想象一种情景：我们有一个组件是用Java写的，而另一个组件是用PHP写的，现在其中一个要调用另一个，我们该怎么办。做个API暴露出接口，然后把调用信息传过去？如果再换一种语言重构了其中一个组件，难道还要写个API。

这里，我们就可以想象一种中间件，来帮助我们解决问题。这个中间件与平台无关，与语言无关，它仅仅作为一个容器，一个通道。可以从一端放入信息，从另一端读取。

有了这个中间件，两端的组件只需要和这个中间件打交道，根本不需要考虑其他事情，降低了耦合，减少依赖，一定程度上减少了资源的占用。

1.MQ

MQ全称为Message Queue（消息队列）是一种应用程序对应用程序的通信方法。应用程序通过读写出入队列的消息（针对应用程序的数据）来通信，而无需专用连接来链接它们。消 息传递指的是程序之间通过在消息中发送数据进行通信，而不是通过直接调用彼此来通信，直接调用通常是用于诸如远程过程调用的技术。排队指的是应用程序通过 队列来通信。队列的使用除去了接收和发送应用程序同时执行的要求。

总结：MQ仅仅是概念上的东西，RabbitMQ是一种具体的实现方式。

2.AMQP

Advanced Message Queuing Protocol

AMQP是一个提供统一消息服务的应用层标准协议，基于此协议的客户端与消息中间件可传递消息，并不受客户端/中间件不同产品，不同开发语言等条件的限制。

总结：Protocol。所以，AMQP只是一个协议，负责统一和规范行为，而RabbitMQ遵循了这个协议。

3.RabbitMQ

RabbitMQ是一个强健的，易用的，可以运行在所有主流系统下的，支持各种开发平台的开源产品。

总结：RabbitMQ就是这样的强大