Apache ActiveMQ

José Luiz S. Teixeira[1], João Henrique S. Farias[1], Marilda S. Pereira[1], Vinicyos S. Pimentel[1]

[1]Universidade Federal do Sul e Suldeste do Pará (UNIFESSPA)

São Geraldo do Araguaia – PA – Brasil

{joseluiz,henrique.farias,marildasp,vinicyos.pimentel}@unifesspa.edu.br

Resumo. *O referente trabalho aborda algumas informações sobre o projeto, ActiveMQ, cuja utilidade se dá em favorecer a comunicação entre diferentes aplicações, necessidade essa tão visível na área da tecnologia da informação o que se faz necessária a implantação de diferentes recursos tecnológicos. Nesse contexto, o mesmo, tem como objetivo, discorrer algumas informações e aspectos direcionados ao ActiveMQ: é um message broker desenvolvido em Java, voltado para comunicação remota entre sistemas que utilizam a especificação JMS (Java Message Service). A fundamentação teórica é baseada em leitura de artigos acadêmicos, periódicos relacionados com o tema, onde o mesmo está organizado em cinco eixos.*

1. Introdução

Em pleno século XXI, ao citar sobre tecnologia da informação, é necessário, considerar o avanço da área, o que tornou possível o surgimento de algumas técnicas direcionadas a atender as necessidades das pessoas. Por essa razão, eis o surgimento de trocas de dados entre diferentes aplicações, sem déficit entre si pois, a comunicação entre as diferentes aplicações, são extremamente necessários nos dias de hoje, e é ainda algo desafiador, efetivar na pratica a interação entre elas.

O referente trabalho, por meio de pesquisa bibliográfica, tem o objetivo de trazer alguns aspectos sobre a comunicação entre as aplicações, especificamente sobre o Active MQ.

O projeto ActiveMQ foi originalmente criado por seus fundadores da LogicBlaze em 2004, como um message broker de código aberto, hospedado pela CodeHaus. O código e a marca registrada ActiveMQ foram doados à Apache Software Foundation em 2007, sendo que os fundadores continuaram a desenvolver o código-base com a comunidade estendida Apache.

ActiveMQ emprega vários modos para alta disponibilidade, incluindo ambos mecanismos de bloqueio de sistema de arquivos e de banco de dados a nível de linha, compartilhamento de armazenamento de persistência, através de um sistema de arquivos compartilhado, ou replicação real usando o Apache ZooKeeper. Um mecanismo robusto de escalamento horizontal, chamado de *Network of Brokers*, também é suportado. No ramo empresarial, o ActiveMQ é celebrado por sua flexibilidade de configuração, e o seu suporte para um número relativamente grande de protocolos de transporte, incluindo OpenWire, Stomp, MQTT, AMQP, REST e WebSockets.

ActiveMQ é usado em implementações de barramento de serviço corporativo, tais como o Apache ServiceMix e Mule. Outros projetos usando ActiveMQ incluem o Apache Camel e Apache CXF em projetos de infraestrutura SOA.

Coincidindo com o lançamento do Apache ActiveMQ 5.3, os primeiros resultados mundiais para as referências do padrão industrial SPECjms2007 foram anunciados. Quatro resultados foram submetidos à Standard Performance Evolution Corporation (SPEC) e aceitos para publicação. Os resultados abrangem diferentes topologias para analisar a escalabilidade do Apache ActiveMQ em duas dimensões.

ActiveMQ está atualmente na versão principal 5, versão secundária 15. A versão 6, chamada ActiveMQ Artemis, é uma reescrita completa do ActiveMQ 5, incorporando a doação do código base HornetQ da Red Hat, e trazendo a implementação JMS do broker à especificação 2.0. Assim, a Red Hat tem controle substancial do projeto devido à sua antiga propriedade do código-base.

2. Definição

O Apache ActiveMQ® é o agente de mensagens baseado em Java, multiprotocolo e de software livre mais popular. Ele suporta protocolos padrão do setor para que os usuários obtenham os benefícios das opções do cliente em uma ampla variedade de idiomas e plataformas. Conecte-se a partir de clientes escritos em JavaScript, C, C++, Python, .Net e muito mais. Integre seus aplicativos multiplataforma usando o onipresente protocolo **AMQP,** troque mensagens entre seus aplicativos da web usando **STOMP** sobre websockets. gerencie seus dispositivos IoT usando **MQTT**. Dê suporte à sua infraestrutura **JMS** existente e além. O ActiveMQ oferece o poder e a flexibilidade para suportar qualquer caso de uso de mensagens.

Existem atualmente dois "Tipos" de ActiveMQ disponíveis - o conhecido corretor "clássico" e o corretor de "próxima geração" codinome *Artemis*. Assim que o Artemis atingir um nível suficiente de paridade de recursos com a base de código "Clássica", ele se tornará a próxima versão principal do ActiveMQ, a documentação inicial da migração está disponível, bem como um roteiro de desenvolvimento para o Artemis.

Proteja seus dados e equilibre sua carga

O ActiveMQ fornece muitos recursos avançados, incluindo balanceamento de carga de mensagens e alta disponibilidade para seus dados, vários brokers "mestres" conectados podem responder dinamicamente à demanda do consumidor movendo mensagens entre os nós em segundo plano. Os intermediários também podem ser emparelhados em uma configuração mestre-escravo para que, se um mestre falhar, o escravo assuma o controle, garantindo que os clientes possam acessar seus dados importantes e eliminando o tempo de inatividade dispendioso.

Padrões fáceis de integração empresarial

Os Padrões de Integração Corporativa descrevem as várias maneiras pelas quais vários aplicativos geralmente interagem e se integram entre si, as mensagens aysnchronous estão no centro dessa integração, e o ActiveMQ facilita o aproveitamento desses padrões por meio de rotas Apache Camel implantadas diretamente no broker.

Implantação flexível

O ActiveMQ é mais comumente implantado como um processo autônomo, essa opção isola o ActiveMQ de qualquer aplicativo específico e fornece flexibilidade máxima para alocação e gerenciamento de recursos, no entanto, o ActiveMQ pode ser configurado para ter um espaço muito pequeno, o que torna viável incorporá-lo em seu aplicativo. Essa opção pode fornecer a um aplicativo uma semântica de mensagens simples e poderosa e também permitir uma troca fácil de mensagens com outros aplicativos.

3. Middleware

Tipo de sincronicidade provida pelo middleware (Síncrono ou Assíncrono)?

Assíncrona

Linguagem de programação para construção das aplicações (Linguagem única ou diversas linguagens)?

Apache ActiveMQ é um message broker escrito em Java com interfaces JMS, REST e WebSocket, porém suporta protocolos como AMQP, MQTT, OpenWire e STOMP que podem ser utilizados por aplicações em diferentes linguagens.

Bibliotecas

- INTERNET
- C (extinto)
- C++
- Erlang
- Vai
- Haskell
- Haxe (extinto)
- Prólogo Jekejeke
- NetLogo
- Node.js
- Perl 5
- Pique
- Pitão
- Raquete
- Ruby nos trilhos
- Tcl/Tk
- Java
- JavaScript

Qual o Domínio de Aplicação que o middleware é utilizado (Tempo real, Adaptativo, Móveis, Redes Sociais)?

O ActiveMQ Real Time é um subprojeto do popular sistema de mensagens Apache ActiveMQ, mas foi projetado especificamente para atender às necessidades de aplicativos colaborativos em tempo real de baixa latência e alto rendimento.

Ele foi projetado para ser usado por vários idiomas em várias plataformas - e tem três objetivos principais:

1. Multicast confiável conectável de baixa latência - com diferentes qualidades de serviço. As complexidades da comunicação em grupo significam que diferentes protocolos multicast confiáveis devem ser usados para atender às necessidades de colaboração de aplicativos em tempo real. ActiveBlaze suporta protocolos conectáveis, que incluem:
    - Protocolos confiáveis baseados em NACK
    - Protocolo do totem
    - Continue Correção de Erro
2. Gerenciamento de associação de grupo ativo para permitir uma verdadeira comunicação ponto a ponto e mensagens de grupo
3. Gerenciamento de cluster para estado compartilhado, tolerância a falhas, confiabilidade e disponibilidade.

A necessidade de baixa latência significa que a API JMS (Java Message Service) nem sempre é adequada e o ActiveBlaze vem com suas próprias APIs sob medida para atingir uma taxa de transferência muito alta. É um objetivo do projeto fornecer uma camada de API JMS opcional em cima das APIs principais do ActiveBlaze para permitir que ela seja uma substituição para implementações de mensagens hub e spoke mais tradicionais.

Para habilitar o suporte a vários idiomas, habilitar o versionamento do formato Wire e ajudar na taxa de transferência rápida - o Apache ActiveBlaze é construído sobre o protobuf do Google - uma estrutura para codificar eficientemente estruturas de dados extensíveis.

Ambiente de execução do middleware (Nuvem, PC, Smartphone, Sensor, "IoT")?

Em nuvem que é utilizado pela Amazon MQ e também pode ser usado no PC.

O Amazon MQ é um serviço de agente gerenciado que facilita a migração para um agente de mensagem na nuvem. Um agente de mensagem permite que aplicações de software e componentes se comuniquem usando várias linguagens de programação, sistemas operacionais e protocolos de sistemas de mensagens formais. Atualmente Amazon MQ é compatível com os mecanismos do Apache ActiveMQ e RabbitMQ.

Tipo de heterogeneidade suportada (Sistema Operacional, Linguagem de Programação, Rede (cabeada, sem fio), Plataforma de Nuvem, Hardware)?

Sistema Operacional, Plataforma em Nuvem.

Qual o Modelo do middleware (Orientado a Objetos, Orientado a Mensagens, Procedural)?

Orientado a Mensagens

Forma de disponibilização do middleware (Proprietário, Aberto)?

Aberto

O middleware é implementado sobre que Camada de Transporte (TCP, UDP)?

TCP

4. Arquitetura

Arquitetura desacoplada e comunicação assíncrona são características do ActiveMQ.

Apache ActiveMQ é um servidor de mensageria de código aberto utilizado por grandes players do mercado que utilizam arquiteturas open-source.

O propósito da Arquitetura Open Source é oportunizar modelos de habitações em plataformas globais na web, as quais qualquer pessoa pode acessar e fazer download dos arquivos, adaptar às suas necessidades, fabricar em laboratórios de fabricação e prototipagem digitais e montar.

4.1 ActiveMQ "Clássico"

Arquitetura estabelecida há muito tempo e infinitamente conectável que atende a muitas gerações de aplicativos.

- JMS 1.1 com implementação de cliente completa incluindo JNDI
- Alta disponibilidade usando armazenamento compartilhado
- Modelo de endereçamento familiar baseado em JMS
- "Rede de corretores" para distribuição de carga
- Opções KahaDB e JDBC para persistência

4.1.1 Recursos

- Suporta uma variedade de clientes e protocolos de linguagem cruzada de Java, C, C++, C#, Ruby, Perl, Python, PHP
    - OpenWire para clientes de alto desempenho em Java, C, C++, C#
    - Suporte Stomp para que os clientes possam ser escritos facilmente em C, Ruby, Perl, Python, PHP, ActionScript/Flash, Smalltalk para conversar com ActiveMQ, bem como qualquer outro Message Broker popular
    - Suporte AMQP v1.0
    - Suporte ao MQTT v3.1 permitindo conexões em um ambiente IoT.
- suporte completo para os Enterprise Integration Patterns tanto no cliente JMS quanto no Message Broker
- Suporta muitos recursos avançados , como Grupos de Mensagens , Destinos Virtuais , Curingas e Destinos Compostos
- Suporta totalmente JMS 1.1 e J2EE 1.4 com suporte para mensagens transitórias, persistentes, transacionais e XA
- Spring Support para que o ActiveMQ possa ser facilmente incorporado em aplicativos Spring e configurado usando o mecanismo de configuração XML do Spring
- Testado em servidores J2EE populares, como TomEE , Geronimo , JBoss, GlassFish e WebLogic
    - Inclui adaptadores de recursos JCA 1.5 para mensagens de entrada e saída para que o ActiveMQ seja implantado automaticamente em qualquer servidor compatível com J2EE 1.4
- Suporta protocolos de transporte conectáveis , como transportes in-VM , TCP, SSL, NIO, UDP, multicast, JGroups e JXTA
- Suporta persistência muito rápida usando JDBC junto com um diário de alto desempenho

- Projetado para clustering de alto desempenho, cliente-servidor, comunicação baseada em pares
- API REST para fornecer API baseada na Web agnóstica e neutra em linguagem para mensagens
- Ajax para suportar suporte de streaming da Web para navegadores da Web usando DHTML puro, permitindo que os navegadores da Web façam parte da malha de mensagens
- Suporte CXF e Axis para que o ActiveMQ possa ser facilmente descartado em qualquer uma dessas pilhas de serviços da Web para fornecer mensagens confiáveis
- Pode ser usado como um provedor JMS na memória, ideal para teste de unidade JMS.

4.2 ActiveMQ Artemis

Arquitetura sem bloqueio de alto desempenho para a próxima geração de aplicativos de mensagens.

- JMS 1.1 e 2.0 + Jakarta Messaging 2.0 e 3.0 com implementações de cliente completas, incluindo JNDI
- Alta disponibilidade usando armazenamento compartilhado ou replicação de rede
- Modelo de endereçamento agnóstico de protocolo simples e poderoso
- Clustering flexível para distribuição de carga
- Implementações avançadas de diário para persistência de baixa latência, bem como JDBC
- Alta paridade de recursos com o ActiveMQ "Classic" para facilitar a migração
- Espelhamento assíncrono para recuperação de desastres
- Balanceamento de carga orientado a dados

4.2.1 Novos Recursos

Recuperação de desastres

Espelhamento de conexões do agente AMQP

Servidor conectado um ao outro agindo como irmãos gêmeos para oferecer suporte à migração fácil do datacenter.

Potencialize seus micros serviços

Mensagens de alto desempenho para micros serviços altamente escaláveis

Os micros serviços geralmente são implementados com HTTP usando um padrão de solicitação-resposta de "bloqueio". Esse padrão pode gerar boa latência em casos de uso de baixa taxa de transferência. No entanto, os padrões baseados em eventos e mensagens assíncronas podem fornecer escalabilidade superior e latência geral mais baixa em casos de uso de alta taxa de transferência. Com throughput potencial medido em *milhões* de mensagens por segundo, o ActiveMQ Artemis tem o desempenho e o conjunto de recursos para trazer esses ganhos para seus aplicativos.

Criação fácil do Docker

Crie imagens do Docker para simplificar a implantação

Os contêineres são uma tecnologia poderosa que você pode usar para simplificar a implantação em qualquer ambiente (dev., teste, produção etc.). O ActiveMQ Artemis fornece alguns scripts simples para começar com o Docker.

Implementações avançadas de diário

Persistência de mensagens flexível e rápida

O diário somente anexo ActiveMQ Artemis vem em vários sabores diferentes. Para o máximo em desempenho e confiabilidade, o AIO no Linux é suportado por meio de uma pequena biblioteca JNI. Para um desempenho ainda melhor com uma ligeira diminuição na confiabilidade em caso de falha de hardware, uma opção de Mapeamento de Memória está disponível. Se nenhuma dessas opções for adequada, uma implementação rápida do Java NIO estará disponível. Por fim, para casos de uso em que um banco de dados é o preferido e o alto desempenho não é uma prioridade, o JDBC é uma opção.

Guia para migração para ActiveMQ Artemis

Migre facilmente do ActiveMQ 5

ActiveMQ Artemis tem análogos para todos os recursos principais do ActiveMQ 5 e suporte completo para o protocolo ActiveMQ 5 Open Wire.

5. Conclusão

ActiveMQ: é um message broker desenvolvido em Java, voltado para comunicação remota entre sistemas que utilizam a especificação JMS (Java Message Service). Sua API é compatível com diversas linguagens, como C, C++, .NET, Perl, PHP, Python, Ruby, etc.

Seu principal objetivo, é fornecer padrões para aplicações orientados a mensagens independente da linguagem utilizada, oferecendo alta disponibilidade, desempenho, escalabilidade, confiabilidade e segurança para ambientes corporativos. Esses requisitos são essências para que as mensagens enviadas alcancem seus destinatários.

6. Referências

1. Apache ActiveMQ – Acesso 07/02/2022
2. Coffe and Tips – Acesso em 07/02/2022
3. Blog do Haylson Martins – Acesso em 07/02/2022
4. Blog Magic Brasil – Acesso em 07/02/2022
5. Repositório UFSM – Acesso em 07/02/2022
6. Amazon MQ – Acesso em 07/02/2022
7. Repositório UFSC – Acesso em 07/02/2022
8. Site Oficial ActiveMQ – Acesso em 07/02/2022