INTERNET OF THINGS

INTRODUCTION

COURSE INFORMATION

- 11 Hours of class
- Theory
- Practical
  - Contiki project (6 weeks after last class)

INTERNET OF THINGS

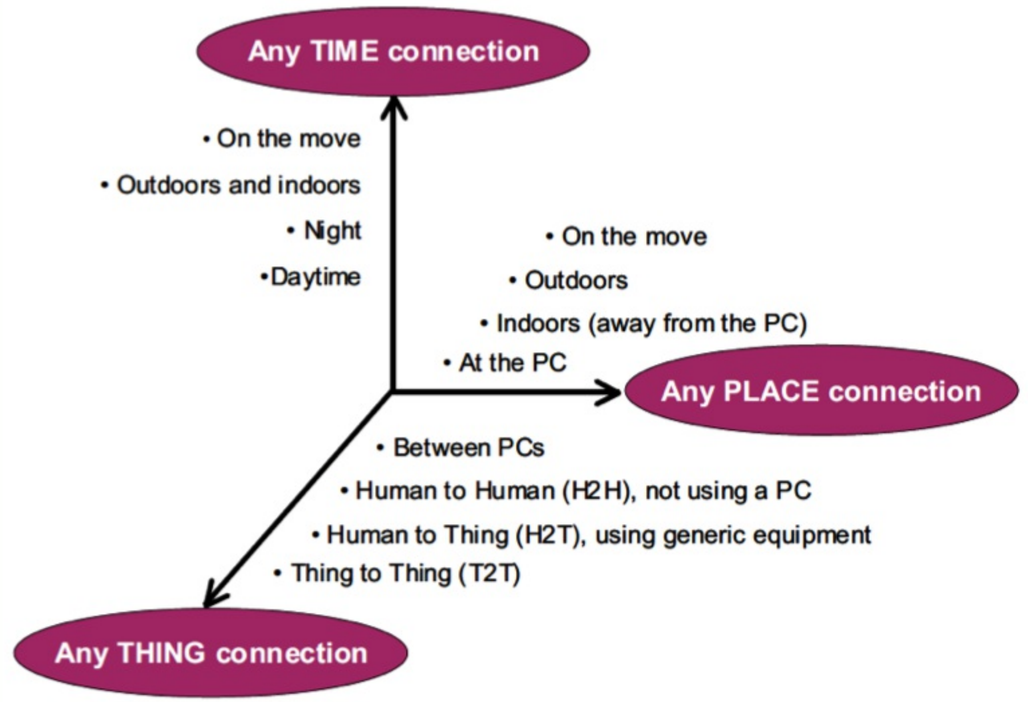
- It is a concept that has attracted much rhetoric, misconception and confusion as to what it means and its implications in a social context
- First proposed by MIT Auto-ID Center and intimately linked to RFID and electronic product code (EPC)
- *... all about physical items talking to each other...*

INTERNET OF THINGS

- Refers to a network between objects, usually it will be wireless and self-configuring, such as household appliances.
- Has come to describe a number of technologies and research disciplines that enable the Internet to reach out into the real world of physical objects.

DIMENSIONS

Figure 1 – A new dimension



Source: ITU adapted from Nomura Research Institute

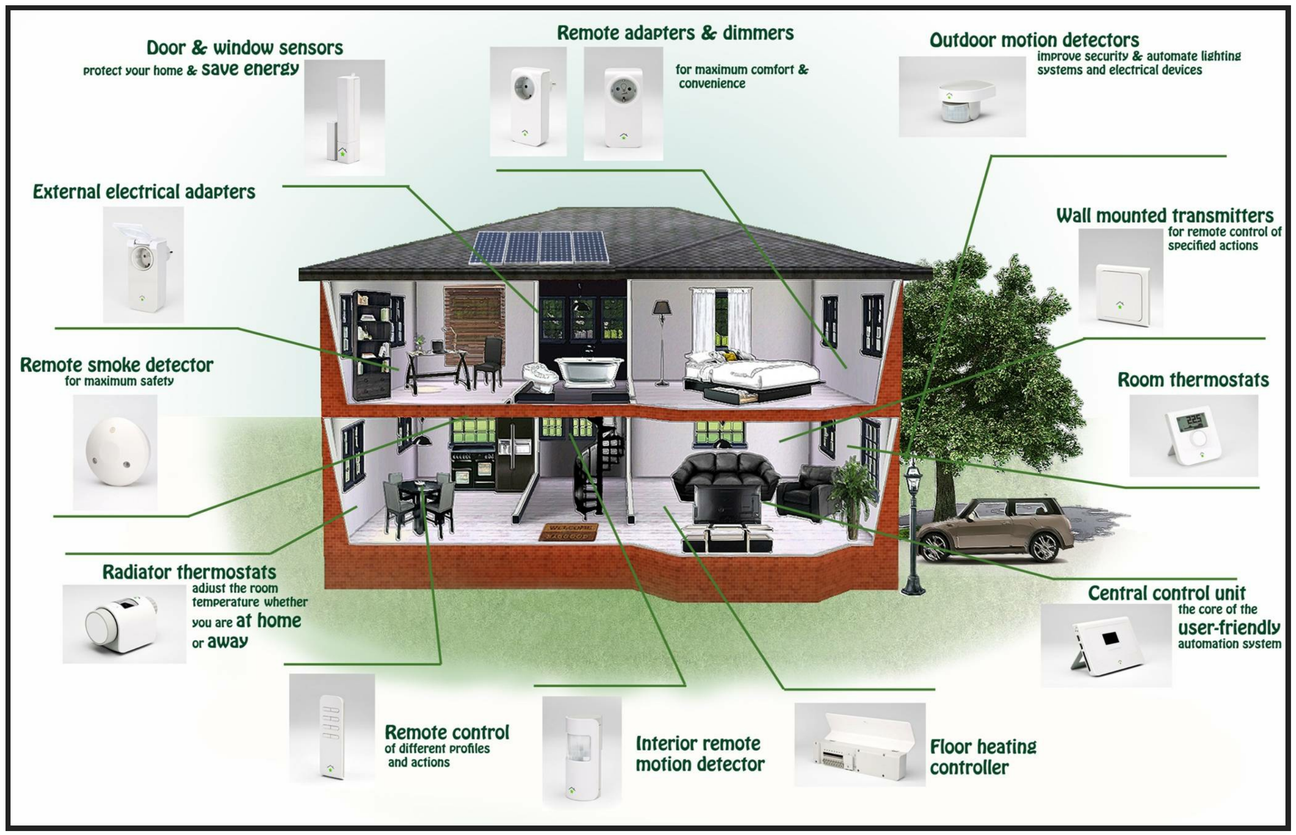
WHY?

- Dynamic control of industry and daily life
- Improve the resource utilization ratio
- Better relationship between human and nature
- Act as a technology integrator

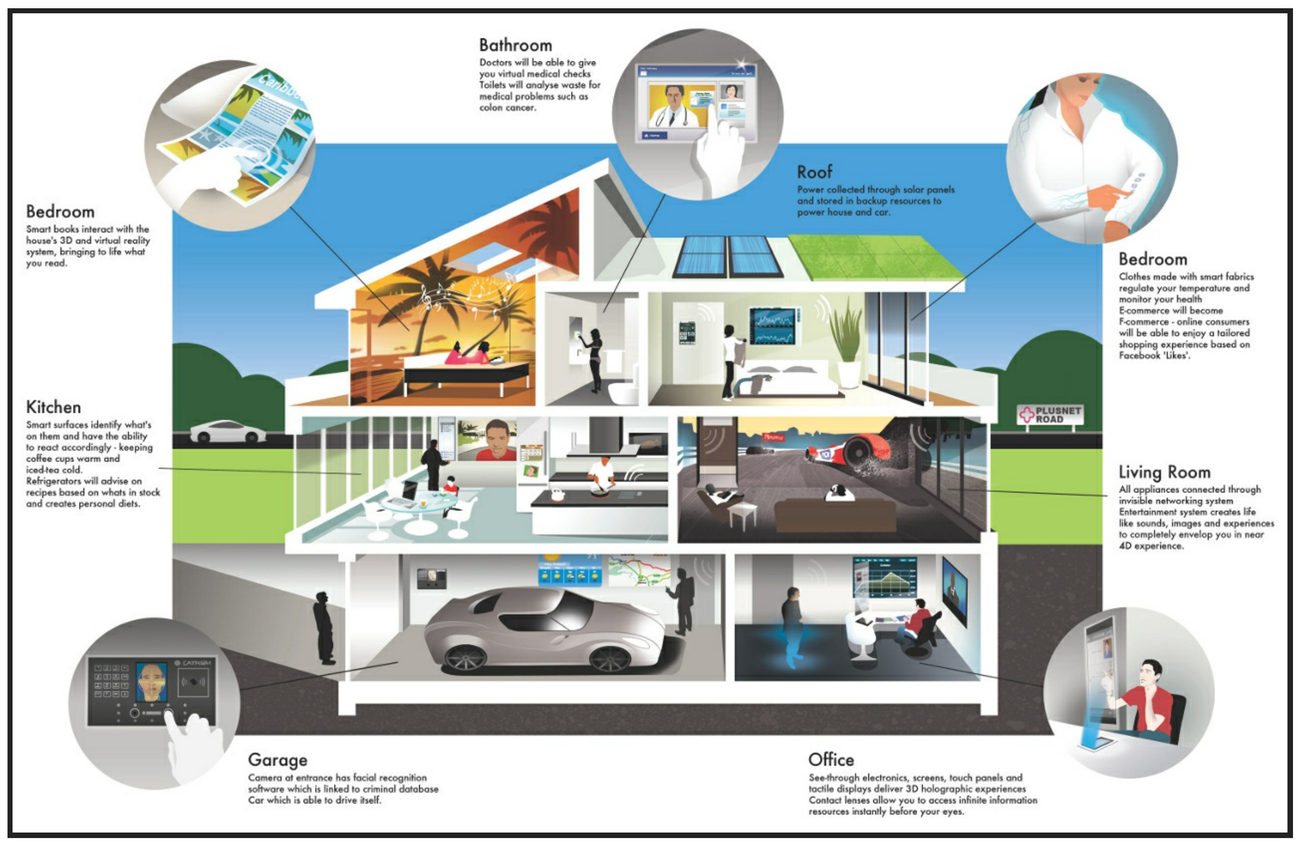
ENABLING TECHNOLOGIES

- RFID: To identify and track the data of things
- Sensors: To collect and process the data to detect the changes in the physical status of things
- Networks: To enhance the power of the network by transferring processing capabilities to different part of the network.
- Nano Tech: To make the smaller and smaller things have the ability to connect and interact.

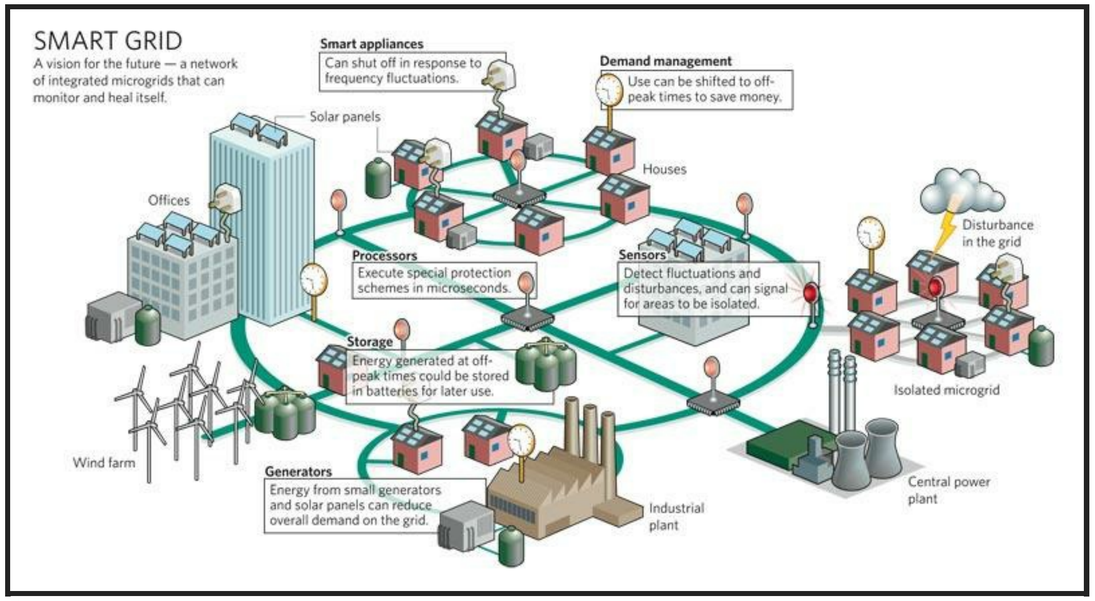
APPLICATIONS. INTELLIGENT HOME



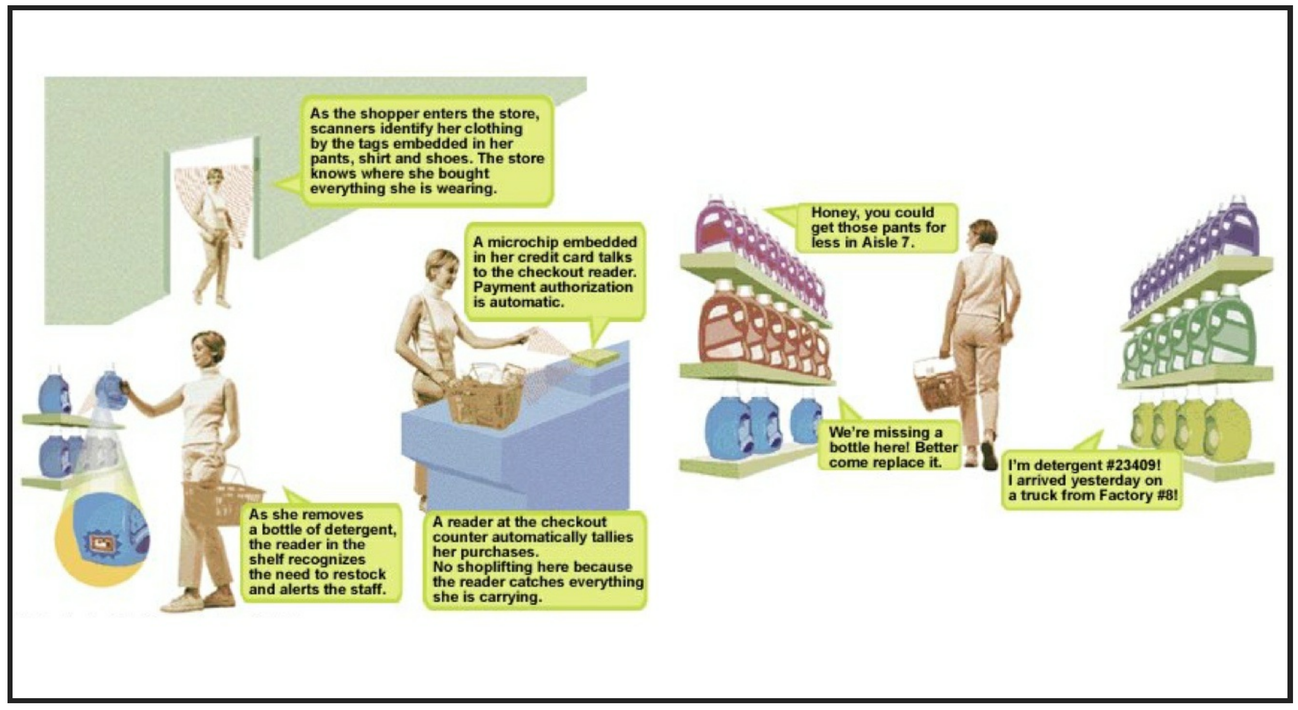
APPLICATIONS. INTELLIGENT HOME 2



APPLICATIONS. SMART GRID



APPLICATIONS. SHOPPING



APPLICATIONS. HEALTH CARE

Mobile health in **2024**

1. Contact lenses

A microscopic camera in the lens takes pictures of the **retina** and matches these to past cases, identifying early symptoms of **diabetic retinopathy**

Fact
1% of global blindness can be attributed to diabetes. Approximately 4,200 people in England are blind due to diabetic retinopathy

2. Fridge

The fridge monitors the **digestive system**: drinks consumed (thirst); vitamin consumption (**deficiencies**); calories/sugar consumption (insulin levels)

Fact
Diabetes is set to cost the NHS £16.9 billion by 2035/6

3. Artificial pancreas

Mini artificial **pancreas** to detect irregular **blood sugar** levels and injects insulin when necessary

Fact
Worldwide in 2013, 382 million people had diabetes; by 2035 this is projected to rise to 592 million

4. Clothes

Smart fibres in all clothes sense a rash or skin condition appearing, signalling the possible onset of diseases such as **skin cancer**

Fact
There are currently almost 13,000 new cases of skin cancer diagnosed each year in the UK

5. Thermometer patch

An electronic stick-on "tattoo", half the width of a human hair in size that detects precise **temperature changes** around the area of skin where it is placed, tracking **heat flow** through the bloodstream. This indicates **cardiovascular activity**

Fact
The number of people who die from cardiovascular diseases, mainly from heart disease and stroke, will increase to 23.3. million by 2030

6. Shoes and socks

Shoes and socks track movement of **feet**, detect when you are too sedentary and update you on **fitness** goals, as well as monitoring your **weight**

Fact
Physical inactivity costs the NHS £900 million annually

7. Nappies

Smart nappies monitor children's **sleeping patterns** and **body temperature** for symptoms of illness such as **dehydration**

Fact
Approximately 440,000 children around the world have diabetes with 70,000 new cases diagnosed each year

8. Toilet

The smart toilet monitors the **liver** and **kidney** by measuring the frequency and amount of urine passed, analysing for **glucose levels**, **dehydration**, **infection** and kidney problems. It also alerts for high **blood pressure**, a symptom of heart disease

Fact
Coronary Heart Disease is the UK's biggest killer with 82,000 deaths annually. Globally, more people die from cardiovascular disease than any other cause

9. Monitoring

Continuous **data collection** and instant **reporting** of fitness mean that prevention of disease can be **incentivised** with rewards for positive behaviour – the "gamification" of healthcare, driving **positive behaviour change**

Fact
Obesity could cost the NHS £9.7 billion more by 2050

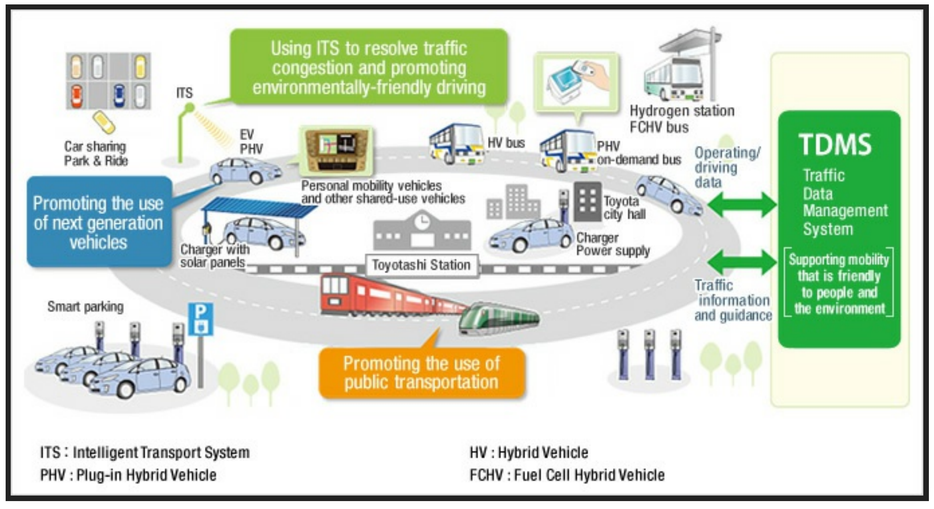
Bupa

www.bupa.com/mhealth

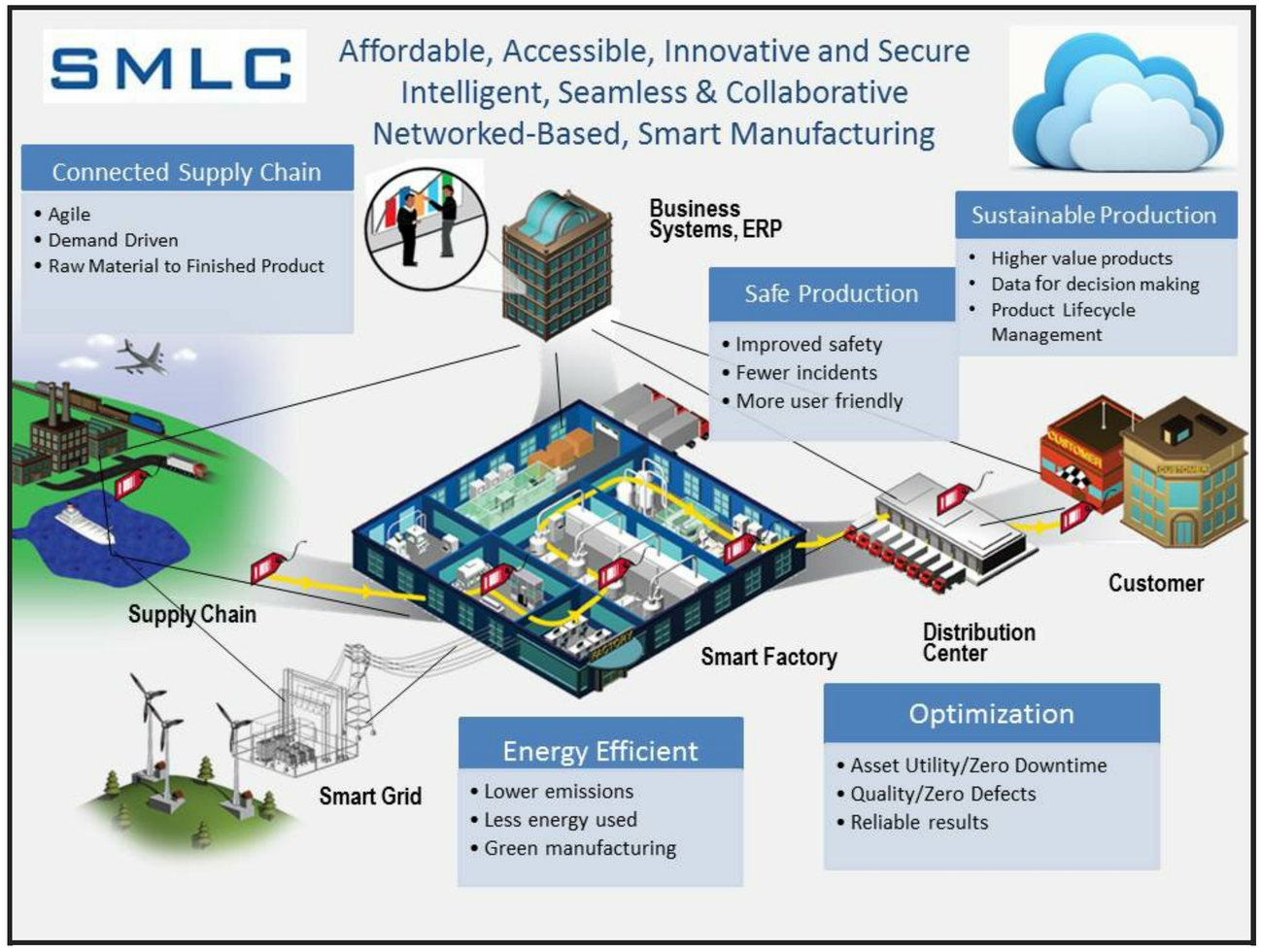
@Bupa BupaHealth Bupa

1: 'Diabetes in the UK 2012', Diabetes UK, **2:** Study in the journal 'Diabetic Medicine', Diabetes UK website, **3:** International Diabetes Federation, **4:** NHS Online, **5:** 'Global atlas on cardiovascular disease prevention and control', World Health Organization, 2011, **6:** British Heart Foundation, 'Physical Activity statistics 2012', **7:** The International Diabetes Federation, **8:** NHS online, **9:** 'Assessing the costs to the NHS associated with alcohol and obesity in Wales', Welsh Assembly Government Social Research, 2011

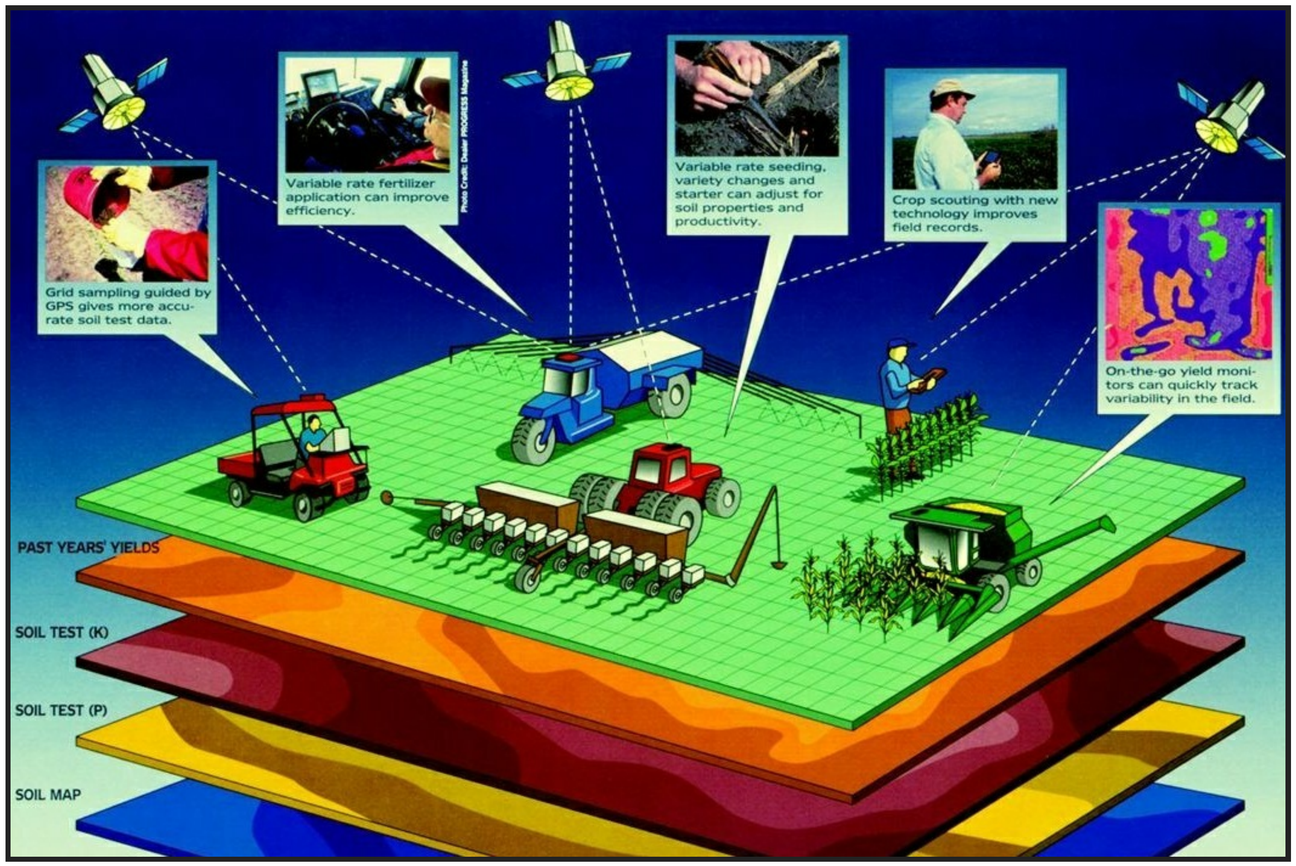
APPLICATIONS. TRANSPORTATION



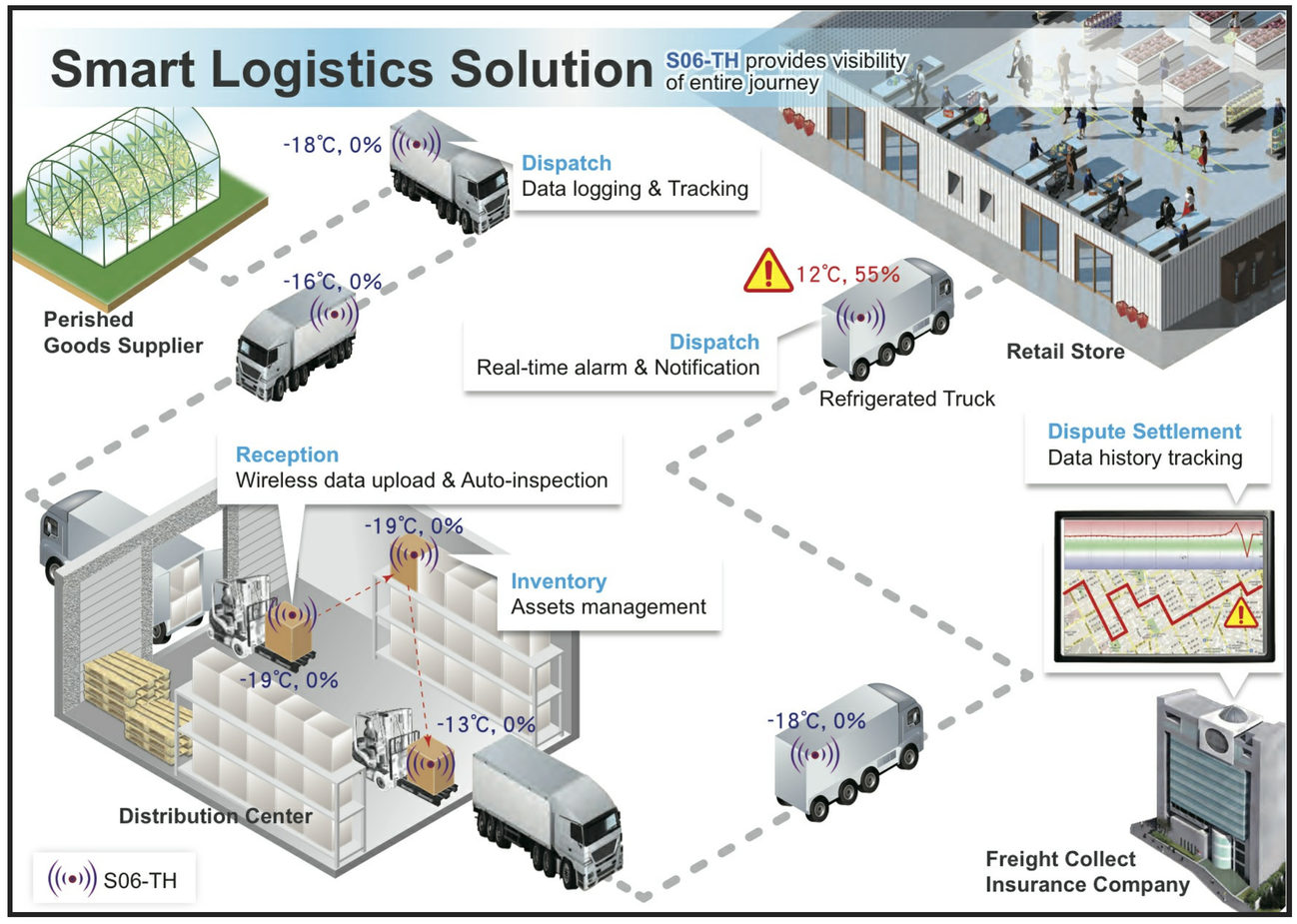
APPLICATIONS. MANUFACTURING



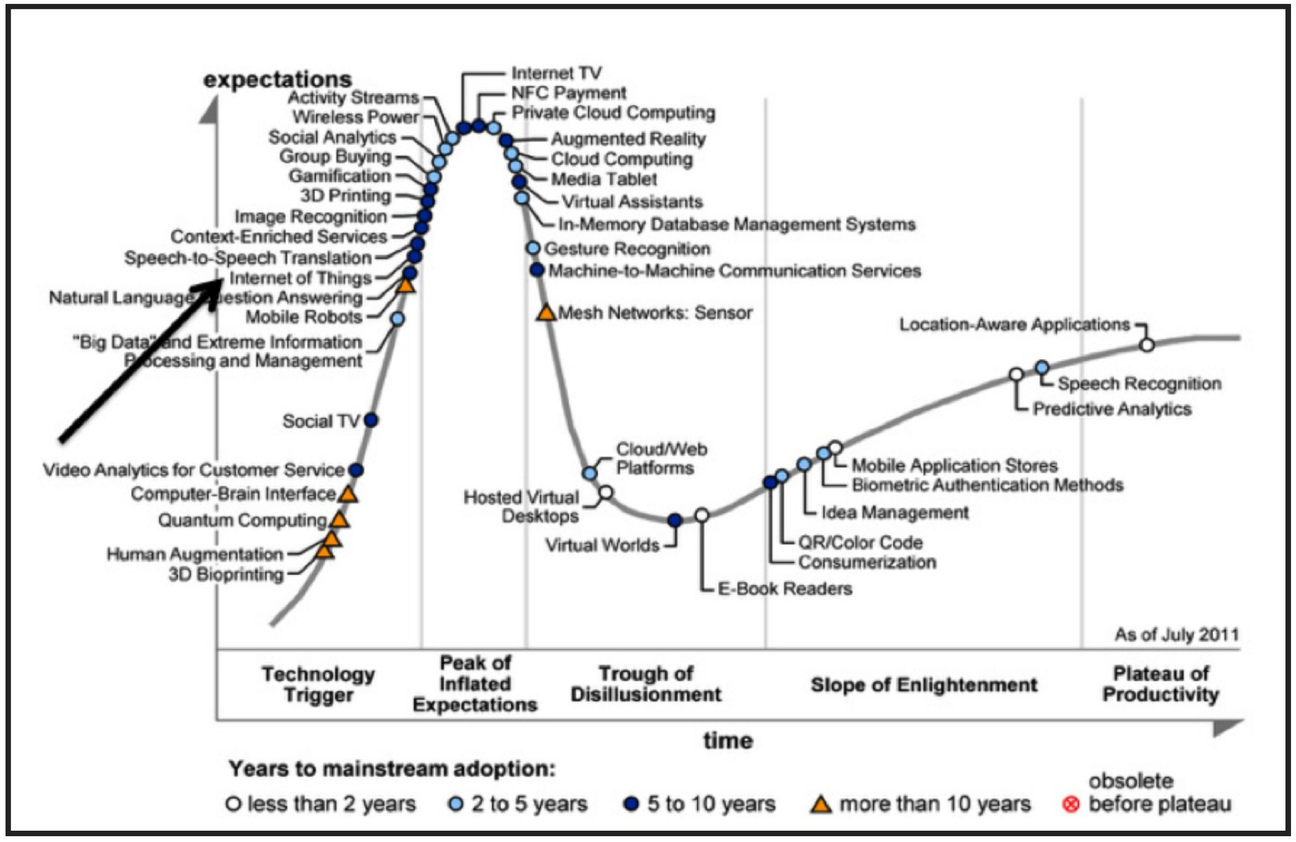
APPLICATIONS. FARMING



APPLICATIONS. LOGISTICS



STATE OF THE ART



CHALLENGES

- Technological Standardization in most areas are still remain fragmented.
- Managing and fostering rapid innovation is a challenge for governments
- Privacy and security
- Absence of governance

EXAMPLES

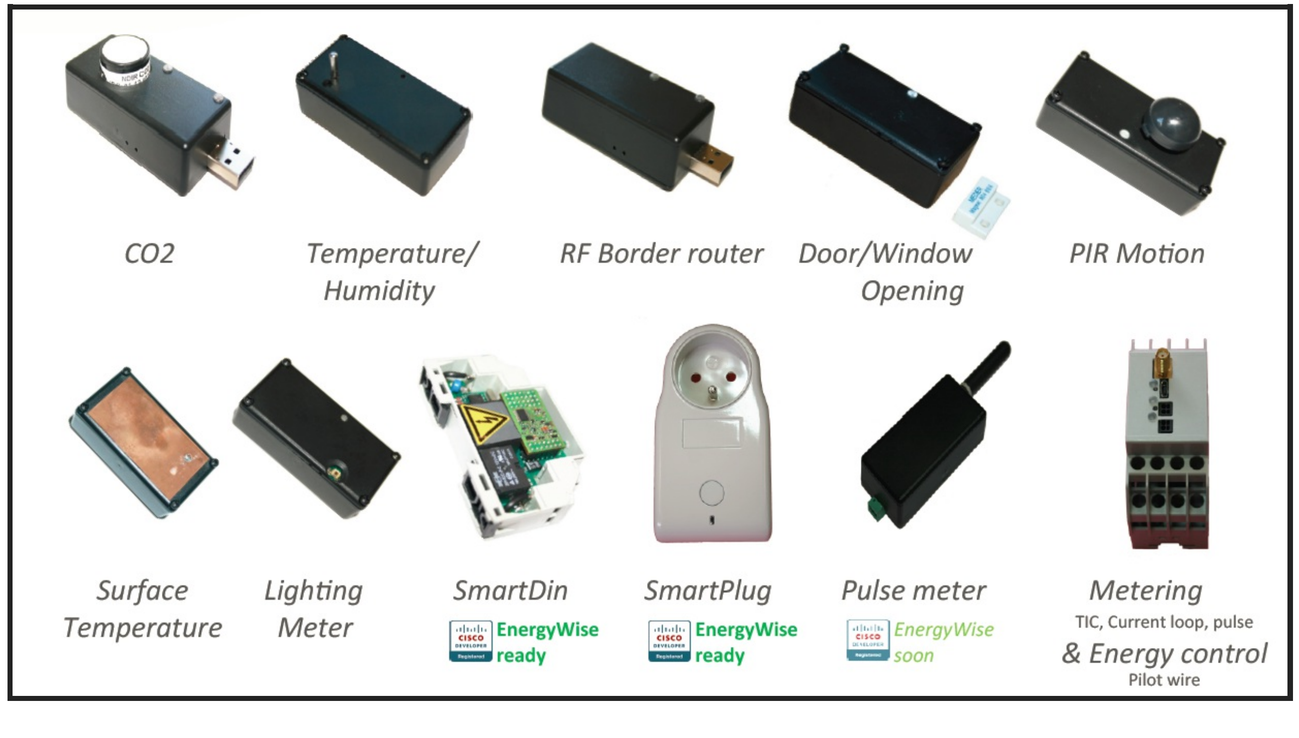
The tado° smart thermostat and intelligent heating system

EXAMPLES

The LIFX WiFi light bulb

EXAMPLES

Watteco IP Sensors



EXAMPLES

Zolertia Zoundtracker. Noise monitoring system

SOME INSPIRATION...

CONTIKI

AN OPERATING SYSTEM

FOR THE INTERNET OF THINGS

INSTALLATION

- Download Instant Contiki
- Download Virtual Box
- Follow this instructions

SETUP SUGGESTIONS

- Spanish keyboard
  - System Settings > Keyboard Layout > Layouts > Spanish
- Install your favourite editor

```
apt get install vim
```

```
apt get install eclipse
```

```
sudo add-apt-repository ppa:webupd8team/sublime-text-3
sudo apt-get update
sudo apt-get install sublime-text-installer
```

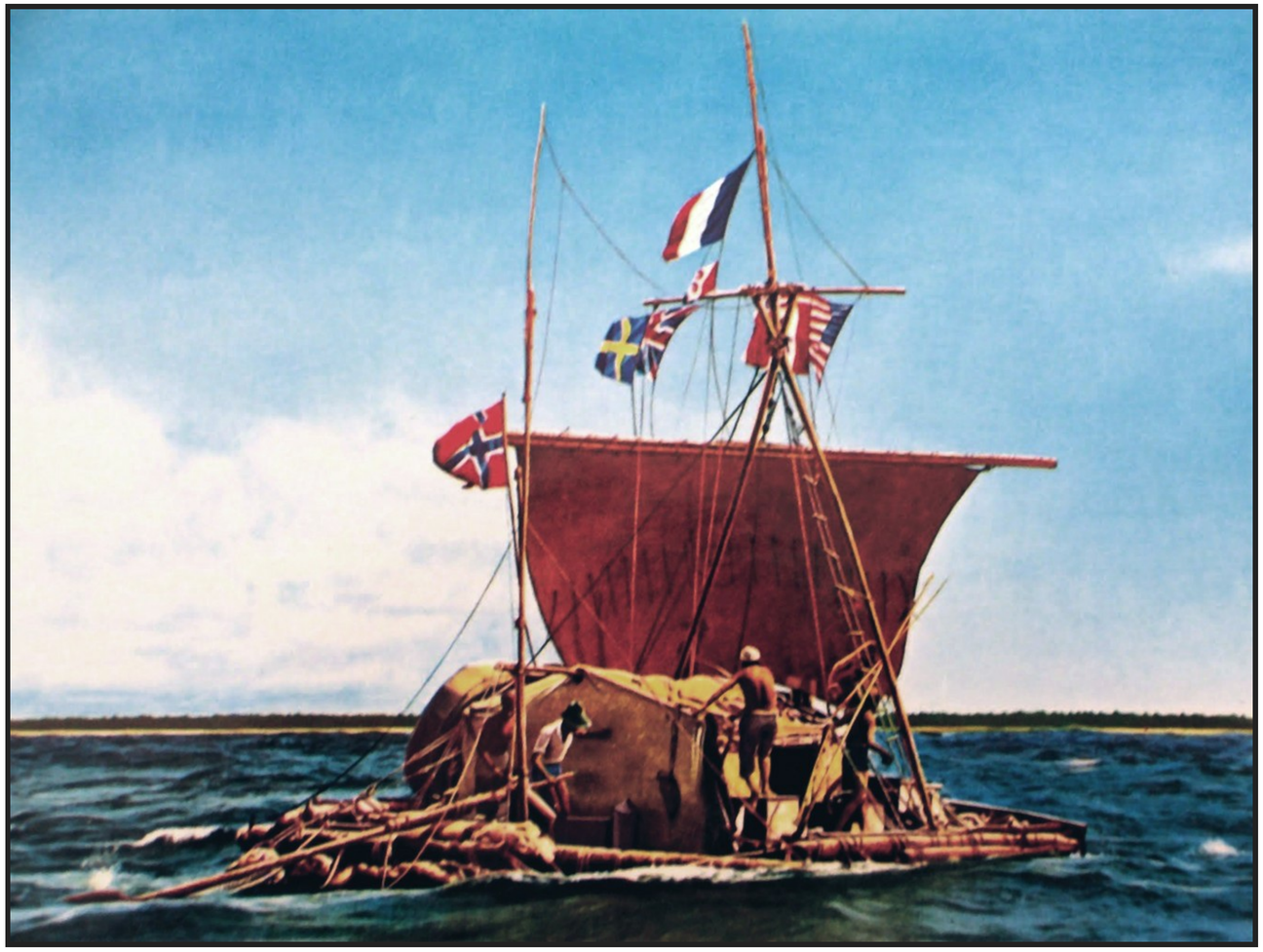
- Add include directories to Eclipse
  - /usr/local/avr/include
  - /home/user/CodeSourcery/SourceryG++Lite/arm-none-eabi/include

CONTIKI

- Contiki – pioneering open source operating system for sensor networks
  - IP networking
  - Hybrid threading model, protothreads
  - Dynamic loading
  - Power profiling
  - Network shell
- Small memory footprint
- Designed for portability
  - 14 platforms, 5 CPUs

NAME

The Kon-Tiki raft sailed across the Pacific Ocean with minimal resources



TARGETS

- Small embedded processors with networking
  - Sensor networks, **smart objects** ...
- 98% of all microprocessors go into embedded systems
- 50% of all processors are 8-bit
  - MSP430, AVR, ARM7, 6502, ...

BACKGROUND: UIP

- uIP – micro IP
- Open source
- ~5k code, ~2k RAM
  - Smallest configuration ~3k code, ~128 bytes RAM
- RFC compliant
  - Order of magnitude smaller than previous stacks
- Bottom-up design
  - Single-packet buffer
  - Event-driven API

UIP: API

- Two APIs
  - The “raw” uIP event-driven API
  - Protosockets, sockets-like programming based on protothreads
- Event-driven API works well for small programs
  - Explicit state machines
- Protosockets work better for larger programs
  - Sequential code

UIP EXAMPLE

```
int smtp_protothread(struct psock *s)
{
    PSOCK_BEGIN(s);
    PSOCK_READTO(s, '\n');
    if(strncmp(inputbuffer, “220”, 3) != 0) {
    PSOCK_CLOSE(s);
    PSOCK_EXIT(s);
}

PSOCK_SEND(s, "HELO ", 5);
PSOCK_SEND(s, hostname, strlen(hostname));
PSOCK_SEND(s, "\r\n", 2);
PSOCK_READTO(s, '\n');
if(inputbuffer[0] != '2') {
    PSOCK_CLOSE(s);
    PSOCK_EXIT(s);
```

PROCESSES

- The Contiki kernel is event-based
  - Invokes processes whenever something happens
    - Sensor events
    - Processes starting, exiting
- Process invocations must not block
- Protothreads provide sequential flow of control in Contiki processes

More info

HELLO WORLD

```
/* Declare the process */
PROCESS(hello_world_process, "Hello world");
/* Make the process start when the module is loaded */
AUTOSTART_PROCESSES(&hello_world);

/* Define the process code */
PROCESS_THREAD(hello_world_process, ev, data) {
    PROCESS_BEGIN(); /* Must always come first */

    printf("Hello, world!\n"); /* Initialization code goes here */
    while(1) { /* Loop for ever */
        PROCESS_WAIT_EVENT(); /* Wait for something to happen */
    }

    PROCESS_END(); /* Must always come last */
}
```

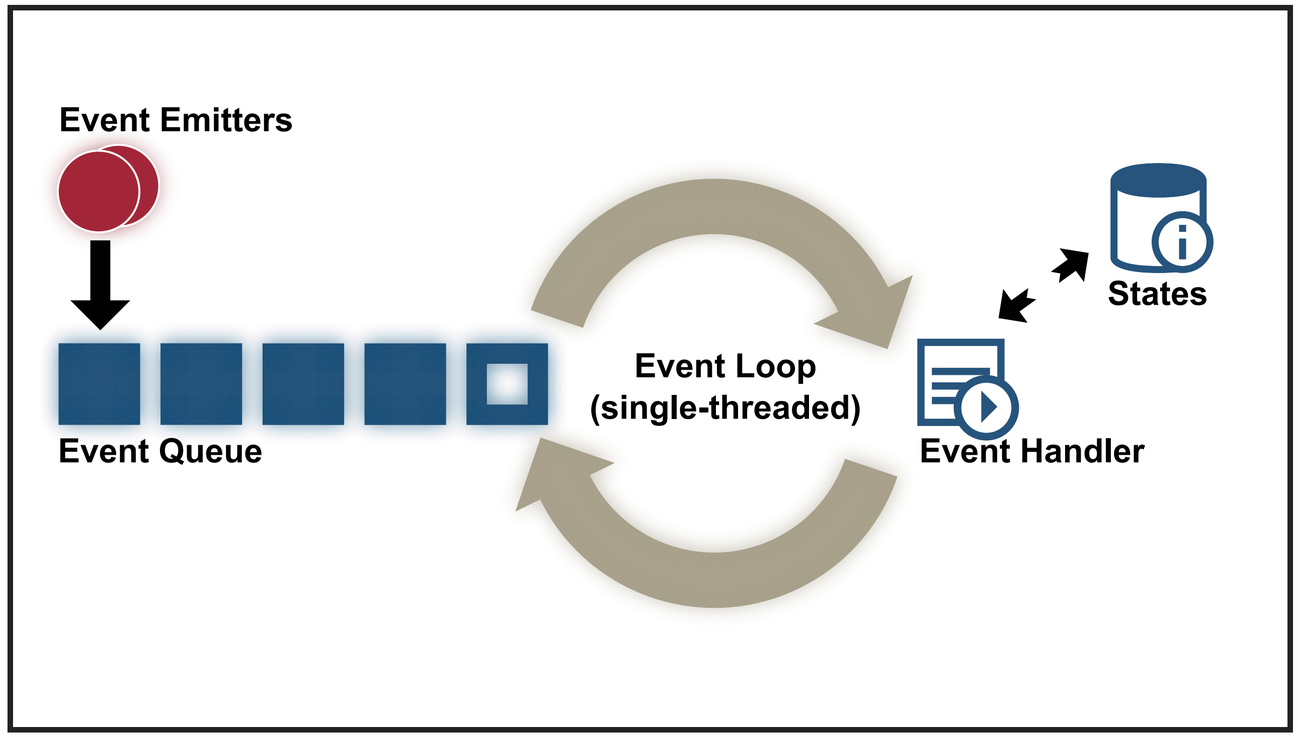
PROCESSES API

```
PROCESS_BEGIN();              // Declares the beginning of a process' pro
PROCESS_END();                // Declares the end of a process' protothre
PROCESS_EXIT();               // Exit the process.
PROCESS_WAIT_EVENT();         // Wait for any event.
PROCESS_WAIT_EVENT_UNTIL(); // Wait for an event, but with a condition.
PROCESS_YIELD();              // Wait for any event, equivalent to PROCES
PROCESS_WAIT_UNTIL();         // Wait for a given condition; may not yiel
PROCESS_PAUSE();              // Temporarily yield the process.
```

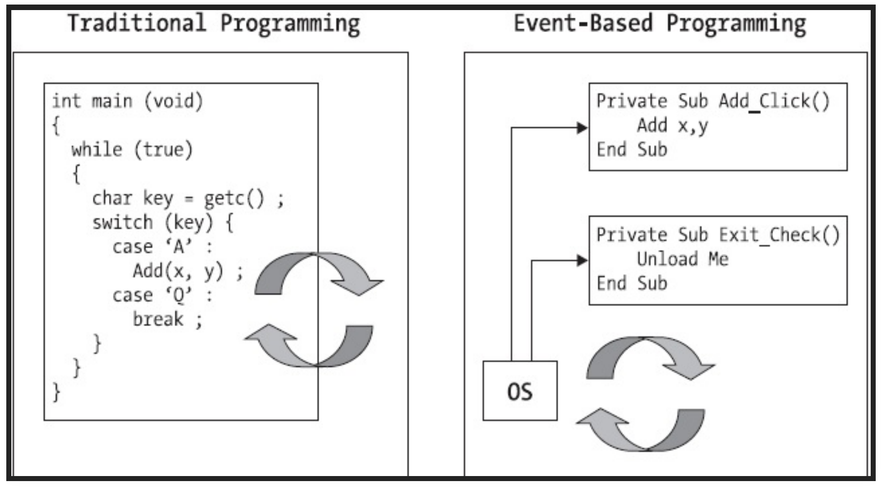
SENDING EVENTS TO PROCESSES

```
static char msg[] = "Data";

 static void
 example_function(void)
 {
   /* Start "Example process", and send it a NULL
      pointer. */

   process_start(&example_process, NULL);

   /* Send the PROCESS_EVENT_MSG event synchronously to
      "Example process", with a pointer to the message in the
      array 'msg'. */
   process_post_synch(&example_process,
                      PROCESS_EVENT_CONTINUE, msg);
   /* Send the PROCESS_EVENT_MSG event asynchronously to
```

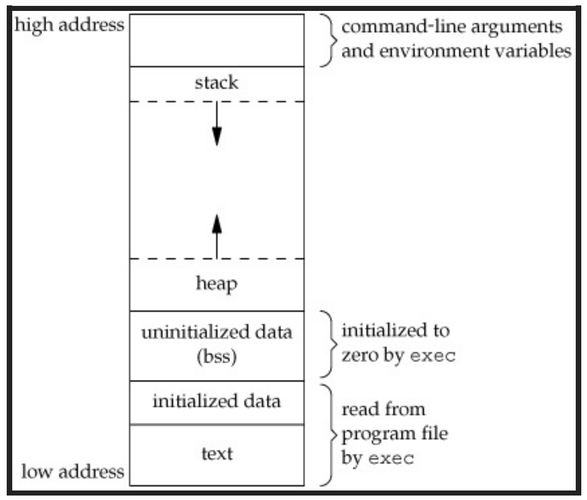
EVENT DRIVEN

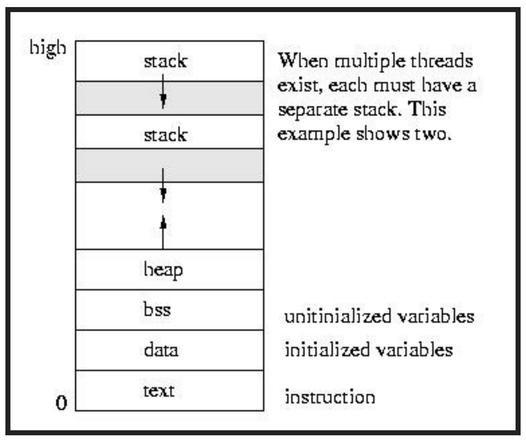


EVENT DRIVEN



EVENT-DRIVEN VS MULTITHREADED





- Multithreading requires per-thread stacks
- Event-driven requires less memory

WHY NOT TO USE THREADS

- Threads require stack memory
  - Unused stack space wastes memory
  - 200 bytes out of 2048 bytes is a lot!
- A multi-threading library quite difficult to port
  - Requires use of assembly language
  - Hardware specific
  - Platform specific
  - Compiler specific

PROTOTHREADS:

A NEW PROGRAMMING ABSTRACTION

- A design point between events and threads
- Programming primitive: conditional blocking wait

```
PT_WAIT_UNTIL(condition)
```

- Single stack
  - Low memory usage, just like events
- Sequential flow of control
  - No explicit state machine, just like threads
  - Programming language helps us: if and while

PROTOTHREAD EXAMPLE

```
int a_protothread(struct pt *pt) {
    PT_BEGIN(pt);
    ...
    PT_WAIT_UNTIL(pt, condition1);
    ...
    if(something) {
        PT_WAIT_UNTIL(pt, condition2);
        ...
    }
    ...
    PT_END(pt);
}
```

PROTOTHREAD LIMITATIONS

- Automatic variables not stored across a blocking wait

```
void f(struct pt *pt) {
    int a=1;
    ...
    PT_WAIT_UNTIL(pt, condition);
    printf("%d\n",a);
    ...
}
```

  - Compiler does produce a warning
  - Workaround: use static local variables instead

```
void f() {
  static int a=1;
  ...
  PT_WAIT_UNTIL(pt, condition);
  printf("%d\n",a);
  ...
}
```

- Constraints on the use of switch() constructs in programs
  - No warning produced by the compiler
  - Workaround: don’t use switches

TIMERS

- struct timer
  - Passive timer, only keeps track of its expiration time
- struct etimer
  - Active timer, sends an event when it expires
- struct ctimer
  - Active timer, calls a function when it expires
  - Used by Rime
- struct rtimer
  - Real-time timer, calls a function at an exact time
  - Be careful ***only 1 rtimer per mote will work!***

More info

And still more

ETIMER

```
PROCESS_THREAD(hello_world_process, ev, data) {
    static struct etimer et; /* Must be static */
    PROCESS_BEGIN();    /* since processes are */
                        /* protothreads */
    while(1) {
        /* Using a struct timer would not work, since
        the process would not be invoked when it expires. */
        etimer_set(&et, CLOCK_SECOND);

        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));

        /* One second has passed :) */
    }

    PROCESS_END();
}
```

NETWORKING

RIME vs IP

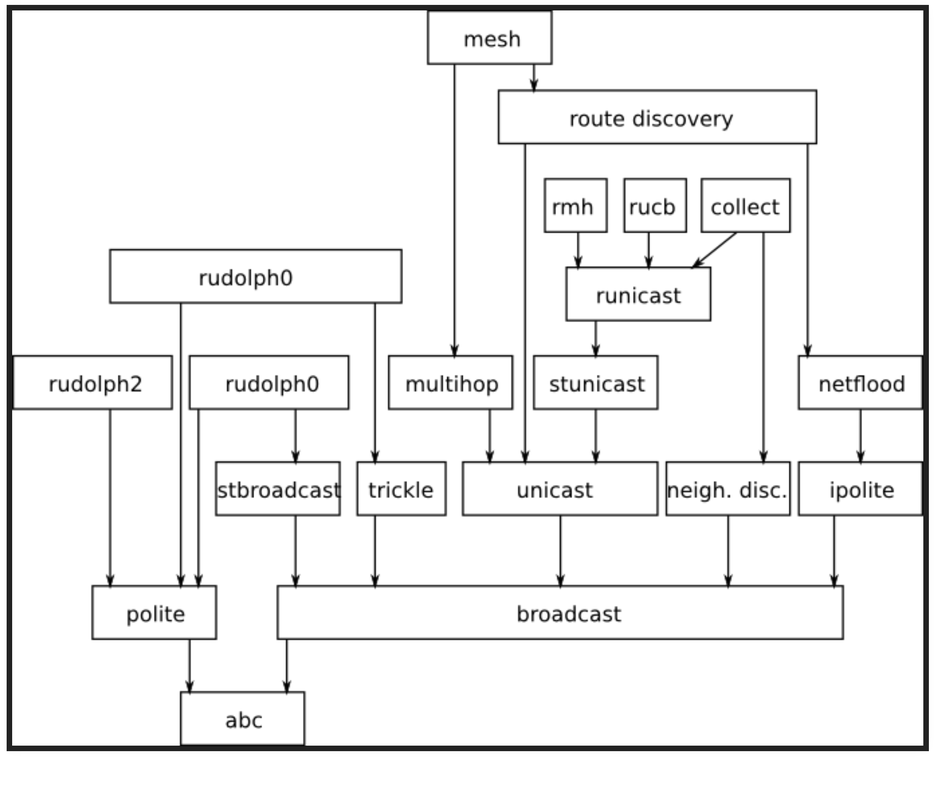
RIME

- A set of communication abstractions (in increasing complexity):
  - Single-hop broadcast (broadcast)
  - Single-hop unicast (unicast)
  - Reliable single-hop unicast (runicast)
  - Best-effort multi-hop unicast (multihop)
  - Hop-by-hop reliable multi-hop unicast (rmh)
  - Best-effort multi-hop flooding (netflood)
  - Reliable multi-hop flooding (trickle)

RIME

- A set of communication abstractions (continued)
  - Hop-by-hop reliable data collection tree routing (collect)
  - Hop-by-hop reliable mesh routing (mesh)
  - Best-effort route discovery (route-disovery)
  - Single-hop reliable bulk transfer (rudolph0)
  - Multi-hop reliable bulk transfer (rudolph1)

RIME


mesh
route discovery
rmh
rucb
collect
rudolph0
runicast
rudolph2
rudolph0
multihop
stunicast
netflood
stbroadcast
trickle
unicast
neigh. disc.
ipolite
polite
broadcast
abc

RIME: DETAILED PRIMITIVES

- abc: the anonymous broadcast, it justs sends a packet via the radio driver, receives all packet from the radio driver and passes them to the upper layer;
- broadcast: the identified broadcast, it adds the sender address to the outgoing packet and passes it to the abc module;
- unicast: this module adds a destination address to the packets passed to the broadcast block. On the receive side, if the packet's destination address doesn't match the node's address, the packet is discarded;

RIME: DETAILED PRIMITIVES

- stunicast: the stubborn unicast, when asked to send a packet to a node, it sends it repeatedly with a given time period until asked to stop. This module is usually not used as is, but is used by the next one;
- runicast: the reliable unicast, it sends a packet using the stunicast module waiting for an acknowledgment packet. When it is received it stops the continuous transmission of the packet. A maximum retransmission number must be specified, in order to avoid infinite sending;

RIME: DETAILED PRIMITIVES

- polite and ipolite: these two modules are almost identical, when a packet has to be sent in a given time frame, the module waits for half of the time, checking if it has received the same packet it's about to send. If it has, the packet is not sent, otherwise it sends the packet. This is useful for flooding techniques to avoid unnecessary retransmissions;
- multihop: this module requires a route table function, and when about to send a packet it asks the route table for the next hop and sends the packet to it using unicast. When it receives a packet, if the node is the destination then the packet is passed to the upper layer, otherwise it asks again the route table for the next hop and relays the packet to it;

RIME CHANNELS

- All communication in Rime is identified by a 16-bit channel
  - Communicating nodes must agree on what modules to use on a certain channel
    - Unicast on channel 155
    - Netflood on channel 130
  - Channel numbers lower than 128 are reserved by the system. Used by the shell, other system apps

EXAMPLE 1. SEND TO NEIGHBOURS

```
void recv(struct broadcast_conn *c) {
    printf("Message received %s\n", (char*)packetbuf_dataptr());
}
```

```
struct broadcast_callbacks cb = {recv}; /* Callback */
struct broadcast_conn c; /* Connection */
```

```
void setup_sending_a_message_to_all_neighbors(void) {
    broadcast_open(&c, 128, &cb); /* Channel 128 */
}
```

```
void send_message_to_neighbours(char* msg, int len) {
    packetbuf_copyfrom(msg, len); /* Copy data to buffer */
    broadcast_send(&c);
}
```

EXAMPLE 2. SEND MESSAGE TO ENTIRE NETWORK

```
void recv(struct trickle_conn *c) { /* callback */
    printf("Message received, length = %d\n", databuf_datalen());
}
```

```
struct trickle_callbacks cb = {recv}; /* Callbacks */
struct trickle_conn c; /* Connection */
```

```
void setup_sending_a_message_to_network(void) {
    trickle_open(&c, CLOCK_SECOND, 129, &cb); /* Channel 129 */
}
```

```
void send_message_to_network(char *msg, int len) {
    packetbuf_copyfrom(msg, len); /* Setup rimebuf */
    trickle_send(&c);
}
```

ADDITIONAL INFORMATION I

Good tutorials

Building the Internet of Things

Day 1 part 1

Day 1 part 2

Day 1 part 3

Day 2 part 1

Day 2 part 2

Day 2 part 3

ADDITIONAL INFORMATION II

Advanced IoT Firmware Engineering

Day 1 part 1

Day 1 part 2

Day 1 part 3

Other stuff

6lowpan Book Exercises

libellium smart world

TABLE EXAMPLE

Tables	Are	Cool
col 3 is	right-aligned	$1600
col 2 is	centered	$12
zebra stripes	are neat	$1