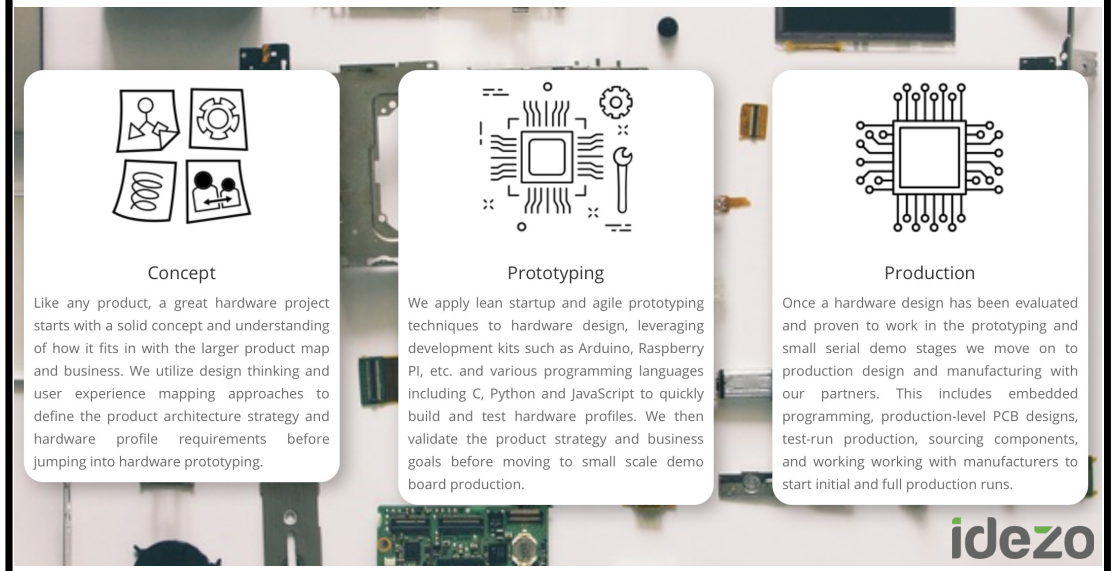
HANDS ON APPROACH TO

Data Science

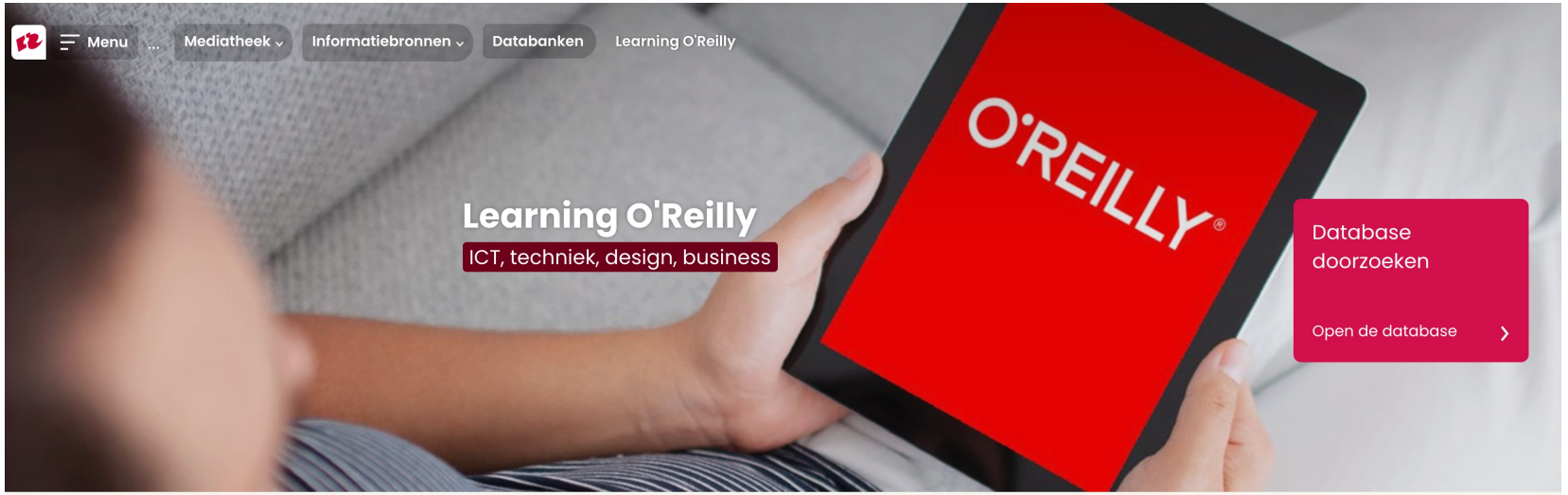
for (the)

IoT



Rob van der Willigen

https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT



Learning O'Reilly is een Engelstalig online leerplatform gericht op ICT vaardigheden. Het biedt tienduizenden e-books en vele video's, case studies en "learning paths", waarbij je zelf je voortgang kunt monitoren.

Toegang

Via de website

De Learning O'Reilly website is gekoppeld aan de Hogeschool Rotterdam login. Kom je toch een O'Reilly inlogscherm tegen, vul dan alleen je @hr.nl e-mailadres in en klik op **Sign In with Single Sign On**. Krijg je een blanco pagina te zien? Schakel eventuele adblockers uit!

Via de O'Reilly app

Leer wat je wil, waar je wil. Met de O'Reilly app download je e-books en ga je verder met een playlist waar je op een ander apparaat gebleven was.

1 GET IT ON Google play Download on the App Store

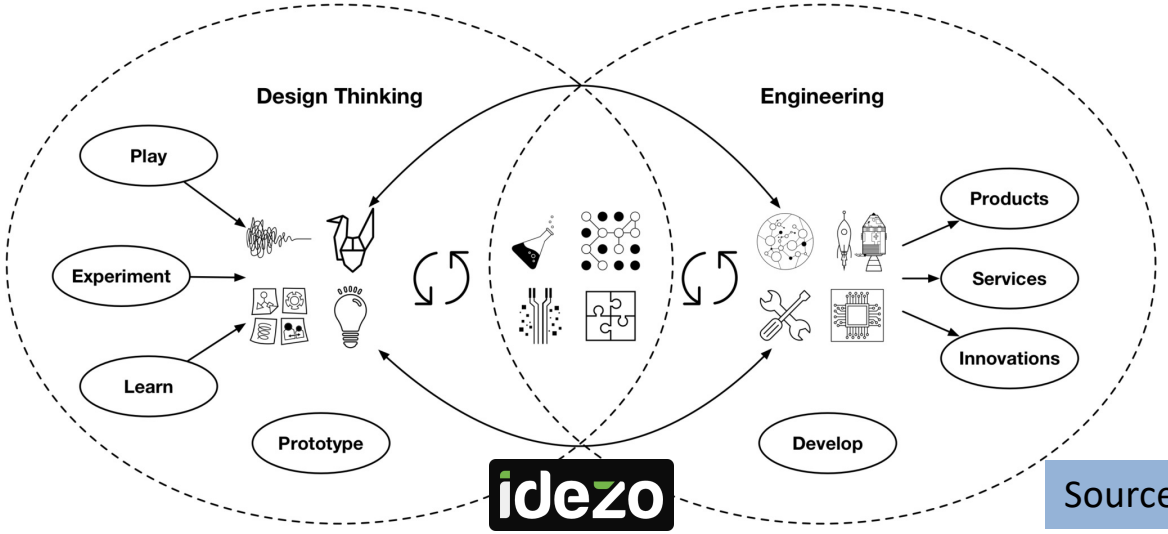
2 Gebruik je @hr.nl e-mailadres om in te loggen.

Inhoud

- Meer dan 8000 instructievideo's en 42.000 (hoofdzakelijk Engelstalige) e-books.

https://www.hogeschoolrotterdam.nl/voorlichting/voorzieningen/mediatheek/informatiebronnen/databases/mtdb/learning_oreilly/184698/

HANDS ON APPROACH TO DATA SCIENCE for (the) IoT



Source: https://idezo.ch

This Data Science Course was developed for Hogeschool Rotterdam **(Rotterdam University of Applied Sciences, RUAS) through the Program for AI & ethics (SPAiCE)**.
If you find errors or omissions, please contact the author, Rob van der Willigen, at r.f.van.der.willigen@hr.nl. Materials of this course and code examples used will become available at:
https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT

Course Setup

HOGESCHOOL ROTTERDAM

Lesson 01:	week 02	**Discovering the IoT Data Science Domain**
Lesson 02:	week 03	**Defining project requirements**
		+ Cost estimate
Lesson 03:	week 04	**Learn to write code**
Lesson 04:	week 05	**Data Science: How to start your own IoT Project**
Lesson 05:	week 06	**IoT Platforms & MiddleWare**
Lesson 06:	week 07	**Core IoT Concepts + Code-testing via IoT** middelware
Lesson 07:	week 08	**Explaining Grading + Summary + Q & A**

Week 08:	**FEEDBACK**
Week 10:	**Submit your IoT-Project via LMS**

About this course

HOGESCHOOL ROTTERDAM

Data Science for (the) IoT is about …

Experimenting, prototyping & Iterating

Concepting, Testing and Debugging

Reusing and Remixing of Code (Python)

Abstracting and Documentation

Keeping a (GitHub) Journal + code commenting

Documentation

Domain knowledge

Prototype-Iteration

Concept + Design Scenarios

Code / Coding (algorithms)

Feedback

IoT-project questions that need to be answered

Show me what you are working on?

What is it about?

What makes it a Data Science for IoT project?

What steps did you take to plan the project?

What coding IDE did you use?

What hardware did you use and how is it connected to the internet?

Which IoT platform + IoT data pipeline did you choose & why?

Did the plan change at all over time?

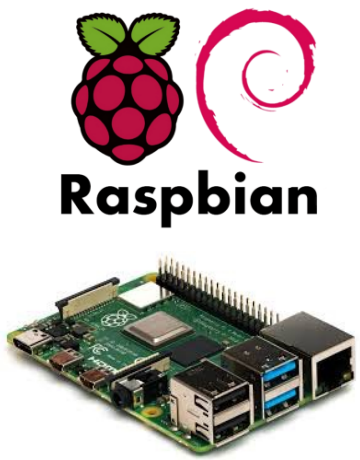
What did you do when you had to change the plan and how did the program then work?

Define for yourself

If someone new came to our class and asked you about data science for IoT:

what it is, what does it do?

how does it work?



Hardware
Microcontroller technology
Sensor technology
Hardware integration
Physical design (3d)
Testing

Software
Web application development
Microcontroller programming
Software integration

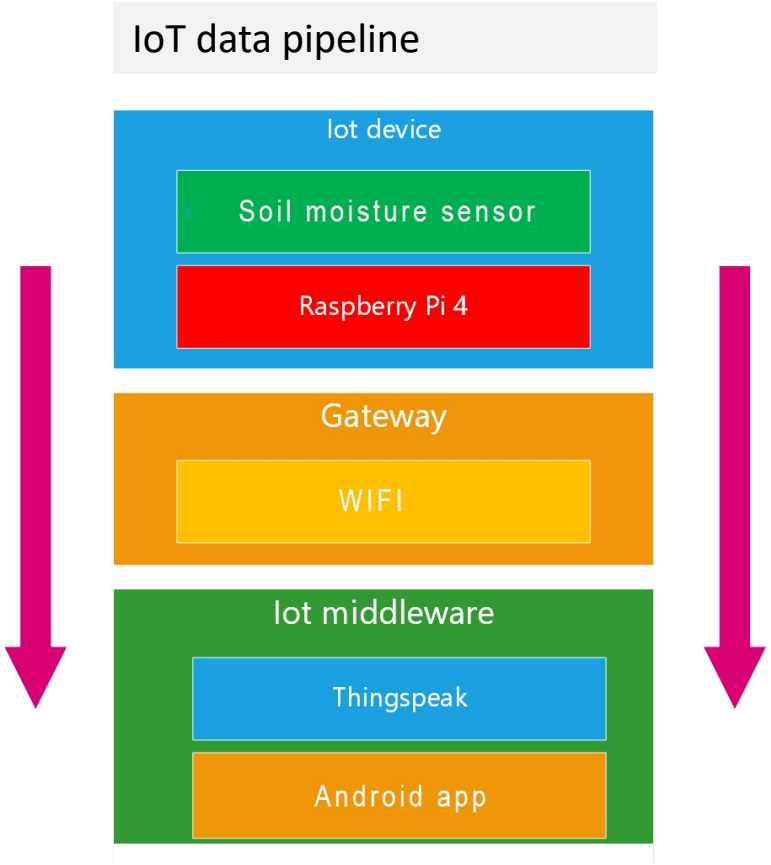
IoT-project → IoT data pipeline

Imagine you would like to know the weather outside in your garden.

In an ideal IoT world, you would have a range of sensors, for example, a temperature sensor that can talk directly to the Internet.

This would be achieved by sending the temperature data from the sensor directly to a backend service via the Internet.

The backend service (IoT middleware) would probably display the data on a web page and would ideally offer the ability to display and analyze the data on your phone.



Learning Objective

During the course, students will:

- Create + Describe a Internet of Things (IoT) concept/scenario
- Become familiar with the concepts surrounding IoT applications by building them such as Data Science & Computational Thinking
- Learn to design and develop simple lightweight IoT prototypes
- Learn to work independently to prepare a interactive prototype of an IoT product

HOGESCHOOL ROTTERDAM

Deliverables:

- A working prototype that includes software (code), hardware and electronics elements.
- Code for the application
- Documentation + Evidence of the IoT project (contributed to GitHub).
- Demonstration of the completed project (annotated video)

Choosing an IoT-project

Think Big / Act Small

The main objective is to choose something you ***care*** about.

Having a project that's genuinely interesting to you is going to be best.

Keep in mind that it should be reasonably well scoped.

Choose something small, discrete and easy to implement

Identify a small solvable problem that illustrate your key IoT idea/concept.

Keep it constrained but conceptually interesting.

Provide a IoT data pipeline

Submitting your IoT-project

Documentation should be posted to GitHub as a new Repository.
Hands-on guide for on working with this platform can be found here.

You should document your project thoroughly including:

- *A description of the concept and goals (include video and/or diagrams as needed.)*
- *A description of relevant prior projects, approaches or methods you researched and that inspired the project. Be thorough and show what informed the project.*
- *The process you underwent to reach the outcome (experiments, hacks, tests, refinments, iterations, failures)*
- *The outcome itself and how it works. Include supporting images, a video of the working prototype, circuit diagrams, etc.)*
- *Outline next steps and future directions.*
- *Add/upload code (with comments) and any supporting documentation and files.*

Grading

The most important thing for this project is to come up with a **compelling concept** - something interesting, informed, aware or critical.

The breakdown of the grades favors your ability to come up with an idea like this. Focus on the idea before implementation!

This IoT course is a way for you to showcase:

- Your engagement and understanding of the Data Science for IoT domain
- Your creative capacity + explanaition of the IoT-concept/scenario
- Your own development in technical (coding-)skills

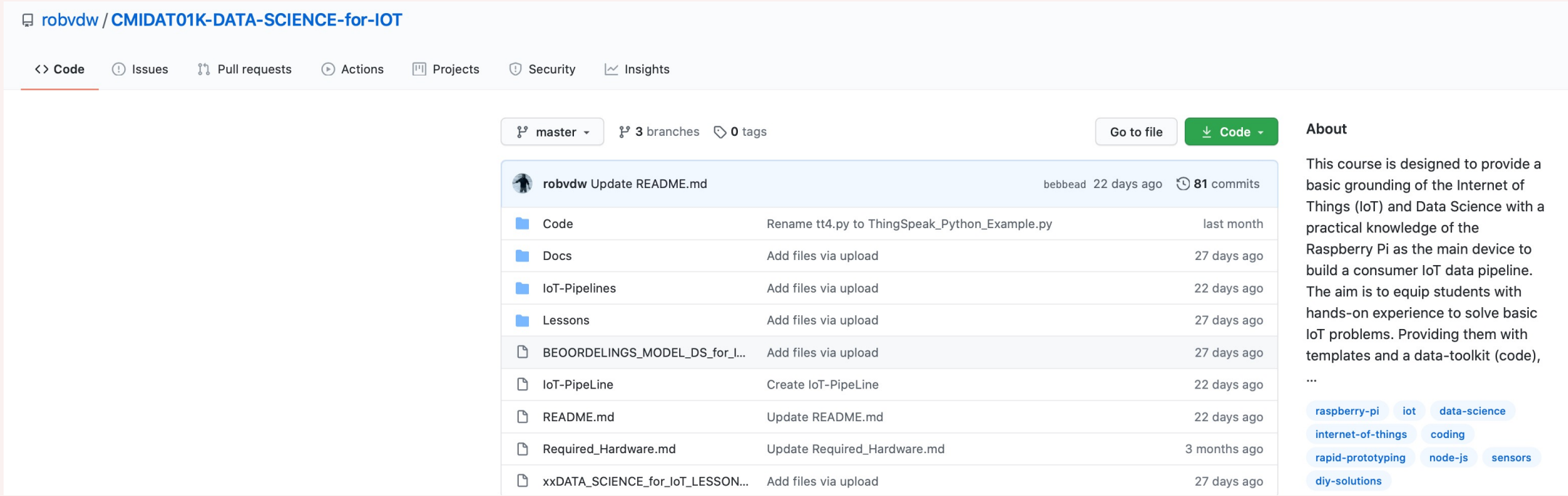
Grading

A strong grade will result by create interesting, well-crafted and well documented projects.

As such, each creative project will be graded as follows:

- 30% - Technical Implementation - Quality of code and execution of the outcome
- 20% - Creativity of Approach and Topic - Merit, Coding, and Context for the outcome/proposal
- 20% - Outcome Documentation - Well illustrated with appropriate use of code, video, diagrams, repeatability, etc.
- 20% - Git-Hub Repository and/or Maker Meet-Up Demonstration - high quality and well narrated demonstration of the solution
- 10% - Process - Description of process (ideation, iteration, etc.) and personal reflection on learning outcomes of technical skills

Materials & Guidelines



https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT

Materials & Guidelines

README.md

Goal & Aims

This Do-it-Yourself (DIY) course is designed to provide a basic grounding of the Internet of Things (IoT).

- The aim is to equip students with the Data Science skills in order to complete basic IoT-scenario's.
- Providing students with templates and a data-toolkit (code), which can be implemented through popular single board microcontrollers (Arduino / Raspberry Pi).

Required Hardware

See: https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT/blob/master/Required_Hardware.md

Scope

A hands-on introductory course exploring the Internet of Things form a (Human-Centered) Data Science point of view.

Creating your own IoT data pipeline https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT/blob/master/IoT-Pipelines/data_pipeline%20example01.png

Learning objectives

- Become familiar with Data Science concepts surrounding IoT applications
- Prepare a (Rapid) prototype for an Internet of Things (IoT) data-pipeline scenario's
- Learn to design and develop simple lightweight IoT-prototypes
- Learn to report on IoT projects (inclusive prototypes + data-pipelines).

Lessons

The content of each lesson will become avaiable each week as a PDF document via:
https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT/tree/master/Lessons

Delivearbles

- A working prototype that includes software (code), hardware and electronics elements.
- Code (commented) for the application
- Documentation of the project (contributed to a personal GitHub Repository).
- An online demo (video or IoT Platform implementation) of the completed IoT data-pipeline/project.

Grading

This course wil be graded acoording to the rules and defintions as oulined in:
https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT/blob/master/Docs/BEOORDELINGS_MODEL_DS_for_IoT_V24_npv_2020.pdf

Data Science Beyond the IoT

For those who want to keep up with novel delelopments within the Data Science domain see my Medium.com Blog: https://robfvdw.medium.com

https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT

lesson one

HOGESCHOOL ROTTERDAM

HOGESCHOOL ROTTERDAM

LES: 01

IoT + DataScience Domain explained

Raspberry Pi

Assignment

Preview Les 02

Data Science

HOGESCHOOL ROTTERDAM

Waarom Data Science?

Veel van de huidige
maatschappelijke en wetenschappelijke vraagstukken zijn
dermate complex
dat zij niet zonder de hulp van computertechnologie opgelost kunnen worden.

Bij deze vraagstukken is de rekenkracht van de computer nodig om tot een oplossing te komen.

Fundamentals

Data Deluge: to much data

1 kilobyte	1,000
1 megabyte	1,000,000
1 gigabyte	1,000,000,000
1 terabyte	1,000,000,000,000
1 petabyte	1,000,000,000,000,000
1 exabyte	1,000,000,000,000,000,000
1 zettabyte	1,000,000,000,000,000,000,000

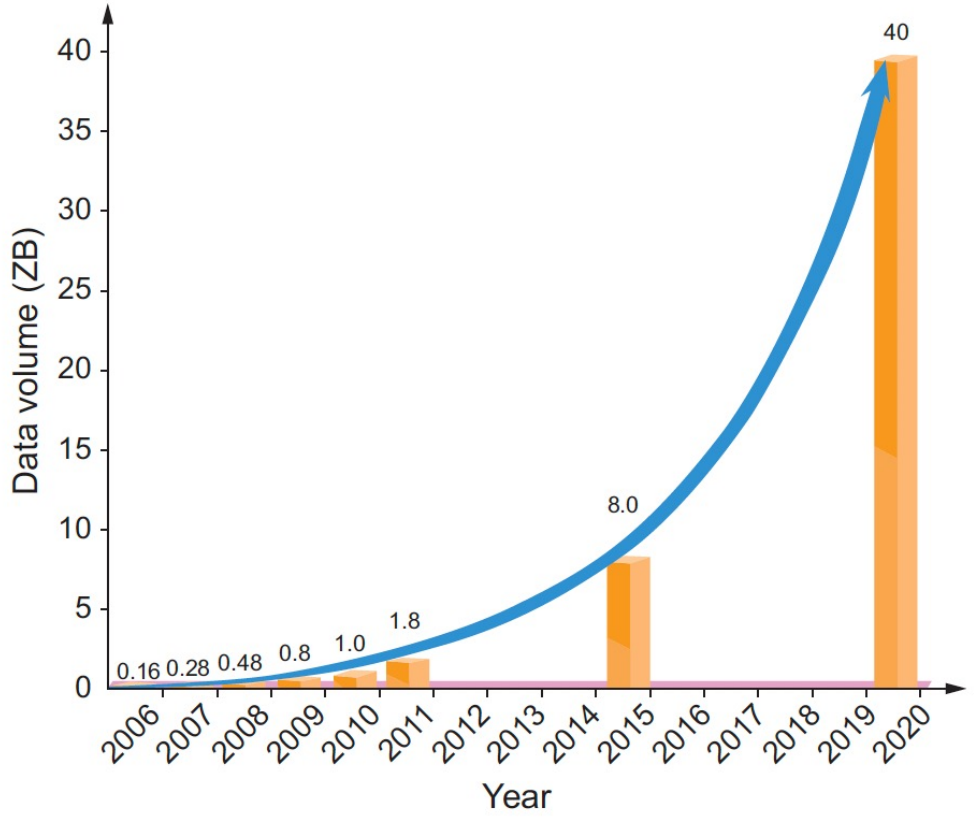
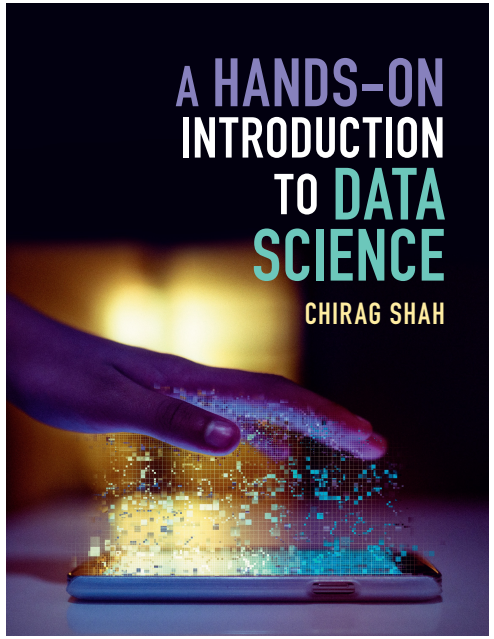
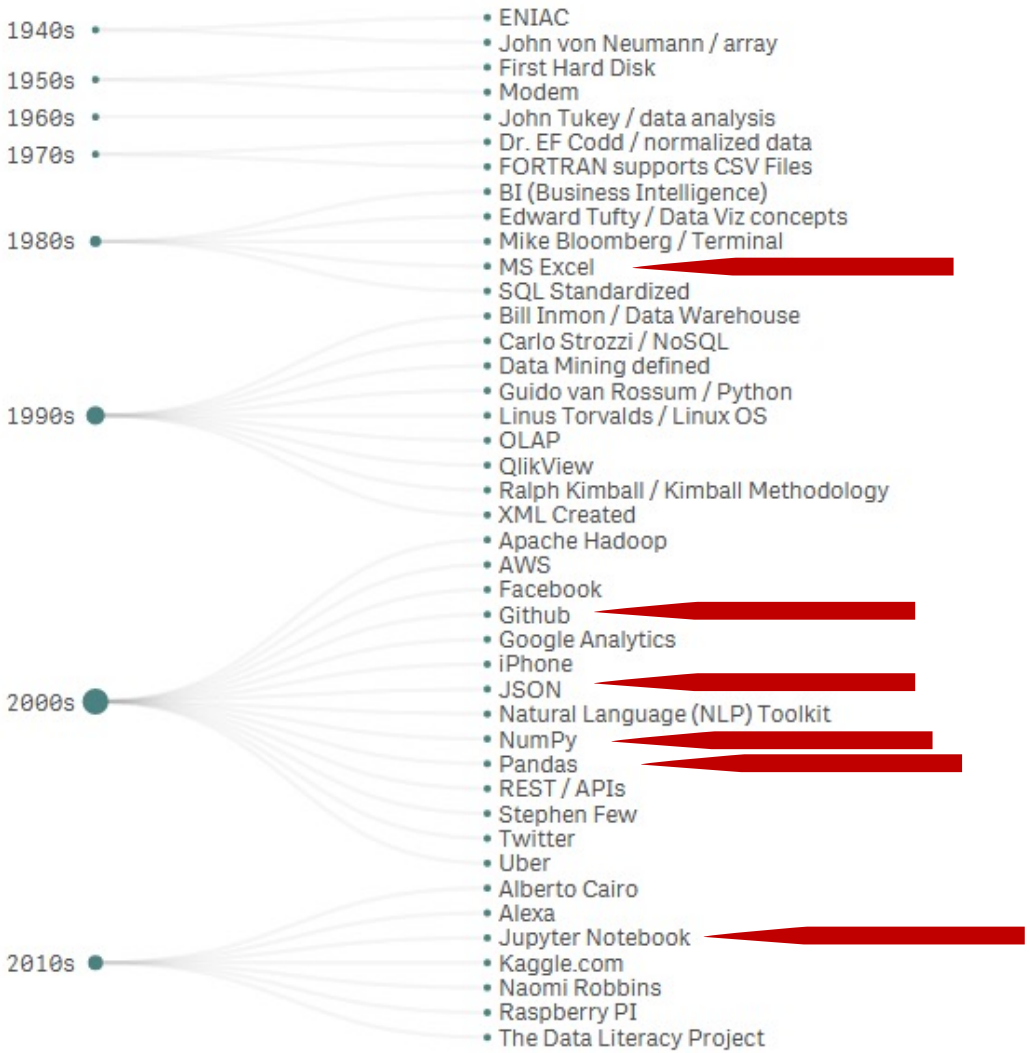


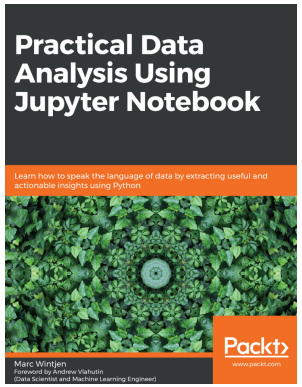
Figure 1.1 Increase of data volume in last 15 years. (Source: IDC's Digital Universe Study, December 2012.[5])



Fundamentals

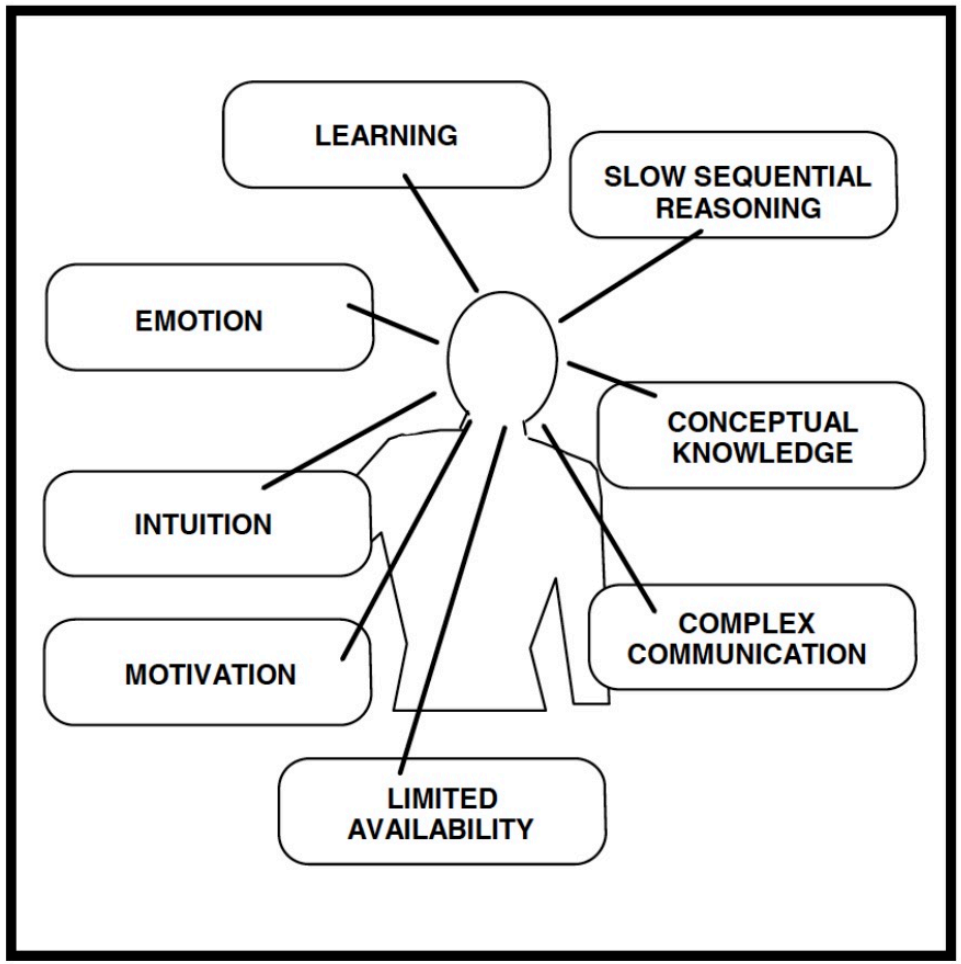
DATA Science Timeline
gives
Historical Context



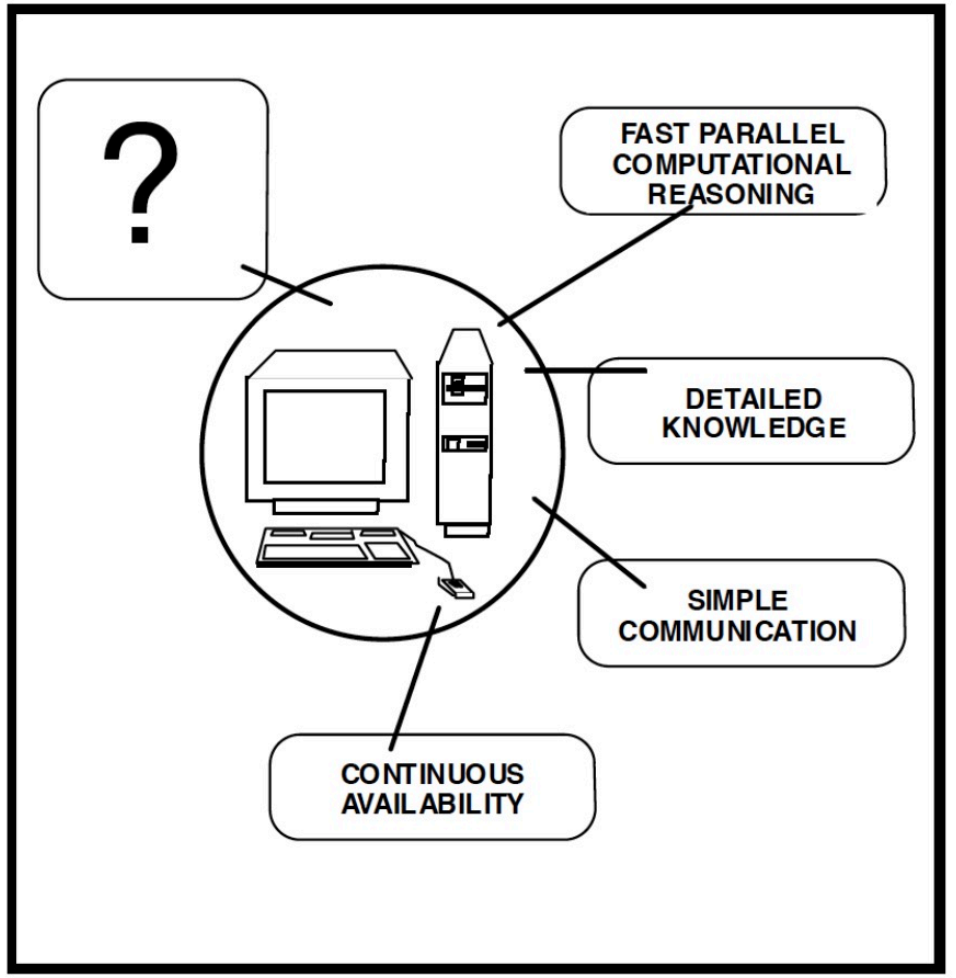


Fundamentals

Humans are pattern-driven



Computers are data-driven



WHAT IS DATA?

HOGESCHOOL ROTTERDAM

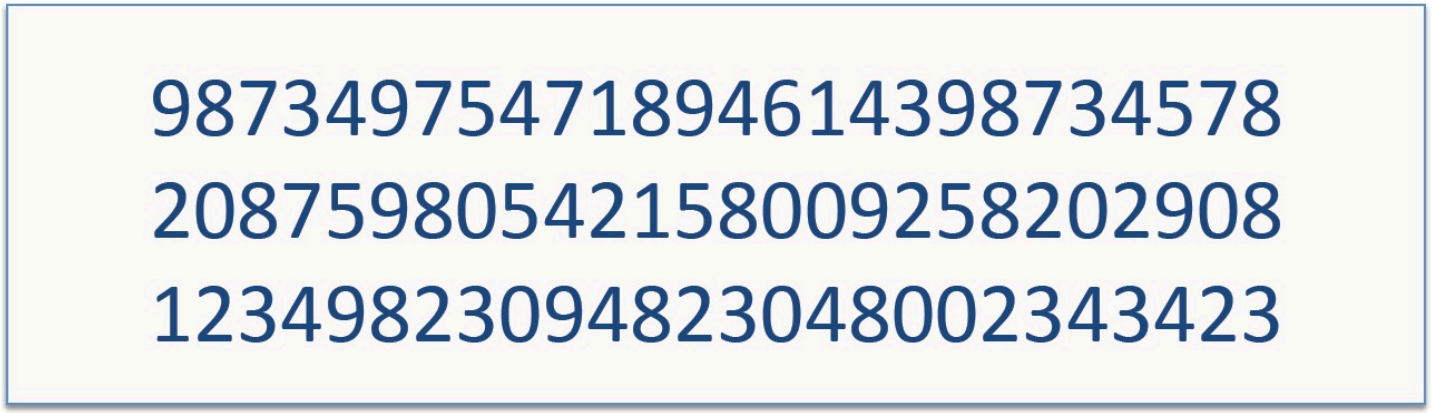
Data [gegevens]

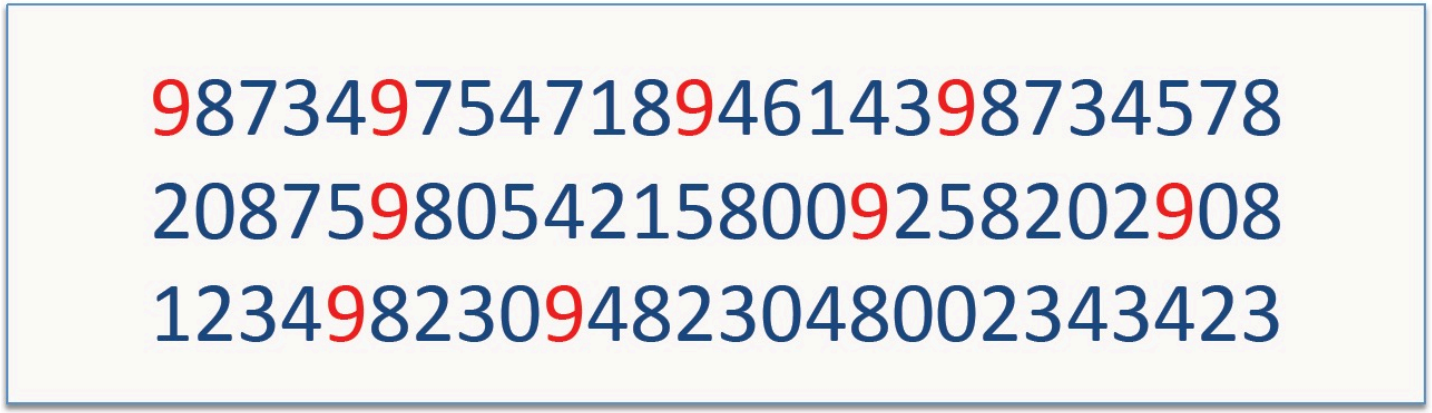
Raw Facts
No Context
Numbers
Symbols

Data comes from the Latin word, "datum," meaning a "thing given."

Although the term "data" has been used since as early as the 1500s, modern usage started in the 1940s and 1950s as practical electronic computers began to input, process, and output data.

https://robfvdw.medium.com/a-generic-approach-to-data-driven-activities-e54144a509a6



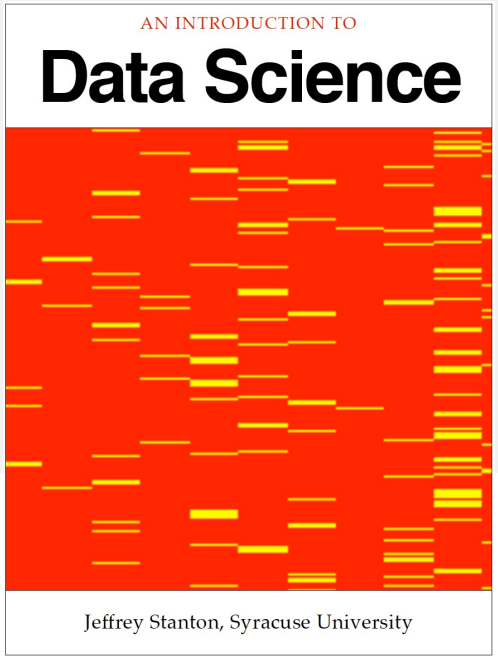


Wat is DATA?

HOGESCHOOL ROTTERDAM

Data [gegevens]

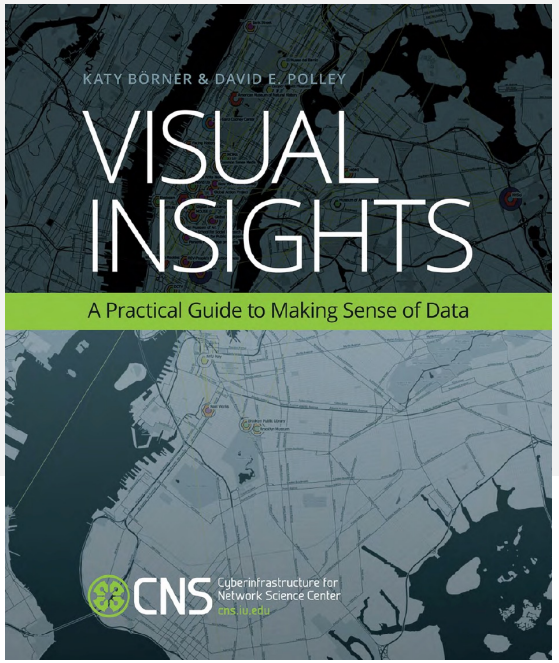
Raw Facts
No Context
Numbers
Symbols



Information [informatie]

Data with structure = processed data
Reduces Uncertainty about Data

- Summarised
- Organised
- Analysed



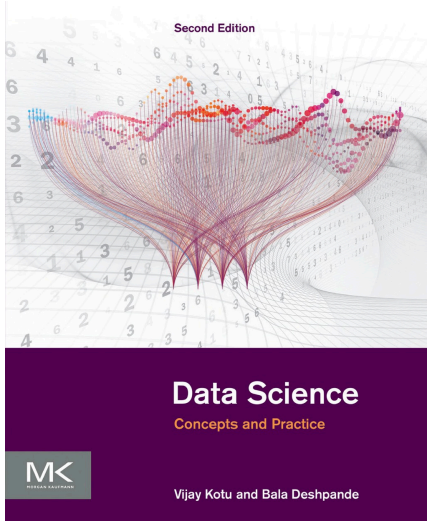
Data Science [**DS**]

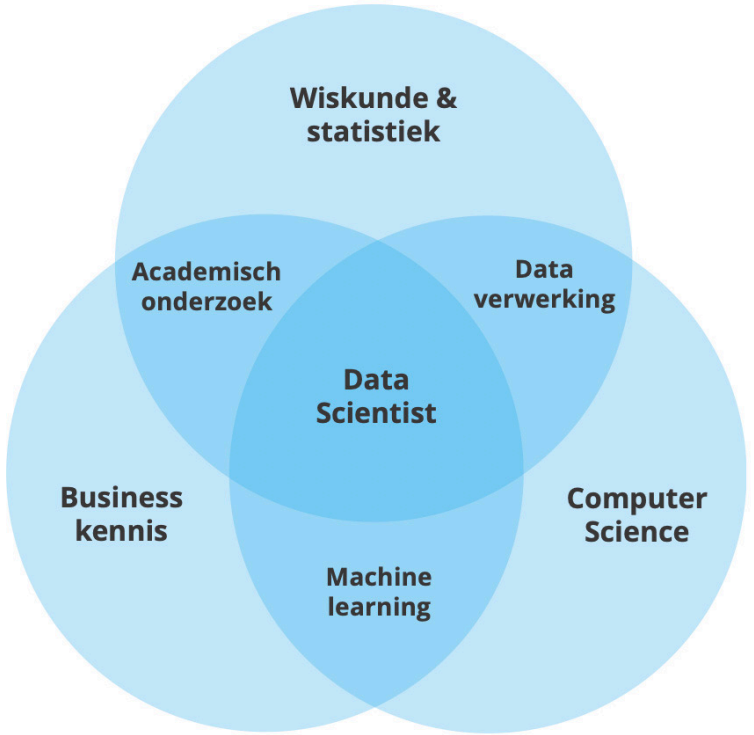
DS is een cross-disciplinair vakgebied dat zich richt op het verkrijgen van inzichten (value) uit data.

Data wetenschappers gebruiken

programmeervaardigheden,
statistische kennis, &
sector- en organisatiekennis

om verbanden te leggen en waardevolle inzichten te destilleren
op basis van **modellen & algoritmen**.


Second Edition
Data Science
Concepts and Practice
Vijay Kotu and Bala Deshpande


Wiskunde & statistiek
Academisch onderzoek
Data verwerking
Data Scientist
Business kennis
Computer Science
Machine learning

Fundamentals

Machine

Man

Data Management

Human Cognition

Data Mining

Perception

Machine Learning

Visualization

StoryTelling

Business Intelligence

Decision Making

Theory

Statistics

Inspired by Daniel Keim, "Visual Analytics: Definition, Process, and Challenges"

HOGESCHOOL ROTTERDAM

Thinking like a data scientist means more than being able to program a computer.

It requires thinking at multiple levels of abstraction called Computational Thinking

Viewpoint | Jeannette M. Wing

Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

Computational thinking builds on the power and limits of computing processes, whether they are executed by a human or by a machine. Computational methods and models give us the courage to solve problems and design systems that no one of us would be capable of tackling alone. Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? and What can computers do better than humans? Most fundamentally it addresses the question: What is computable? Today, we know only parts of the answers to such questions.

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, what is appropriately incestuous about this vision is that computing and computers facilitate the spread of computational thinking.

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.

Having to solve a particular problem, we might ask: How difficult is it to solve? and What's the best way to solve it? Computer science rests on solid theoretical underpinnings to answer such questions precisely. Stating the difficulty of a problem accounts for the underlying power of the machine—the computing device that will run the solution. We must consider the machine's instruction set, its resource constraints, and its operating environment.

In solving a problem efficiently, we might further ask whether an approximate solution is good enough, whether we can use randomization to our advantage, and whether false positives or false negatives are allowed. Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation.

Computational thinking is thinking recursively. It is parallel processing. It is interpreting code as data and data as code. It is type checking as the generalization of dimensional analysis. It is recognizing both the virtues and the dangers of aliasing, or giving someone or something more than one name. It is recognizing both the cost and power of indirect addressing and procedure call. It is judging a program not just for correctness and efficiency but for aesthetics, and a system's design for simplicity and elegance.

Computational thinking is using abstraction and decomposition when attacking a large complex task or designing a large complex system. It is separation of concerns. It is choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable. It is using invariants to describe a system's behavior succinctly and declaratively. It is having the confidence we can safely use, modify, and influence a large complex system without understanding its every detail. It is

LISA HANEY

HOGESCHOOL ROTTERDAM

Computational thinking is a way of solving problems, designing systems, & understanding human behavior that draws on concepts fundamental to data science.

Viewpoint | Jeannette M. Wing

Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

Computational thinking builds on the power and limits of computing processes, whether they are executed by a human or by a machine. Computational methods and models give us the courage to solve problems and design systems that no one of us would be capable of tackling alone. Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? and What can computers do better than humans? Most fundamentally it addresses the question: What is computable? Today, we know only parts of the answers to such questions.

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, what is appropriately incestuous about this vision is that computing and computers facilitate the spread of computational thinking.

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.

Having to solve a particular problem, we might ask: How difficult is it to solve? and What's the best way to solve it? Computer science rests on solid theoretical underpinnings to answer such questions precisely. Stating the difficulty of a problem accounts for the underlying power of the machine—the computing device that will run the solution. We must consider the machine's instruction set, its resource constraints, and its operating environment.

In solving a problem efficiently, we might further ask whether an approximate solution is good enough, whether we can use randomization to our advantage, and whether false positives or false negatives are allowed. Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation.

Computational thinking is thinking recursively. It is parallel processing. It is interpreting code as data and data as code. It is type checking as the generalization of dimensional analysis. It is recognizing both the virtues and the dangers of aliasing, or giving someone or something more than one name. It is recognizing both the cost and power of indirect addressing and procedure call. It is judging a program not just for correctness and efficiency but for aesthetics, and a system's design for simplicity and elegance.

Computational thinking is using abstraction and decomposition when attacking a large complex task or designing a large complex system. It is separation of concerns. It is choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable. It is using invariants to describe a system's behavior succinctly and declaratively. It is having the confidence we can safely use, modify, and influence a large complex system without understanding its every detail. It is

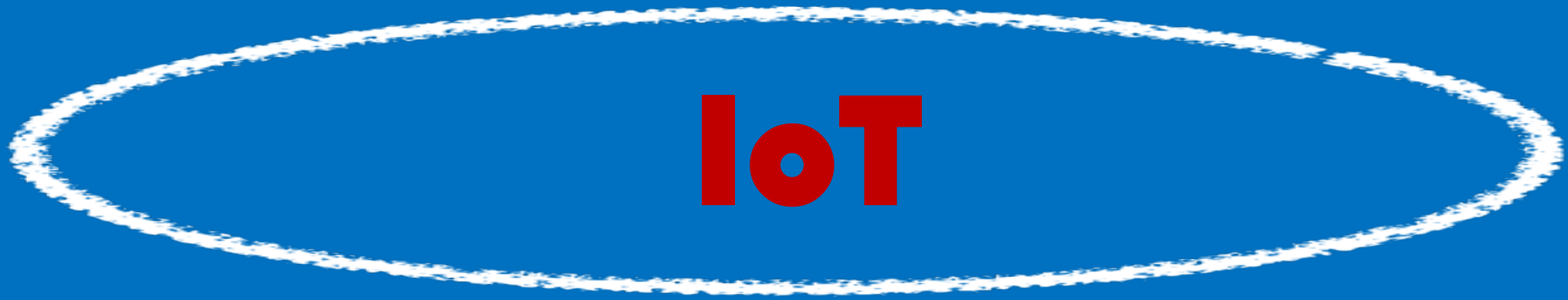
LISA HANEY

Computational Thinking

Computational thinking richt zich op de *vaardigheden*
die essentieel zijn om *problemen op te lossen*
waarbij veel data, code, variabelen en rekenkracht nodig zijn.

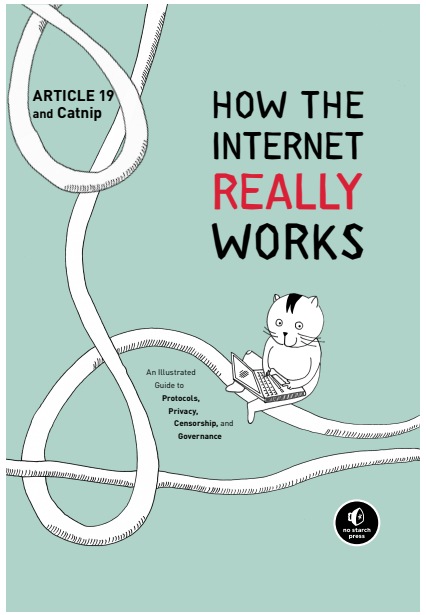
Het is daarbij belangrijk om te begrijpen hoe informatie tot stand komt
zodat je computersystemen kan benutten om problemen op te oplossen,
voor het denken in stappen en daarmee in voorwaardelijkheden
voor volgorde van de benodigde gegevens.

Computertechnologie gebruiken bij het zoeken naar oplossingen ***---Data Science---***
betekent inzicht krijgen in algoritmes (een reeks instructies om vanaf een beginpunt een bepaald doel te bereiken)
en procedures (een verzameling activiteiten die in een bepaalde volgorde moet worden uitgevoerd).



HOGESCHOOL ROTTERDAM

https://medium.com/@smartrac/the-deep-web-the-dark-web-and-simple-things-2e601ec980ac



The Layers of the Internet

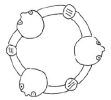
The internet is made of several layers, one on top of the other and interacting with each other. Now that we have learned about the concrete functions and components of the internet, it's time to conceptualize the internet as a whole system, including the people and institutions that operate it. Each layer operates on the layer below it and serves the layer above it.

Social Layer

The most relatable layer of the internet is made up of the entities that use it and the human relationships that govern it.

citizens

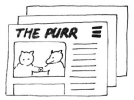
companies

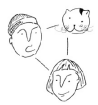


nonprofit organizations

Content Layer

The content layer, what data is accessible and available over the internet, is perhaps the most recognizable for users.





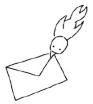
websites (news sites, social media, blogs)

entertainment

Application Layer

Applications are the ways content is served.

web-based platforms

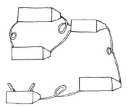


email

instant messaging

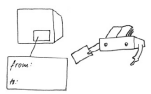
Logical Layer

The logic of the interoperable internet, or its standard protocols, support the connections between devices and the applications running on them.



networking

routing

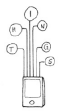


addressing
(Domain Name System)

Infrastructural Layer

Internet infrastructure is the material basis of the IP network, or the physical components across which the logical layer can send information from one place to another.

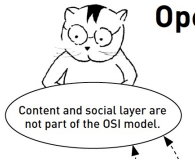
computers
(servers, personal computers, mobile devices)



internet of things (IoT)

routers
(gateways, switches)

data centers



governments

virtual Reality (VR)

voice over IP

traffic controls

telecommunication cables, Wireless networks, Satellite networks

Open Systems Interconnection (OSI) Model

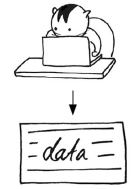
In 1984, the International Organization for Standardization (ISO) published the **Open Systems Interconnection (OSI) model**, a conceptual model that characterized and standardized the technical communication of a telecommunication or computing system. As a concept, in which any layer can take for granted underlying layers, it enables the interoperability of the internet as a diverse communication system across its governance space.

Layer 7: Application

This layer is the layer with which the user directly interacts using client applications, for example a web browser.

Layer 6: Presentation

This layer translates and transmits data between applications, such as text encoding or audio/visual compression.



Application Layer

Layer 5: Session

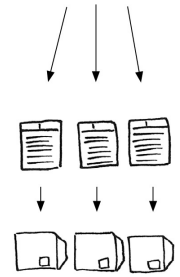
This layer controls the connections between computers by opening the connection, or session, and closing it, according to various session layer protocols.

Layer 4: Transport

This layer ensures the reliable transmission of packets, grouped in so-called messages, segments (TCP), or datagrams (UDP).

Layer 3: Network

This layer defines an addressing scheme and how packets are routed over the network.



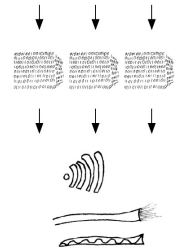
Logical Layer

Layer 2: Data-Link

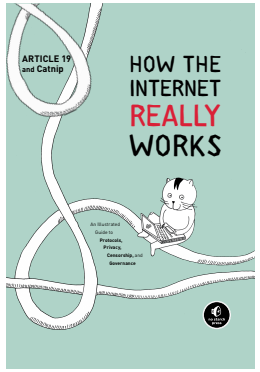
This layer defines the transmission of data frames between two nodes directly connected in the physical layer.

Layer 1: Physical

This layer defines electrical and physical specifications of the data connection. At this layer, physical media (electrical cable, optical glass fiber, radio frequency spectrum) send and receive raw bit streams. Network adapters, repeaters, and modems operate only at this low level.



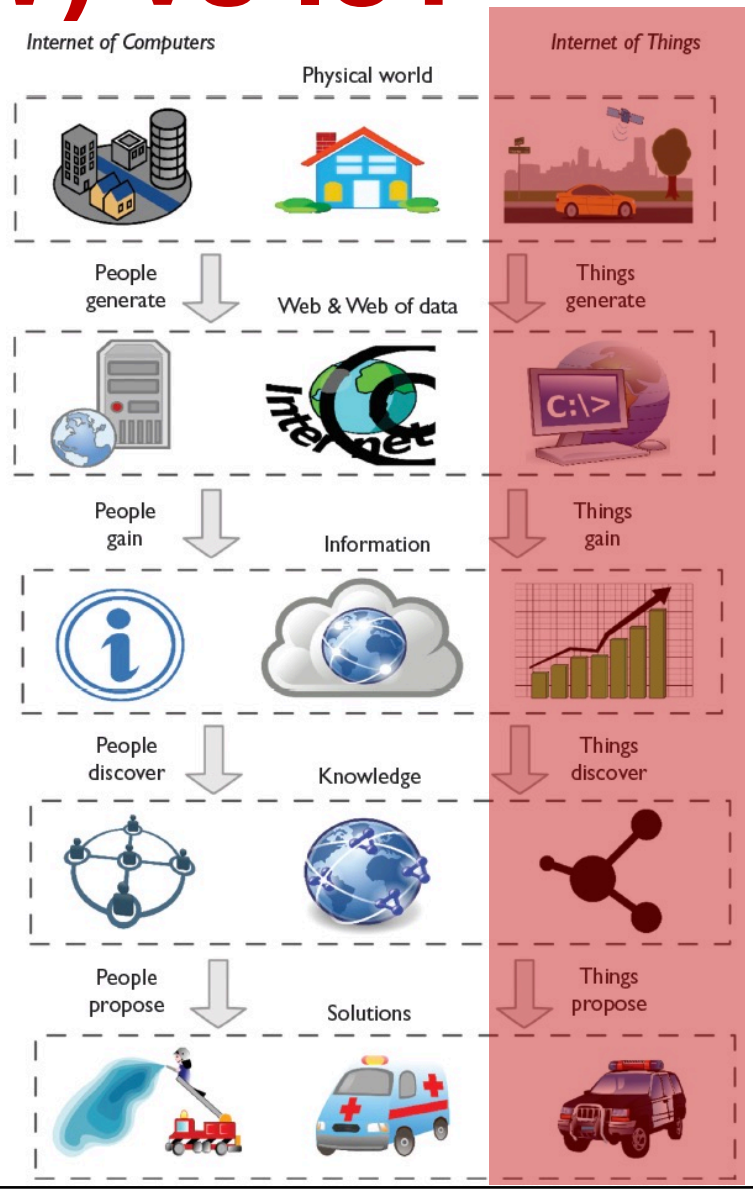
Infrastructural Layer



World Wide Web (WWW) vs IoT

WWW:
In the **Internet of Computers (WWW)**, the main **data producers** and consumers are **human beings**.

IoT:
The main actors become **things**, where things are the majority of **data producers** and consumers.

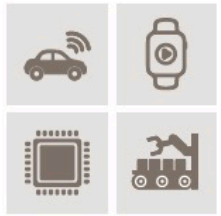


Qin, Y., Sheng, Q. Z., & Curry, E. (2015). Matching over linked data streams in the internet of things. *IEEE Internet Computing*, *19*(3), 21-27.

3 Pillars of the IoT

1 things

- Sensors
- Actuators
- Machines

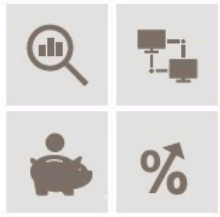


2 Connectivity

- Network
- Protocol
- Security

4 Data Products

- Outcome driven
- Product as a service
- Leverage existing investment



3 People + Processes

- Storage
- Analytics
- Visualization
- Integration

HOGESCHOOL ROTTERDAM

IoT User-Types

Hobbyists

Consumer

Industrial

Industry

IoT Application Domains

Consumers

- Connected gadgets
- Appliances
- Wearables
- Domestic robots
- Participatory sensing
- Social Web of Things

Automotive Transport

- Autonomous vehicles
- Multimodal transport
- Logistics
- Traffic management

Retail Banking

- Micro payments
- Retail logistics
- Product life-cycle info
- Shopping assistance

Environmental

- Pollution
- Air, Water, Soil
- Weather, Climate
- Noise
- Erosion, fires

Infrastructures

- Buildings, Homes
- Roads, Rail

Utilities

- Smart Grid
- Water management
- Gas, Oil, Renewables
- Waste management
- Heating, Cooling

Health Well-being

- Remote monitoring
- Assisted living
- Behavioral change
- Treatment compliance
- Sports, Fitness

Smart Cities

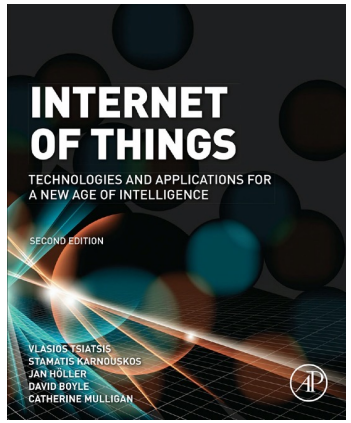
- Integrated environments
- Optimized operations
- Convenience
- Socioeconomics
- Sustainability
- Inclusive living

Process industries

- Robotics
- Manufacturing
- Natural resources
- Remote operations
- Automation
- Heavy machinery

Agriculture

- Forestry
- Crops and farming
- Urban agriculture
- Livestock, Fisheries



IoT

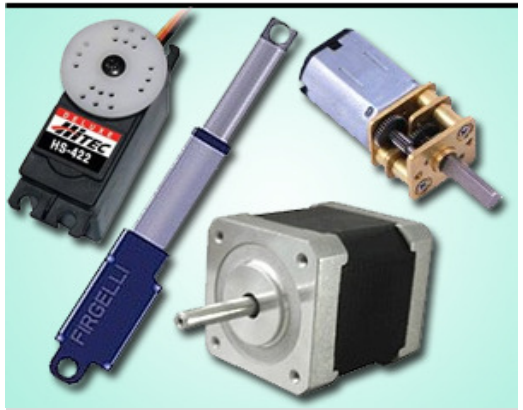
HOGESCHOOL ROTTERDAM

Smart Systems and the Internet of Things are driven by a combination of:

http://postscapes.com/what-exactly-is-the-internet-of-things-infographic

1 things

ACTUATORS

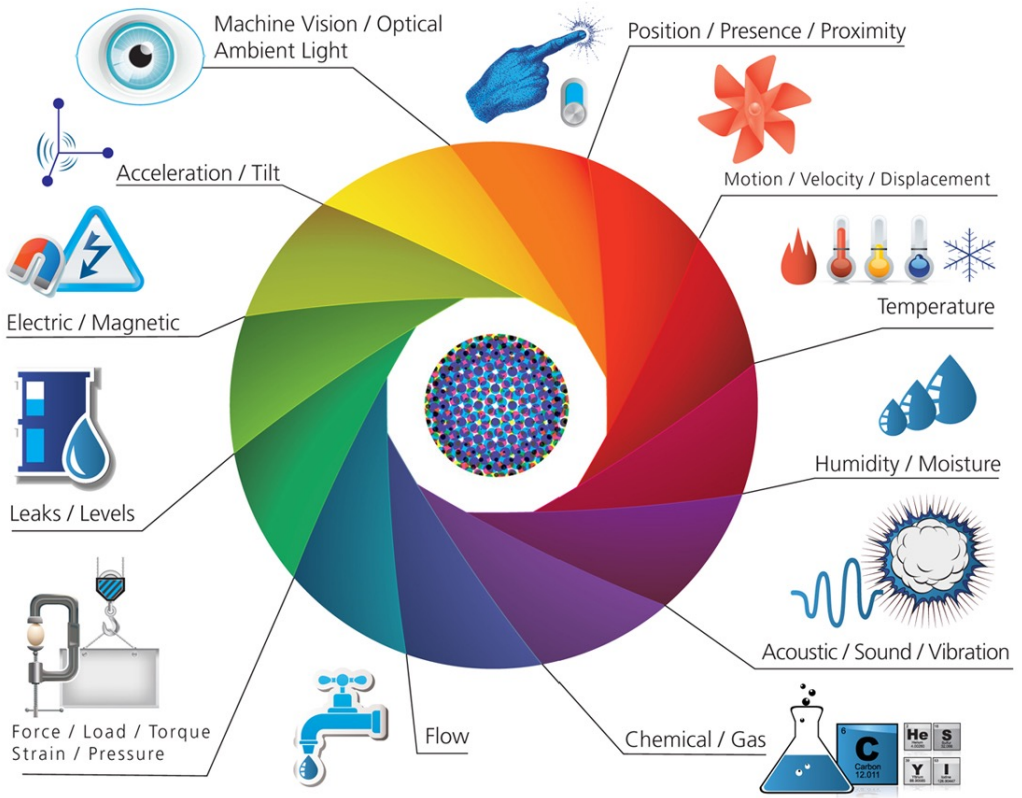


An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system.
An actuator requires a control signal and a source of energy.

The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power.

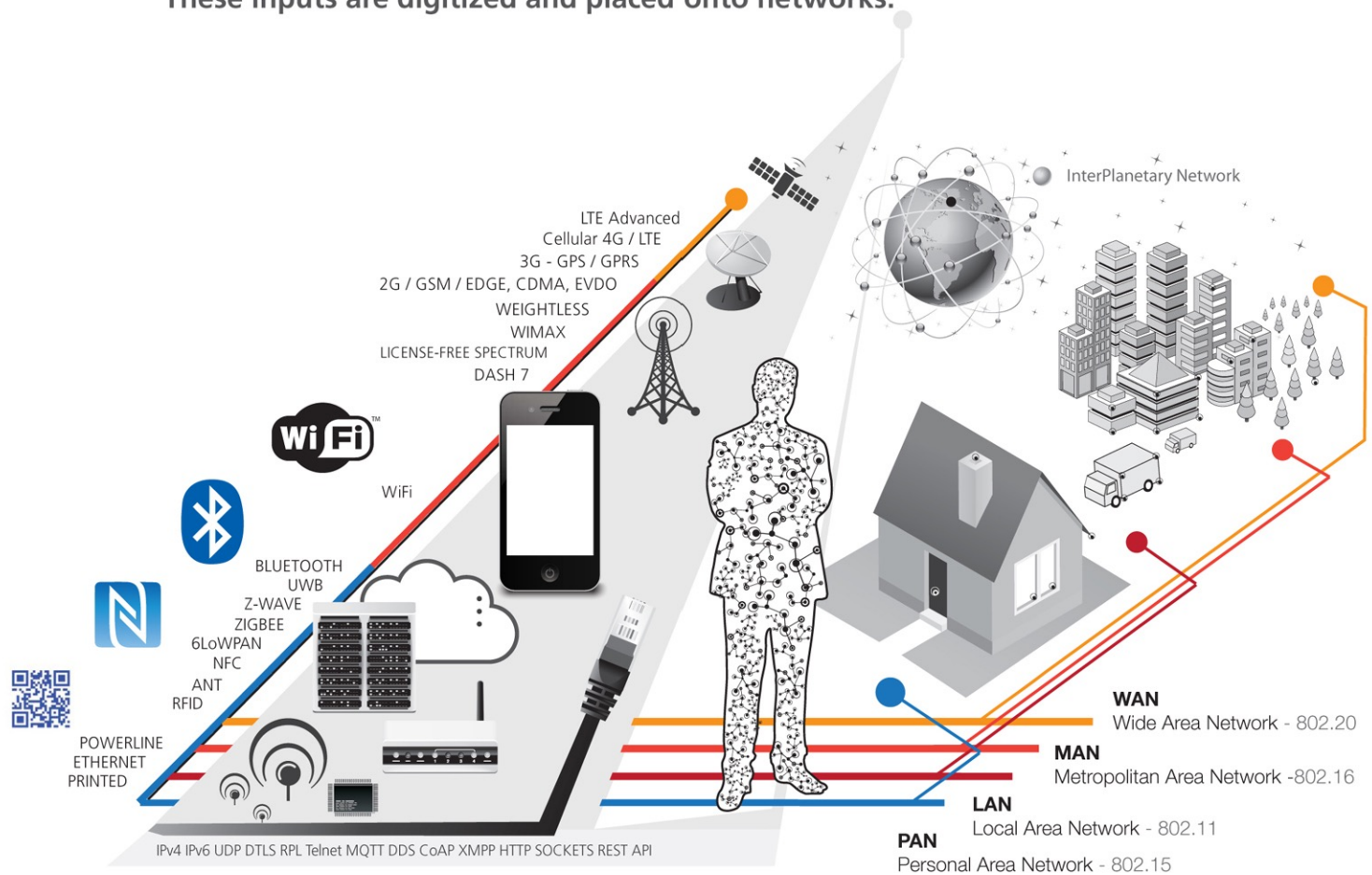
1 SENSORS & *ACTUATORS*

We are giving our world a digital nervous system. Location data using GPS sensors. Eyes and ears using cameras and microphones, along with sensory organs that can measure everything from temperature to pressure changes.



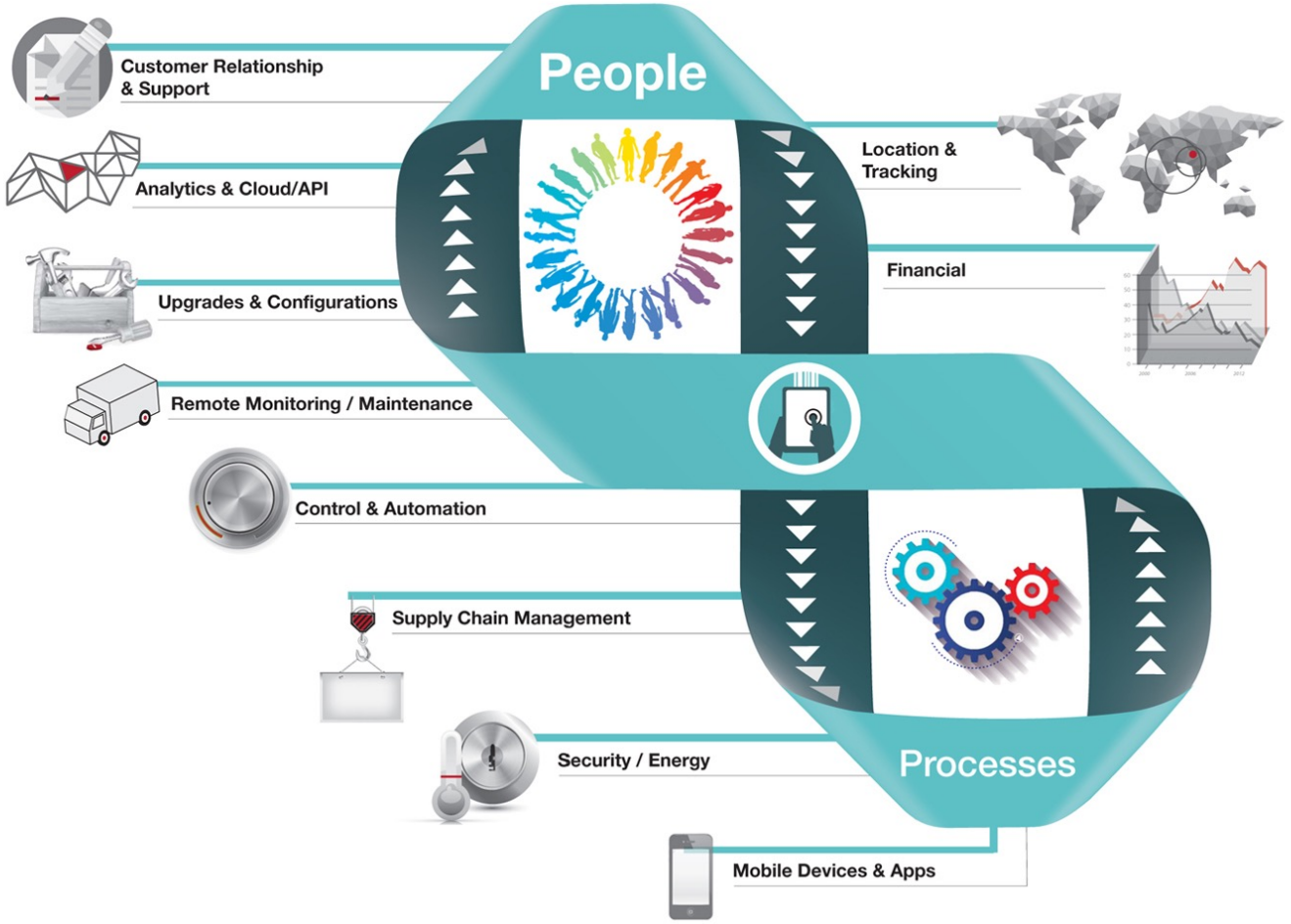
2 CONNECTIVITY

These inputs are digitized and placed onto networks.



3 PEOPLE & *PROCESSES*

These networked inputs can then be combined into bi-directional systems that integrate data, people, processes and systems for better decision making.

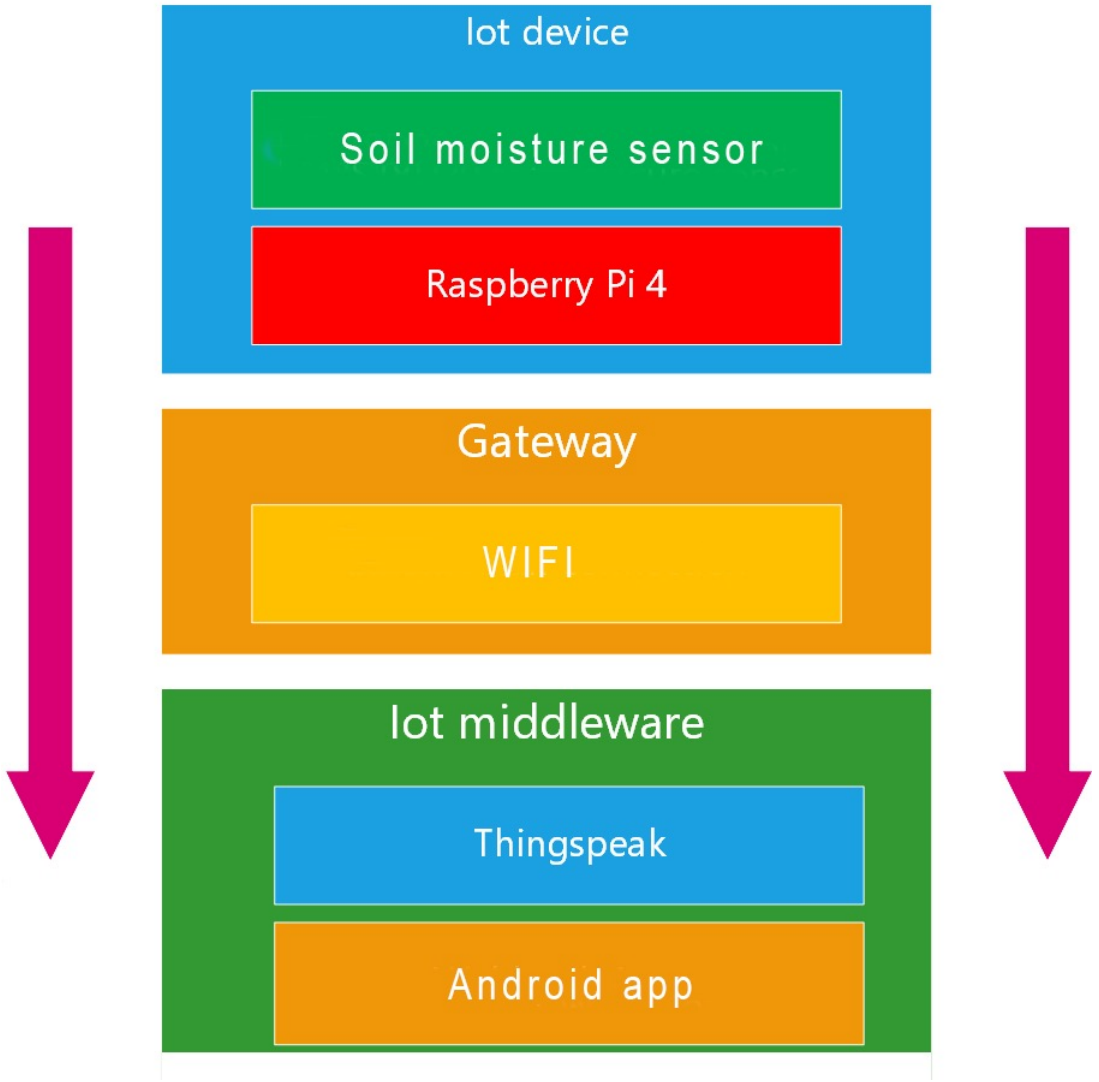


IoT Data Pipeline

Collect

Transform

Store & Analyse



Technologies enabling IoT

WIRELESS TECHNOLOGIES

802.11, 802.15.4, Zigbee, Bluetooth, BLE, CDMA, GSM, LTE

NETWORK TECHNOLOGIES

Ethernet, CAN, rs485/rs422/rs232, 1-wire, i2c, SPI, ModBus/MudBusRTU, IPMI, iSCSI

WEB SERVICE TECHNOLOGIES

SOAP, REST, WSDL, XML, JSON, UDDI, WebSockets

TCP/IP PROTOCOLS

Sockets, IPv4, IPv6, TCP, UDP, ICMP, QOS, SNMP, IMAP, POP3, IPMI, etc.

MOBILE TECHNOLOGIES

Android SDK, Qt, iOS SDK, Objective C, Java, Swift

PROTOCOL TECHNOLOGIES

HTTP, JMS, AMQP, D-Bus

AUTHORIZATION & AUTHENTICATION TECHNOLOGIES

Oauth2, PAM, LDAP

PLATFORMS AND CPU ARCHITECTURES

ARM, X86, PowerPC, AVR, PIC

PROGRAM / SCRIPTING LANGUAGES

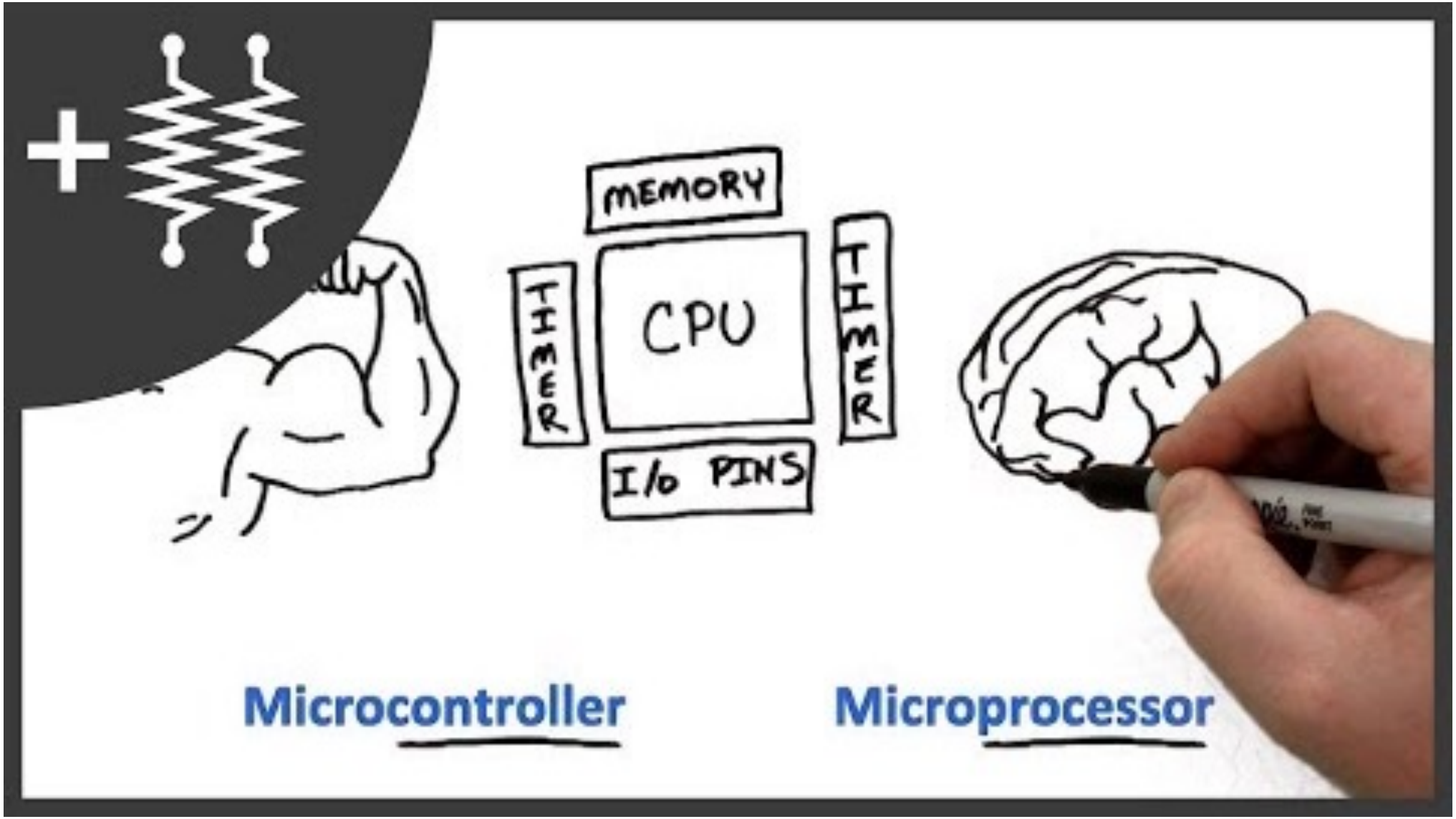
Java, C/C++, C#, JavaScript, Ruby, Groovy, Python, Tcl/Tk, ASM, Bash

https://doi.org/10.15779/Z38WW0V

https://doi.org/10.1016/j.eswa.2019.05.014

Getting to Know Your IoT Device

HOGESCHOOL ROTTERDAM



https://www.youtube.com/watch?v=7vhvnaWUZjE

Raspberry Pi

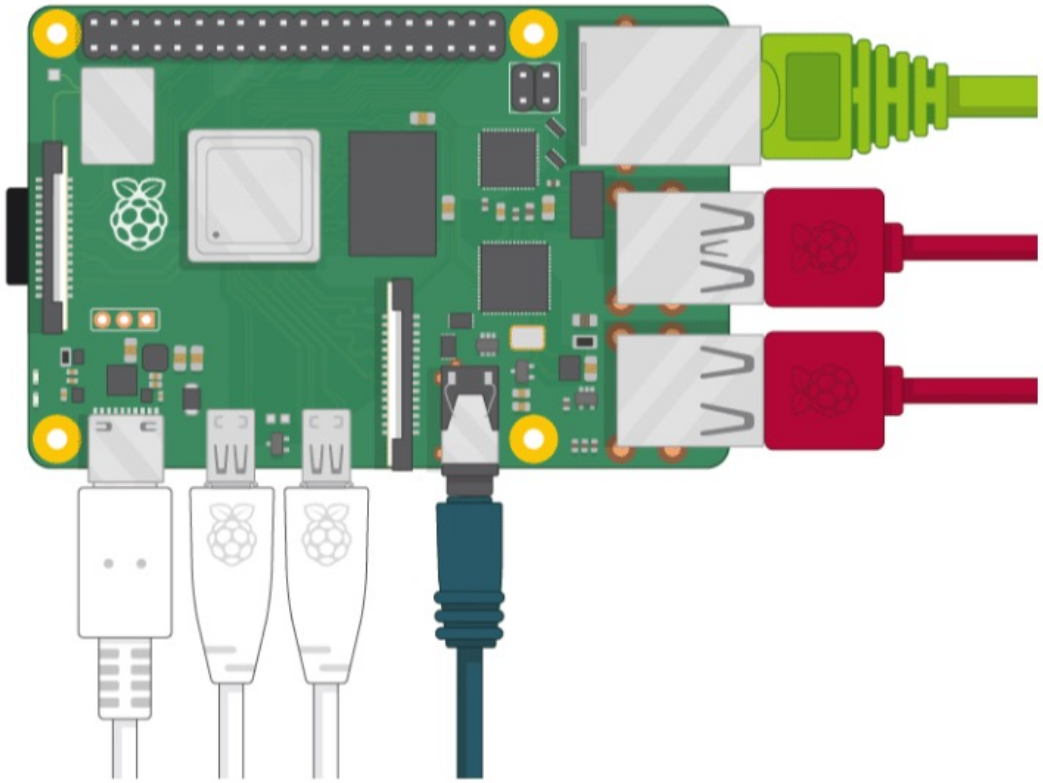
https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

HOGESCHOOL ROTTERDAM

- Introduction
- What you will need
- Set up your SD card
- Connect your Pi
- Start up your Pi
- Finish the setup
- Where to find help
- What next?

Introduction

Here you'll learn about your Raspberry Pi, what things you need to use it, and how to set it up.

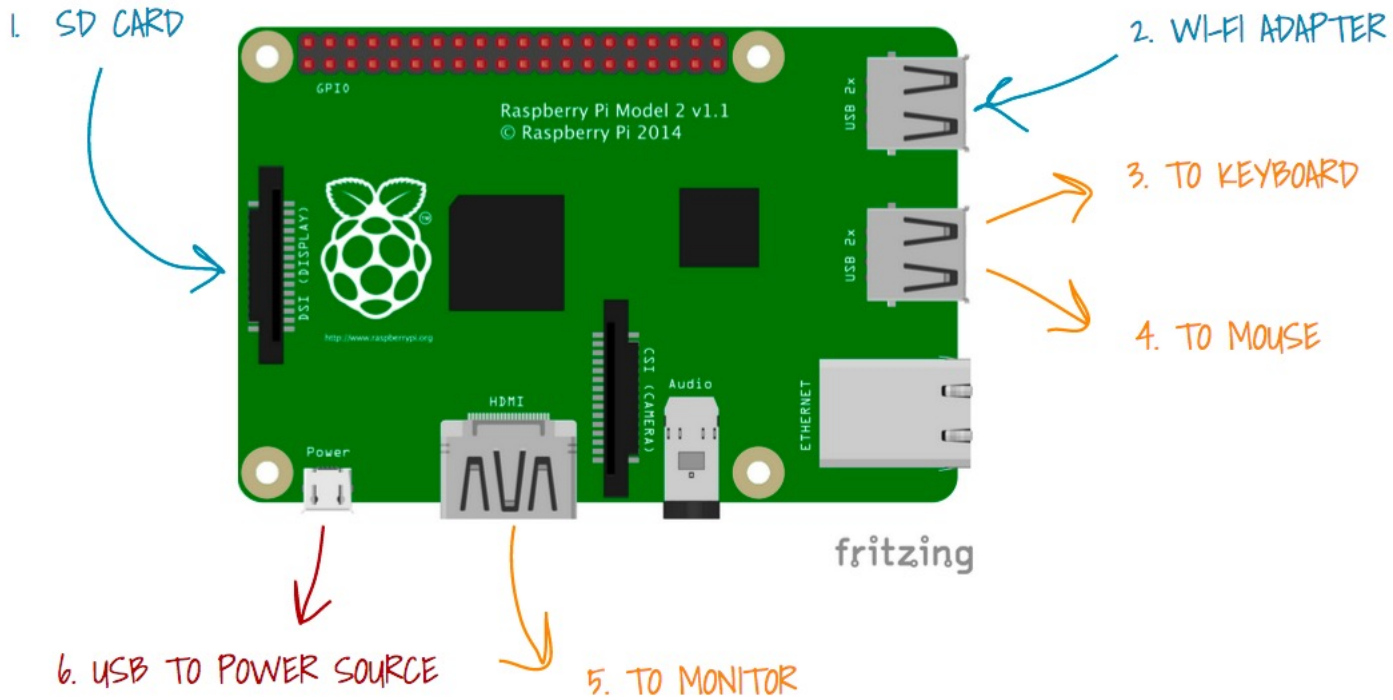


What is a Raspberry Pi?

The Raspberry Pi is a low cost, **credit-card sized computer** that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

What's more, the Raspberry Pi has the ability to interact with the outside world, and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work.

Main Components Raspberry Pi

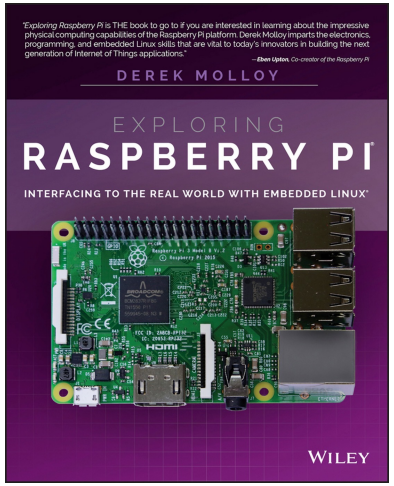


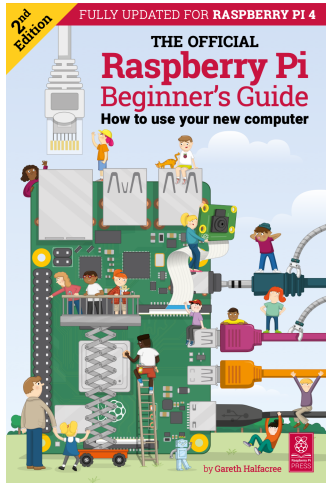
https://www.pubnub.com/blog/internet-of-things-101-getting-started-w-raspberry-pi/

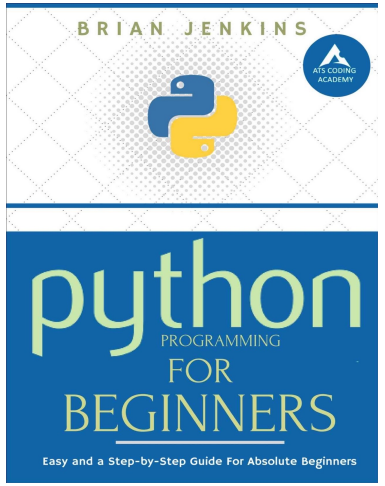
Things you should Do

HOGESCHOOL ROTTERDAM

Start Reading TEXTBOOKS!







How to get Access to TEXTBOOKS!

Learning O'Reilly is een Engelstalig online leerplatform gericht op ICT vaardigheden. Het biedt tienduizenden e-books en vele video's, case studies en "learning paths", waarbij je zelf je voortgang kunt monitoren.

Toegang

Via de website

De Learning O'Reilly website is gekoppeld aan de Hogeschool Rotterdam login. Kom je toch een O'Reilly inlogscherm tegen, vul dan alleen je @hr.nl e-mailadres in en klik op **Sign In with Single Sign On**. Krijg je een blanco pagina te zien? Schakel eventuele adblockers uit!

Via de O'Reilly app

Leer wat je wil, waar je wil. Met de O'Reilly app download je e-books en ga je verder met een playlist waar je op een ander apparaat gebleven was.

1 Google play App Store

2 Gebruik je @hr.nl e-mailadres om in te loggen.

Hardware YOU NEED:

Raspberry Pi 3+ or higher generation:

De **Raspberry Pi 3 Model A+** board (2018) heeft 512MB RAM
Beschikt net over een 64-bit quad core processor op 1.4GHz, dual-band WiFi (2.4GHz en 5GHz) en Bluetooth 4.2/BLE.

Specificaties
Processor: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Geheugen: 512MB LPDDR2 SDRAM
Connectiviteit:
- 2.4GHz en 5GHz 802.11b/g/n/ac WiFi
- Bluetooth 4.2 / BLE
- 1x USB 2.0 poort

Uitbreiding: 40-pin GPIO header
Video & Geluid:
- 1x full-size HDMI
- DSI display poort
- DSI camera poort
- 4-polige 3.5mm jack voor stereo audio en composiet video

Multimedia: H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 g
Stroomvoorziening:
- 5V / 2.5A DC via microUSB
- 5V DC via GPIO header

Afmetingen: 65x56x8.5mm
Gewicht: 29g

https://www.kiwi-electronics.nl/raspberry-pi-3-model-a-plus

	Raspberry Pi 4 B	Raspberry Pi 3 Model A+	Raspberry Pi 3 B+	Raspberry Pi Zero WH
Image				
Release date	2019 Jun 24	2018 Nov 15	2018 Mar 14	2018 Jan 12
Description				Same as Raspberry Pi Zero W with header already soldered
Product details				
Price	US$35.00	US$25.00	US$35.00	US$15.00
SOC				
SOC Type	Broadcom BCM2711	Broadcom BCM2837B0	Broadcom BCM2837B0	Broadcom BCM2835
Core Type	Cortex-A72 (ARM v8) 64-bit	Cortex-A53 64-bit	Cortex-A53 64-bit	ARM1176JZF-S
No. Of Cores	4	4	4	1
GPU		VideoCore IV	VideoCore IV	VideoCore IV
CPU Clock	1.5 GHz	1.4 GHz	1.4 GHz	1 GHz
RAM	1 GB , 2 GB, 4 GB	512 MB DDR2	1 GB DDR2	512 MB
Wired Connectivity				
USB	✓ 2x USB3.0 + 2x USB2.0	✓ 1xUSB 2.0	✓ 4x USB2.0	✓ micro & micro OTG
Ethernet	✓ Gigabit	✗	✓ Gigabit - Over USB 2.0	✗
SATA Ports	✗	✗	✗	✗
HDMI port	✓ 2x micro HDMI	✓	✓	✓ mini
Analog Video Out	✓ shared with audio jack	✓ shared with audio jack	✓ shared with audio jack	✓ via unpopulated pin
Analog Audio Out	✓ 3.5mm jack	✓ 3.5mm jack	✓ 3.5mm jack	− HDMI audio
Analog Audio In	✗	✗	✗	✗
SPI	✓	✓	✓	✓
I2C	✓	✓	✓	✓
GPIO	✓	✓	✓	✓
LCD Panel	✓	✓	✓	✗
Camera	✓	✓	✓	✓
SD/MMC	✓ microSD	✓ microSD	✓ microSD	✓ microSD
Serial	−	−	− RX/TX UART	−
Wireless Connectivity (On-Board)				
Wi-Fi	✓ 2.4GHz and 5GHz 802.11 b/g/n/ac	✓ 2.4GHz and 5GHz 802.11 b/g/n/ac	✓ 2.4GHz and 5GHz 802.11 b/g/n/ac	✓ 802.11n
Bluetooth®	✓ 5.0	✓ 4.2, BLE	✓ 4.2, BLE	✓ 4.1
Dimensions				
Height	3.37 in (85.6 mm)	2.55 in (65 mm)	3.37 in (85.6 mm)	1.18 in (30 mm)
Width	2.22 in (56.5 mm)	2.20 in (56 mm)	2.22 in (56.5 mm)	2.55 in (65 mm)
Depth	0.43307 in (11 mm)	0.43307 in (11 mm)	0.66929 in (17 mm)	0.51181 in (13 mm)
Weight		1.02 oz (29 g)	1.58 oz (45 g)	0.42328 oz (12 g)
Power				
Power ratings			1.13 A @5V	180 mA
Power sources	USB-C	microUSB, GPIO	microUSB, GPIO	microUSB or GPIO
Power Over Ethernet	− with PoE Hat	✗	− with PoE Hat	✗

Create Your own github

HOGESCHOOL ROTTERDAM

1st STEP Managing Data Science for IoT Projects

➔➔➔Getting started with GitHub🡐🡐🡐

A Computational Thinking culture has an intellectual dimension, engaging with a set of creative concepts and practices. It has a physical dimension, encouraging interactions with others through the placement of desks, chairs, and computers. Most importantly, it has an affective dimension, cultivating a sense of confidence and fearlessness.

Tip

Programming involves working with many different file types, each of which is indicated by its **extension** (the letters after the `.` in the file name, such as `.pdf`). It is useful to specify that your computer should show these extensions in File Explorer or Finder; see instructions for Windows[a] or for Mac[b] to enable this.

[a] https://helpx.adobe.com/x-productkb/global/show-hidden-files-folders-extensions.html

[b] https://support.apple.com/kb/PH25381?locale=en_US

LEARNING OBJECTIVES

Students will:

+ be introduced to the concept of computational Thinking, in the context of GitHub
+ be able to imagine possibilities for their own GitHub-based computational thinking
+ become familiar with resources that support their computational thinking
+ prepare for creating GitHub projects by establishing a GitHub account, exploring GitHub, creating design journals

KEY WORDS, CONCEPTS, & PRACTICES

+ IDE
+ GitHub
+ Visual Studio Code

learn to operate your Raspberry Pi

HOGESCHOOL ROTTERDAM

1 INSTALLING GIT

One of the most important aspects of doing data science is keeping track of the changes that you (and others) make to your code. `git` is a *version control system* that provides a set of commands that you can use to manage changes to written code, particularly when collaborating with other programmers (version control is described in more detail in Chapter 3).

`git` comes pre-installed on Macs, though it is possible that the first time you try to use the tool you will be prompted to install the *Xcode command line developer tools* via a dialog box. You can choose to install these tools, or download the latest version of `git` online.

On Windows, you will need to download[2] the `git` software. Once you have downloaded the installer, double-click on your downloaded file, and follow the instructions to complete installation. This will also install a program called *Git Bash*, which provides a command line (text-based) interface for executing commands on your computer. See Section 1.1.2 for alternative and additional Windows command line tools.

On Linux, you can install `git` using `apt-get` or a similar command. For more information, see the download page for Linux.[3]

[2] **`git` downloads**: https://git-scm.com/downloads

[3] **`git` download for Linux and Unix**: https://git-scm.com/download/linux

2 CREATING A GITHUB ACCOUNT

GitHub[4] is a website that is used to store copies of computer code that are being managed with `git`. To use GitHub, you will need to create a free GitHub account.[5] When you register, remember that your profile is public, and future collaborators or employers may review your GitHub account to assess your background and ongoing projects. Because GitHub leverages the `git` software package, you don't need to install anything else on your machine to use GitHub.

[4] **GitHub**: https://github.com

[5] **Join GitHub**: https://github.com/join

3 SELECTING A TEXT EDITOR

Visual Studio Code[9] (or VS Code; not to be confused with Visual Studio) is a free, open source editor developed by Microsoft—yes, really. While it focuses on web programming and JavaScript, it readily supports lots of languages, including Markdown and `R`, and provides a number of extensions for adding even more features. It has a similar command palette to Atom, but isn't quite as nice for editing Markdown specifically. Although fairly new, it is updated regularly and has become one of the authors' main editors for programming.

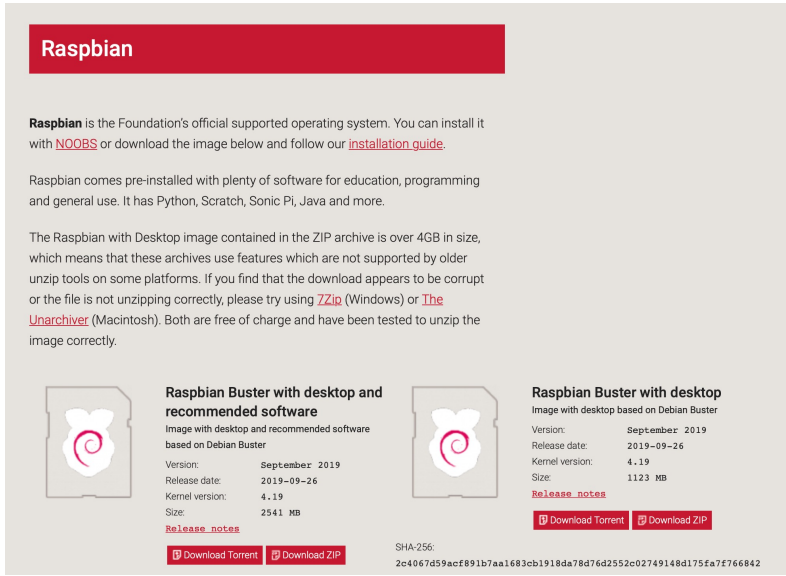
[9] **Visual Studio Code**: https://code.visualstudio.com

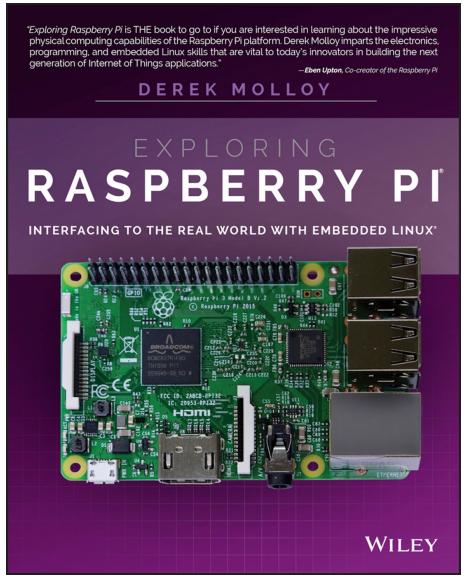
2nd STEP Managing Data Science for IoT Projects

HOGESCHOOL ROTTERDAM

→→→ Getting to know the Raspberry Pi ←←←

Use the. **Raspberry Pi** Beginner's Guide + https://www.raspberrypi.org/downloads/raspbian/ to install Raspbian on your own device.





LEARNING OBJECTIVES

Students will:

+ be introduced to the concept of computational Thinking, in the context of **Raspberry Pi**
+ be able to imagine possibilities for their own **Raspberry Pi** -based data science for IoT project
+ become familiar with Raspbian
+ Programming with Python using the Thonny Pyton IDE

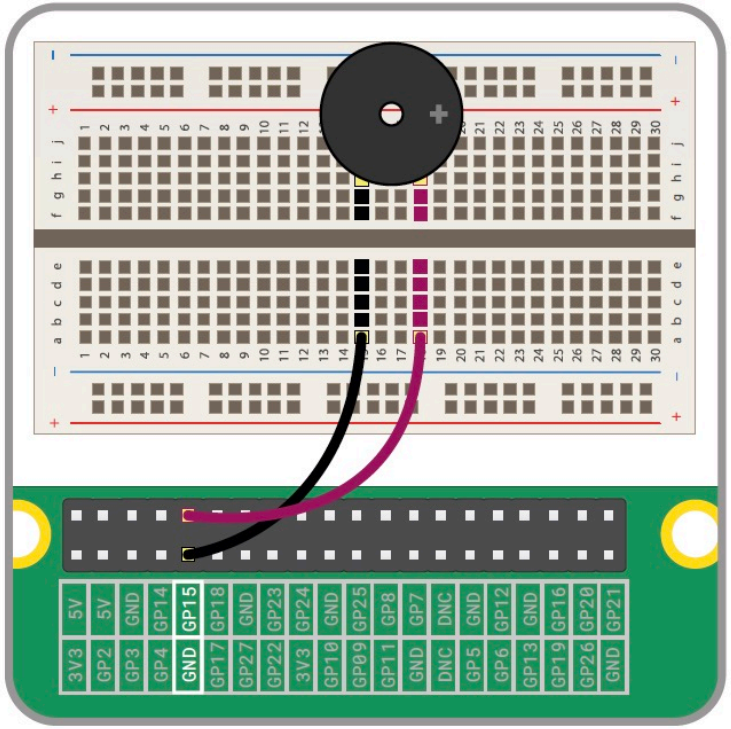
KEY WORDS, CONCEPTS, & PRACTICES

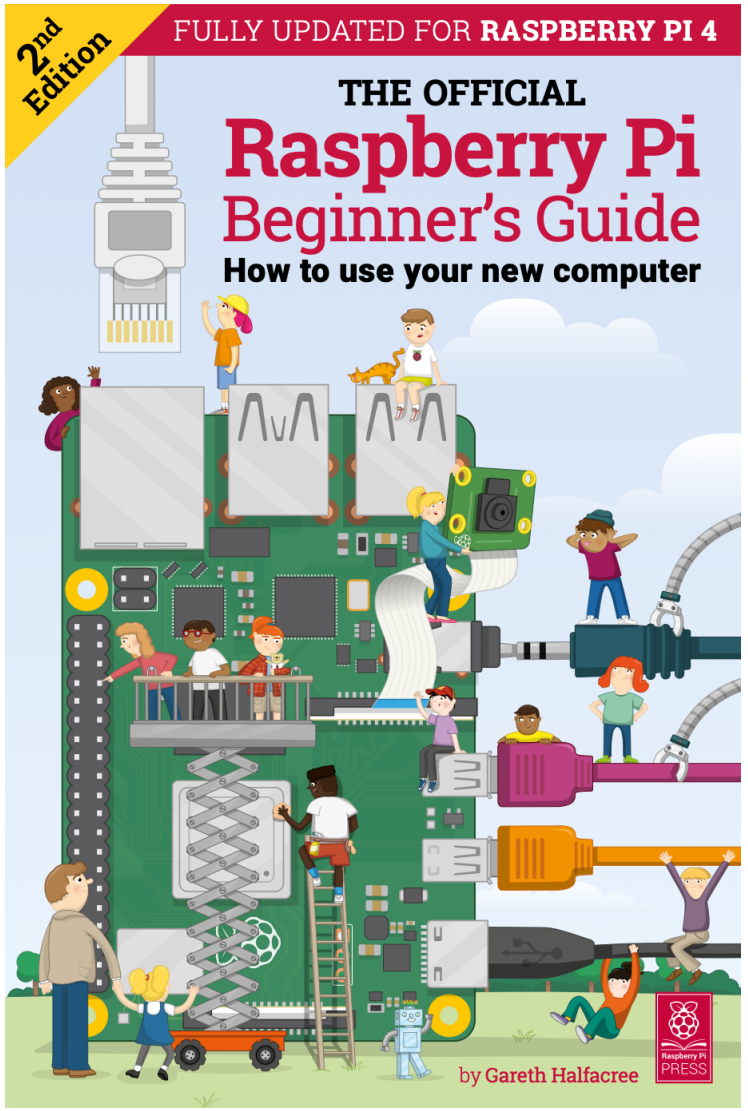
+ IDE
+ Python
+ Raspbian

Preview
Lesson Two

HOGESCHOOL ROTTERDAM

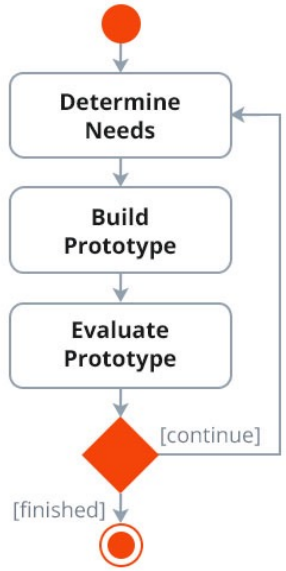
NEXT TIME
Rapid Prototyping

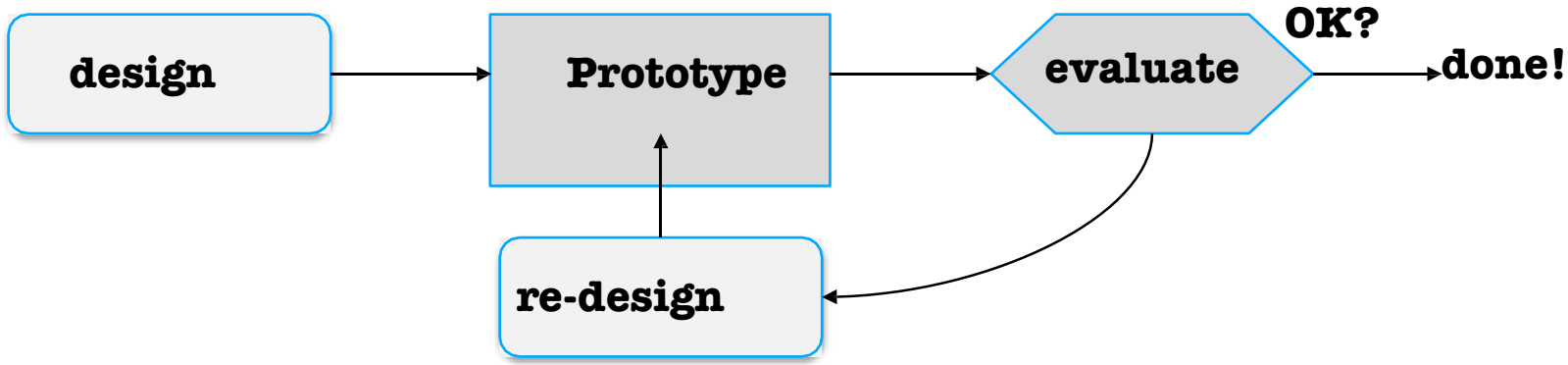




PROTOTYPING

- You never get it right first time
- If at first you don"t succeed ...





PITFALLS OF PROTOTYPING

- Moving little by little … but to where

1. need a good start point
2. need to understand what is wrong