Universidade Católica de Pelotas
Engenharia de Computação

Elementos Arquitetônicos

Gabriel Harter Zoppo

Disciplina: Sistemas Distribuídos
Professora: Adenauer Correa Yamin

Pelotas, abril de 2022.

Perguntas importantes para o entendimento de SD:

- Quais entidades que estão se comunicando num sistema distribuído?
- Como elas se comunicam ou, mais especificamente qual paradigma de comunicação usado?
- Quais funções e responsabilidades(possivelmente variáveis) estão relacionadas a eles na arquitetura global?
- Como eles são mapeados na infraestrutura distribuída física(qual a sua localização)?

Entidades em comunicação:

- **Sistema:** Geralmente claros por as entidades que se comunicam serem processos, levando a visão de um SD como processos acoplados por um paradigma de comunicação apropriado. Possui algumas advertências:
  - Em alguns ambientes primitivos os SO talvez não suportem abstração de processo, sendo assim as entidades são nós.
  - Na maioria dos ambientes os processos são complementados por threads.

Entidades em comunicação:

- **Objetos:** Os objetos foram introduzidos para permitir e estimular o uso de estratégias orientadas a objeto nos sistemas distribuídos. Os objetos são acessados por meio de interfaces, com uma linguagem de definição de interface (ou IDL, interface definition language) associada fornecendo uma especificação dos métodos definidos neste objeto.
- **Componentes:** Os componentes se parecem com objetos, pois oferecem abstrações voltadas ao problema para a construção de sistemas distribuídos e também são acessados por meio de interfaces. A principal diferença é que os componentes torna todas as dependências explícitas e fornecendo um contrato mais completo para a construção do sistema.
- **Serviços Web:** Os serviços Web são parcialmente definidos pelas tecnologias baseadas na Web que adotam. Outra distinção importante resulta do estilo de uso da tecnologia. Os serviços Web podem ser implementados por diferentes provedores e usar diferentes tecnologias.

Paradigmas de comunicação:

- **Comunicação entre processos:** Se refere ao suporte de nível relativamente baixo para comunicação entre processos nos sistemas distribuídos, incluindo primitivas de passagem de mensagens, acesso direto à API oferecida pelos protocolos Internet (programação de soquetes) e também o suporte para comunicação em grupo* (multicast).
- **A invocação remota:** representa o paradigma de comunicação mais comum nos sistemas distribuídos, cobrindo uma variedade de técnicas baseadas na troca bilateral entre as entidades que se comunicam em um sistema distribuído e resultando na chamada de uma operação, um procedimento ou método remoto.
- **Protocolos de requisição-resposta:** Os protocolos de requisição-resposta são efetivamente um padrão imposto em um serviço de passagem de mensagens para suportar computação cliente-servidor.

Paradigmas de comunicação:

- **Chamada de procedimento remoto:** Representa uma inovação intelectual importante na computação distribuída. Na RPC, procedimentos nos processos de computadores remotos podem ser chamados como se fossem procedimentos no espaço de endereçamento local.
- **Invocação de método remoto:** A invocação de método remoto (RMI, Remote Method Invocation) é muito parecida com as chamadas de procedimento remoto, mas voltada para o mundo dos objetos distribuídos. Com essa estratégia, um objeto chamador pode invocar um método em um objeto potencialmente remoto, usando invocação de método remoto

Funções e responsabilidades:

- **Cliente Servidor:**
  - Arquitetura mais citada quando se discute os sistemas distribuídos.
  - Os servidores Web, e a maioria dos outros serviços Internet, são clientes do serviço DNS, que mapeia nomes de domínio Internet a endereços de rede (IP).
  - Um mecanismo de busca é tanto um servidor como um cliente: ele responde às consultas de clientes navegadores e executa Web crawlers que atuam como clientes de outros servidores Web.

- **Peer-to-peer:**
  - Todos os processos envolvidos em uma tarefa ou atividade desempenham funções semelhantes, interagindo cooperativamente como pares sem distinção entre processos clientes e servidores, nem entre os computadores em que são executados.
  - O objetivo da arquitetura peer-to-peer é explorar os recursos (tanto dados como de hardware) de um grande número de computadores para o cumprimento de uma dada tarefa ou atividade. Um dos exemplos mais antigos desse tipo de arquitetura é o aplicativo Napster, empregado para o compartilhamento de arquivos de música digital.

Mapeamento das entidades:

- **Mapeamento de serviços em vários servidores:**
  - Os serviços podem ser implementados como vários processos servidores em diferentes computadores hospedeiros, interagindo conforme for necessário, para fornecer um serviço para processos cliente.
  - Um tipo de arquitetura em que ocorre uma interação maior entre vários servidores, e por isso denominada arquitetura fortemente acoplada, é o baseado em cluster, conforme apresentado no Capítulo 1. Um cluster é construído a partir de várias, às vezes milhares, de unidades de processamento, e a execução de um serviço pode ser particionada ou duplicada entre elas.

- **Uso de cache:**
  - Na prática, o emprego de caches é bastante comum. Por exemplo, os navegadores Web mantêm no sistema de arquivos local uma cache das páginas recentemente visitadas e, antes de exibi-las, com o auxílio de uma requisição HTTP especial, verifica nos servidores originais se as páginas armazenadas na cache estão atualizadas.
  - O objetivo dos servidores proxies é aumentar a disponibilidade e o desempenho do serviço, reduzindo a carga sobre a rede remota e sobre os servidores Web. Os servidores proxies podem assumir outras funções, como, por exemplo, serem usados para acessar servidores Web através de um firewall.

Mapeamento das entidades:

- **Código móvel:**
  - Os applets representam um exemplo bem conhecido e bastante utilizado de código móvel – o usuário, executando um navegador, seleciona um link que aponta para um applet, cujo código é armazenado em um servidor Web e posteriormente executado. Uma vantagem de executar um código localmente é que ele pode dar uma boa resposta interativa, pois não sofre os atrasos nem a variação da largura de banda associada à comunicação na rede.

- **Agentes móveis:**
  - Um agente móvel é um programa em execução (inclui código e dados) que passa de um computador para outro em um ambiente de rede, realizando uma tarefa em nome de alguém, como uma coleta de informações, e finalmente retornando com os resultados obtidos a esse alguém.
  - Os agentes móveis (assim como o código móvel) são uma ameaça em potencial à segurança para os recursos existentes nos computadores que visitam. O ambiente que recebe um agente móvel deve decidir, com base na identidade do usuário em nome de quem o agente está atuando, qual dos recursos locais ele pode usar. A identidade deve ser incluída de maneira segura com o código e com os dados do agente móvel.

Referências:

- **Livro texto da disciplina em português:**
  - https://www.dropbox.com/s/ovnda64rhy4j9i2/Couloris_Sistemas-Distribuidos-Conceitos-e-Projeto-Quinta-Edicao.pdf?dl=0
- **Livro texto da disciplina em inglês:**
  - https://www.dropbox.com/s/a3s8x0k5xeaepx4/DistributedSystems-Concepts-and-Design-5th-ed.pdf?dl=0

Universidade Católica de Pelotas
Engenharia de Computação

Elementos Arquitetônicos

Gabriel Harter Zoppo

Disciplina: Sistemas Distribuídos
Professora: Adenauer Correa Yamin

Pelotas, abril de 2022.