Os sistemas de mensagens das empresas permitem que os programas comuniquem através da troca de mensagens, tal como as pessoas comunicam ao trocarem emails. Mas ao contrário dos emails, os sistemas de mensagens das empresas providenciam garantias de entrega, velocidade, segurança e liberdade de spam. Até ao surgimento do AMQP, não havia nenhum standard aberto para sistemas de mensagens de empresas, portanto os programadores tinham de escrever os seus próprios, ou utilizar sistemas proprietários que são dispendiosos.

O AMQP é o primeiro sistema aberto para mensagens de empresas. Foi desenhando para suportar trocas de mensagens para qualquer aplicação distribuída. O encaminhamento pode ser flexivelmente configurável, facilmente suporta paradigmas de mensagens comuns, tais como: ponto a ponto, fanout, publish-subscribe e pedido-resposta.

O AMQP define não só a interface, mas também o comportamento dos clientes e servidores, por forma a possibilitar uma verdadeira interoperabilidade entre diferentes fabricantes.

Define o formato das mensagens a serem enviadas através do canal de comunicação.

Opera segundo dois ideias:

- Message Queues: Podem dar diferentes tipos de qualidade de serviço, exemplo: garantia de persistência de dados elevada, ou armazenamento apenas em RAM. Cliente pode especificar qual a qualidade de serviço que deseja, no acto da criação da queue.

- Exchanges: Encaminham as mensagens para queues. Têm semânticas: one-to-one, one-to-many, one-to-one-of-many. Os bindings são conjuntos de regras que definem o comportamento dos Exchanges, podendo definir que todas as mensagens são encaminhadas para uma queue, ou, que o encaminhamento é baseado no conteúdo das mensagens.

- Bindings: Um Binding especifica como as mensagens são encaminhadas de um Exchange para uma Queue.

As mensagens circulam entre brokers e clientes através de métodos assíncronos e síncronos.

Entidades relevantes, do ponto de vista de comunicação:

- Message broker: Entidade a que os utilizadores se conectam. Pode ser distribuído por várias replicadas organizadas em cluster (rede de brokers).

- Utilizador: Autenticações de acesso ao middleware

- Conexão: Ligação física, exemplo: TCP/IP

- Canal: Ligação lógica associada a uma conexão, através desta consegue-se manter estado numa comunicação.

As entidades envolvidas nas comunicações são declaradas e têm um nome, este é o atributo que permite que identificar uma entidade. Os nomes das entidades são codificados em UTF-8 e podem ter entre 1 a 255 caracteres.

## Exchanges

Entidade para onde as mensagens são enviadas.

Propriedades:

* passive: the exchange will not get declared but an error will be thrown if it does not exist.
* durable: the exchange will survive a broker restart.
* auto-delete: the exchange will get deleted as soon as there are no more queues bound to it. Exchanges to which queues have never been bound will never get auto deleted.

## Queues

As queues têm nomes e são o destino das mensagens. O message broker pode entregar as mensagens, das queues, aos clientes (push), ou os clientes podem recolher as mensagens das queues (pull). A ordem de entrega de mensagens da queue é igual à ordem de entrada das mensagens na queue.

* alternate-exchange: when message are rejected by a subscriber or orphaned by queue deletion, its message get routed to this exchange and get removed from the queue.
* passive: the queue will not get declared but an error will be thrown if it does not exist.
* durable: the queue will survive a broker restart.
* exclusive: there can only be one client for this specific queue.
* auto-delete: the queue will get deleted as soon as no more subscriptions are active on it. This shares the same constraint as the auto-delete property for exchanges: if no subscription has been ever active on the queue it will not get auto-deleted. An exclusive queue however will always get auto-deleted when the client terminates its session.

## Messagens

As mensagens não têm nomes e são publicadas para Exchanges. As mensagens são constituídas por um cabeçalho, que contém propriedades utilizadas no encaminhamento e opcionalmente pelas aplicações, e corpo, de um tamanho variável, que apenas é interpretado pelas aplicações.

* routing-key: this field is used in ways dependent on the type of the exchange.
* immediate: the message will get handled as unroutable if at least one of the queues which would receive the message has no subscription on it.
* delivery-mode: indicates that a message might need persistence. Only for such messages the broker makes a best-effort to prevent a loss of the message before consumption. If there is uncertainty on the broker's end about the successful delivery of a message (e.g. in case of errors) it might deliver a message more than once. Non persistent delivery modes do not show this kind of behavior.
* priority: an indicator (a range between 0 and 9) that a message has higher precedence than others.
* expiration: the duration in milliseconds before the broker may handle the message as unroutable.

## Bindings

Um Binding especifica como as mensagens são encaminhadas de um Exchange para uma Queue.

* Unconditional - the binding has no properties and requests "all" messages from the exchange.
* Conditional on a fixed string - the binding has one property, the **routing key** and requests all messages that have an identical routing key.
* Conditional on a pattern match - the binding has one property, the **routing key** and requests all messages that match the routing key using a pattern-matching algorithm. Arbitrary pattern syntaxes could be used. AMQP implements topic matching.
* Conditional on multiple fixed strings - the binding has a table of properties, the **arguments** and requests all messages whose headers match these arguments, using logical ANDs or ORs to combine matches.
* Conditional on multiple patterns - the binding has a table of properties, the **arguments** and requests all messages whose headers match these arguments, using a pattern matching algorithm and logical combinations.
* Conditional on algorithmic comparison - the binding has an algorithmic expression (like an [SQL](http://en.wikipedia.org/wiki/SQL) SELECT WHERE clause) and requests all messages whose headers match that expression.
* Conditional on content inspection - the binding specifies arbitrary criteria that are resolved by inspection of the actual message content.

## Encaminhamento de mensagens de Exchanges, seguindo Bindings, para Queues

As mensagens são entregues a Exchanges. Estes estão associados a Bindings, que definem o critério de correspondência com mensagens e a Queue de destino das mensagens correspondidas.

Tipos de correspondência:

* a direct exchange matches when the routing key property of a message and the key of the binding are identical.
* a fanout exchange always matches, even on bindings without a key.
* a topic exchange matches the routing key property of a message on binding key words. Words are strings which are separated by dots. Two additional characters are also valid: the *\**, which matches 1 word and the *#*, which matches 0..N words. Example: *\*.stock.#* matches the routing keys *usd.stock* and *eur.stock.db* but not *stock.nasdaq*.
* a headers exchange matches on the presence of keys as well as key–value pairs which can be concatenated with logical and–or connections in a messages header. In this case the routing key is not a criterion for matching that is considered by the exchange. Neither does the binding carry a single routing key but special format which contains header keys and / or key-value-pairs which match on the header key being present or the header key being present and the value being the same respectively.

A existência de Bindings permite tornar que as Queues e Exchanges sejam independentes. É possível ligar uma Queue, com vários Bindings, a um ou mais Exchanges. Se vários consumidores utilizarem uma mesma Queue, cada mensagem será recebida apenas por um dos consumidores. Por outro lado, se cada consumidor definir uma Queue e utilizar um Binding comum a vários consumidores, cada um dos consumidores irá receber uma cópia das mensagens correspondidas pelo Binding nas respectivas Exchanges.

## AMQP Wire-level Format

O AMQP wire-level é formato binário com características modernas: multicanal, negociado, assíncrono, seguro, portável, neutro e eficiente.

Este formato é dividido em duas camadas, sendo elas: funcional e transporte. A camada funcional define um conjunto de comandos, agrupados em classes lógicas de funcionalidade, que fazem trabalho útil em nome da aplicação. A camada de transporte é a que transporta estes métodos da aplicação para o servidor, e no sentido inverso, e que trata da multiplexagem do canal, do encapsulamento, da codificação do conteúdo, dos sinais “heart-beat”, da representação de dados e do tratamento de erros. Tanto a camada de transporte como as camadas de níveis superiores são utilizados em pirâmide, o que permite a evolução do protocolo e a adopção de tecnologias emergentes.