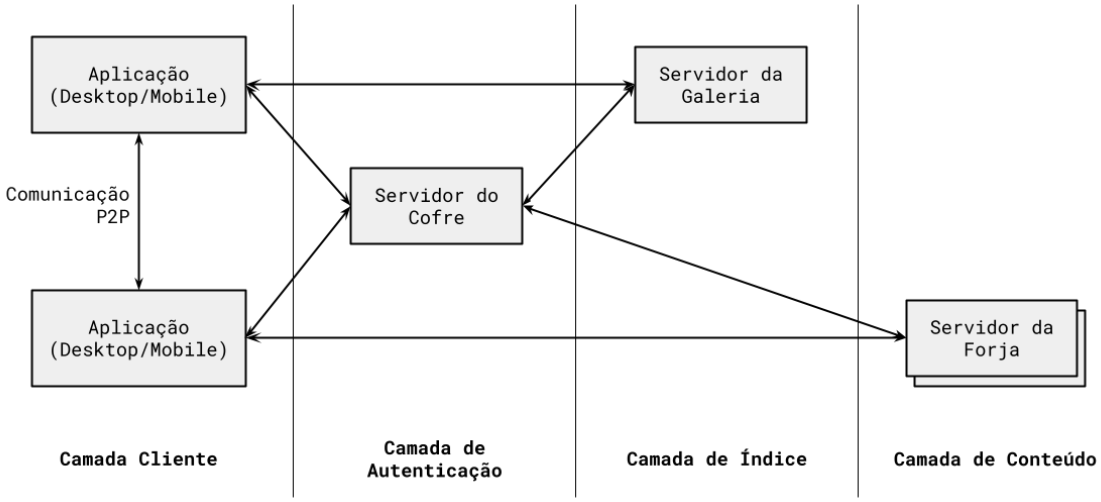

Figura 3: Camadas físicas do sistema. Por simplicidade, algumas das setas entre as diferentes instâncias da aplicação cliente e outras camadas foram omitidas, mas todos os clientes possuem o mesmo comportamento.

Em nosso sistema distribuído, podemos destacar uma camada física para cada uma das quatro funcionalidades gerais que pretendemos fornecer, como mostra a Figura 3. Essa divisão permitiria inclusive que múltiplos servidores existam em cada camada, em especial a camada de conteúdo, que poderia gerar dados de diferentes itens dependendo do servidor.

Quanto às **camadas lógicas**, com cada uma utilizando os serviços oferecidos pela camada imediatamente inferior por abstrações de software, seguiremos o padrão apresentado na disciplina. Considerando o fácil suporte a múltiplas plataformas das linguagens permitidas para a implementação (Python ou Java), não precisamos nos preocupar com as camadas lógicas de plataforma. No caso específico da primeira entrega, não teremos a camada de *middleware*, pois usaremos diretamente a API de soquetes. Nas demais entregas, essa camada terá o gRPC como *middleware* de chamada de procedimento remoto, e depois trocaremos para um *middleware* para o protocolo SOAP.

Uma proposta possível de separação de camadas lógicas é apresentada na Figura 4, na qual a camada de aplicação é separada em apresentação – apresenta dados para usuário numa interface gráfica, ou emite *logs* nos servidores –, negócio – se ocupa com a lógica interna das funcionalidades – e comunicação – faz a interação com o *middleware* subjacente.

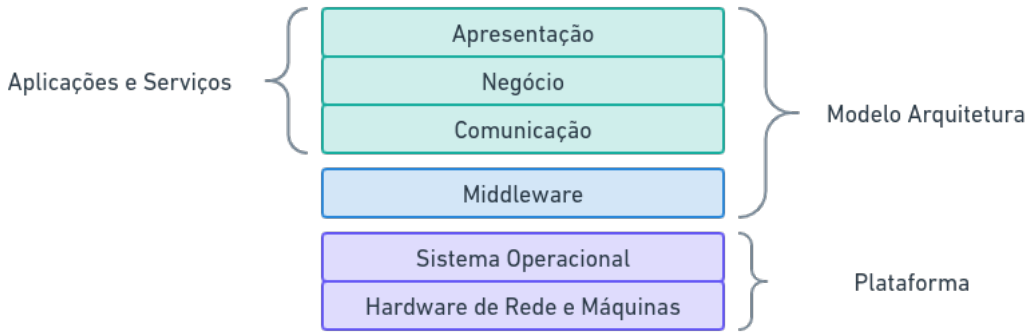

Figura 4: Camadas lógicas do sistema.

2.3 Modelos Fundamentais

Para o **modelo de interação**, consideramos que a taxa de transmissão de dados não é um fator relevante ao sistema, visto que todas as transferências de dados são muito pequenas; os maiores dados compartilhados serão as imagens das gemas, que são pequenos conjuntos de imagens de 16x16 pixels, contendo no máximo 500 B cada arquivo. A latência e *jitter* também não são de grande relevância, pois mesmo que algumas mensagens demorem muito para ser transmitidas, ou se houver muitas oscilações na rede, não se trata de informações críticas ou que necessitam chegar em alguma ordem ou tempo específicos. Considerando esse levantamento, nosso sistema será assíncrono, pois não há preocupação com comunicação em tempo real em nenhuma das interações.

Abordando o **modelo de falhas**, o principal tipo de falha que pode ocorrer e que podemos tratar com o uso do TCP, são as falhas por omissão. Se necessário, podemos também definir janelas de *timeout* razoáveis, e permitir ao usuário solicitar retransmissão de suas mensagens no caso dos processos clientes. Em caso de falhas arbitrárias que ocorrerem no canal de comunicação, também a escolha do protocolo TCP é capaz de tratar. Já as falhas de temporização não entram no contexto, visto que o sistema não é síncrono.

Por fim, no **modelo de segurança**, é interessante que haja proteção aos processos, no sentido de que a identidade dos demais processos seja confirmada. Propomos um esquema básico de criptografia assimétrica para isso, e por consequência também obtemos um mecanismo de proteção ao canal. Requerendo que todos os processos do sistema possuam um par de chaves

pública e privada, é possível usar autenticação por chave pública e identificar os processos, e também é possível criptografar toda a comunicação feita. O servidor de autenticação (Cofre) seria o responsável por gerenciar essas chaves públicas.

3 Conclusão

Através deste levantamento, foi possível obter uma visão mais clara da organização do projeto, além de revisar os conceitos de modelos de sistemas distribuídos. Uma modelagem adequada torna possível realizar futuramente uma melhor implementação do sistema, e esperamos obter êxito nesse sentido.

Referências

[Boucadair et al., 2013] Boucadair, M., Penno, R., and Wing, D. (2013). Universal Plug and Play (UPnP) Internet Gateway Device-Port Control Protocol Interworking Function (IGD-PCP IWF). *RFC 6970*.

[Ford et al., 2005] Ford, B., Srisuresh, P., and Kegel, D. (2005). Peer-to-Peer Communication Across Network Address Translators. In *USENIX Annual Technical Conference, General Track*, pages 179–192.

[Hu, 2005] Hu, Z. (2005). Nat Traversal Techniques and Peer-to-Peer Applications. In *HUT T-110.551 Seminar on Internetworking*, pages 04–26. Citeseer.

[Wu et al., 2006] Wu, M.-W., Huang, Y., Chen, I.-Y., Lu, S.-K., and Kuo, S.-Y. (2006). A scalable port forwarding for p2p-based wi-fi applications. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 26–37. Springer.