Internet of Things
IO 4041

Week 2

Overview of Computer Networks
&
Reference Models

Computer Network

- A group of computers and associated devices that are connected by communication facilities
- Local Area Network (LAN)
- Wide Area Network (WAN)
- Metropolitan Area Network (MAN)

- Intranet
- Extranet

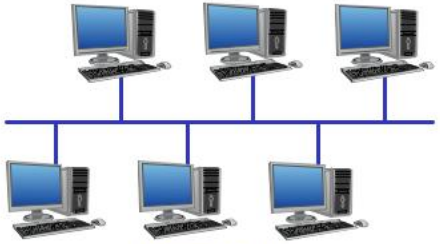
- WLAN: wireless LAN (Wi-Fi)
- WPAN: wireless personal area network (Bluetooth)

Network benefits

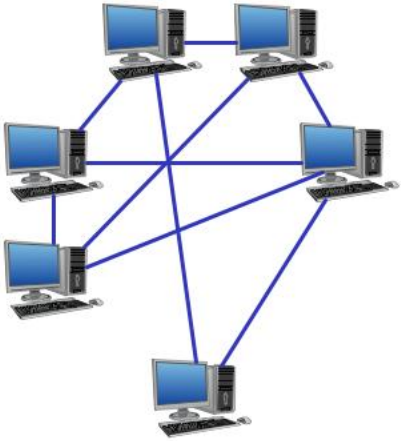
- Information sharing: Authorized users can use other computers on the network to access and share information and data. This could include special group projects, databases, etc.
- Hardware sharing: One device connected to a network, such as a printer or scanner, can be shared by many users.
- Software sharing: Instead of purchasing and installing a software program on each computer, it can be installed on the server. All of the users can then access the program from a single location.
- Collaborative environment: Users can work together on group projects by combining the power and capabilities of diverse equipment.

Network topologies

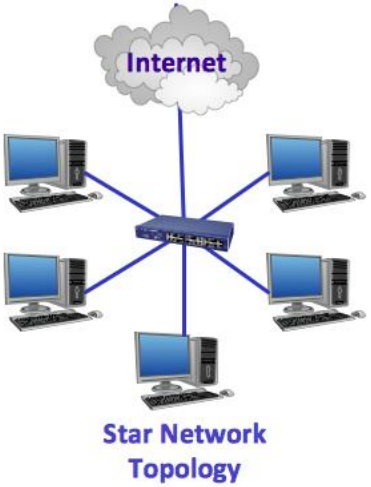
Fully Connected Network Topology



Common Bus Topology



Mesh Network Topology

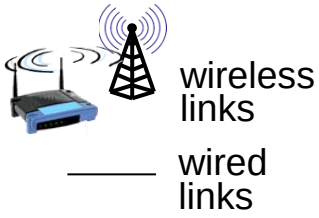


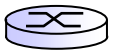
Star Network Topology

Ring Network Topology

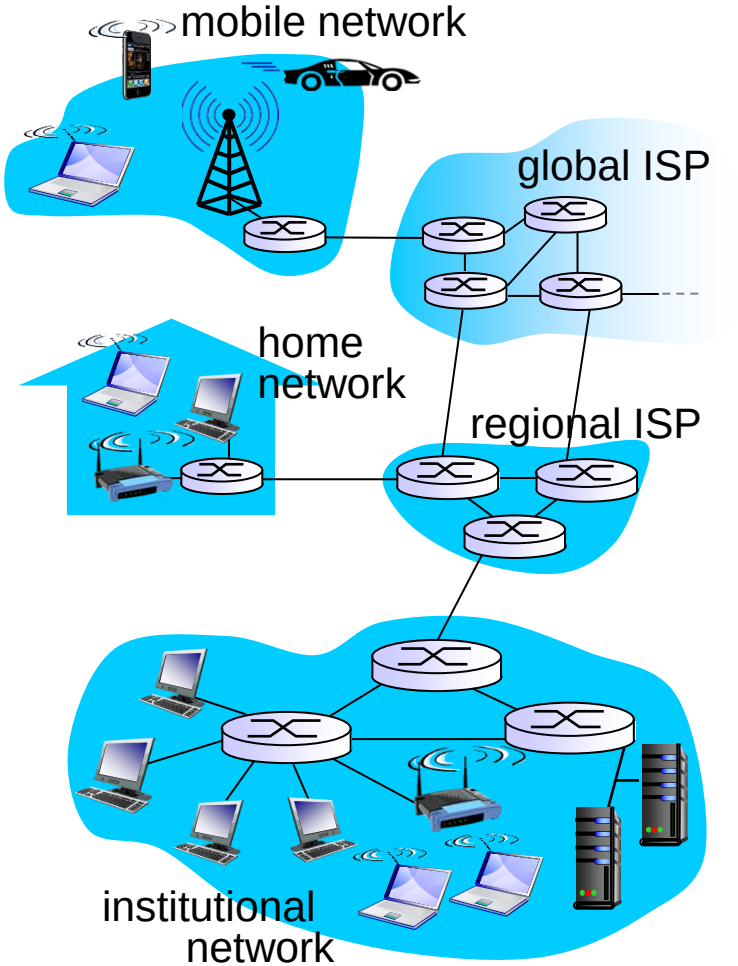
What's the Internet: "nuts and bolts" view

- millions of connected computing devices:
  - *hosts* = *end systems*
  - running *network apps*
- *communication links*
  - fiber, copper, radio, satellite
  - transmission rate: *bandwidth*



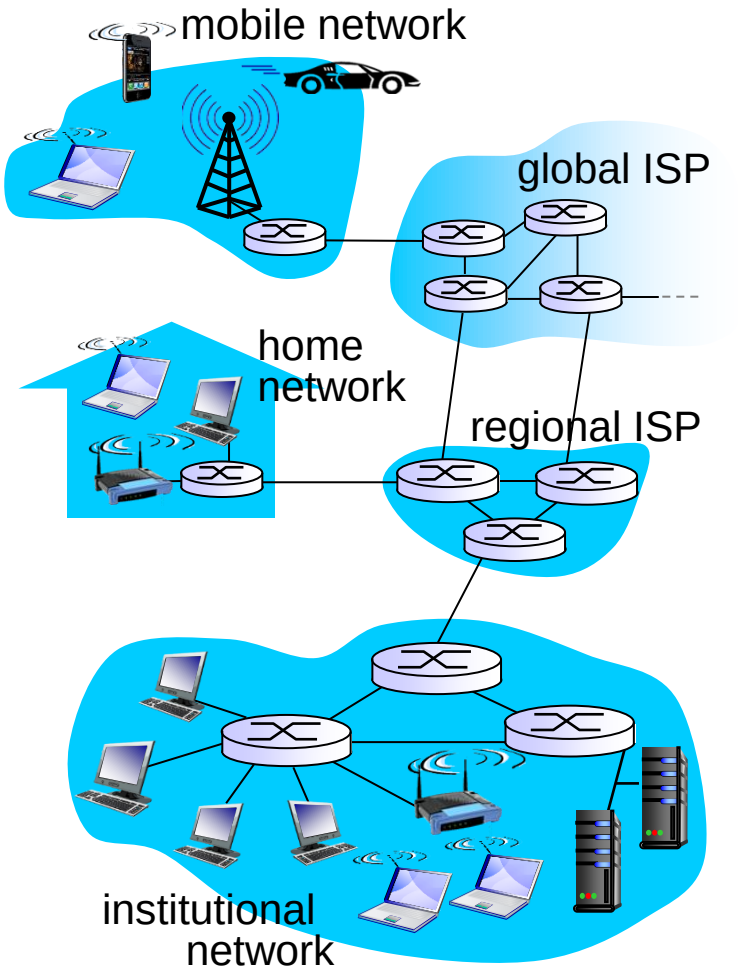


- *Packet switches:* forward packets (chunks of data)
  - *routers* and *switches*



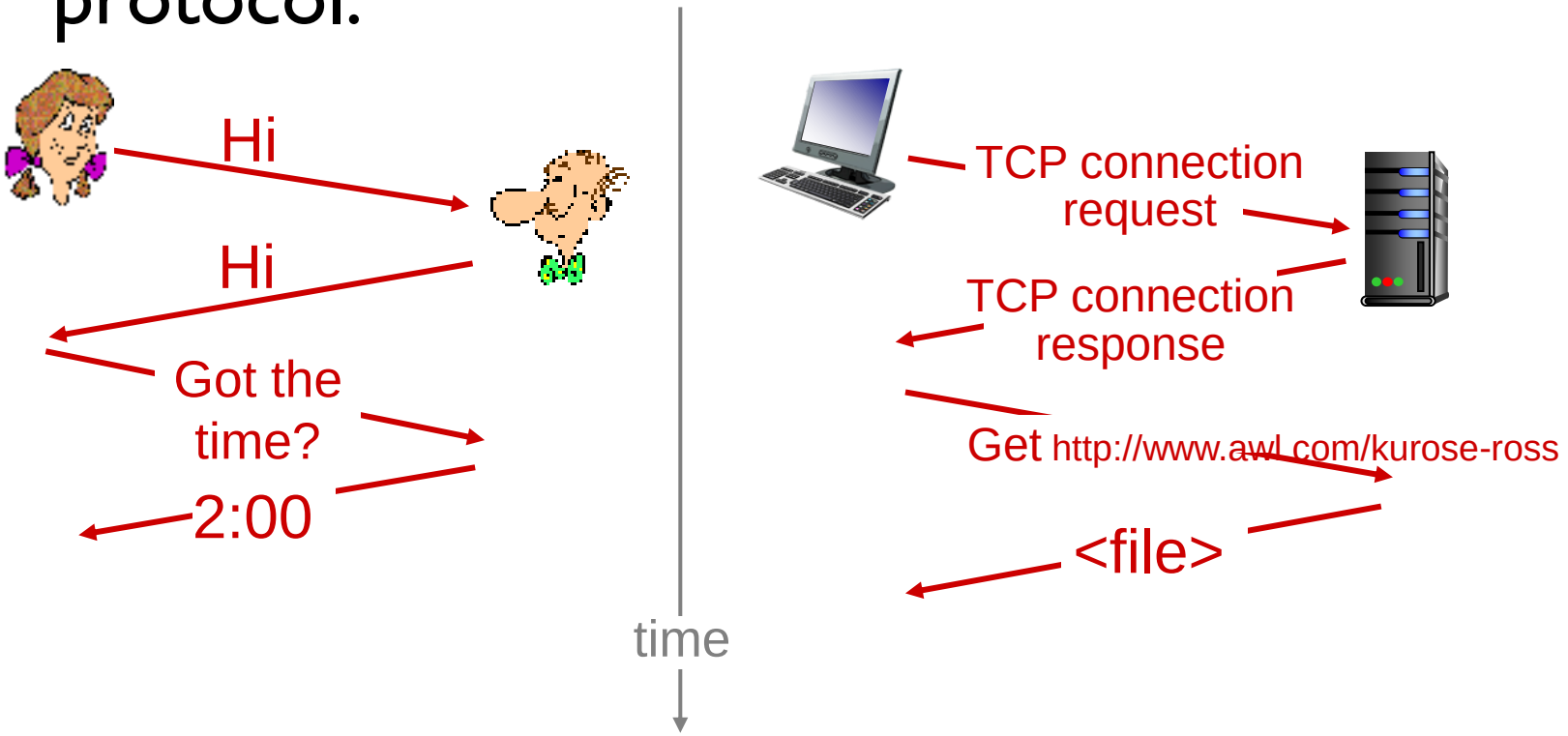
What's the Internet: "nuts and bolts" view

- *Internet:* "network of networks"
  - Interconnected ISPs
- *protocols* control sending, receiving of msgs
  - e.g., TCP, IP, HTTP, Skype, 802.11
- *Internet standards*
  - RFC: Request for comments
  - IETF: Internet Engineering Task Force



What's a protocol?

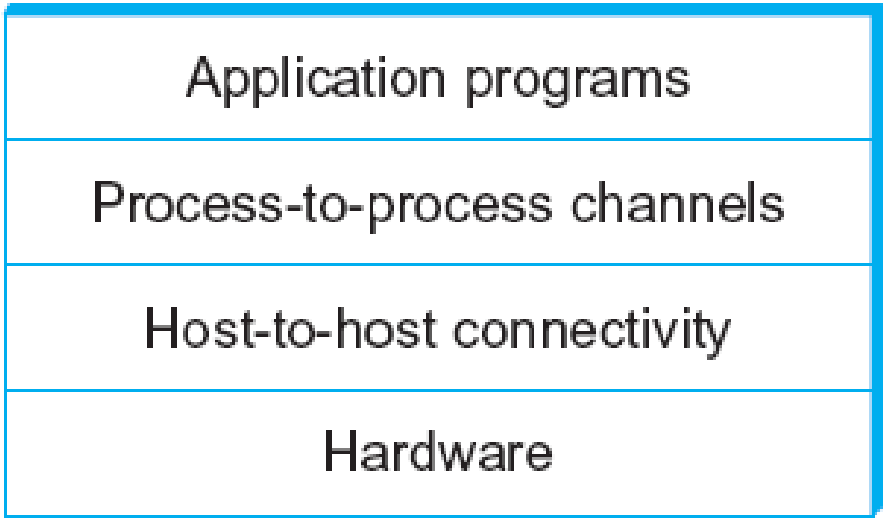
a human protocol and a computer network protocol:



Network Protocols

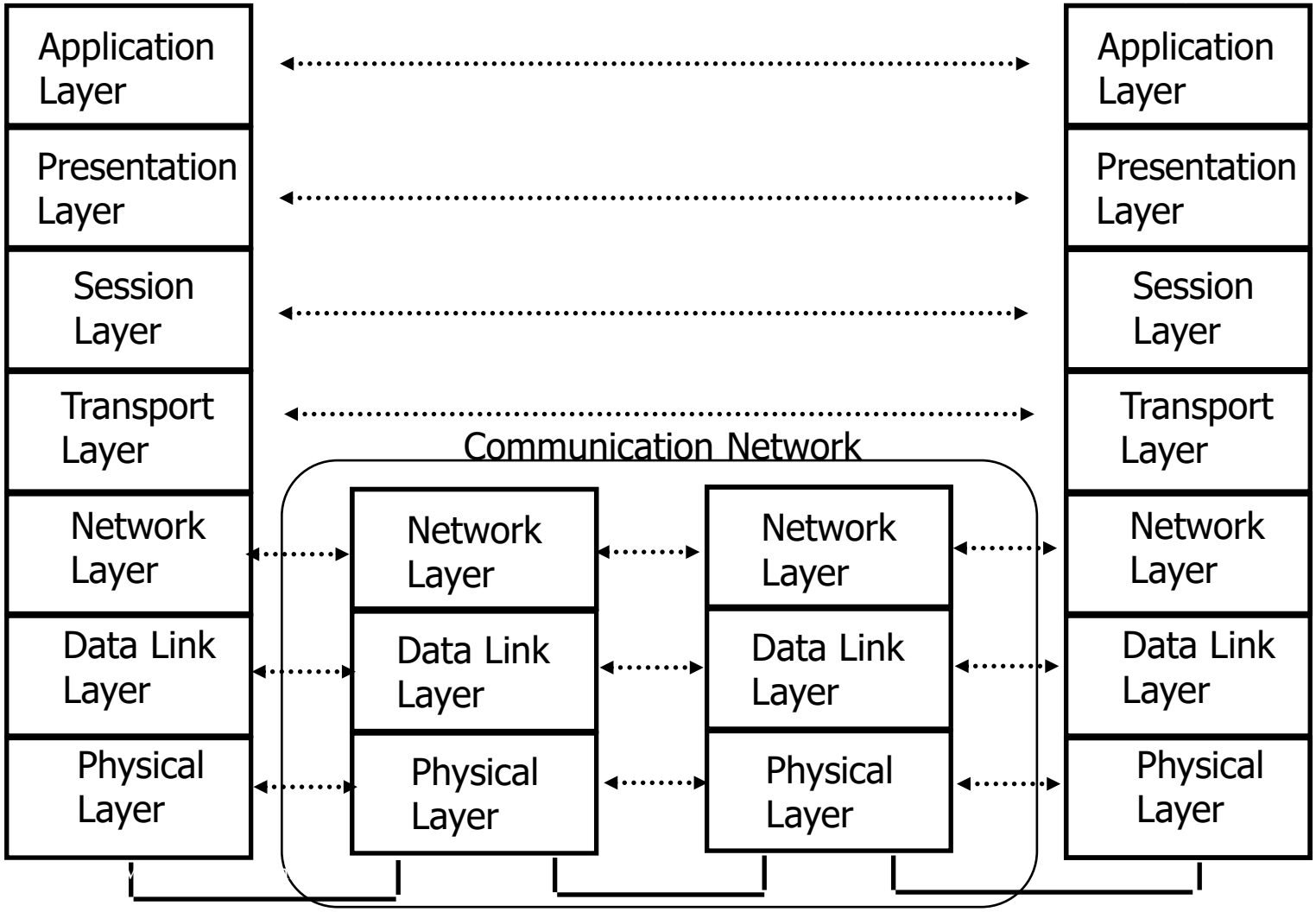
- Protocols are the **building blocks** of a network architecture
- Formal standards and policies enabling communication
- IEEE (Institute of Electrical and Electronics Engineers): standardization
  - Example: Project 802
    - 802.3: Ethernet
    - 802.11: WLAN
    - 802.15: WPAN

Network Architecture

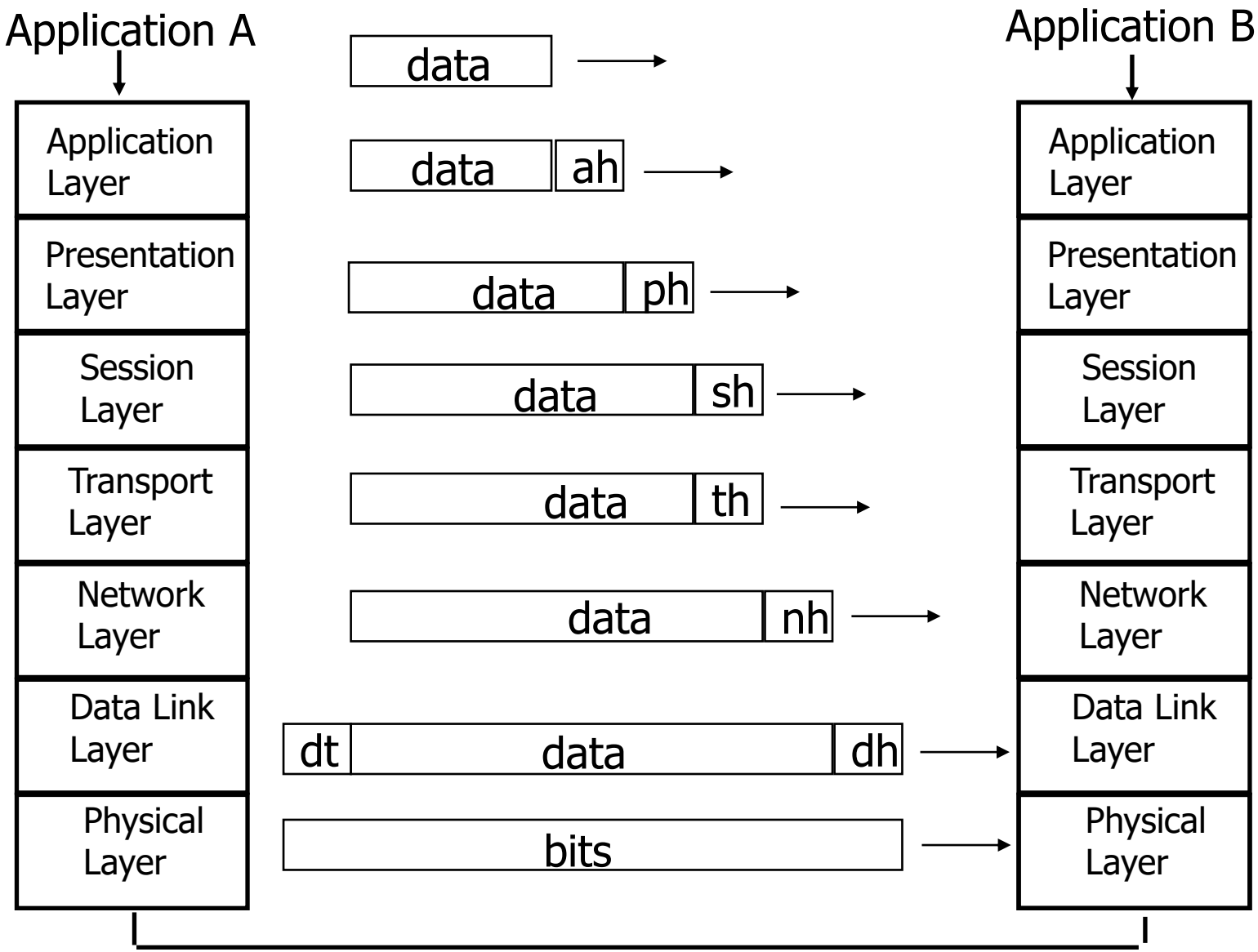


Example of a layered network system

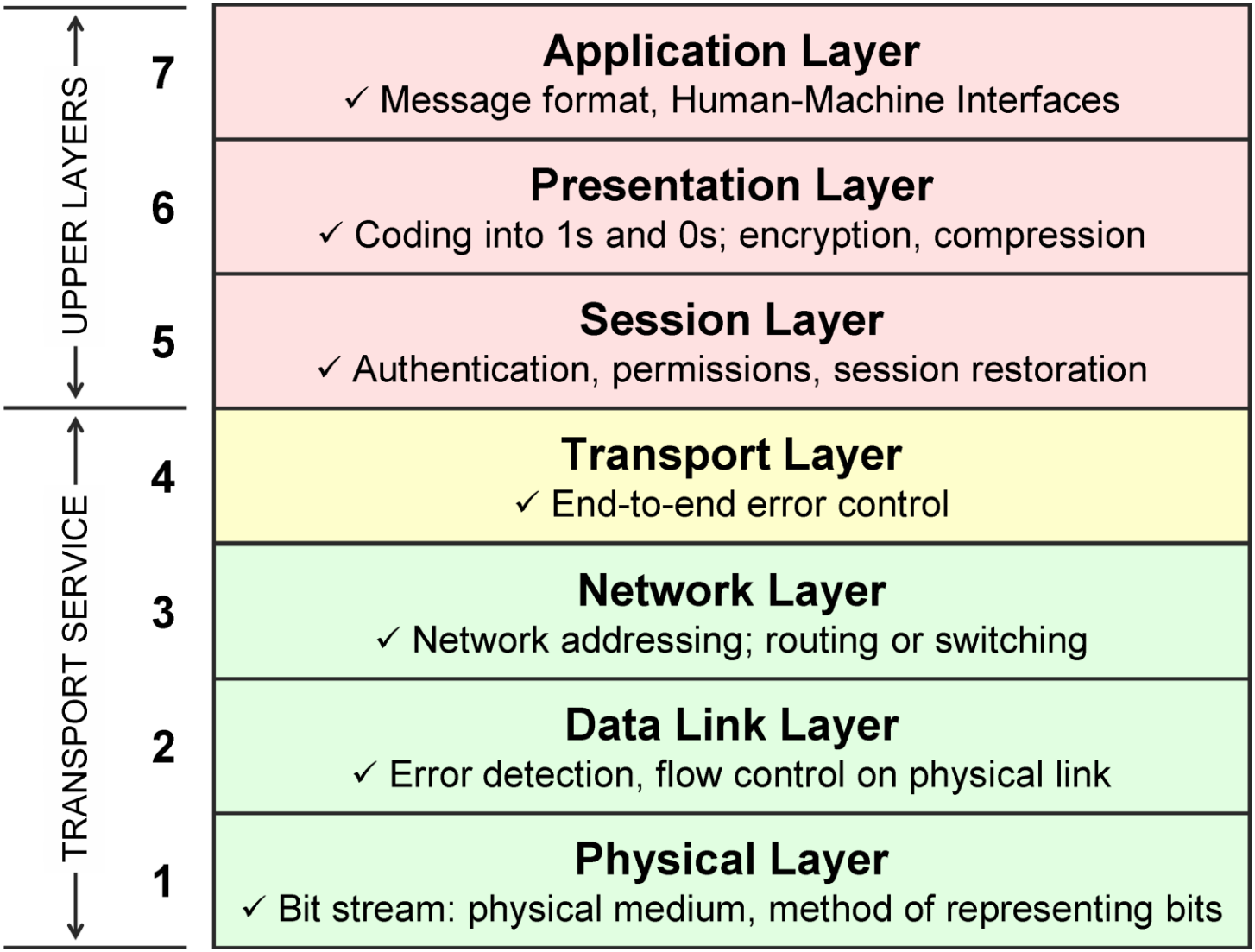
OSI reference model



Network protocol headers



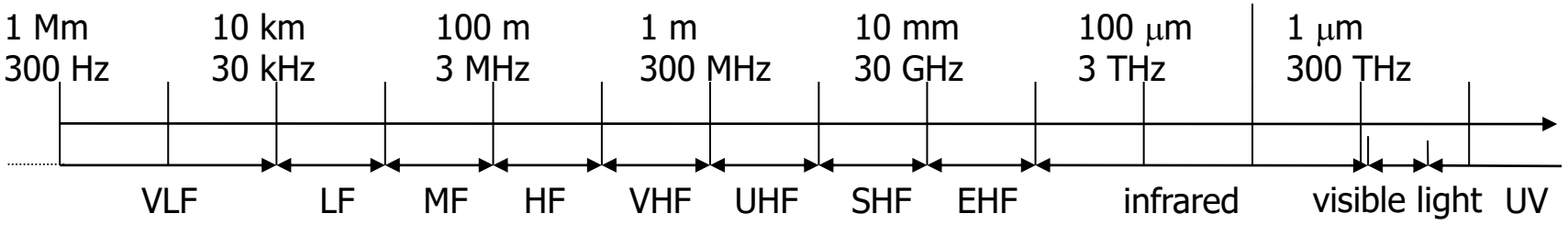
Open System Architecture



Wireless Characteristics

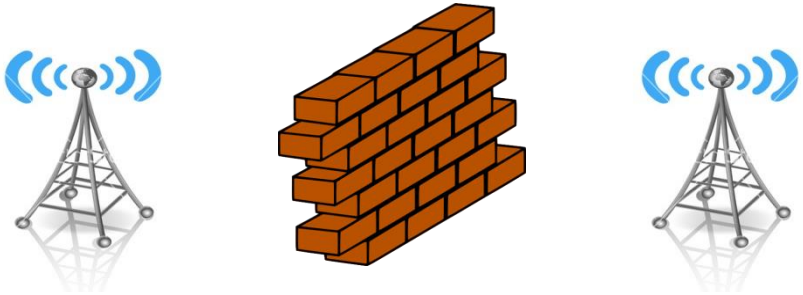
- VLF = Very Low Frequency
- LF = Low Frequency
- MF = Medium Frequency
- HF = High Frequency
- VHF = Very High Frequency
- UHF = Ultra High Frequency
- SHF = Super High Frequency
- EHF = Extra High Frequency
- UV = Ultraviolet Light

- Frequency and wave length
  - $\lambda = c/f$
  - wave length λ, speed of light $c \cong 3 \times 10^8$m/s, frequency f



Frequencies for Mobile Communication

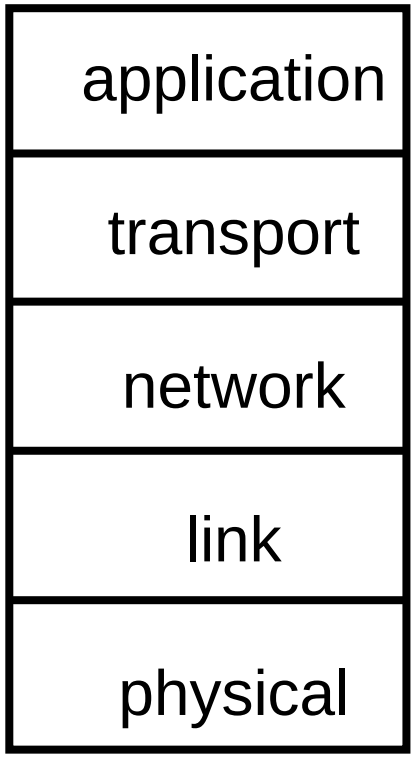
- Low Frequencies:
  - low data rates
  - travel long distances
  - follow Earth's surface
  - penetrate objects and water (submarine communication)



- High Frequencies:
  - high data rates
  - short distances
  - straight lines
  - cannot penetrate objects ("**Line of Sight**" or **LOS**)

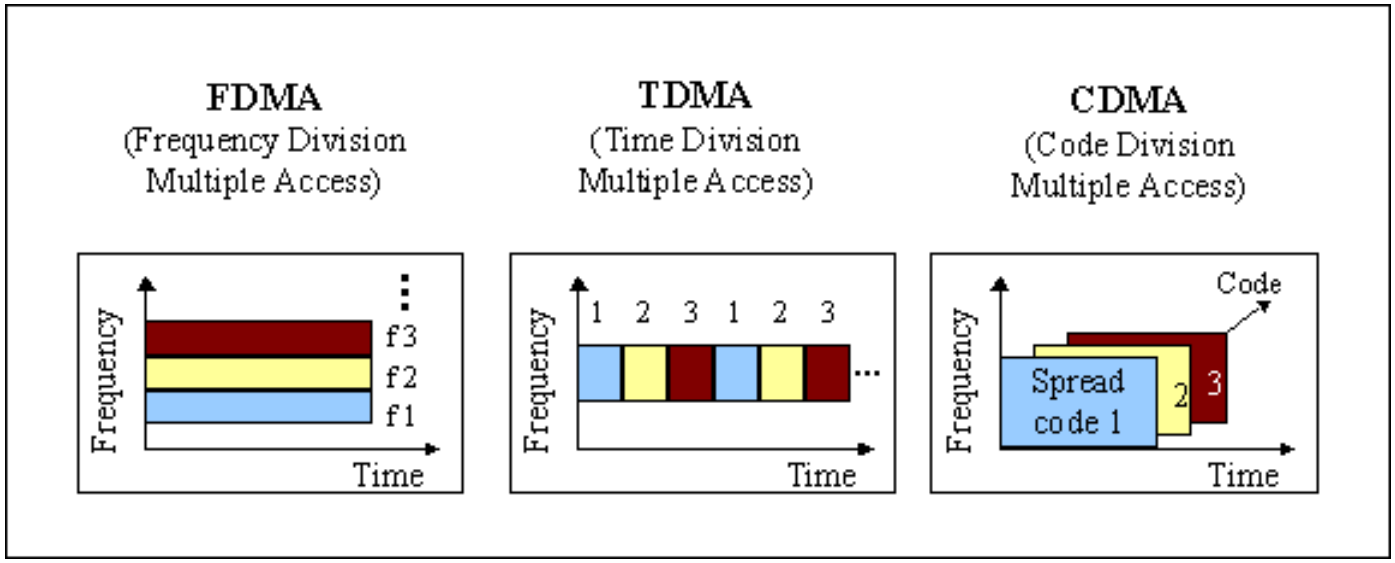
Internet protocol stack

- *application:* supporting network applications
  - FTP, SMTP, HTTP
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.111 (WiFi), PPP
- *physical:* bits "on the wire"

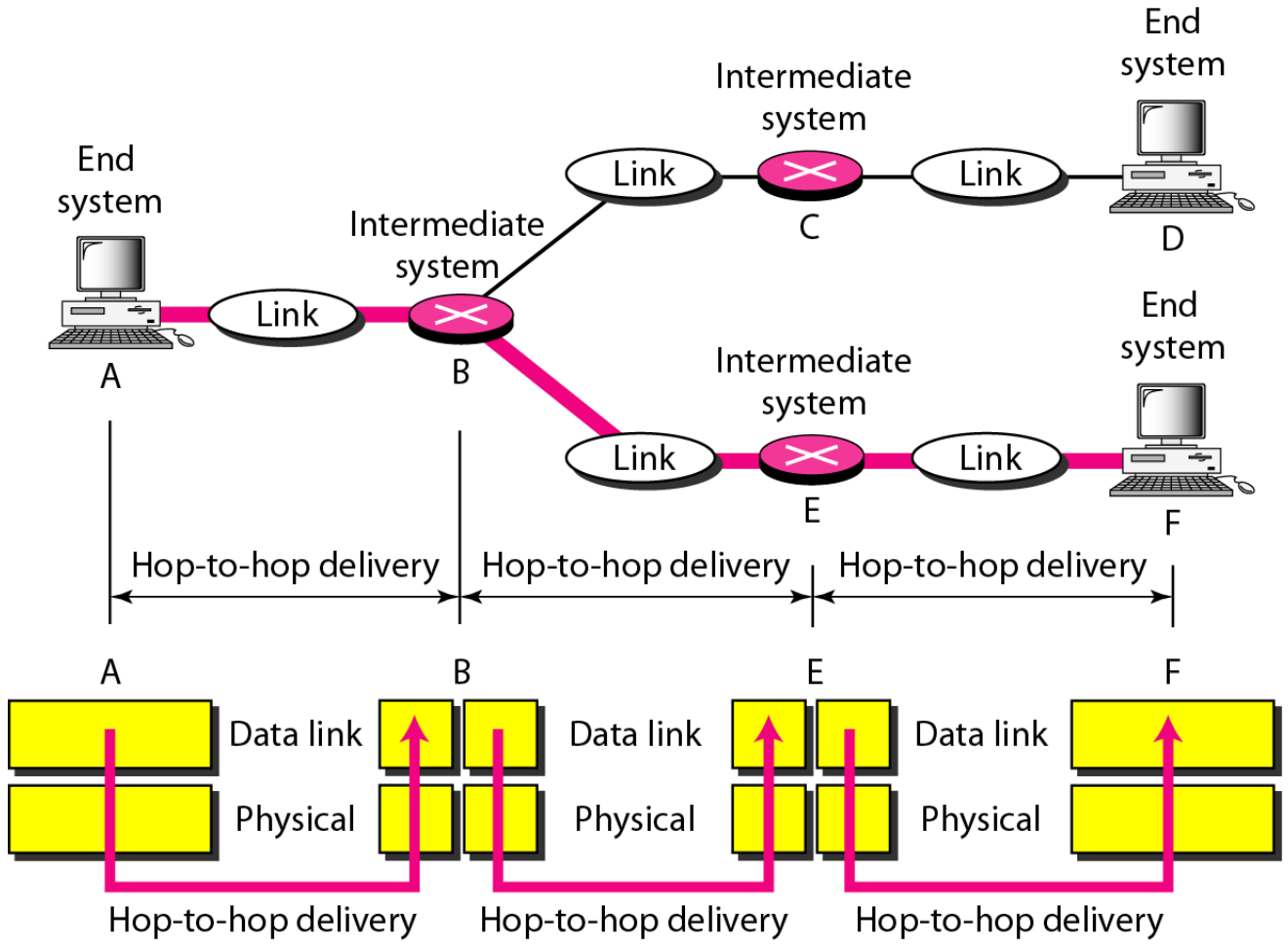


Data link layer

- Defines when/how medium will be accessed for transmission
- Units typically called “frames”; error detection/correction; divided into sub-layers, including: MAC = Medium Access Control (MAC address 6f:00:2b:23:1f:32)
- Cell phone example:



Hop-to-hop delivery



Example: Ethernet

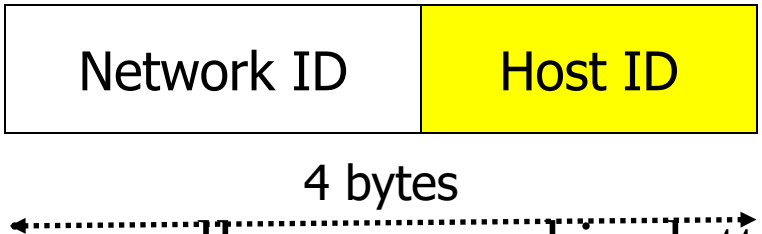
- Medium Access Control (MAC) protocol
- **CSMA/CD** Protocol
  - **C**arrier **S**ense
  - **M**ultiple **A**ccess
  - **C**ollision **D**etection

Example: Wi-Fi (802.11)

- **CSMA/CA** Protocol
  - **C**arrier **S**ense
  - **M**ultiple **A**ccess
  - **C**ollision **A**voidance

Network Layer (Layer 3)

- **Dominant protocol: IP = Internet Protocol**
- Addressing and routing (sender & receiver IP address)
- Uses 32 bit **hierarchical address space** with location information embedded in the structure



- IPv4 address is usually expressed in dotted-decimal notation, e.g.:

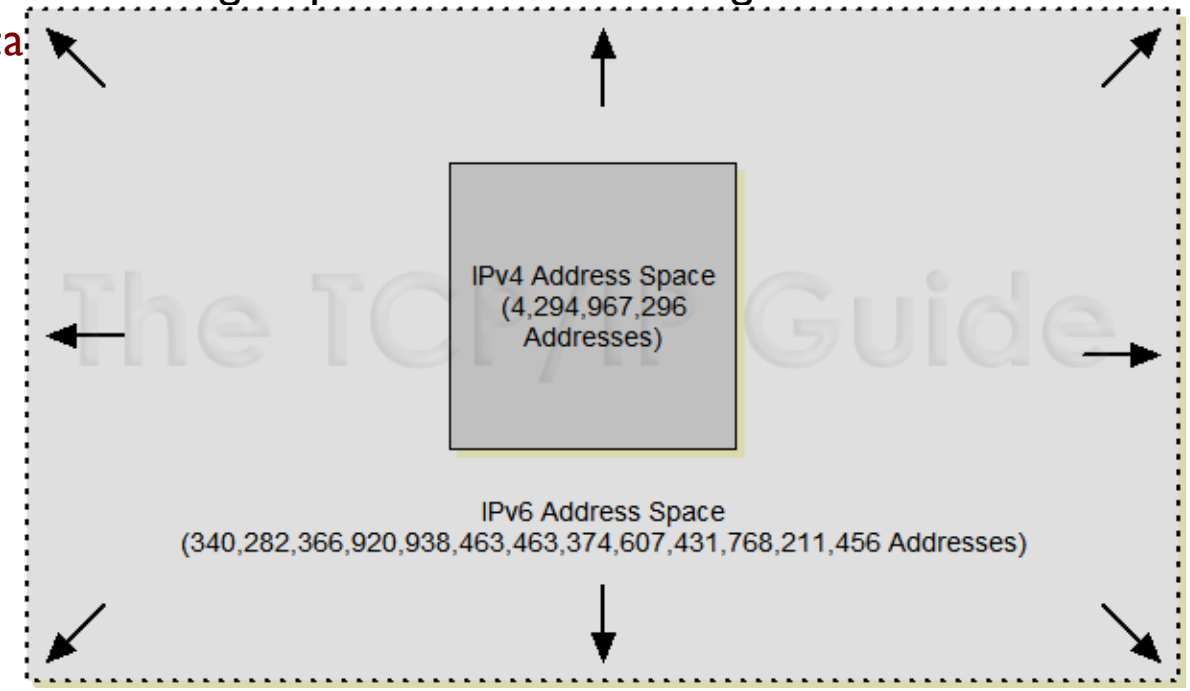
128.100.11.56

IPv6

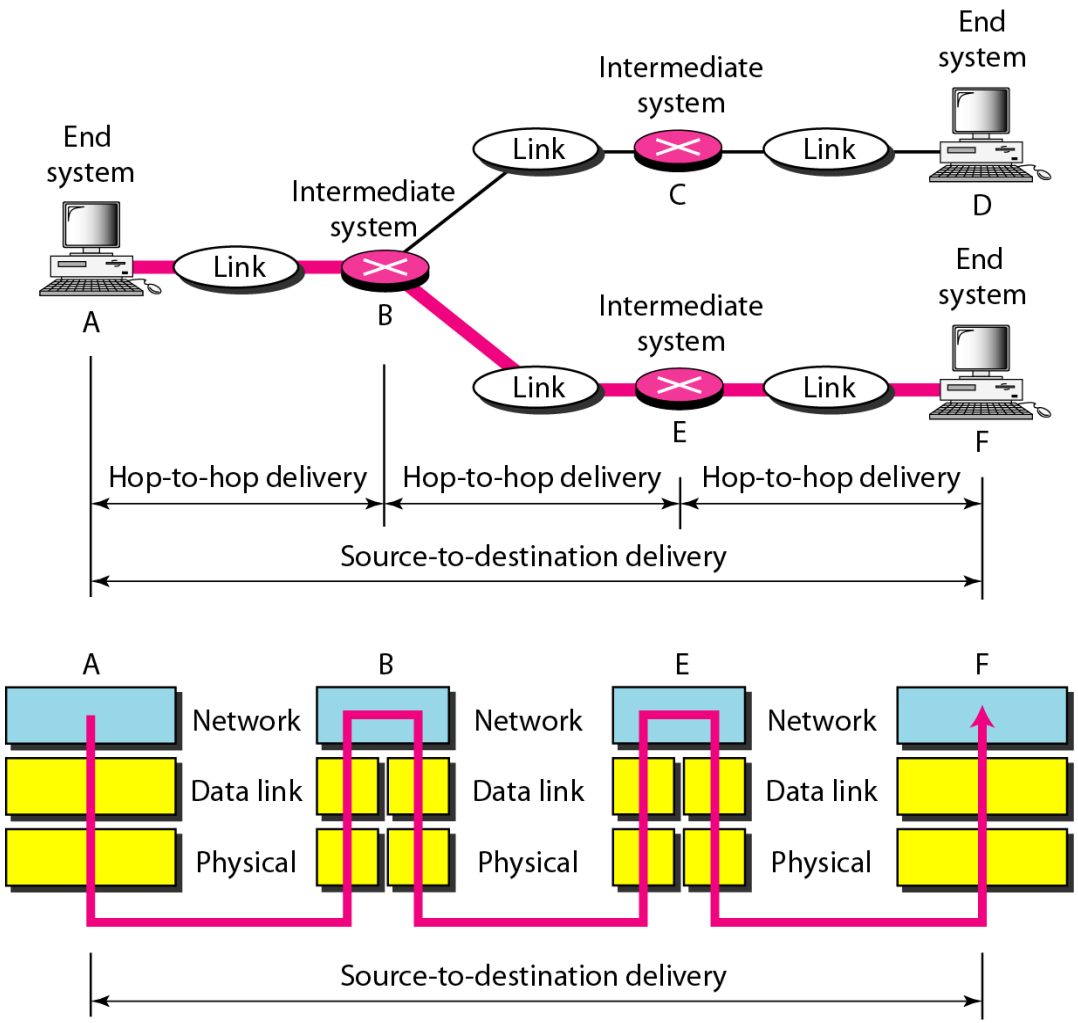
- IPv6 addresses are 128 bits long
- 16 bytes of IPv6 address are represented as a group of hexadecimal digits, separated by colons, e.g.:

  2000:fdb8:0000:0000:0001:00ab:853c:39a1
- Shorthand – leave out groups of zeros and leading zeros:

  2000:fdb8:::1:a



Source-to-destination delivery



Transport Layer (Layer 4)

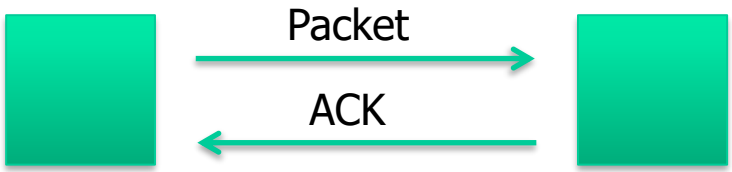
- **UDP** (User Datagram Protocol)

- Adds more addressing: "**ports**"
  - IP address tell you which computer
  - Ports tell you which application on that computer
  - Example: a web server "listens" to requests on port 80
  - Web browser: http://www.google.com:80 = http://216.58.216.100:80
    - ":80": optional

  - **Unreliable!**
    - Packets can get lost; packets can arrive out of order

Transport Layer

- **TCP** (Transmission Control Protocol)
- **Reliable** protocol!
- Adds ports (just like UDP), but also provides:
  - In-order delivery of packets (using sequence numbers)
  - Reliable delivery: using acknowledgment (ACK) packets

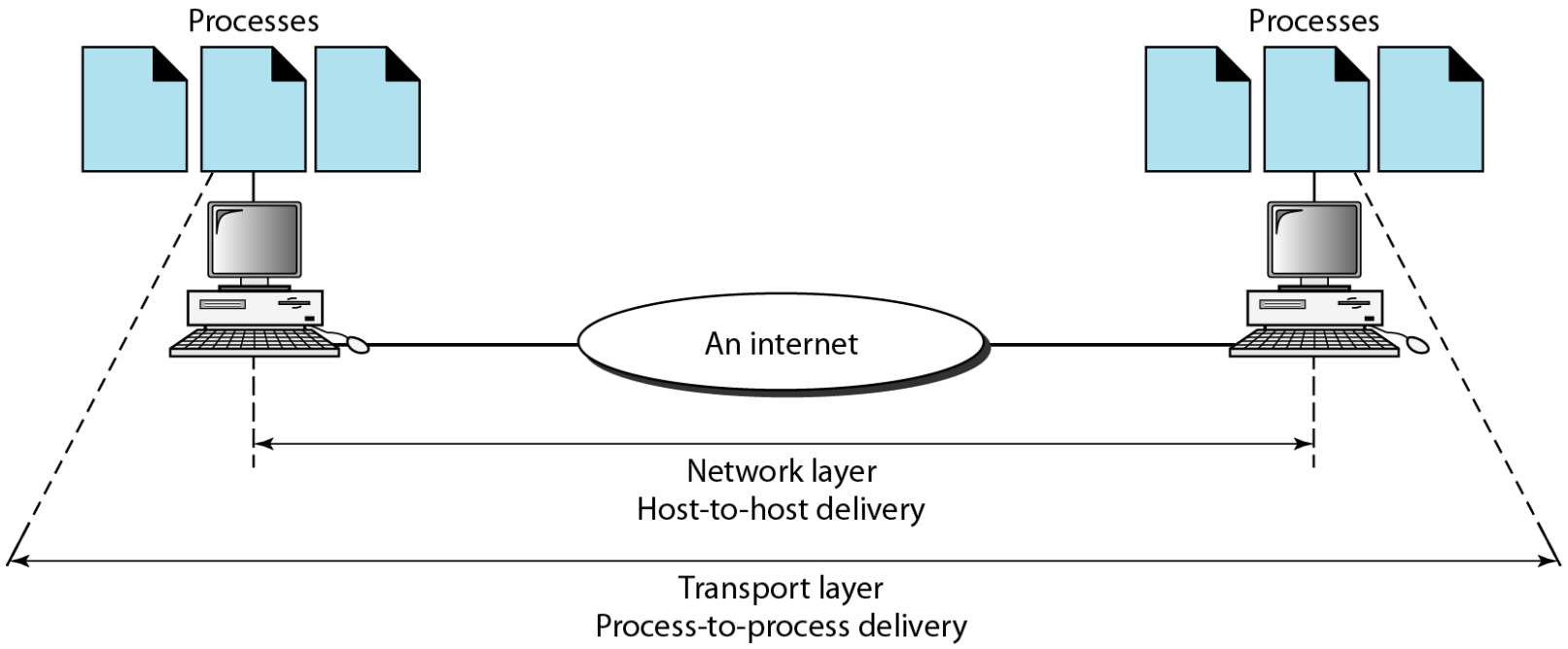


  - **Flow control & congestion control:**
    - Allows receiver to slow down sender
    - Allows "network" to slow down sender

UDP vs TCP

- TCP:
  - typical choice of most applications
  - do not want to lose data, out-of-order arrival, etc.
  - email, web traffic, financial transactions, etc.
- UDP:
  - can be "faster"
    - no flow/congestion control "slowing down" traffic
    - no retransmissions
    - good for "real-time" traffic
  - out-of-order arrival: can also "reorder" at application level
  - loss of data: can be acceptable
    - missing frames in video/audio stream

Reliable process-to-process delivery of a message



APPLICATION LAYER

The application layer is responsible to allow access to network resources (to send and receive data)

- Interface between user applications and lower network services

All applications and utilities communicating with the network fall in this layer

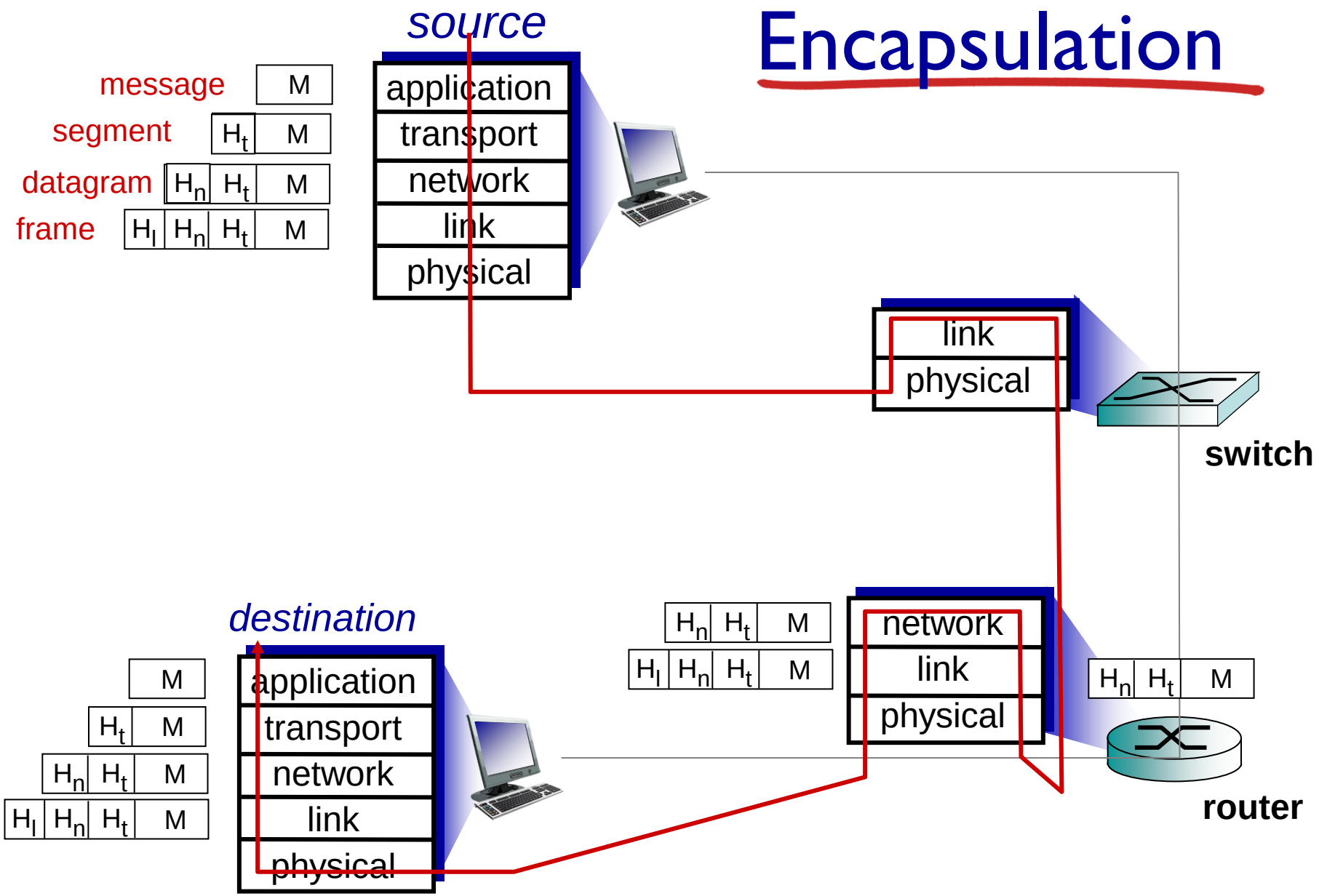
Examples: Browsers, Email clients (outlook express, Opera mail), FTP clients (WinSCP, Filezila)

The two application layers exchange *messages* between each other as though there were a bridge between the two layers.

- However, the communication is done through all the layers.

Protocols: Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), *Simple Network Management Protocol (SNMP)*, *Domain Naming System (DNS)*, Telnet etc.

Encapsulation



IoT

What is the Internet-of-Things?

How Does My Fridge Do That?

- You are leaving the home (sense user)
- There's no milk in fridge (sense object)
- Use this information to make a decision (process)
- Inform user of decision (communicate)

How Does My Fridge Do That?

- **You are leaving the home (sense user)**
  - What type of sensor?
  - Distinguish between parent and child
  - Identify reason for leaving home
  - Identify other contexts (e.g., store hours)
- There's no milk in fridge (sense object)
- Use this information to make a decision (process)
- Inform user of decision (notify)

How Does My Fridge Do That?

- You are leaving the home (sense user)
- **There's no milk in fridge (sense object)**
  - What type of sensor?
  - Is milk needed?
  - No milk or "little" milk? (prediction)
- Use this information to make a decision (process)
- Inform user of decision (notify)

How Does My Fridge Do That?

- You are leaving the home (sense user)
- There's no milk in fridge (sense object)
- **Use this information to make a decision (process)**
  - Where is processor?
  - What are the rules?
  - Fixed rules versus dynamic rules (learning)
- Inform user of decision (notify)

How Does My Fridge Do That?

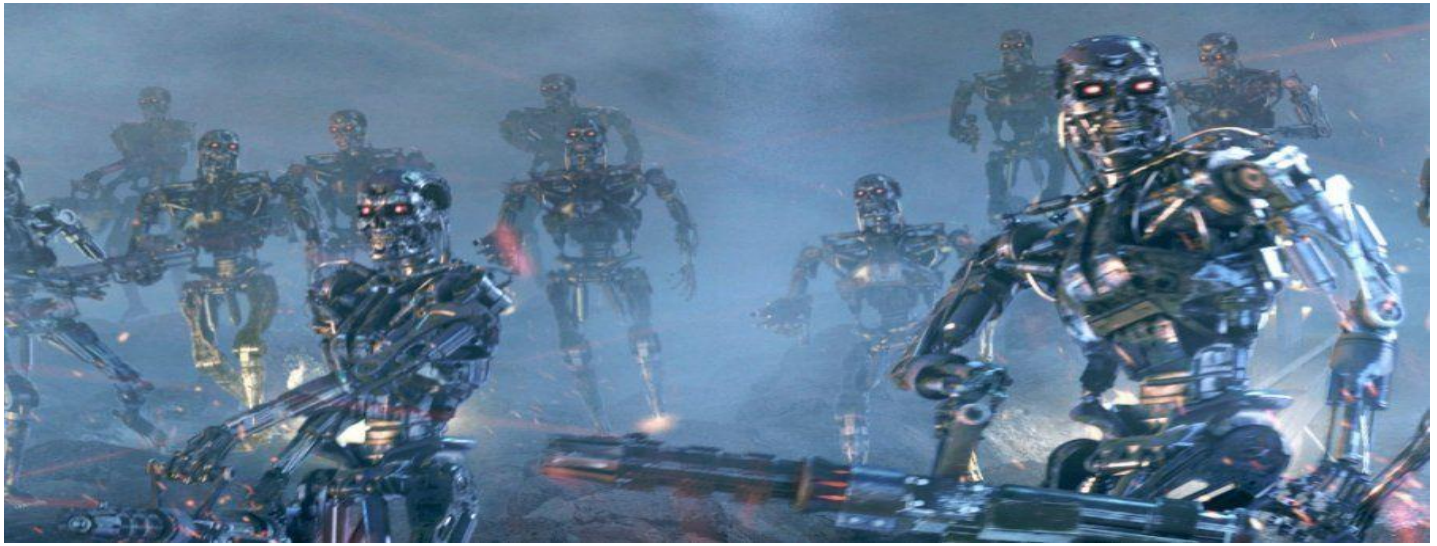
- You are leaving the home (sense user)
- There's no milk in fridge (sense object)
- Use this information to make a decision (process)
- **Inform user of decision (notify)**
  - How?
  - When?
  - Privacy?
  - Subtleness?
  - Information overflow?

Related Areas/Terminology

- **Embedded systems**: not necessarily connected
- **Sensor networks**: collection of sensor devices connected through wireless channels
- **Real-time systems**: focus on time constraints
- **Pervasive/ubiquitous computing**: focus on anytime/anywhere computing

Related Areas

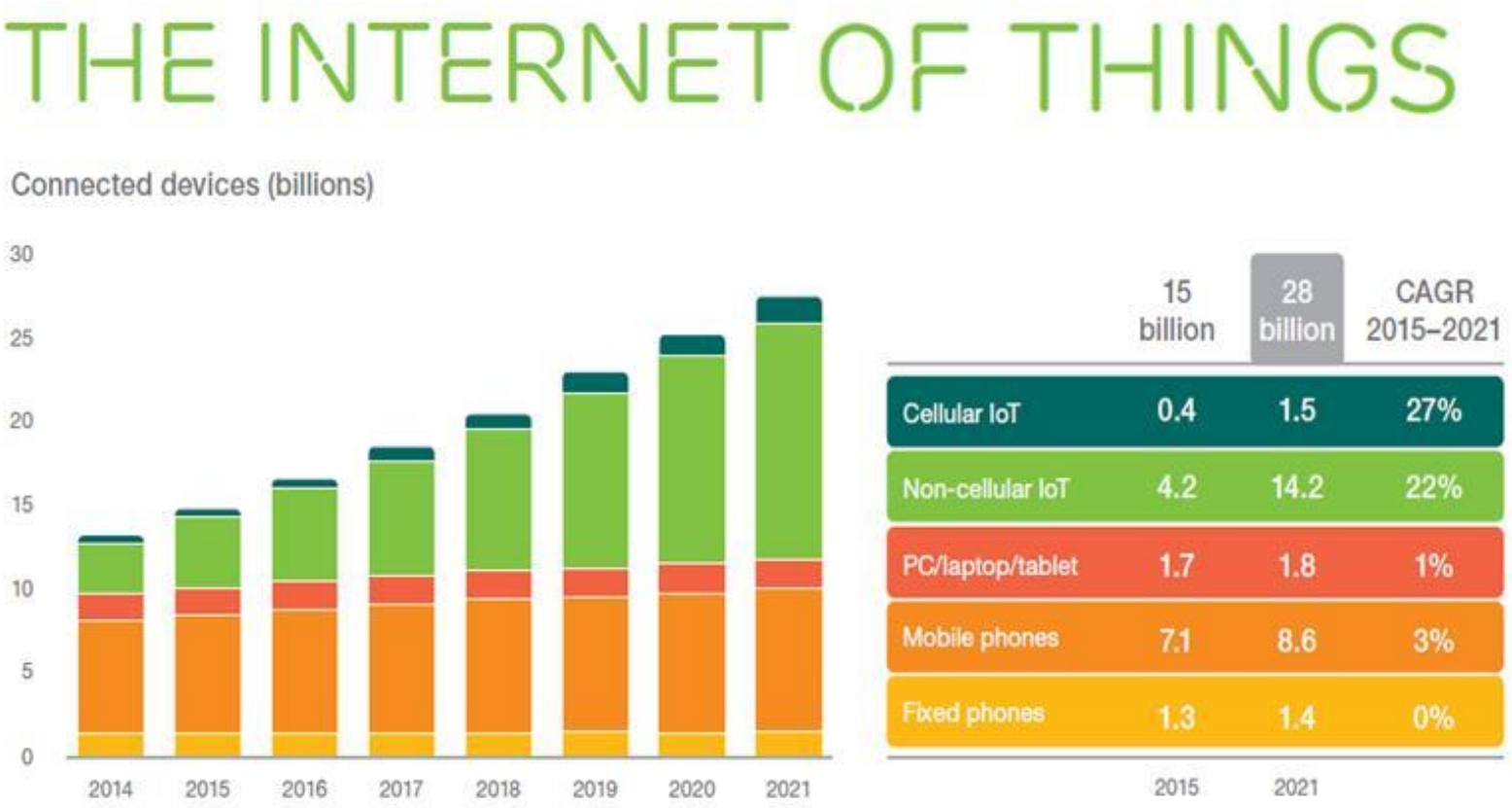
- Machine-to-machine (M2M) communications
- Internet of Everything (Cisco Systems)
- “Skynet” (Terminator movie)



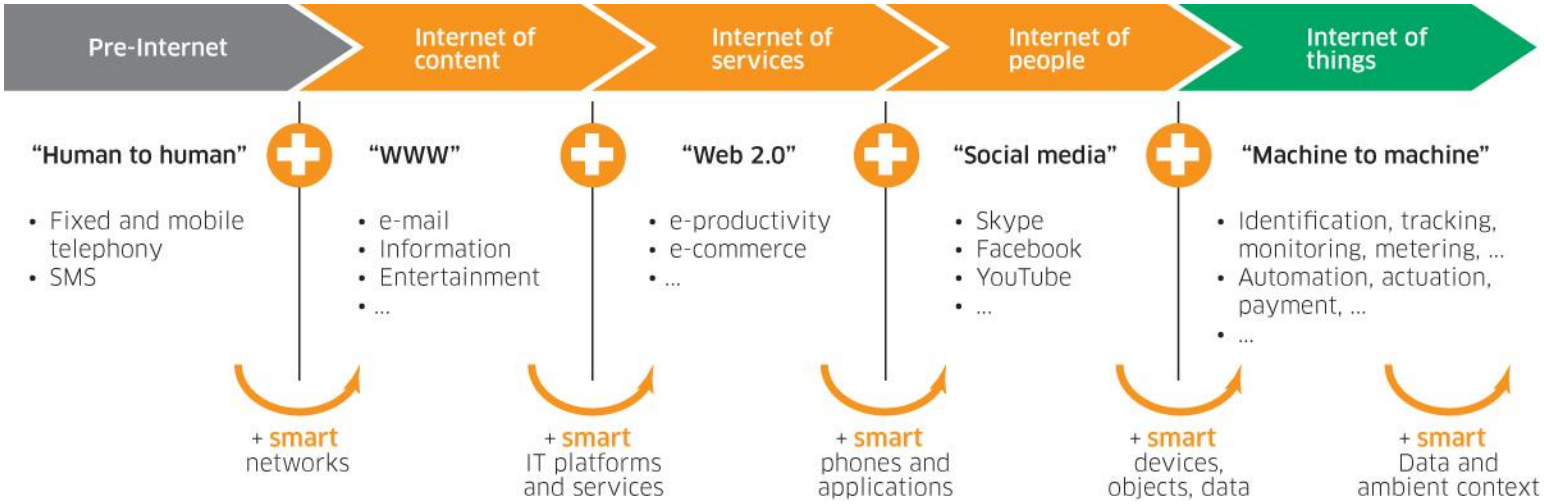
"Internet-of-Things"

- Term coined by British entrepreneur Kevin Ashton, while working at MIT Auto-ID Labs
- Referred to (and envisioning) a future global network of objects connected specifically by RFID (radio-frequency identification)
- Complete automation of data collection
- First article about IoT in 2004 from MIT; called "Internet 0"

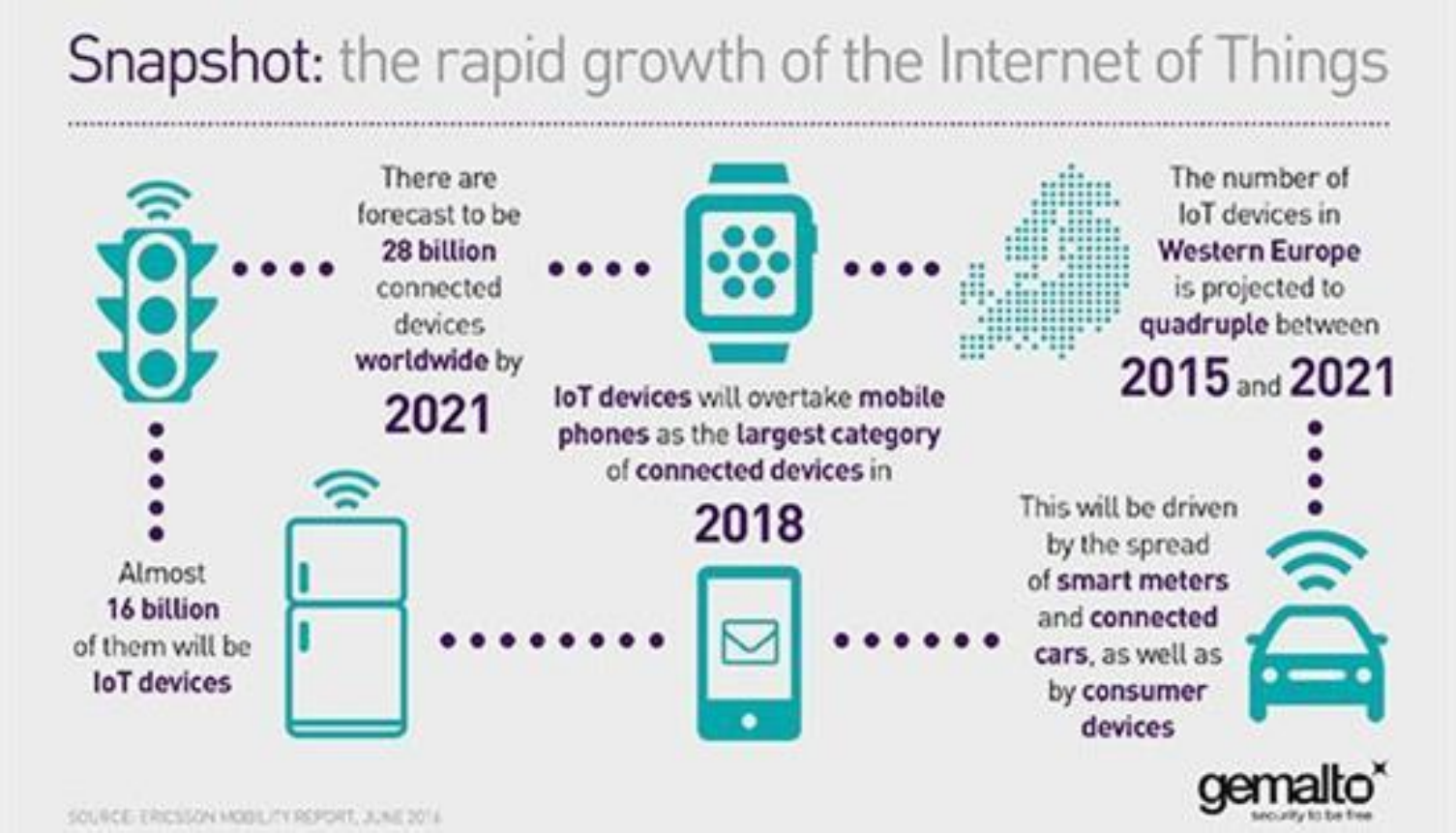
Internet-of-Things Vision & Growth



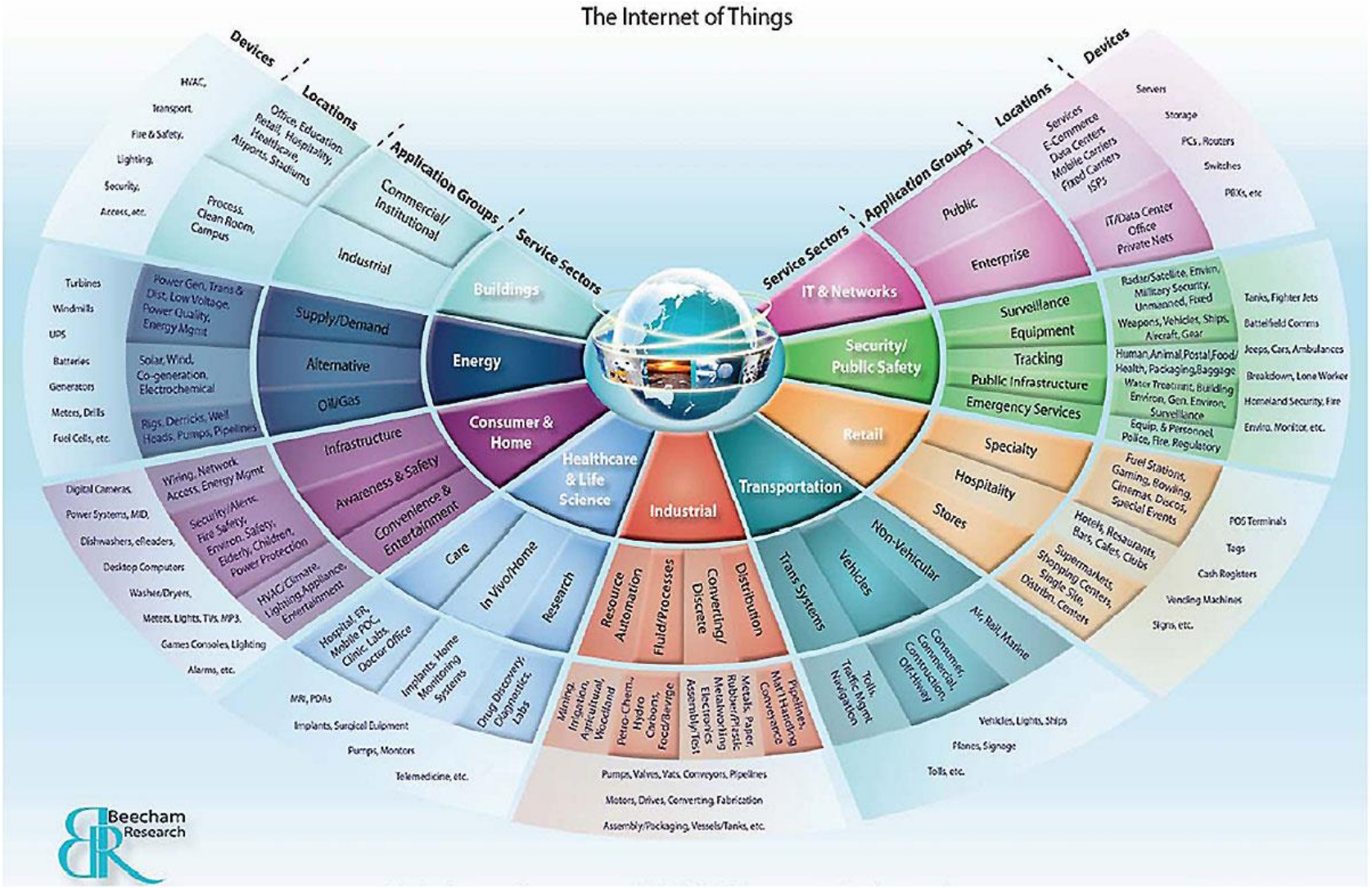
Internet-of-Things Vision & Growth



Internet-of-Things Vision & Growth



Internet-of-Things Vision & Growth

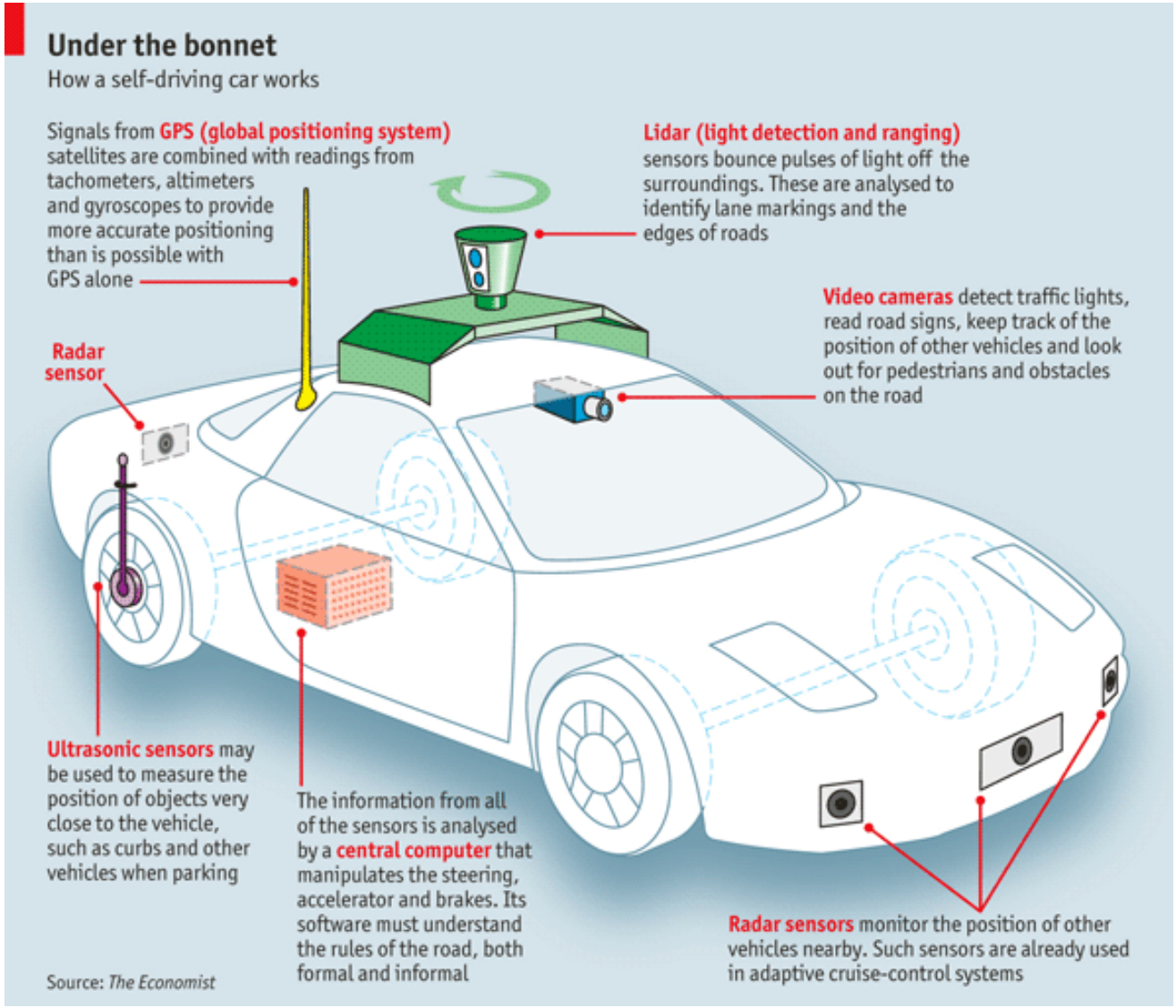


Augment Existing Things

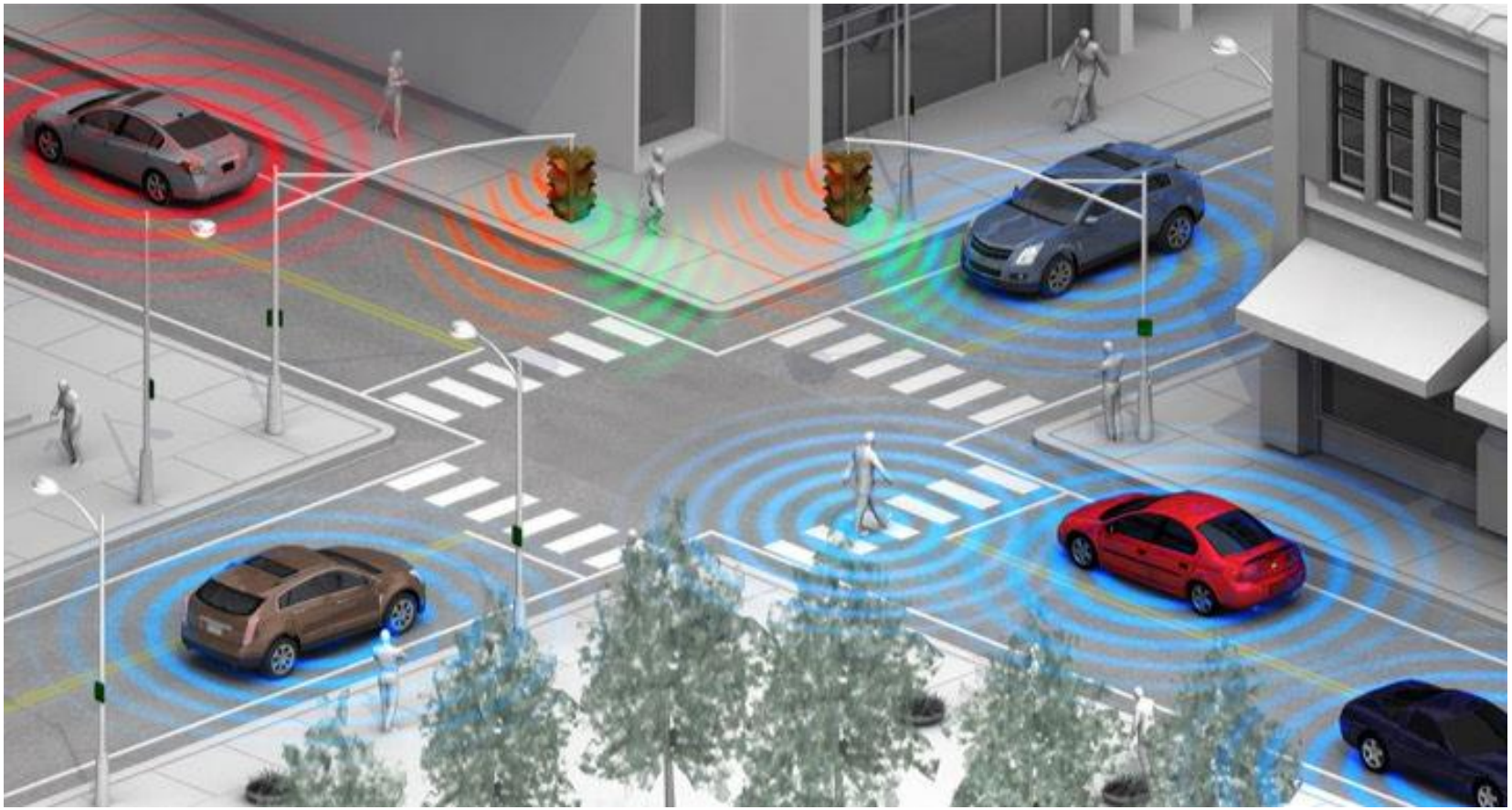
Example: Connected Roadways

- US DoT Statistics for 2012:
  - 5.6million crashes
  - About 31,000 fatalities (26,500 in EU)
  - Over 1.6M injuries
- 1trillion USD in economic loss
- 5.5billion hours of travel delays per year
- CO_2 emissions

Example: Connected Roadways



Example: Connected Roadways



State of Self-Driving Car

Example: Smart & Connected Buildings

- Energy management
- Lighting
- Safety
- HVAC
- Building automation
- Smart spaces

Example: Connected Factory

- New product and service introductions faster
- Increasing production, quality, uptime
- Mitigating unplanned downtime
- Protecting from cyber threats
- Worker productivity and safety

Enablers: Portability

- Reducing the size of hardware to enable the creation of computers that could be physically moved around relatively easily

Enablers: Miniaturization

- Creating new and significantly smaller mobile form factors that allowed the use of personal mobile devices while on the move

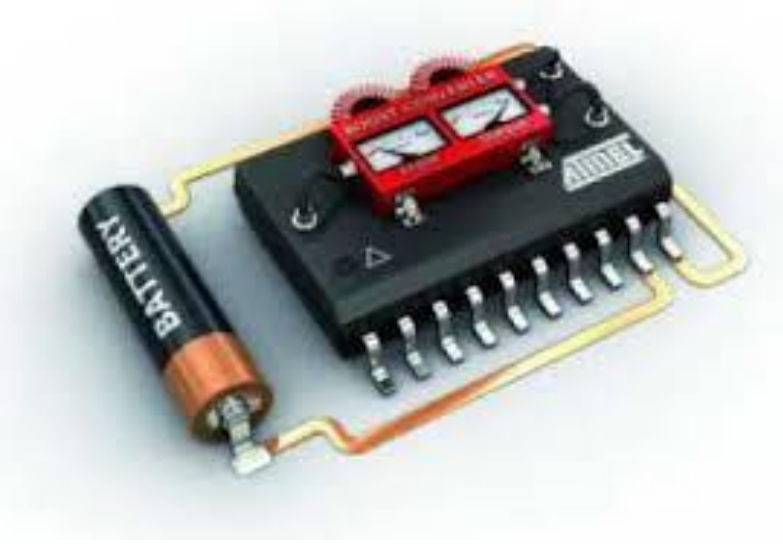
50mm x 50mm

35mm x 35mm

15mm x 15mm

Enablers: Low Power and Low Heat

- Low power architectures
- Low power radios
- Sleep modes
- Energy harvesting



Enablers: Connectivity

- Developing devices and applications that allowed users to be online and communicate via wireless data networks while on the move

Wi Fi

Bluetooth®

4G LTE

Enablers: Convergence

- Integrating emerging types of digital mobile devices, such as Personal Digital Assistants (PDAs), mobile phones, music players, cameras, games, etc., into hybrid devices

Enablers: Ecosystems

- The emerging wave of *digital ecosystems* is about the larger wholes of pervasive and interrelated technologies that interactive mobile systems are increasingly becoming a part of

Example: Smartphone

- Portability: carry it anywhere you want
- Miniaturization: make it possible to build device to fit in your pocket
- Connectivity: Wi-Fi, LTE/4G, cellular, Bluetooth
- Convergence: phone, camera, gaming device, movie streaming, music player, …
- Digital Ecosystem: cloud, social networks, software development kits, app stores, big data, standardization …

Challenges for Smart Objects

- Node Level Challenges
  - Physical size, cost and power consumption
- Network Level Challenges
  - Scale of nodes in network
  - Power and memory constraints of nodes
- Standardization
- Interoperability

A demo with Arduino Uno

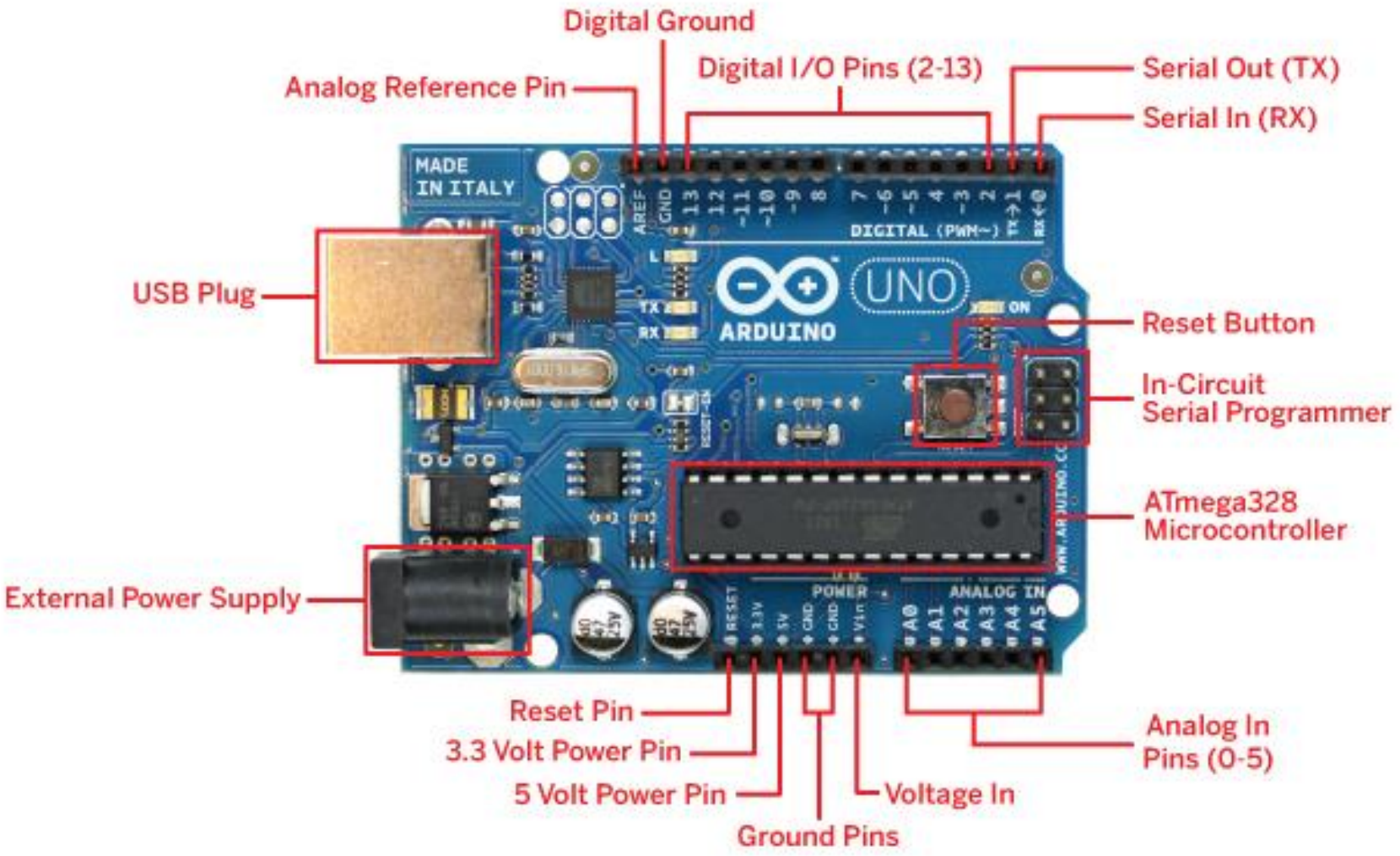
Arduino - Overview

- A Prototype platform based on an easy-to-use hardware and software (open source)

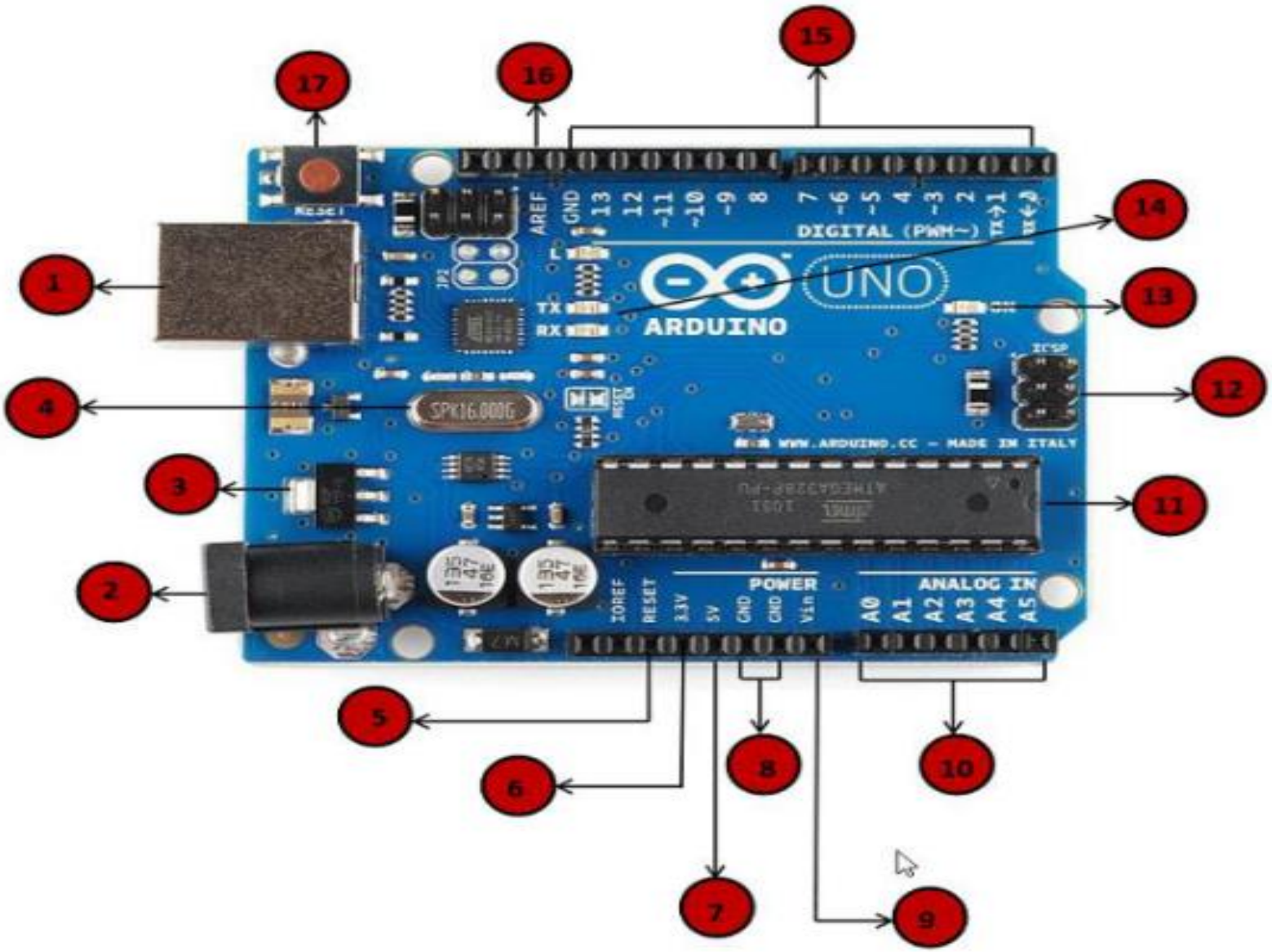
- It consists of a
  - Circuit board (**microcontroller**) which can be programmed

  - Arduino IDE, a ready made software,
    - Used to write and upload the computer code to the physical board

Arduino


Digital Ground
Analog Reference Pin
Digital I/O Pins (2-13)
Serial Out (TX)
Serial In (RX)
USB Plug
Reset Button
In-Circuit
Serial Programmer
ATmega328
Microcontroller
External Power Supply
Reset Pin
3.3 Volt Power Pin
5 Volt Power Pin
Ground Pins
Voltage In
Analog In
Pins (0-5)
MADE
IN ITALY
DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN

Arduino


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
DIGITAL (PWM~)
UNO
ARDUINO
TX
RX
POWER
ANALOG IN
WWW.ARDUINO.CC – MADE IN ITALY

Summary

- Microcontroller: ATmega328 (11)
  - Brain of the board
- Power: can be powered via
  - the USB connection (1)
  - with an external power supply (2)

- Operating Voltage 5V

  Input Voltage (recommended) 7-12V
  Input Voltage (limits) 6-20V
- Voltage Regulator (3):

  to control the voltage given to the board and stabilize the DC voltages used by the processor and other elements

Summary

- Pins: **(3.3, 5, GND, Vin) (6,7,8,9)**
  - 5V (7): outputs a regulated 5V from the regulator on the board
  - 3.3V (6) – Supply 3.3 output volt
  - GND (8)(Ground) – There are several GND pins on the Arduino, any of which can be used to ground your circuit.
  - Vin (9) – This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

Summary

- **Input and Output**
  - **14 Digital** I/O Pins (15): of which 6 provide 8-bit PWM (Pulse Width Modulation) output
    - PWM: 3, 5, 6, 9, 10, and 11
  - 6 **Analog Input** Pins (A0 through A5) (10)
    - can read the signal from an analog sensor (like humidity or temperature) and convert it into a digital value that can be read by the microprocessor.
  - DC Current per I/O Pin 40 mA (recommended 20 mA)
  - DC Current for 3.3V Pin 150 mA

Summary

TX and RX LEDs (14)

- Serial: Two labels: TX (transmit) and RX (receive)
- 0 to receive (RX) and 1 to transmit (TX) TTL serial data
- They appear in two places on the UNO board.
  - First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication.
  - Second, the TX and RX led (13).
- The TX led flashes with different speed while sending the serial data.
  - The speed of flashing depends on the baud rate used by the board.
- -RX flashes during the receiving process.

Components

Crystal Oscillator (4)

- Clock Speed 16 MHz

- To help Arduino in dealing with time issues
- Clock Speed or frequency is 16 MHz

Arduino Reset (5,17)

- reset your UNO board (i.e., start program from the beginning) in two ways.
  - First, by using the reset button (17) on the board.
  - Second, connect an external reset button to the Arduino pin labelled RESET (5).

Components

Power led indicator (13)

- This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

AREF (16)

- Analog Reference: used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Components

- Memory
  - Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader SRAM
  - 2 KB (ATmega328) EEPROM
  - 1 KB (ATmega328)
- Resistors
- Capacitors
- LEDs

Arduino - Features

- Able to read analog or digital input signals from different sensors and
  - turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- Allows to control board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (i.e., via uploading software).
- Allows to load a new code onto the board simply via USB cable)no need of an extra piece of hardware)
- uses a simplified version of C++, making it easier to learn to program.

Installation

- **Step 1:** get Arduino board and USB cable
- **Step 2:** Download Arduino IDE Software
- **Step 3:** Power up your board
- **Step 4:** Launch Arduino IDE
- **Step 5:** Open the project
  - Create a new one or
  - Open an existing one
- **Step 6:** Select Arduino board
- **Step 7:** select serial port
- **Step 8:** upload the program to your board

Program structure

- **Sketch:** new terminology in Arduino
- Program can be divided into 3 main parts
  - Structure
    - Setup () function
    - Loop () function
  - Values
  - Functions
  -

Program structure

- Structure
  - Setup () function
    - called when a sketch starts
    - Use it to initialize the variables, pin modes, start using libraries, etc
    - It will only run once, after each power up or reset of the Arduino board
  - Loop () function
    - the **loop()** function loops consecutively, allowing your program to change and respond.
    - Use it to actively control the Arduino board.
- Values
- Functions