AULA 04 - IoTA

INTRODUÇÃO À INTERNET DAS COISAS 2
O software por trás da IoT 2
O que é software? 2
Como fazer um software? 2
Para que servem as linguagens de programação? 2
Onde tudo começou? 3
Curiosidades 4
Evolução das Linguagens de Programação 4
Como as coisas na Internet realmente se conversam? 5
Arquitetura Cliente-Servidor 5
Protocolo TCP/IP 6
REFERÊNCIAS 10

INTRODUÇÃO À INTERNET DAS COISAS

O software por trás da IoT

https://youtu.be/PdiTSHsSWFI

O que é software?

Software é um conjunto de instruções que controlam o funcionamento de um computador.
Software é uma palavra da língua inglesa e é utilizada em oposição a hardware. Lembre que mencionamos na aula anterior que a palavra em Inglês "hard" significava algo rígido? Então, a palavra em Inglês "soft" significa o oposto de "hard", algo que não é rígido.
O Software, ao contrário do Hardware, é a parte do computador que nós não podemos tocar. São os programas e aplicativos que executam no computador, nos smartphones, tablets, etc.
Agora vamos falar um pouco sobre como criar um software.

Como fazer um software?

Como o software é um conjunto de instruções, "fazer um software" consiste em escrever instruções para que um computador as execute.
Este site do CodeIoT, por exemplo, foi programado com instruções de computador. Quando você clica em "Próximo", ele vai para a próxima página do curso. Isso acontece porque alguém escreveu instruções para que o site fizesse isso.
Mas e quem foi esse "alguém que escreveu instruções"?
As pessoas que escrevem as instruções de software são chamados programadores ou programadoras, eles criam os programas.
Agora, vamos pensar um pouco sobre como esse programador poderia dar instruções ao computador. Antes de mais nada, o programador precisa se comunicar com o computador. Para se comunicar, ele precisa usar algum tipo de linguagem -- assim como nós humanos usamos línguas faladas ou de sinais para nos comunicar. Bom, mas já que o programador precisa se comunicar com o computador, ele utiliza uma linguagem de programação.
Então é isso, uma linguagem de programação não é nada mais do que uma ferramenta que permite que o programador escreva programas. O botão de "Próximo" desta página, por exemplo, foi programado usando uma linguagem de programação chamada "Javascript". No curso "Aprendendo a Programar" da plataforma CodeIoT você vai aprender como usar a linguagem de programação "Scratch" para programar o computador.
Agora que você já sabe que para fazer um software você só precisa usar alguma linguagem de programação para instruir o computador, vamos aprender um pouco sobre a história da programação.

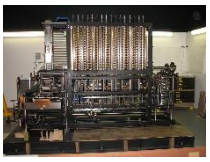
Para que servem as linguagens de programação?

Você já ouviu a expressão: "o computador é burro"? Essa expressão não é totalmente mentira, pois, um computador não é capaz de tomar decisões, ele apenas executa comandos fornecidos por pessoas que são programadores. O computador entende o que chamamos de linguagem de máquina: uma sequência de números 1 ou 0.
Todas as instruções dadas a um computador são essencialmente sequências numéricas compostas por 0 e 1 (Ex: 01101100).
Você já parou para pensar no trabalho que dá escrever um programa inteiro usando instruções compostas por combinações de 0 e 1?

É para isso que existem as linguagens de programação: para facilitar a comunicação entre pessoas e computadores, entre programadores e o hardware do computador.
O programador escreve instruções em uma linguagem bem próxima da que as pessoas usam para se comunicar. Em seguida um segundo programa, um compilador, traduz o que o programador escreveu para a linguagem de máquina, sequências compostas por 0 e 1. Esta linguagem o computador entende e consegue executar.
Há ainda uma outra situação. O programador escreve instruções em uma linguagem próxima da língua falada e um outro programa, um interpretador, traduz cada instrução e o computador a executa.
Mas quando precisamos de um compilador e quando precisamos de um interpretador? Isso depende da linguagem que utilizamos. Existem linguagens que necessitam de um compilador e outras de um interpretador. São linguagens compiladas ou interpretadas.
Agora vamos responder à pergunta inicial: Para que servem as linguagens de programação?
As linguagens de programação servem para programadores se comunicarem com computadores passando instruções com uma linguagem próxima da língua que as pessoas usam para se comunicar.

Onde tudo começou?

Você sabia que apesar de atualmente as mulheres serem minoria na Computação, foi uma mulher a primeira pessoa a desenvolver um conjunto de instruções para um computador?
O primeiro programa de computador foi criado em 1843 por Augusta Ada Byron, Condessa de Lovelace, nascida em 1815, conhecida como Ada Lovelace.
O computador para qual a Ada desenvolveu seu algoritmo ficou conhecido como a Máquina Analítica de Babbage, que é considerado o primeiro computador mecânico da história. O inventor da Máquina Analítica foi Charles Babbage, que era professor de matemática em Cambridge. Como em 1830, a taxa de erros humanos em cálculos matemáticos era gigantesca, Babbage se empenhou na criação de uma máquina que fizesse cálculos, para acabar com esse problema. Ada ajudou Charles no desenvolvimento da Máquina Analítica e escreveu um artigo com um método passo a passo para calcular a sequência de Bernouilli, conhecida também como a Lei dos Grandes Números, na máquina de Babbage.



A Máquina Analítica de Babbage era movida a vapor, executava até 60 cálculos por minuto e utilizava cartões perfurados para fazer a sua programação. Os cartões perfurados foram utilizados como forma de programar os primeiros computadores, já que eles não tinham ainda monitor e teclado.

Mas a ideia de utilizar cartões perfurados veio da área têxtil! Algumas décadas antes da criação da Máquina Analítica de Babbage e da Ada escrever o seu algoritmo, ainda em 1804, Joseph-Marie Jacquard inventou o Tear Mecânico.
Naquela época, os tecelões de seda franceses tinham que ficar trocando os fios e as linhas a cada passagem da máquina laçadeira. Isso era uma tarefa manual e muito repetitiva para os tecelões. Jacquard percebeu que as mudanças dos fios seguiam uma certa lógica e inventou um processo de cartões perfurados que definiam padrões nas laçadeiras e podiam fazer desenhos complexos. Assim, o trabalho do tecelão poderia ser trocado para algo automatizado.

O Tear funcionava da seguinte forma: um conjunto de cartões avançavam sobre uma folha giratória. Onde havia um orifício, a agulha conseguia passar, caso contrário, a agulha não passava. Dessa forma, somente trabalhavam as agulhas

coincidentes com os furos. Trata-se, portanto, de um sistema binário. Estes padrões em cartão perfurado podem ser considerados programas.
Os primeiros computadores foram inspirados pelo tear mecânico e utilizaram esta mesma estratégia. Eles tinham uma interface que recebia programas em cartões perfurados para computar informações. Há quem considere que a Máquina de Tear seja a mais antiga forma de programação criada por Jaquard em 1804, na França.

Curiosidades

Vale a pena saber...
Em 1953, a Máquina de Babbage foi redescoberta e o algoritmo que Ada propôs foi implementado. Os dois entraram para a História como o primeiro computador e o primeiro software, respectivamente.
E, em homenagem a Ada Lovelace, a primeira programadora da história, o Departamento de Defesa dos Estados Unidos registrou em 1980, a linguagem de programação ADA.

Evolução das Linguagens de Programação

Quando os computadores surgiram, a sua programação não era nada fácil. As primeiras linguagens de programação que surgiram demandavam muitos detalhes. Para poder escrever instruções para o computador, o programador precisava entender como ele foi projetado, quais módulos ele tinha e como eram suas memórias. Em outras palavras, ele precisava utilizar linguagens de programação que exigiam muito detalhamento. Essas linguagens são chamadas linguagens de baixo nível.
O Assembly, por exemplo, é uma linguagem de baixo nível, que requer muito tempo para criar um programa.
Com o tempo, as linguagens de programação evoluíram, surgiram as linguagens de programação de alto nível, que estão mais próximas à linguagem humana. Nelas, os programadores não precisam conhecer os detalhes de hardware do computador, concentrando-se mais no funcionamento do programa e menos em como ele será executado.
Linguagens como Pascal, Python, Java por exemplo, reúnem e simplificam a linguagem de máquina e fazem determinadas ações só por comandos que geralmente são palavras em inglês. Um exemplo é o comando ´print´ da linguagem Python, que é usado para exibir uma mensagem na tela do usuário.

```
print("Hello World")
```
```
Hello World
```

Neste exemplo, o comando ´print´ é utilizado para exibir a frase ´Hello World´ na tela do usuário.
Atualmente existem linguagens de programação visual, onde ao invés de usar comandos de textos em inglês, imagens e elementos gráficos são utilizados. Eles podem ser arrastados e montados em blocos que se encaixam.

Scratch é um exemplo de linguagem de programação visual. Você vai conhecer e aprender a usar esta linguagem no próximo curso do CodeIoT.
Não há dúvidas que o hardware que usamos é extremamente útil e isso graças à programação, ao processo de criação do software, que dá utilidade aos aparelhos que usamos.
Por isso, quando for usar algum software para resolver algum problema ou simplesmente quando você jogar algum game, não esqueça de agradecer aos mestres (Ada Lovelace e Jacquard), pois foram eles que geraram a ideia que serviu de base para a programação atual.

Como as coisas na Internet realmente se conversam?

https://youtu.be/WwNsbaZ-Tik

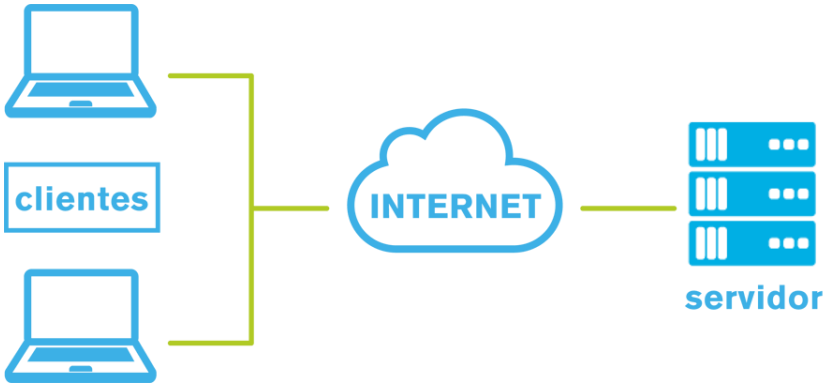
Arquitetura Cliente-Servidor

Para um computador se comunicar com outro ou um objeto inteligente trocar informações com outro, precisaremos utilizar uma arquitetura. Lembrando que uma arquitetura é um modelo que auxilia o desenvolvimento de um projeto e facilita sua implementação.
Começaremos com a arquitetura cliente-servidor. Vamos entender melhor seus conceitos e funcionamento.
A arquitetura cliente-servidor é um modelo que distribui tarefas entre fornecedores de um recurso ou serviço, denominados servidores, e os requerentes dos serviços, chamados de clientes.
Servidores e clientes são computadores ou equipamentos de hardware. Servidores prestam serviços para muitos clientes.
A arquitetura cliente-servidor parte do princípio de que um cliente inicia um pedido enviando uma solicitação para um servidor que a processa e retorna com os dados solicitados. Com base nesta resposta, o cliente pode fazer novas requisições.

- Um cliente pode ser um celular, um tablet ou um computador.
- Um servidor é um computador com grande capacidade de processamento.

Geralmente um servidor oferece vários serviços, pois isso elimina a necessidade de aquisição de vários servidores, já que esses computadores são especiais e, portanto, custam caro. Por isso, quando o cliente faz uma requisição ao servidor, ele indica uma determinada porta. Cada porta responde a um serviço.
Quando o cliente e o servidor se comunicam por meio da Internet chamamos estes serviços de serviços Web. Um dos serviços Web é o fornecimento de páginas de Internet -- ou, resumindo, sites --, as quais são acessadas por meio dos navegadores. O funcionamento é bem simples. Vamos entender por meio de um exemplo.
Um usuário acessa o endereço de uma página por meio do seu navegador web. Então seu computador, o cliente, envia uma requisição para o endereço da página que ele quer carregar. O servidor, ao receber a requisição, irá processá-la e retornar a página que o usuário quer

visualizar. O usuário pode interagir com essa página, clicando em links por exemplo, enviando assim uma nova requisição ao servidor, iniciando todo o processo novamente.
É fácil notar no exemplo que o cliente e o servidor não estão no mesmo computador. O servidor pode estar em outra cidade ou outro continente. Por isso a comunicação entre eles é feita através de uma rede, neste caso: a Internet.



Um servidor pode também fazer uma requisição a outro servidor, atuando neste caso como cliente. Mas como isso é possível? Vamos ver um exemplo.
Imagine que uma pessoa está usando o Google. Ela digita "previsão do tempo", e aperta a tecla 'Enter'. O computador dessa pessoa irá atuar como cliente e enviar uma requisição para o serviço Web no servidor do Google. Este serviço, por sua vez, irá agir como um cliente e pesquisar na Internet por sites de meteorologia. Eventualmente ele irá descobrir quais websites são relevantes e devolver uma lista com estes sites para o cliente que fez o pedido.
É importante notar nesse exemplo, que a busca nos sites é realmente executada pelo serviço Web do Google, sem interface de usuário, ou seja, é um serviço se comunicando com outro serviço. Do ponto de vista da pessoa que digitou e apertou 'Enter', vai parecer que o seu computador cliente fez a mágica de encontrar os sites, porém a verdade é que essa mágica aconteceu nos servidores do Google, que se comunicaram com outros servidores por meio da Internet.
Mas como os clientes e os servidores se comunicam? Eles utilizam uma língua como português ou inglês? Ou eles usam linguagens de programação como aprendemos na aula passada? Os clientes e os servidores se comunicam fazendo uso de protocolos de comunicação, especificamente o protocolo TCP/IP (Transmission Control Protocol/ Internet Protocol).
Em resumo:

- pessoas se comunicam entre si usando línguas como português ou inglês;
- pessoas comunicam instruções a computadores usando linguagens de programação como Pascal ou Python;
- clientes e servidores se comunicam usando o protocolo TCP/IP.

Vamos nos aprofundar um pouco mais no protocolo de comunicação TCP/IP?

Protocolo TCP/IP

Vamos entender melhor o protocolo TCP/IP.
TCP/IP é a forma de comunicação entre computadores conectados à Internet. Vamos ver os detalhes relembrando alguns conceitos que trabalhamos na primeira seção deste curso.
Todos os computadores possuem um endereço numérico único chamado endereço IP e além deste endereço, possuem também inúmeras portas onde as aplicações e os processos se comunicam.
Imagine que o cliente, com endereço IP 177.175.79.80 queira, através da porta 65, iniciar uma conexão com o servidor de endereço IP 185.187.87.88 na porta 80 para obter a página inicial do Facebook.

Como será que o computador cliente sabe o endereço IP e a porta de serviço que deve se conectar no computador servidor para obter a página inicial do Facebook se nada disso foi informado?
A única informação passada foi o endereço do Facebook na barra de endereços do navegador, ou seja, www.facebook.com
Acontece o seguinte: quando um computador está conectado à Internet, ele está configurado para acessar um servidor especial chamado servidor de nomes ou servidor DNS (Domain Name System), como é mais conhecido. Este servidor funciona como uma lista de endereços.
Quando digitamos o endereço www.facebook.com na barra de endereços, estamos informando o endereço URL (Uniform Resource Locator) do site.
Se o navegador não conhecer o endereço IP para esta URL, então ele se conecta com o servidor DNS e pergunta:

- *“Olá. Tenho a URL www.facebook.com, você pode me informar qual é o endereço IP dela?”*

O servidor então responde:

- *"Tenho sim, o endereço IP desta URL é 185.187.87.88."*

Portanto, o servidor DNS pega o endereço URL e converte em um endereço IP para descobrir em qual computador essa página está armazenada.

Mas e a porta? Como o cliente sabe em qual porta deve se conectar?
Para responder essa pergunta, imagine o seguinte cenário:
Uma pessoa passa pelo seu bairro e te pergunta: “Onde fica o metrô?”. E você prontamente responde: “O metrô fica no final desta rua.”. Isso é automático para você, pois o metrô sempre esteve no mesmo lugar desde quando você era criança e dificilmente vai mudar de lugar. E se isso acontecer, os interessados são sempre informados.
O mesmo ocorre com as portas disponíveis no computador. Elas são conhecidas de acordo com o serviço que oferecem.

Veja alguns exemplos:

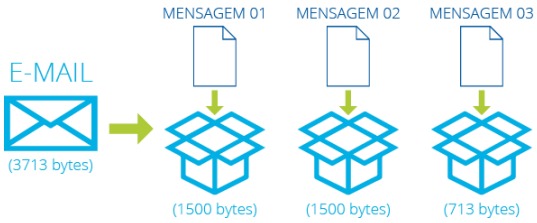
- Se precisar de um serviço de transferência de arquivos, ele pode ser encontrado na porta 21.
- Se precisar de um serviço de entrega de e-mail, ele estará na porta 25.
- Se precisar de um serviço de entrega de páginas Internet, ele estará na porta 80.

E por aí vai...
Agora que o computador do cliente já conhece o endereço IP do destino e a porta na qual ele deseja se conectar, o cliente precisa estabelecer uma conexão com o servidor.

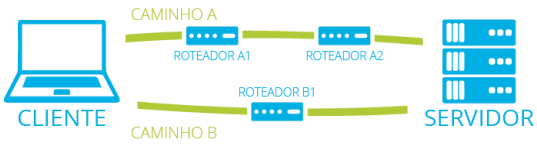
A conexão é estabelecida da seguinte forma:

- Cliente: Olá 185.186.87.88, desejo estabelecer uma conexão na porta 80.
- Servidor: Olá 177.178.79.80. Pode realizar a conexão.
- Cliente: Ok, iniciarei a conexão.

Depois de estabelecer uma conexão entre cliente e servidor, eles trocam mensagens. As mensagens trocadas ocorrem por meio de pacotes. Os pacotes são fragmentos menores dos dados que trafegam na rede. Por exemplo, uma mensagem de e-mail pode ser dividida em vários pacotes. Cada pacote contém informações como o endereço de origem, o endereço de destino e a sequência que os pacotes devem ser reconstruídos ao chegar no seu destino.



Uma característica importante é a garantia de entrega, graças ao protocolo TCP/IP, pois todos os pacotes que saem da origem possuem a garantia de que chegarão ao seu destino e que serão entregues de forma ordenada e sem modificações.
Cada pacote pode viajar pela rede por diferentes caminhos, sendo enviados de um computador para o outro, mais ou menos na direção do seu destino.

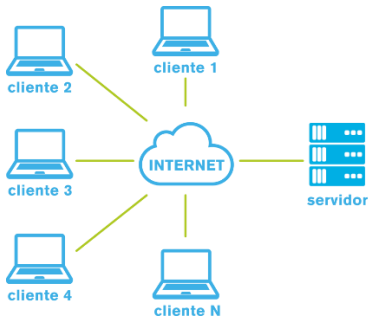


Mas quem decide qual é o melhor caminho? Os roteadores. São dispositivos capazes de realizar a comunicação entre várias redes que compõem a Internet. O roteador analisa cada pacote de informação e decide a melhor rota até o seu destino. Quando os pacotes chegam ao seu destino, a informação original é reconstituída.

O TCP/IP, portanto, é um protocolo de rede que permite a comunicação entre computadores. Uma conexão deve ser estabelecida antes do início do envio de pacotes. Trata-se de um protocolo ponto-a-ponto, possui garantia de entrega de pacotes de forma ordenada e sem modificações e possui controle de fluxo.
Vamos agora falar de servidores Web.

Servidor Web

Existe um serviço conhecido como Servidor Web (também conhecido como servidores HTTP) que recebe e manipula todos os pacotes que vem pela porta 80, ou seja, são responsáveis pela troca de solicitações de páginas de Internet entre usuário e servidor.
Servidores Web permitem armazenar um conjunto de páginas Web. É por meio destes servidores que sites são disponibilizados para os usuários que os acessam.
Em outras palavras, o Servidor Web é o responsável por disponibilizar as páginas Web de cada site.

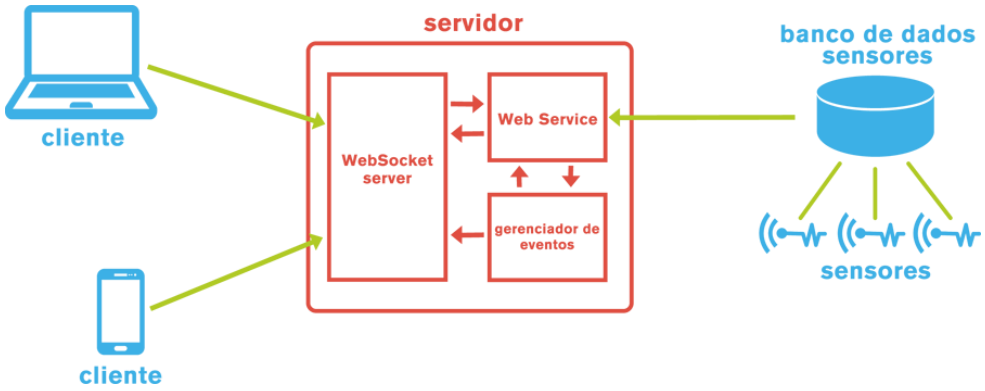


Funciona de forma bem simples. Vamos entender com um exemplo.

Quando um usuário abre um navegador, digita o endereço de uma página Web e pressiona uma tecla para carregá-la, o navegador envia uma requisição HTTP para o endereço digitado. O Website hospedado no servidor recebe a requisição HTTP e responde com a página solicitada. Essa página é o conteúdo de um arquivo de extensão .html ou .php que é interpretada pelo navegador e exibida na tela do usuário.
Geralmente um Servidor Web é implementado com diferentes tecnologias como PHP (Hypertext PreProcessor) ou servidores ASP (Active Server Pages) que permitem criar aplicações com conteúdo dinâmico, juntamente com um servidor de Banco de Dados (como MySQL ou Oracle) responsável pelo armazenamento de informações. Se o servidor Web fosse implementado sozinho ele seria capaz apenas de exibir páginas HTML estáticas.
Vamos ver agora o que é a arquitetura Publish/Subscribe.

Publish/Subscribe

Como vimos, na arquitetura cliente/servidor tradicional o cliente é o responsável por fazer uma requisição e aguardar a resposta do servidor. Já na arquitetura Publish/Subscribe, o conceito é outro. Vamos entender como funciona.
A arquitetura Publish/Subscribe é um modelo de comunicação entre computadores. Neste modelo consumidores (subscribers – assinantes) expressam seu interesse em um ou mais serviços, para que possam vir a ser notificados quando informações geradas nesses serviços por um produtor (publisher – publicador) combinam com as informações de seu interesse.
Podemos dizer que a arquitetura Publish/Subscribe inverte o modelo tradicional do cliente solicitando um serviço. Neste modelo é o servidor que inicia a comunicação e envia dados aos clientes que assinaram o serviço. Neste modelo, é o servidor que empurra os dados para os clientes, utilizando uma arquitetura padrão chamada Publish/Subscribe.



Vamos ver um exemplo.
Produtores de informações como sensores são os publishers e publicam suas leituras em um Banco de Dados. As leituras são enviadas do Banco de Dados para um gerenciador de eventos localizado no servidor. Consumidores de informações interessados assinam esse gerenciador indicando para quais sensores desejam receber leituras. E desta forma, se tornam assinantes ou subscribers. O gerenciador de eventos recebe as leituras dos sensores e as envia para seus respectivos clientes assinantes.
O uso de um gerenciador de eventos como mediador proporciona um desacoplamento entre produtores (publishers) e assinantes (subscribers). Esse desacoplamento elimina qualquer dependência entre eles, aumenta a escalabilidade (a adição ou remoção de produtores ou assinantes é independente da interação) e o ambiente dinâmico resultante se adapta bem em redes móveis onde há frequentes conexões e desconexões.

REFERÊNCIAS

- Material integralmente extraído e adaptado do curso **Introdução à Internet das Coisas**, da plataforma 'Code IoT', criado em parceria com a Samsung e LSI-TEC Escola Politécnica da USP - https://codeiot.org.br/courses/course-v1:LSI-TEC+IOT101+2021_OC/about - Acessado em 22/02/2022.
    - Licença disponível em https://creativecommons.org/licenses/by-nc-nd/4.0/deed.pt_BR - Acessado em 22/02/2022.
    - Plano de aula disponível em https://codeiot.org.br/assets/courseware/v1/bb8646d5c21172edc03e5c48a639ecba/asset-v1:LSI-TEC+IOT101+2021_OC+type@asset+block/Plano_de_Aula_IoT_Semana_3.pdf - Acessado em 22/02/2022.
    - Internet das Coisas: da Teoria à Prática http://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf