**UNIVERSIDADE POSITIVO**

**CURSO DE GRADUAÇÃO SISTEMAS DE INFORMAÇÃO**

THIAGO DE LIMA - 1524295

AIRTON SILVA - 1716623

MARCOS AURÉLIO - 1504293

**PROTOCOLOS**

**CURITIBA**

**2019**

**SUMÁRIO**

[**Introdução** 3](#_Toc22853861)

[**1.** **AMQP – Advanced Message Queuing Protocol** 4](#_Toc22853862)

[**1.1** **Vantagens** 4](#_Toc22853863)

[**1.2.** **Onde o protocolo é utilizado** 5](#_Toc22853864)

[**1.3.** **Rede que opera** 6](#_Toc22853865)

[**1.4.** **Informações para se utilizar** 6](#_Toc22853866)

[**2.** **MQTT – Message Queue Telemetry Transport** 9](#_Toc22853867)

[**2.1.** **Vantagens** 9](#_Toc22853868)

[**2.2.** **Onde o protocolo é utilizado** 10](#_Toc22853869)

[**2.3.** **Rede que opera** 10](#_Toc22853870)

[**2.4.** **Informações para se utilizar** 10](#_Toc22853871)

[**3.** **STOMP – Streaming de texto simples / Orientada Messaging Protocol** 13](#_Toc22853872)

[**3.1.** **Vantagens** 13](#_Toc22853873)

[**3.2.** **Onde o protocolo é utilizado** 13](#_Toc22853874)

[**3.3.** **Rede que opera** 15](#_Toc22853875)

[**3.4.** **Informações para se utilizar** 16](#_Toc22853876)

[**Conclusão** 18](#_Toc22853877)

[**Referências** 19](#_Toc22853878)

## **Introdução**

Trabalho desenvolvido com a intenção de demonstrar tipos de protocolos de Mensageria (comunicação entre máquinas), identificando as principais características, vantagens, onde é utilizada, a rede em que os mesmos operam, além das informações para se utilizar de três tipos de protocolos: AMQP, STOMP e MQTT.

## **AMQP – Advanced Message Queuing Protocol**

Este protocolo foi desenvolvido por usuários que atuavam no mercado financeiro visando permitir a comunicação entre sistemas de forma transparente (interoperabilidade). O AMQP é um protocolo pertencente à camada de transporte que roda acima do TCP, o mesmo permite tanto o envio quanto o recebimento de mensagens de forma assíncrona (quando não há uma resposta imediata) independente do hardware, sistema ou linguagem. O protocolo pode ser visto como um assíncrono equivalente ao HTTP sendo capaz do mesmo comunicar-se com um broker que se encontra no caminho.

Devido o protocolo de mensageria ser avançado o mesmo pode: realizar a restrição do acesso das filas, gerir a profundidade, anotações, cabeçalhos, e propriedade de mensagem tornando o protocolo robusto no mundo corporativo.

O AMQP pode ser dividido em duas camadas:

* **Camada funcional:** conjunto de métodos e comandos que trabalham de forma útil em prol da aplicação para suportar as diversas formas de trocas de mensagens.
* **Camada de transporte:** o protocolo cobre o conteúdo de forma relativa ao meio de transporte, multiplexação de canais, pacote de dados (framing), encoding do conteúdo, heart-heating, representação dos dados, além do tratamento de erro.

## **Vantagens**

O AMQP por se tratar de um protocolo binário traz consigo características, como sendo: multicanal, negociável, assíncrono, seguro, neutro, portável e eficiente. Contudo o mesmo traz também vantagens, como:

* **Escalabilidade a nível atômico:** todo serviço/ação pode ser escalado de forma individual, tirando do mesmo uma maior capacidade e eficiência do hardware.
* **Manutenibilidade:** cada microserviço possui processos independentes desde a concepção à implantação, e com isso todo o esforço empregado de recursos humanos pode ser mais aproveitado.
* **Reutilização:** o mesmo microserviço pode ser consumido por aplicações diferentes, tendo uma diminuição do tempo para o desenvolvimento das aplicações com recursos iguais.
* **Resistência à falhas:** somente recursos que dependam do microserviço que se encontra indisponível ou falhando serão comprometidos.
* **Flexibilidade de tecnologia:** devido o AMQP se comportar como uma lingua entre microserviços, nada impede que cada um seja desenvolvido em uma linguagem/plataforma diferente tendo como único requisito se comunicar com os demais.
* **Padrão:** o mesmo produz um padrão aberto para os protocolos.
* **Interoperabilidade:** entre tecnologias e plataformas.

## **Onde o protocolo é utilizado**

O AMQP foi criado com o intuito de ser um substituto aos protocolos antigos existentes e Middlewares proprietário, contudo o mesmo tem como objetivo a confiabilidade e interoperabilidade sendo muito utilizado para mensageria e com isso, alguns dos Message Brokers que utilizam este tipo de protocolo são: RabbitMQ, ActiveMQ, Apache Kafka, Azure Service Bus e AmazonMQ.

O protocolo foi idealizado por usuários que tinham como necessidade conectar seus sistemas de maneira confiável e segura para realizar seus negócios, diminuindo o consumo, melhorando a conectividade, além de posicionar a empresa em benefício de novas inovações. Portanto o AMQP é utilizado para conectar sistemas críticos em telecomunicações, manufatura, internet, entre outros segmentos de mercado, tendo como usuários notáveis empresas como:



Fonte: AMQP < <https://www.amqp.org/> >

## **Rede que opera**

O AMQP é operado através de redes wireless, que são chamados de dispositivos IoT (Internet of Things, traduzido significa Internet das Coisas), a mesma é definida como a comunicação realizada de máquina a máquina (M2M) via internet fisicamente ou virtualmente, contudo esta rede possui 7 camadas, aonde o AMQP roda acima do protocolo TCP pertencente à camada de transporte. O protocolo é considerado um intermediário muito confiável em relação as entregas de mensagens assíncronas.

## **Informações para se utilizar**

Como já relacionado neste artigo, o AMQP é um protocolo aberto para sistemas corporativos visando a troca de mensagens, devido a isso os recursos principais do AMQP, são:

* **Servidor:** conhecido como “broker”, o mesmo aceita conexões dos clientes e inserem as funções de “filas” de mensagens e roteamento.
* **Filas:** local de armazenagem das mensagens sendo em memória/disco, ou as enviadas em sequência para as aplicações. Estas filas podem ser e ter como propriedades as seguintes características: privadas/pública, armazenadas de forma durável/transitório, existência permanente/temporária, filas que armazenam/encaminham, entre outros. Todas as filas são independentes umas das outras.
* **Exchanges:** entidades internas do servidor que recebem as mensagens e roteiam as mesmas para as filas, levando-se em consideração os critérios pré-definidos do bindings.
* **Bindings:** relação ocorrida entre exchanges e filas, aonde definirá como será realizado o roteamento das mensagens.
* **Virtual hosts:** os mesmos são domínios independentes no servidor, mas que compartilham o mesmo ambiente tanto para a autenticação quanto para a segurança. Ou seja, são coleções de exchanges, filas e outros objetos.

Abaixo se encontra uma visão geral dos componentes do protocolo:

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Uso de AMQP para Transporte de Mensagens entre Atores Remotos

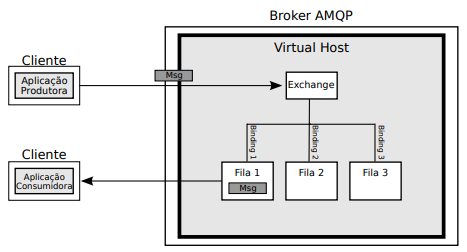
< <https://www.ime.usp.br/~reverbel/students/qualif_exams/qualificacao-thadeu.pdf> >

O modelo AMQP separa as tarefas acima e atribui em locais distintos sendo eles exchanges ou filas, tendo como funções:

1. Receber as mensagens e fazer o roteio para as filas.
2. Armazenar e encaminhar para as aplicações.

Contudo para que ocorra o envio de uma mensagem deve-se ter uma determinada Exchange e que seja de um virtual host, sendo realizado uma rotulação tendo uma informação de roteamento, assim como dados no corpo da mesma. Quando a mensagem tiver sido recebida no servidor AMQP, o roteamento ocorre para filas do virtual host individuais ou conjuntos delas, caso se tenha a possibilidade de rotear a mensagem, o comportamento do que deve ser feito com a mensagem será definido pelo “produtor”. Se a mensagem for depositada em alguma fila, a mesma deverá passá-la para a aplicação consumidora e caso não seja possível a mesma deverá ser armazenada e posteriormente feita uma nova tentativa.

Quando a mensagem for finalmente entregue, a mesma é removida da fila. A aceitação do recebimento da mensagem, poderá ocorrer imediatamente ou poderá ser processada após um tempo. Abaixo é demonstrado como funcionaria todo o processo descrito acima:



Fonte: Uso de AMQP para Transporte de Mensagens entre Atores Remotos

< <https://www.ime.usp.br/~reverbel/students/qualif_exams/qualificacao-thadeu.pdf> >

## **MQTT – Message Queue Telemetry Transport**

O protocolo foi desenvolvido pela IBM e a Eurotech nos anos 90 sendo voltado inicialmente somente para sistemas de aquisição e supervisão de dados, após um período, o MQTT teve como função também a troca de mensagens que suportariam a comunicação assíncrona.

Portanto o MQTT usa um tipo de topologia para a publicação dos dados e assinatura/leitura (publish/subscribe) dos mesmos para envio destas mensagens, tendo também como funcionalidades os seguintes aspectos:

* Ser um protocolo simples e fácil de implementar.
* Aplicação leve para comunicação remota entre dispositivos quando os dados são limitados, garantindo uma alta performance.
* Seguro em redes instáveis.
* Suporta diversos tipos de aplicações tanto para IoT, quanto para dispositivos.
* Biblioteca suportada por diversas línguas de programação

Todas as características acima motivam a implementação do MQTT pelas empresas em seus dispositivos e sistemas, devido demonstrarem a versatilidade e compatibilidade para se tornar uma boa opção.

## **Vantagens**

O MQTT é vantajoso há empresas que tenham como foco um fluxo mais amplo além da possiblidade de conexão entre máquinas e sensores, o protocolo é muito utilizado devido sua facilidade e simplicidade, além das seguintes vantagens:

* **Codificação simples:** funciona em sistemas não muito modernos e até mesmo com falhas de armazenamento, útil em redes com internet limitada.
* **Sistema não sobrecarregáveis:** somente o necessário passa pelo protocolo MQTT, assim não ocorre falhas.
* **Domínio Público:** traz mais flexibilidade e a instalação pode realizada em qualquer rede ou hardware, além de ser personalizado.
* **Segurança e qualidade**
* **Padronização:** comunicação realizada por rede cabeada ou Wi-Fi.

## **Onde o protocolo é utilizado**

O MQTT em relação a outros tipos de protocolos de mensagens corporativas tem como vantagem o seu tamanho, devido o mesmo ser menor que os demais tornando-se ideal tanto para os aplicativos móveis quanto para o IoT. Portanto, empresas como o Facebook utilizam-se deste tipo de protocolo em parte de seus aplicativos móveis, tendo em vista que o consumo de energia empregado é mais baixo. Abaixo se encontra outras empresas que utilizam o MQTT:

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Stackshare < <https://stackshare.io/stackups/mqtt-vs-#integrations> >

.

## **Rede que opera**

Assim como o AMQP, o MQTT também é operado através de redes wireless nos dispositivos IoT (Internet of Things, traduzido significa Internet das Coisas), a mesma é definida como um protocolo para troca de mensagens através da rede ela foi desenvolvida pela IBM e se tornou o principal protocolo que trabalha com a comunicação da IoT devido sua simplicidade, segurança e facilidade de implantação.

## **Informações para se utilizar**

Como já relacionado no artigo, o MQTT é um protocolo que utiliza-se do modelo publish/subscribe (pub/sub) em sistemas visando a troca de mensagens, portanto o protocolo compreende duas funcionalidades na rede: o broker (servidor) e os clientes que enviam as mensagens ao broker, que fica responsável de rotear as mesmas para outros clientes. Ou seja, o servidor broker tem como responsabilidade receber, enfileirar e disparar as mensagens recebidas deste modelo dos “pub” para os “sub”, um ponto que o desenvolvedor ressalta é que qualquer coisa que seja capaz de fazer essa interação com o broker e receber as mensagens é considerado um cliente. Portanto a relação entre estes dois pontos é baseada em:

1. Conexão realizada pelo cliente ao broker através da TCP/IP aonde o mesmo assina um “tópico” qualquer no servidor.
2. Publicação das mensagens no tópico através do envio da mensagem e o tópico ao broker.
3. Encaminhamento da mensagem pelo broker aos clientes do tópico.

Abaixo se encontra a arquitetura do protocolo MQTT:

Uma imagem contendo captura de tela

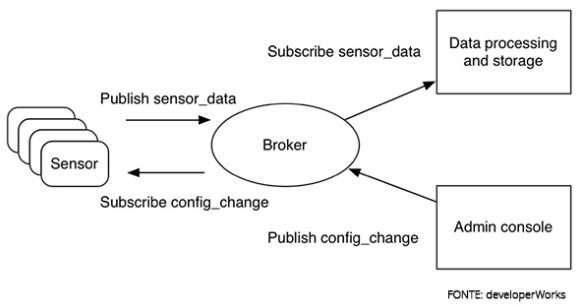
Descrição gerada automaticamente

Fonte: Buteco Open Source

< <https://butecoopensource.github.io/mqtt-parte-1-o-que-e-mqtt/> >

Desta forma, é visto que o broker implementa as estruturas de armazenagem (tópicos) destes dados, e os mesmos fazem uma referência de publicação (publish) dos dados, assim como uma leitura (subscribe) deles. Mas para que todo este processo funcione precisa-se de uma conexão que funcione, como por exemplo a TCP/IP citada acima.

De qualquer forma, a flexibilidade tida na utilização do protocolo MQTT permite ao desenvolvedor limitar a interação dos clientes, como conforme abaixo:



Fonte: IBM < <https://developer.ibm.com/components/mqtt/articles/iot-mqtt-why-good-for-iot?mhsrc=ibmsearch_a&mhq=mqtt> >

A imagem acima demonstra como a interação do cliente pode ser limitada, como por exemplo:

1. As leituras publicadas são realizadas no tópico “sensor\_data” e a assinatura é feita no tópico “config\_change”
2. Os processadores de dados salvam os mesmos em um banco de dados (back-end) com assinatura de tópico “sensor\_data”
3. Recebimento de comandos do administrador do sistema, para ajustes de configurações, sensibilidade e frequência de amostra, além de publicar todas as alterações no tópico denominado “config\_change”.

## **STOMP – Streaming de texto simples / Orientada Messaging Protocol**

O protocolo foi desenvolvido devido a necessidade das empresas de se conectarem aos intermediários de mensagens corporativas com linguagem em script, como por exemplo o Ruby, Python e Perl.

No ambiente do STOMP ocorrem operações relacionadas com as mensagens, como enviar uma com segurança e depois desconectar ou até mesmo consumir todas as mensagens de um destino, o mesmo acaba sendo uma opção entre outros tipos de protocolos de mensageria aberta e até mesmo de intermediários.

Portanto o protocolo é muito utilizado em integrações decorrentes de mensagens. A utilização de um broker que suporte este tipo de protocolo faz com que os clientes se comuniquem com o mesmo independendo da plataforma utilizada, pois a comunicação realizada é trocada a abstração textual.

## **Vantagens**

O STOMP é um protocolo baseado em texto (script), sendo comparado ao HTTP, o mesmo assim como AMQP possui um cabeçalho e corpo para a mensagem enviada em sua arquitetura. Além disso o protocolo é considerado leve e simples, sendo utilizado por diversos tipos de servidores como o RabbitMQ que tem o WEB STOMP.

O protocolo possui as seguintes vantagens:

* Projetado para ser leve e de fácil implementação.
* Servidor com variedade de idiomas.
* Interoperável.
* Não possui muitas restrições em relação a arquitetura dos servidores e não se precisa de muitos recursos.
* Nomeação de destinos.
* Semântica de confiabilidade.

## **Onde o protocolo é utilizado**

Conforme já relacionado, o protocolo assume uma rede de transmissão bidirecional, aonde o cliente e o servidor se comunicam através dos quadro STOMP enviados pelos fluxos no servidor. Abaixo se encontram alguns servidores compatíveis com o STOMP:



Fonte: Stomp < <http://stomp.github.io/implementations.html> >

E abaixo se encontram alguns clientes com as bibliotecas compatíveis com o protocolo STOMP:

Uma imagem contendo texto

Descrição gerada automaticamente

Fonte: Stomp < <http://stomp.github.io/implementations.html> >

## **Rede que opera**

Assim como o AMQP e o MQTT, o STOMP também é operado através de redes wireless, mas o mesmo é baseado em quadros, e estes mesmos quadros são modelados em HTTP, portanto um mesmo quadro pode ser um: comando, conjunto de cabeçalhos e um corpo opcional.

Toda a codificação deste protocolo é realizado em UTF-8, aonde o mesmo aceita também codificações alternativas. O servidor do protocolo é moldado como um aglomerado de destinos tendo como finalidade ser o local aonde as mensagens devem ser enviadas.

Levando-se em consideração isso, o cliente STOMP pode atuar de dois modos simultâneos, como:

* **Produtor:** envia as mensagens para um destino no servidor através do “send”.
* **Consumidor:** envia um “subscribe” para um destino e recebe mensagens do servidor como “message”.

## **Informações para se utilizar**

Como já citado no artigo, o STOMP é muito utilizado para realizar comunicações distribuídas entre computadores e aplicativos, além disso todas as mensagens são orientadas através de fluxo contínuo ou texto. Portanto, as mensagens podem ser tanto criadas, quanto enviadas por um lado e recebidas ou processadas por outro, estas mensagens podem ser qualquer tipo de texto como: xml, flat, ison e etc.

As mensagens recebidas pelo protocolo são enfileiradas, como por exemplo quando um site qualquer solicita informações do cliente e o mesmo após preencher envia ao “servidor”, após isso é enviado um e-mail com o resumo da solicitação do cliente, contudo caso aja muitos envios de formulários o servidor automaticamente diminuíra o envio dos e-mails de resumo e os clientes verão que a página está lenta, o motivo disto é que o período do envio da mensagem está sendo maior e o servidor está tentando enviar todos de forma enfileirada. Abaixo se encontra como seria a arquitetura do STOMP:

Uma imagem contendo texto, mapa

Descrição gerada automaticamente

Fonte: HojaLibre <<https://hojalibre.wordpress.com/2010/12/08/el-protocolo-de-colas-de-mensajes-stomp/>>

Se levarmos em consideração que o servidor envie a mensagem para o servidor de correio colocando-a em uma fila, o servidor da web poderá exibir uma resposta ao cliente que a mesma está sendo liberada para outros usuários para o envio de seu formulário. Só de se utilizar deste tipo de façanha o servidor da web já não ficaria lento tão rapidamente, contudo a mensagem ainda se encontrará em fila e após o processamento o serviço enviará um e-mail com as instruções.

Portanto o STOMP facilita e muito o processa de gravação de mensagens no servidor, pois o mesmo enfileira e as despacha para outro processo conforme recursos disponíveis.

## **Conclusão**

Após ser realizado a pesquisa, foi constatado que estes três tipos de protocolos com brokers são os mais utilizados no mercado atual em diferentes tipos de necessidade de serviços tido pelas empresas, como por exemplo: o AMQP é muito utilizado em empresas que realizam transações devido ter todos os recursos e ser um pacote completo de protocolo; o MQTT já é muito utilizado em dispositivos IoT (embarcados) sendo um protocolo com poucos recursos e enxuto; já o STOMP é baseado em texto, e normalmente utilizando em conjunto websockets o que permite uma visualização das mensagens na web.

Todos estes três protocolos possuem suas vantagens, e todos ganham suporte de programas de mensageria grandes como o ActiveMQ e o RabbitMQ. Devemos levar em consideração também que todos os protocolos possuem como vantagem em comum a interoperabilidade, e as diferenças entre eles são relacionadas ao tipo de projeto empregado, escala, segurança, largura de banda e etc., portanto o cliente que deve decidir qual é o mais adequado com base no que o mesmo quer.

## **Referências**

* **AMQP**

<https://www.embarcados.com.br/amqp-protocolo-de-comunicacao-para-iot/>

<https://dnunes.blog/2018/07/06/protocolos-de-menssageria-amqp-mqtt-e-stomp/>

vitormeriat.com.br/2014/11/21/mensageria-e-os-protocoloschoose-your-poison/

<http://tsdn.tecnospeed.com.br/blog-do-desenvolvimento-tecnospeed/post/arquitetura-de-microservicos-utilizando-amqp>

<https://www.devmedia.com.br/introducao-ao-amqp-com-rabbitmq/33036>

<https://medium.com/totvsdevelopers/diferen%C3%A7as-do-amqp-1-0-para-as-vers%C3%B5es-anteriores-9db828cc9e3e>

<https://www.amqp.org/product/solve>

<https://www.amqp.org/about/examples>

<https://www.amqp.org/product/overview>

<https://www.embarcados.com.br/amqp-protocolo-de-comunicacao-para-iot/>

<https://www.ime.usp.br/~reverbel/students/qualif_exams/qualificacao-thadeu.pdf>

* **Iot**

<https://news.sap.com/brazil/2016/05/o-que-e-iot-a-internet-das-coisas/>

* **MQTT**

<https://blogmasterwalkershop.com.br/outros/conhecendo-o-protocolo-mqtt/>

<https://engprocess.com.br/mqtt-broker/>

<https://engprocess.com.br/mqtt/>

<https://stackshare.io/stackups/mqtt-vs-#integrations>

<https://blogs.vmware.com/vfabric/2013/02/choosing-your-messaging-protocol-amqp-mqtt-or-stomp.html>

<https://butecoopensource.github.io/mqtt-parte-1-o-que-e-mqtt/>

<https://developer.ibm.com/components/mqtt/articles/iot-mqtt-why-good-for-iot?mhsrc=ibmsearch_a&mhq=mqtt>

* **STOMP**

<https://stomp.github.io/stomp-specification-1.2.html>

<http://thiagocolen.blogspot.com/2012/08/habilitando-stomp-transport-connector.html>

<https://unpocodejava.com/2013/08/23/eligiendo-protocolo-de-mensajeria-amqp-mqtt-o-stomp/>

<https://www.vitormeriat.com.br/2014/11/21/mensageria-e-os-protocoloschoose-your-poison/>

<http://stomp.github.io/>

<http://stomp.github.io/implementations.html>

<https://dnunes.blog/2018/07/06/protocolos-de-menssageria-amqp-mqtt-e-stomp/>

<http://thiagocolen.blogspot.com/2012/09/um-pouco-de-stomp.html>

<https://blogs.vmware.com/vfabric/2013/02/choosing-your-messaging-protocol-amqp-mqtt-or-stomp.html>

<https://hojalibre.wordpress.com/2010/12/08/el-protocolo-de-colas-de-mensajes-stomp/>