The University Of Sheffield

“Click Here to Kill Everyone”... introducing the Internet of Things

COM3505, Lecture 1
Prof Hamish Cunningham

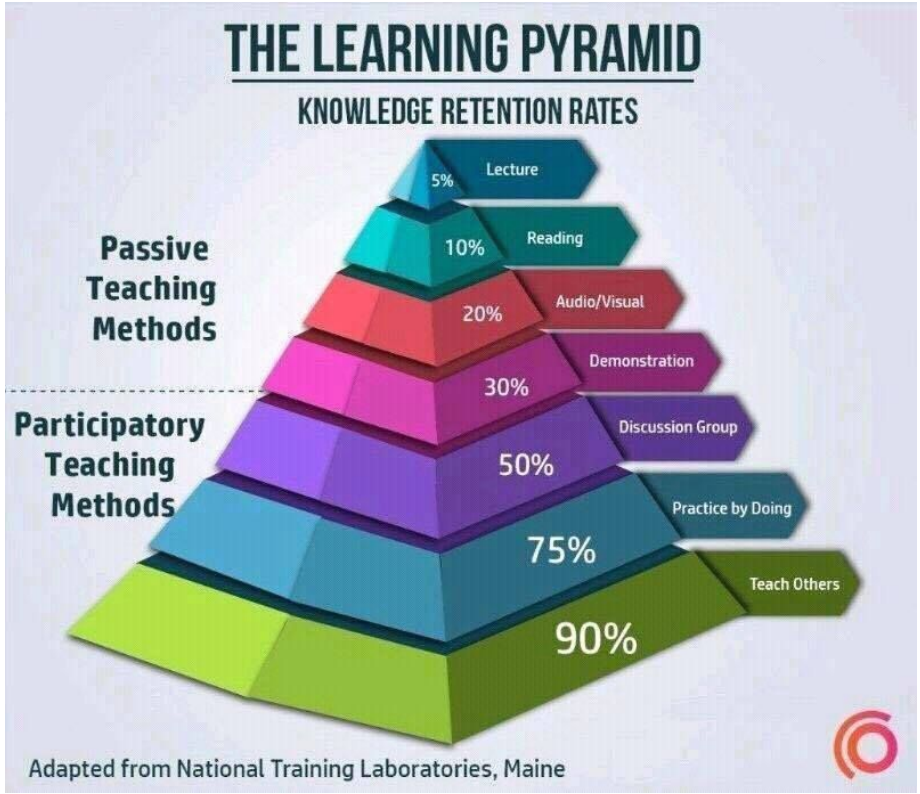
Who's going to learn the most...

... out of everyone in this room?

Me!

Help yourself to learn by:

- building stuff (this is a *practical* course!)
- figuring out what I've missed or got wrong! (and telling me)
- discussing with your peers
- *teaching* others about what you've learned (but not doing their work for them!)
- reading *critically* around the subject



The University Of Sheffield

What is the Internet of Things (IoT)?

A world robot?!

> ...the Internet of Things has **three parts**. There are the **sensors** that collect data about us and our environment... Then there are the "**smarts**" that figure out what the data means and what to do about it... And finally, there are the **actuators** that affect our environment. … You can think of the sensors as the **eyes and ears** of the internet. You can think of the actuators as the **hands and feet** of the internet. And you can think of the stuff in the middle as the **brain**. We are building an internet that senses, thinks, and acts. This is the classic definition of a robot. We're building **a world-size robot**, and we don't even realize it.
>
> (Schneier 2017, *Click Here to Kill Everyone*)

At the other extreme…:

The University Of Sheffield

What is… (2)?

More prosaically:

At the other extreme the IoT is about what becomes possible when:

- networked microcontrollers* become **cheap** enough to embed in very many everyday contexts,
- from central heating thermostats to garage doors.
- These devices face **tight constraints** of power usage and cost, and
- concomitant challenges to their **security** and **functionality**.
- They also quickly become extremely **numerous**, driving work on
  - **big data** analytics and
  - **cloud** computing.

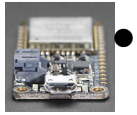
(*__MCU__: i/o, memory and small CPU on a chip — smallest single die unit of computation;
cf. **SoC**, System on a Chip, adds peripherals like video, USB, etc. etc.)

History, Antecedents

The tech: networked devices go back perhaps 50 years or more; the term **IoT** itself is often credited to Kevin Ashton in 1999 (working at the Auto ID Lab at MIT).

Related and predecessor fields include:

- **embedded systems**: computation built into devices with specific purposes (i.e. **not** general purpose computers)
- **ambient computing** or **ubiquitous computing**: the trend for computation to move into more and more devices (Schneier: "*We no longer have things with computers embedded in them. We have computers with things attached to them.*")
- **physical computing** (or **cyber-physical systems**): computation dependent on sensor input and/or producing actuator output
- **distributed computing**: jobs performed by multiple machines
- **utility computing**: start of the as-a-service model, e.g. software as a service, data as a... ("XAAS")
- **the cloud**: utility distributed computing

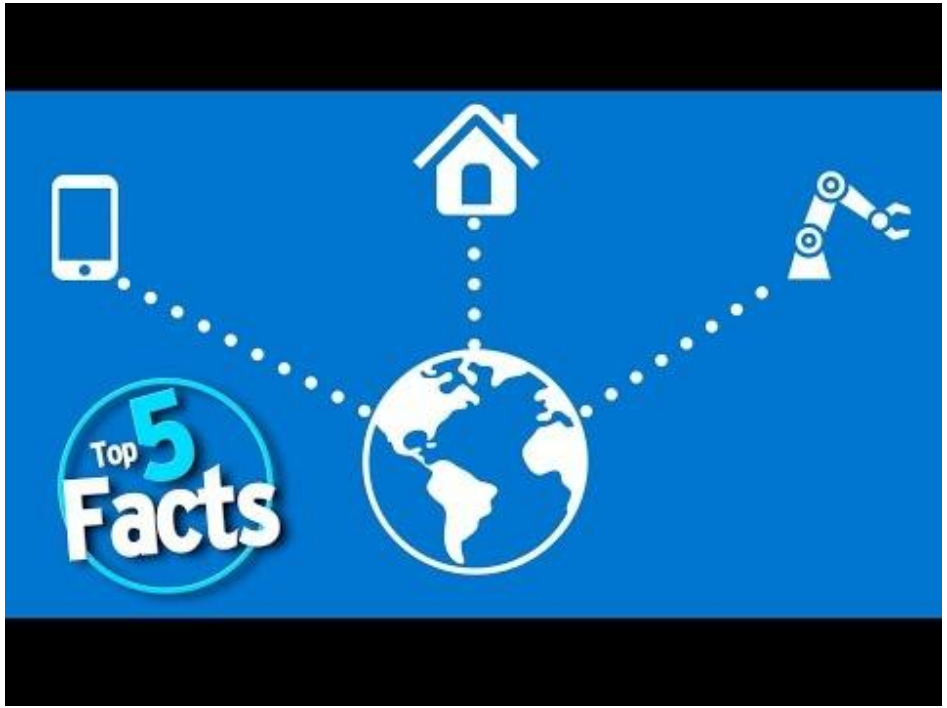


The University Of Sheffield

IoT: Double Plus Ungood or Bellyfeel Goodthink?

Spot the references?

(See also: Gattaca, Elysium, …)



Rise of the machines: who is the ‘internet of things’ good for? (Greenfield) https://goo.gl/uIUCrD

An IoT view from IBM

Notes:

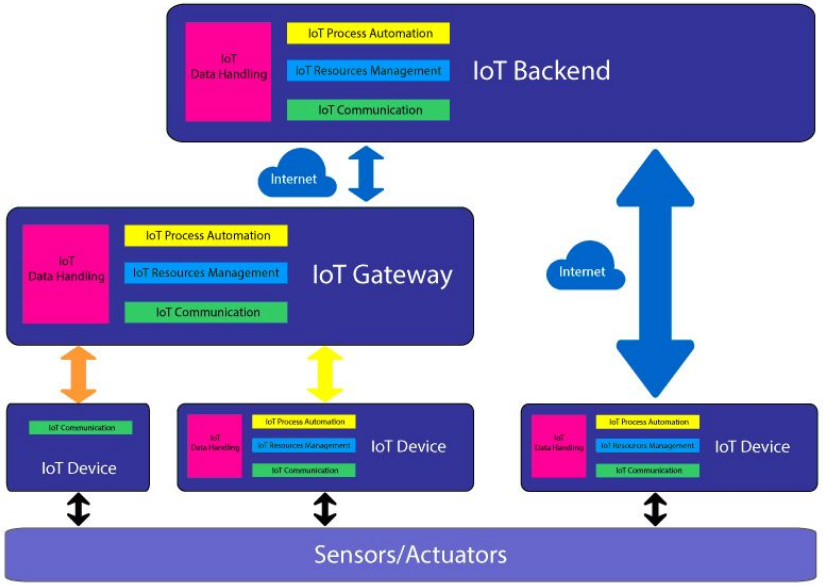
- 0.40: platform?
- 1.45: gateway?
- 2.20: asset management sys?

Remember: read (or watch) *critically*!

Software Architecture and the IoT (“fog”?!)

Aside from the devices themselves (connected microcontrollers running sensor/actuator rigs), IoT systems will typically also involve one or more of:

- a platform, or backend (e.g. IFTTT, AWS IoT or Adafruit.io)
- a gateway (e.g. Google Home, Phillips Hue Bridge, Amazon Echo, Siemens Mindsphere MindConnect, a Pi...)
- communications protocols (e.g. MQTT COAP, HTTP, ...)
- big data processing (Hadoop, Storm, Samza, Spark, Flink…)



(forklog.net)

The University Of Sheffield

How to pass COM3505

Assessment:

- 30% exams (MOLE quizzes)
- 70% coursework, split across:
  - lab work checking by teaching assistants
  - cloud data push, validated by our server
  - code pushed to github
- do the optional assignments for extra marks

Notes:

- lab assignments go off when we provide the solutions: try not to be late!
- quizzes depend on the lectures and reading materials
- equipment is handed out at the lectures
- be aware of the University regulations on plagiarism
- if in doubt: ASK!

Course programme

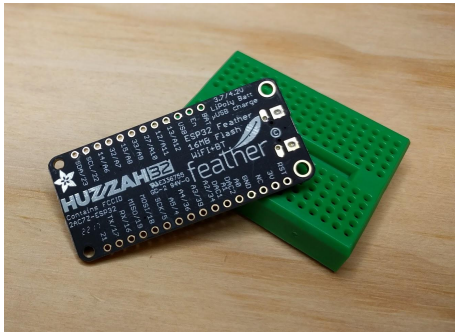
1. *Click here to kill everyone.*
2. *Revolutionary code: from MIT printers to the Arduino.*
3. *Small but perfectly formed... digging into the ESP32.*
4. *Sensing and responding.*
   Quiz 1.
5. *Cloudside.*
6. Reading week A

Labs: creating connected devices

7. *Country of the blind: networking devices without user interfaces.*
   Quiz 2.
8. *Projects.*
9. *Your house, my botnet? Securing the IoT.*
10. *Applications (1).*
    Quiz 3.
11. *Applications (2).*
12. Reading week B

Labs: projects: robot, air q, panic button, voting machine, …

Hardware: the ESP32



New wave wifi microcontrollers from fabless chip manufacturer Espressif:

- started making a splash with the ESP8266 in 2014
- an order of magnitude cheaper and more powerful than equivalent AVR-based boards (though new entrants are catching up!)
- Arduino-compatible layer made programming it much easier, and made Arduino sensor/actuator code reusable
- ESP32 released 2016 adding:
  - dual core (e.g. wifi etc. runs on one, user code on the other)
  - Bluetooth LE
  - more GPIO
  - touch and temperature sensors

Your boards: the Adafruit Huzzah ESP32 Feather

- a robust board with built-in LiPo charger, USB to serial, automatic bootloader reset
- part of the Feather range, which includes add-on motor drivers, RF, TFT and OLED screens, relays, midi synth...

The University Of Sheffield

Firmware: Arduino C++

Arduino is both a hardware platform (most often based on AVR microcontrollers) and an IDE — arduino.cc — that has lead the way on open embedded computation since 2005.

C++ is a high (or medium) level language layered on the C systems programming language.

The ESP32 is **not** an AVR-based device, but there is a compatibility layer that interfaces it to the Arduino IDE. This makes lots of code for sensors and actuators and etc. magically available for the ESP.

Quick! Learn C! Some resources to help:

- www.learn-c.org
- www.tutorialspoint.com/cprogramming
- www.tutorialspoint.com/cplusplus
- www.learncpp.com

On GNU Linux or Ubuntu:

- www.network-theory.co.uk/docs/gccintro
- askubuntu.com/questions/36520/how-could-i-begin-c-programming-on-ubuntu

You're all already programmers: you no longer need to study every bracket and comma of the syntax of new languages.

You're nearly professionals: so you need to get ready to beg, borrow and steal (i.e. google it, **copy** it, **share** your improvements) your next bit of code inspiration.

Software: various cloud services

Commercial services:

- if this then that: IFTTT
- adafruit.io
- ...

Sending data to the COM3505 cloud server

- http://com3505.gate.ac.uk:9191/com3505?...data...
- data format: key=value, e.g.: email=hamish@gate.ac.uk
- it requires a .ac.uk email address, and a MAC address — in week 01 please pass your GitHub ID as the “MAC”
- so to send your email to our server do:
- http://com3505.gate.ac.uk:9191/com3505?email=me@sheffield.ac.uk&mac=MeOnGitHub

The University Of Sheffield

Infrastructure: GitHub

Linus Torsvald's Very Short Christmas Card List...

- 2005: BitKeeper access for Linux Kernel development withdrawn, Linus decides to implement his own…
- “Git is a term of insult with origins in British English denoting an unpleasant, silly, incompetent, stupid, annoying, senile, elderly or childish person. As a mild oath it is roughly on a par with prat and marginally less pejorative than berk.”
- Linus: "I'm an egotistical bastard, and I name all my projects after myself. First 'Linux', now 'git'."

Git is...

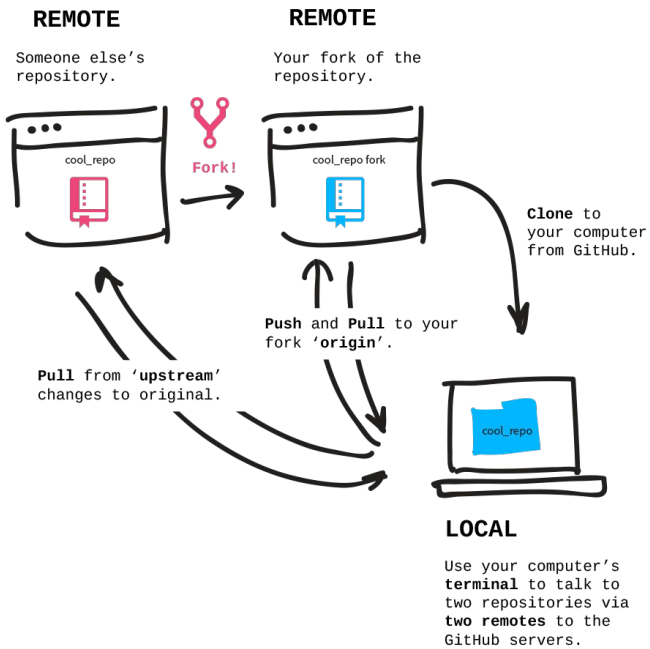
- entirely appropriate for at least one project in the world (the Linux Kernel: 15 million lines of code, thousands of developers, >a billion installations...)
- a bit like swatting a fly with a pile driver for most other projects?!

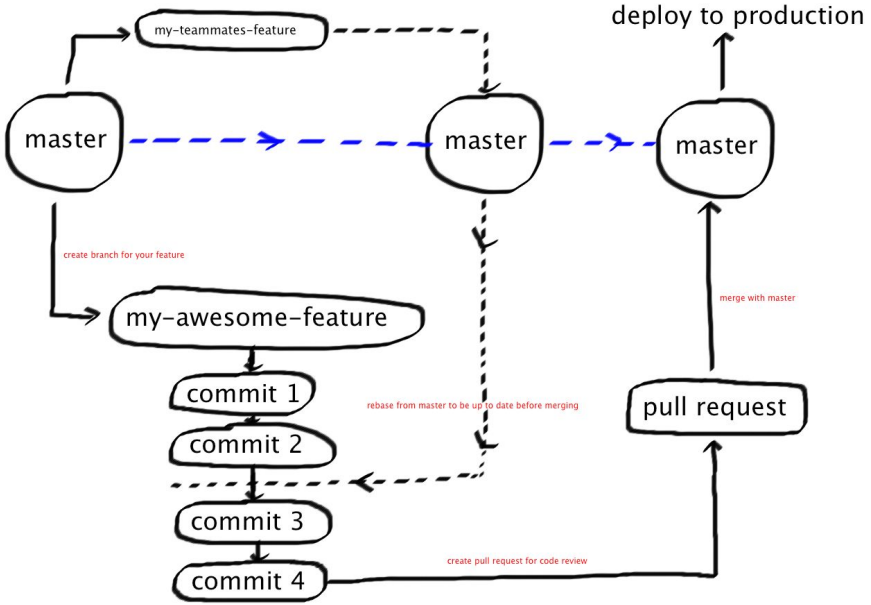
Fortunately, GitHub is...

- a user-friendly service wrapping Git
- full of handy collaboration tools, supporting the defacto standard development workflow: fork / branch / pull request
- useful for running code-based courses like this one

The University Of Sheffield

GitHub: fork, [branch], commit, push, [pull request]





GitHub in COM3505

GitHub classroom creates an “assignment”, which has a link that will trigger initialisation of a repository for you

Each week this repository will be updated with exercises for your lab classes, and example code solutions for previous exercises

After the lecture:

- assignments, example code and notes are on github.com
- if you don’t have an account, set one up
- go to this link: tinyurl.com/com3505-2017-b
- this will create a repository for you — clone it into your own filespace
- see Notes/Week01.mkd to get started
- all your code **MUST** be checked in and pushed as soon as complete

Weekly schedule

- the lecture
- we push notes, description of the exercises (in `Notes/`) and example code (in `Thing/`) into your github repository
- you pull
- you code up the exercises (in `MyThing/`)
  - where requested your device pushes data to a cloud server
- you push your code
- repeat

weeks 1 and 2 are different!

- two lectures and two labs in week 1
- one lab in week 2
- you get extra lab time this week to help get started
  - Lecture 1: **Mon 12**, DIA 2LT6
  - Lab week 01 (part 1): **Tues 10**, NC PC
  - Lecture 2: **Thurs 12**, DIA LT8
  - Lab week 01 (part 2): **Fri 9**, DIA 2.02
  - Lab week 02: **Mon (2nd) 9**, DIA 2.02
- then nothing until week 3...

The University Of Sheffield

LiPo alert: please don’t kill yourself (or anyone else)!

- We will be supplying you with a Lithium Polymer (LiPo) battery.
- These batteries frequently contain as much energy as a hand grenade.
- Their proper usage and management is essential.
- Even big companies with skilled engineering teams find them difficult (e.g. Samsung’s exploding Galaxy Note 7).
- You **MUST** follow our instructions about their care and usage diligently.
- If in doubt ask!

LiPos (2) — don’t try this at home!

Your **TODO** list for week 1:

Note: you **should** be able to use the University managed desktop (windows) machines, or the Uni Linux boot, or your own machines. We will **actively support** the use of the Linux boots first and foremost!

- Get the assignments: tinyurl.com/com3505-2017-b (**WRITE THIS DOWN!!!**)
- Clone your GitHub repository (all your work **must** be pushed to this repository promptly each week)
- Read and digest Notes/Week01.mkd
- Send your email address & github ID to our cloud server
- Do the reading
- Make sure you understood the lecture; review the slides tinyurl.com/com3505l1 if needed
- See you in the lab! (note: working in 2s on Friday; 1s after)