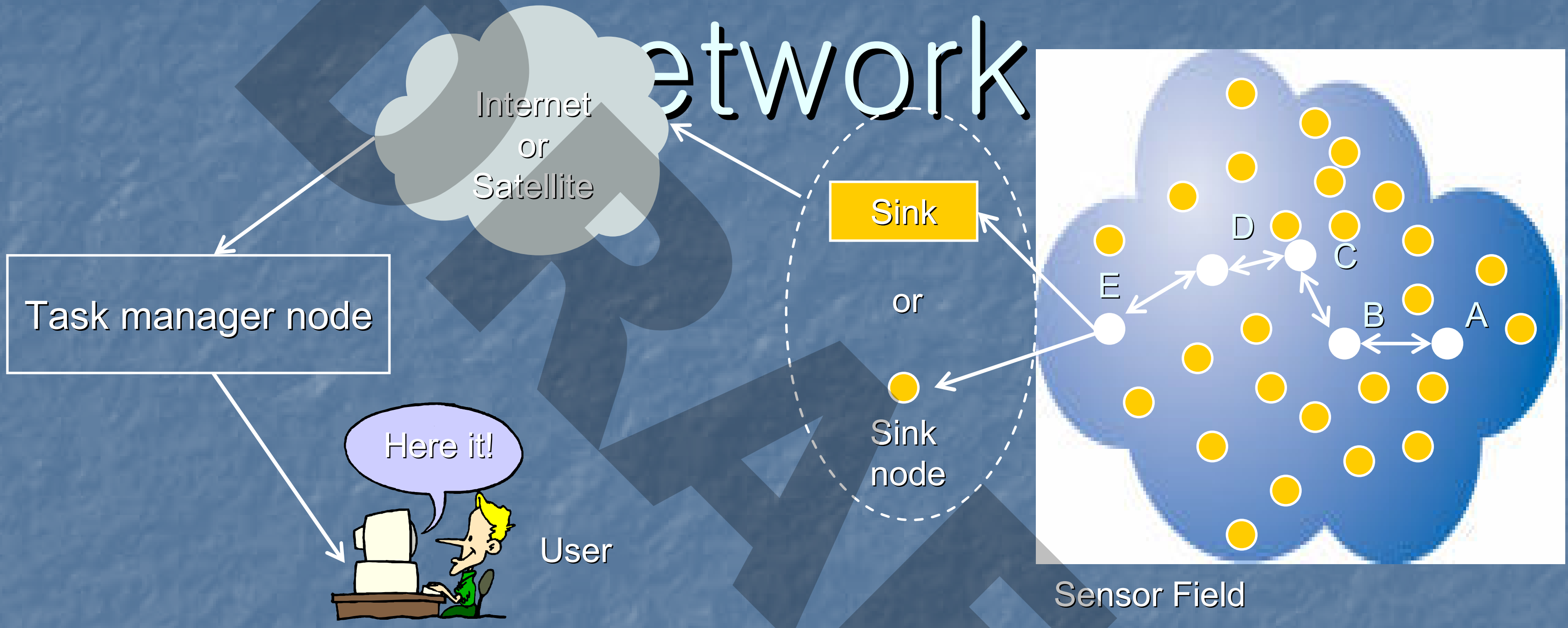
Sensor network and disaster prevention(1)

ChangDuk Jung
Prof. of Korea University
E-mail:jcd1234@korea.ac.kr

Ubiquitous research

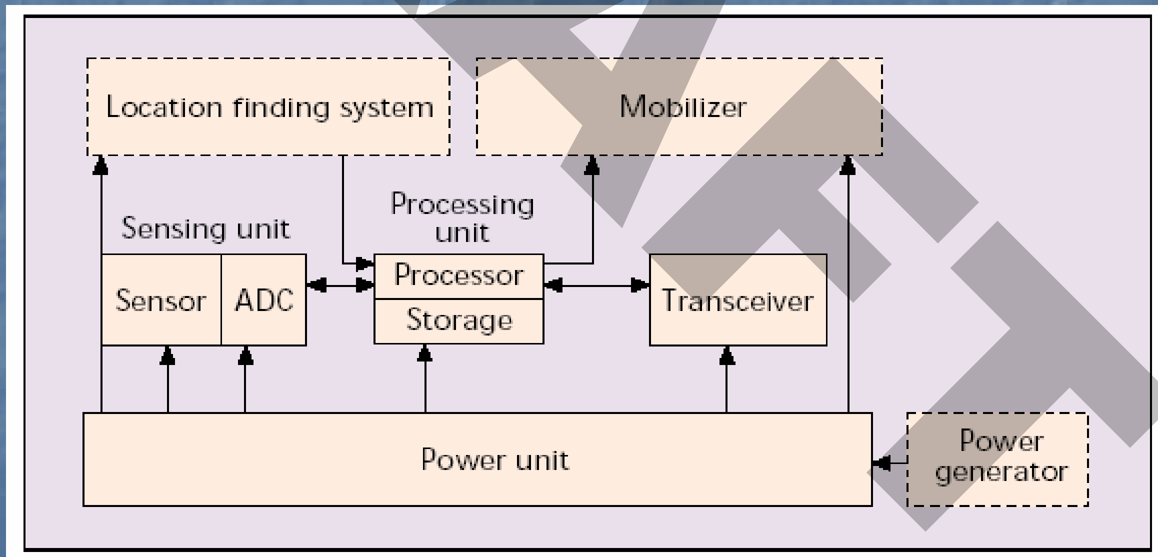
Ubiquitous sensor network



- Role of sensor
  - collect data
  - route data back to the sink by a multi-hop infrastructure-less architecture

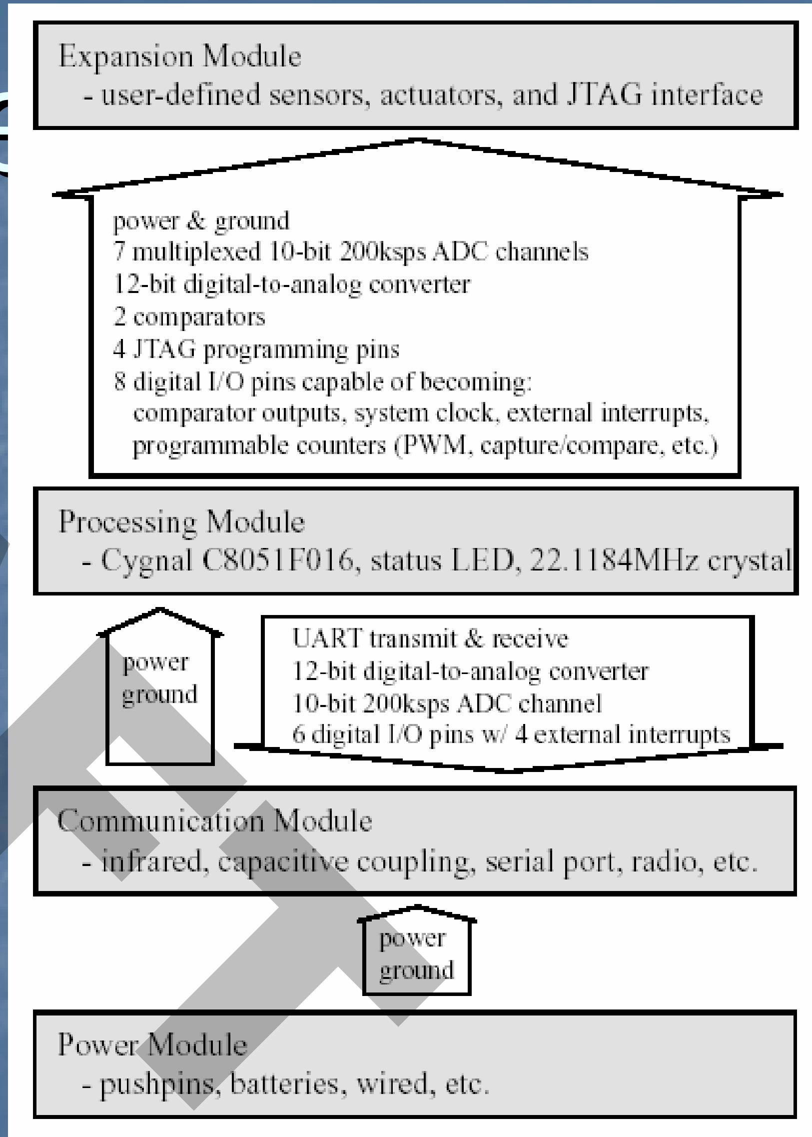
Ubiquitous sensor network

- Hardware constraints
  - 4 elements
    - sensing unit, processing unit, transceiver unit, power unit
  - component
    - location finding system, power generator, mobilizer



sensor ne

- Hardware stack of PushPin
  - power module
    - pushpins
    - batteries
    - wired
  - communication module
    - IR
    - serial port
    - radio
  - processing module
  - expansion module
    - user-define sensors
    - actuators
    - JTAG interface

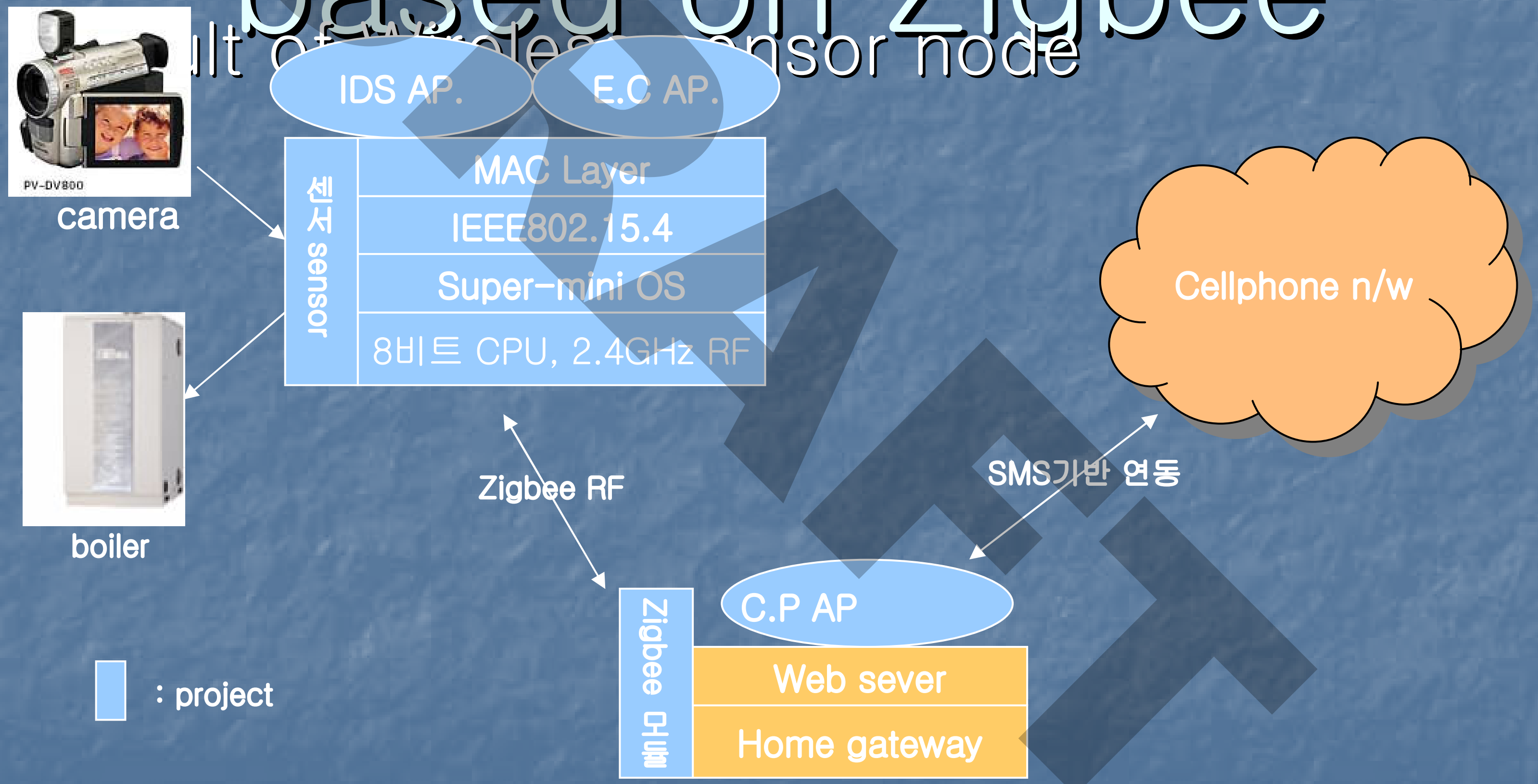


Zigbee Alliance

- Stack requirements
  - 8 bit micro processor e.g. 8051
  - Full protocol stack < 32K
  - Simple node only stack ~ 4K
  - Coordinator requires extra RAM
    - Node device database
    - Transaction table
    - Pairing table

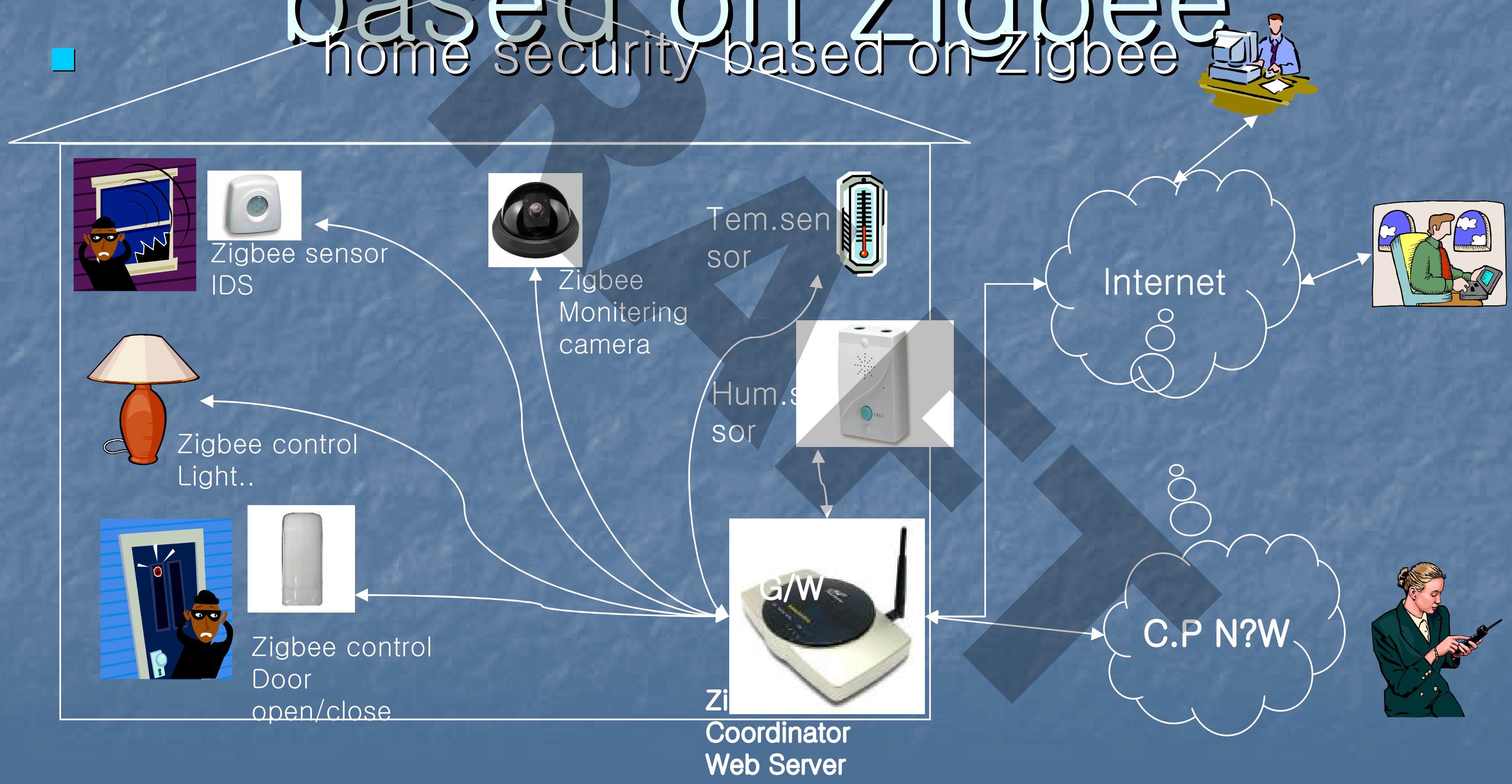
Project :Development of equipment and sensor based on Zigbee

- Result of Wireless sensor node



Project :Development of equipment and sensor based on Zigbee

- home security based on Zigbee



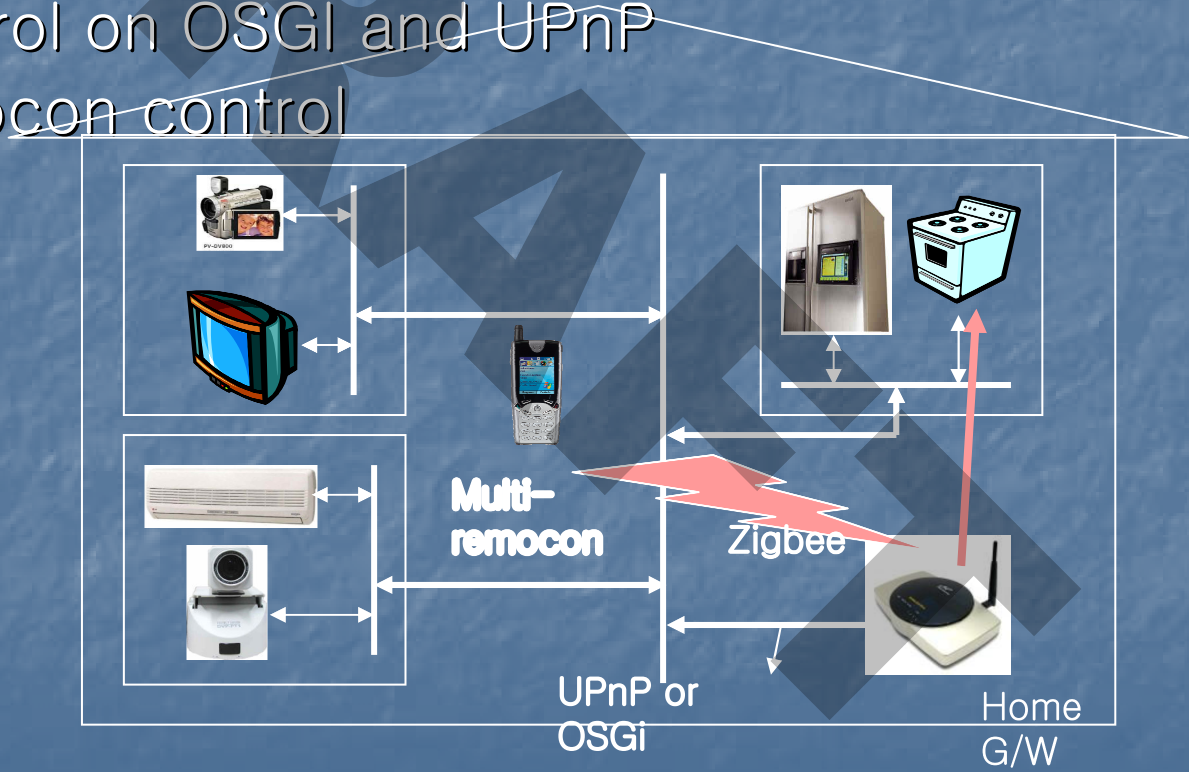
Project :Development of equipment and sensor based on Zigbee

- Result of ubiquitous sensor network
  - Multi -remocon
    - User interface design

Multi-remotech

Project :Development of equipment and sensor based on Zigbee

- control on OSGI and UPnP
- Remocon control

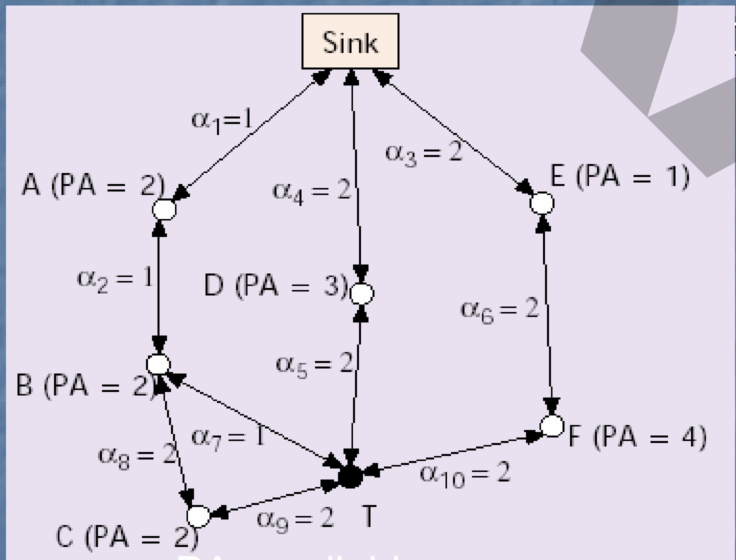


Ad-Hoc Routing

- Hierarchical Routing Protocols
  - When the network size increases
    - “flat” routing schemes become infeasible
  - based on
    - organizing nodes in groups
    - assigning nodes different functionalities inside a nd outside a group
- 4 protocols
  - CGSR (Clusterhead-Gateway Switch Routing)
  - HSR (Hierarchical State Routing)
  - ZRP (Zone Routing Protocol)
  - LANMAR (Landmark Ad Hoc Routing Protocol)

Ad-Hoc Routing

- Power Awareness Routing
  - power efficiency is still an important consideration
  - an ideal SN has to be based on an attribute-based …tion awareness



PA=available power

α = energy required

- energy-efficient route
  - route1: Sink-A-B-T, total PA=4, total α=3
  - route2: Sink-A-B-C-T, total PA=6, total α=6
  - route3: Sink-D-T, total PA=3, total α=4
  - route4: Sink-E-F-T, total PA=5, total α=6
- maximum PA route : route 4
- minimum energy route : route 1
- minimum hop route : route 3
- maximum minimum PA node route : route 3

Sensorware

- A user sends a query to the SN
  - the query is a script, a state machine in its simplest form, which is injected to one or more sensor nodes
  - the script describe how to populate (replicate or migrate) itself to other nodes
    - the process of population can continue depending on events and the current state
  - the populated scripts will collaborate among themselves in order to extract the information needed by the user, and eventually send back to the user
- So, it allows user interaction that goes

Sensorware

- Again, in the proactive distributed model, the scripts will look mostly like state machines that are influenced by external events

```
# These are data that the script carries with it. The particular ones inform
#the current instance about its parent node (the one which sent the script)
set send_node [Agent_memory_read 0];
set send_node_neighbors [Agent_memory_read 1];
# update these data for use in the next replication
Agent_memory_write 0 [getNodeID];
Agent_memory_write 1 [getNodeNeighbors];
#based on the above info find nodes that probably do not have the script
#and store their list in the variable $remaining nodes (commands omitted)
# set a timer called RT with the initial value 200ms
setTimer RT 200
# the big loop starts
while {1} {
      set ans [wait -msg * -data * -until RT]
#wait cmd returned, find out type and body of event.
      set type [lindex $ans 0];   set body [lrange $ans 1 end]
      switch $type {
              w {   # a timer expired, do something.
                    setTimer TT  100
              }
              s {   # data was sensed.
#wait for sensing data threshold to be passed, within 5 ms
                    set ans [wait -data -threshold 10 -until 5]
                    set type [lindex $ans 0];
                    if { $type = "s" } { set ready 1 }
              }
              n {   # a network message was received.
                     If { $ready}  {Agent_replicate $remaining_nodes; exit; }
              }
}
```

IEEE1451

- Distributed Measurement and Control(DMC)

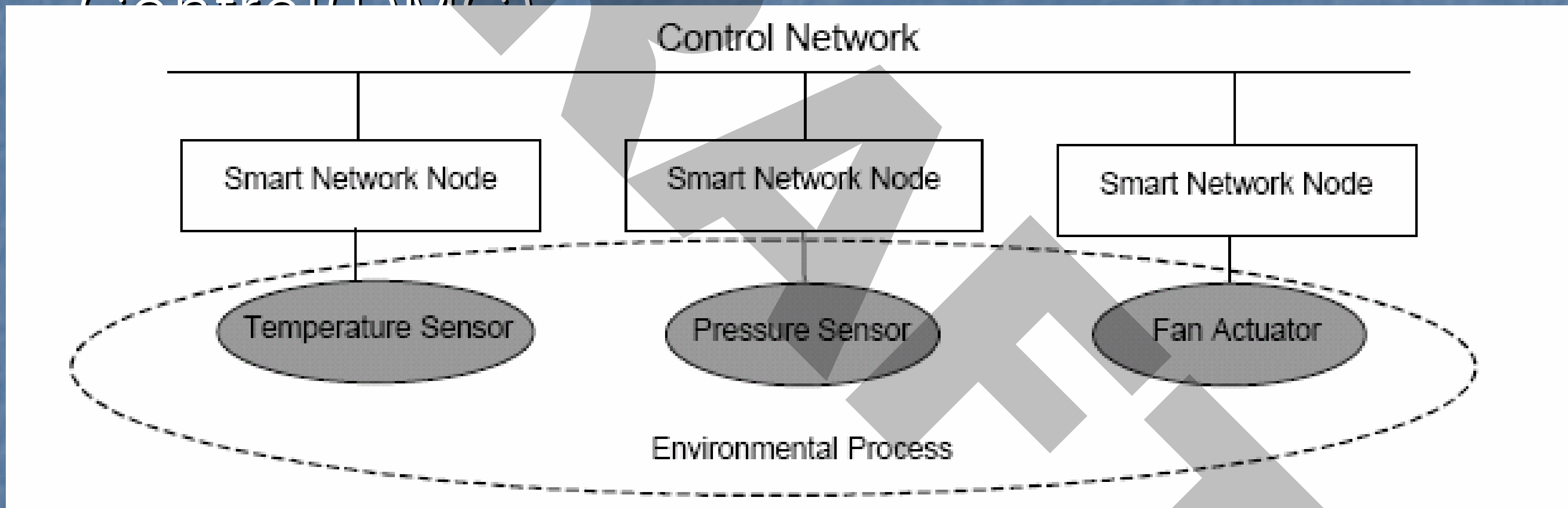
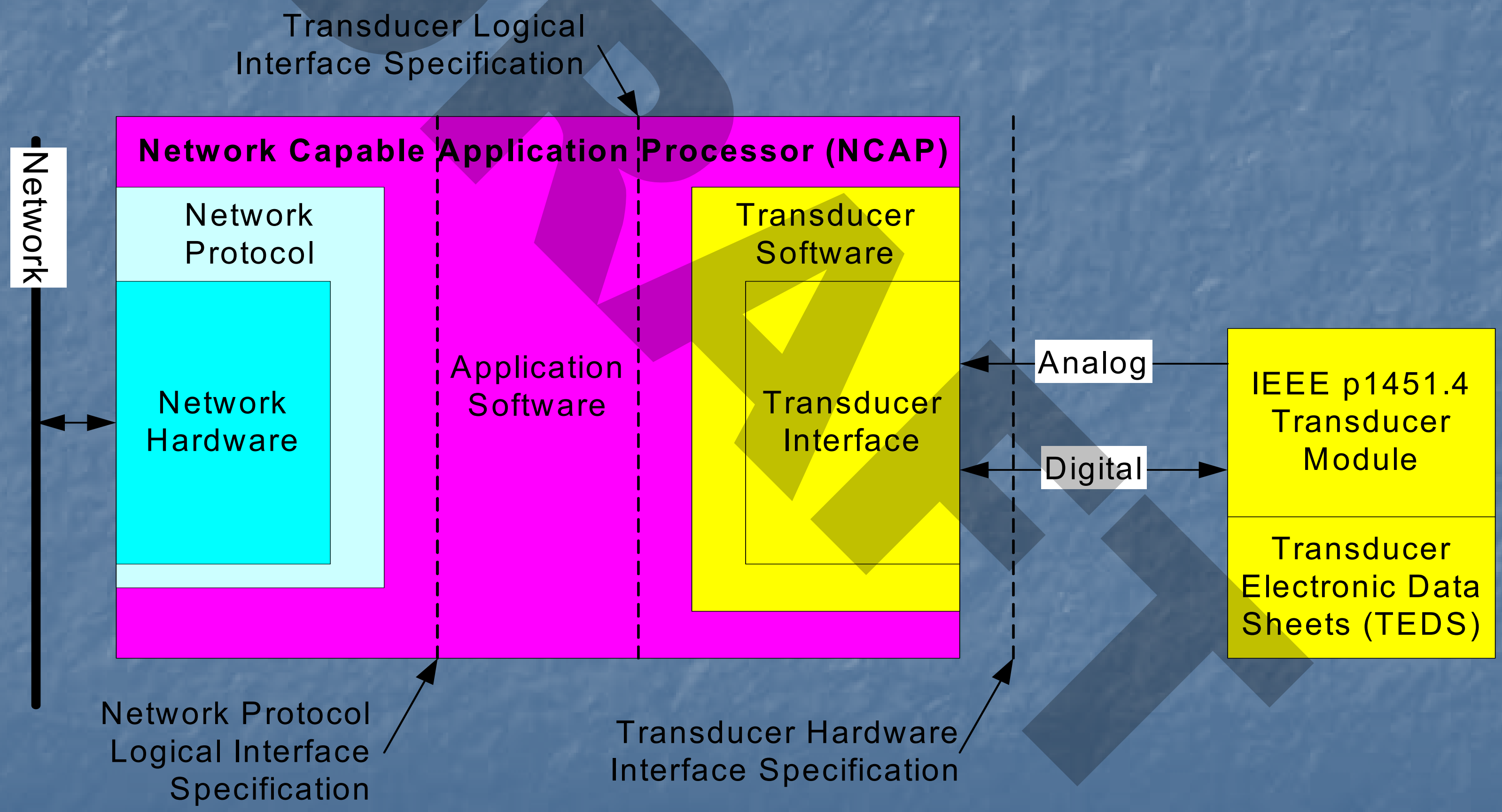


Figure 1: A basic distributed measurement and control system.

IEEE1451



Transducer Logical Interface Specification
Network Capable Application Processor (NCAP)
Network
Network Protocol
Network Hardware
Application Software
Transducer Software
Transducer Interface
Analog
Digital
IEEE p1451.4 Transducer Module
Transducer Electronic Data Sheets (TEDS)
Network Protocol Logical Interface Specification
Transducer Hardware Interface Specification

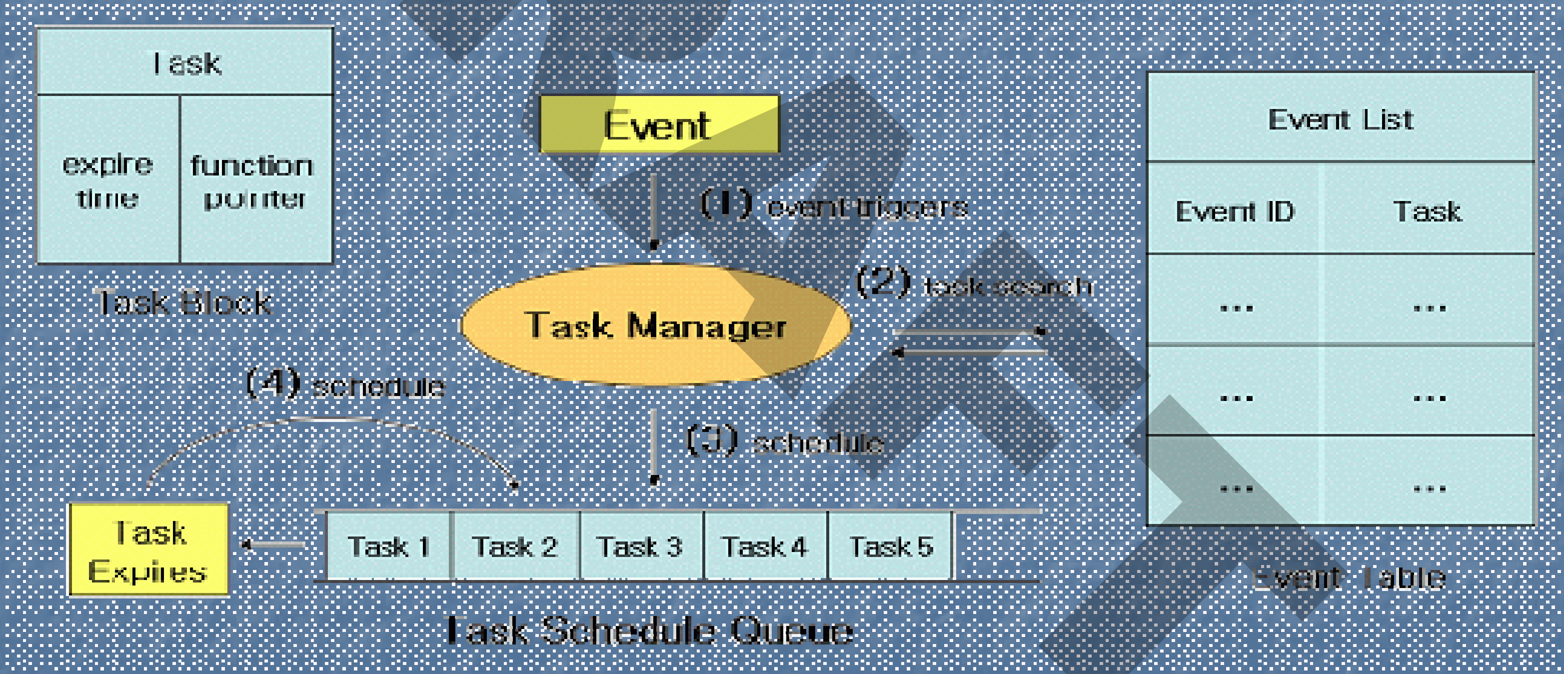
Project :Nano-X

- Project: NanoX
  - 8bitCPU
    - AVR co.:ATmega128
    - RAM: 4KB, Flash: 128KB
    - Radio: UK Bim-433, 433MHz
    - Power Supply < 15mA(tx or rx)
    - Half Duplex upto 40kbit/s
  - RF protocol
  - MAC protocol
    - Multi-hop routing protocol

Sensors

- Key
  - Single Stack Multi-Tasking OS
  - Short Lived Tasks
    - thread based tasks
    - event-driven nonpreemptive
    - no wait within a task
    - Parameter passing is possible, but restricted

Sensors



Task
expire time
function pointer
Task Block
Event
(1) event triggers
Task Manager
(2) task search
(3) schedule
(4) schedule
Task Expires
Task 1
Task 2
Task 3
Task 4
Task 5
Task Schedule Queue
Event List
Event ID
Task
...
Event Table

Sensors

- Registration of task
- typedef struct {

```
        int eventID;
        void (*function)(void);
} Task;

Task eventTable[MAX_TASK];

void TaskRegistrate(void)
{
            RegistEventTask(EVENT_ADC, HandleADC);
            RegistEventTask(EVENT_Timer, OneTick);
            RegistEventTask(EVENT_RADIO_RECV,
                                        HandleRecvData);
}
```

Sensors

- subroutine

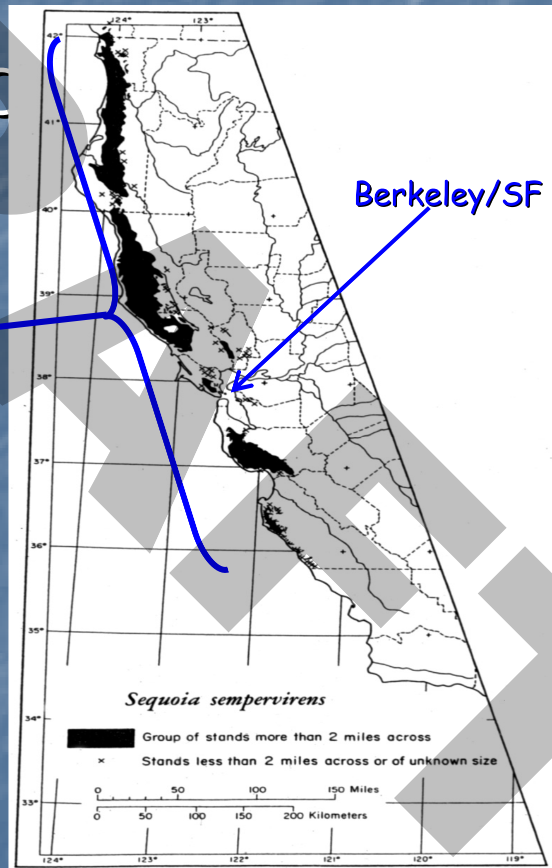
```
SIGNAL( SIG_OVERFLOW0 )
{
      EvnetTrigger(EVENT_TIMER, NOW);
}
SIGNAL( SIG_UART0_RECV )//RF
{
    if( (0x02 & inp(PINC) ) == 0x00) //when carrier ditected
    {
        EventTrigger(EVENT_RADIO_RECV, NOW);
    }

}
SIGNAL( SIG_ADC )
{
        EventTrigger(EVENT_ADC, NOW);
}
```

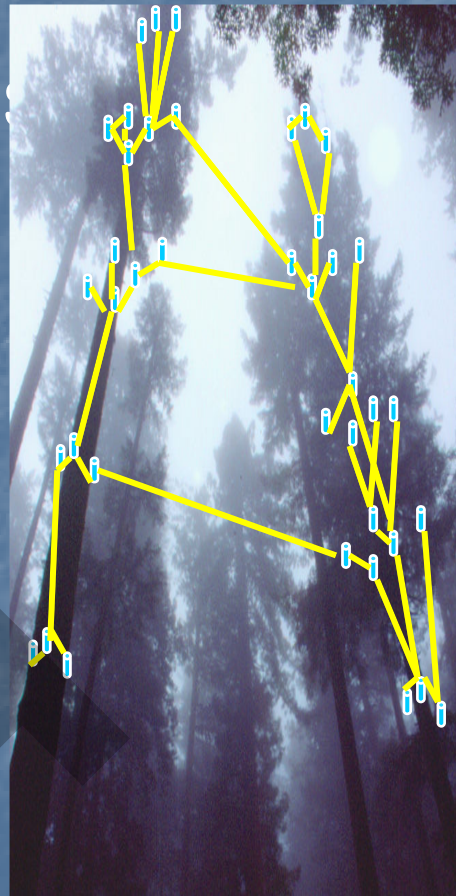
Sensor network and disaster prevention(2)

ChangDuk Jung
Prof. of Korea University
E-mail:jcd1234@korea.ac.kr

Ubiquitous research

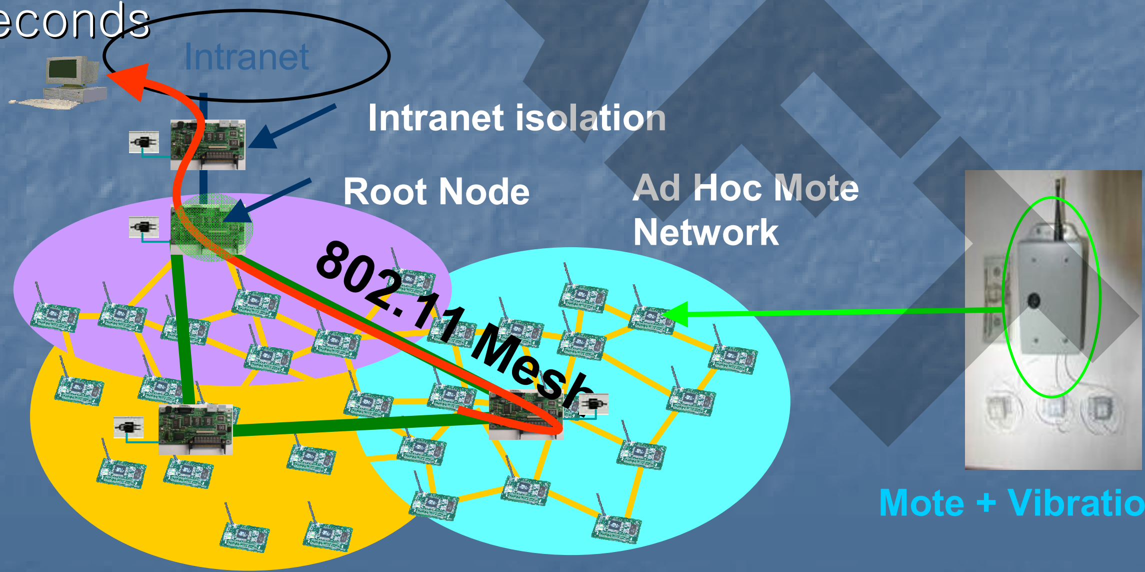
Research motivation





Application Request

- Monitoring Spaces(Equipment,Flood..)
  - Equipment failures in production fabs is very costly
    - Predict and perform preemptive maintenance
  - Typical fab has ~5,000 vibration sensors
    - Pumps, scrubbers, …
    - Electricians collect data by hand few times a year
    - Sample: 10's kilohertz, high precision, few seconds

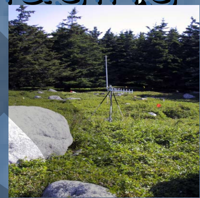


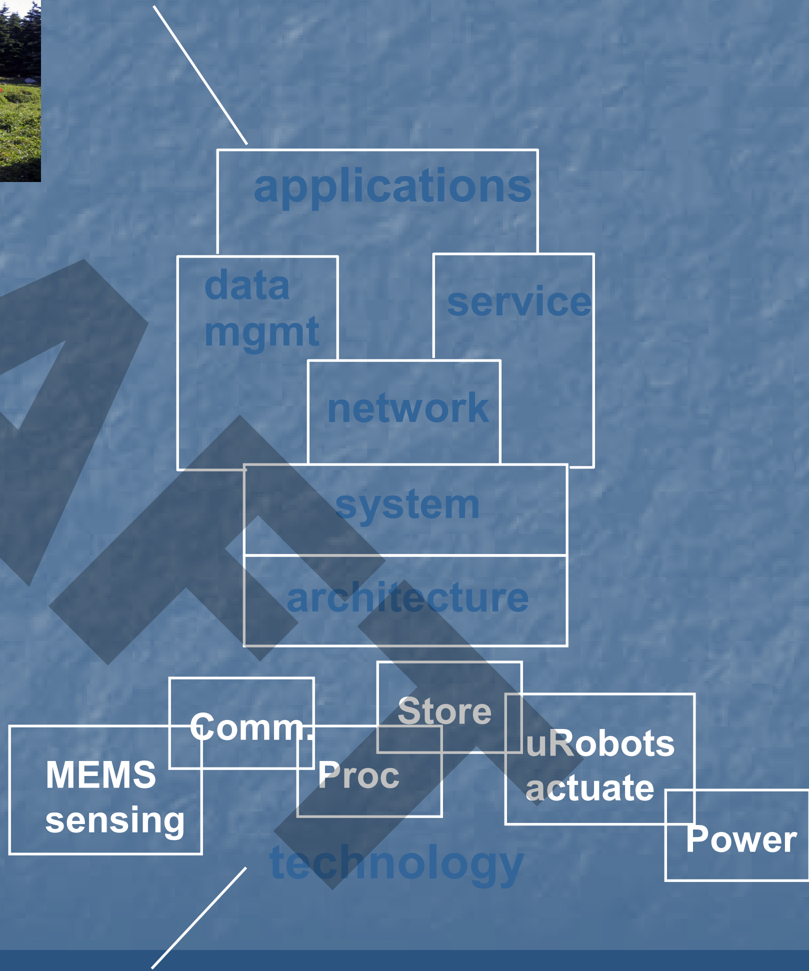
Mote + Vibration Sensors

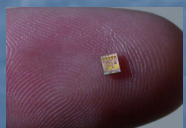
System Challenges(Site,Flood..)

Monitoring & Managing Spaces and Things



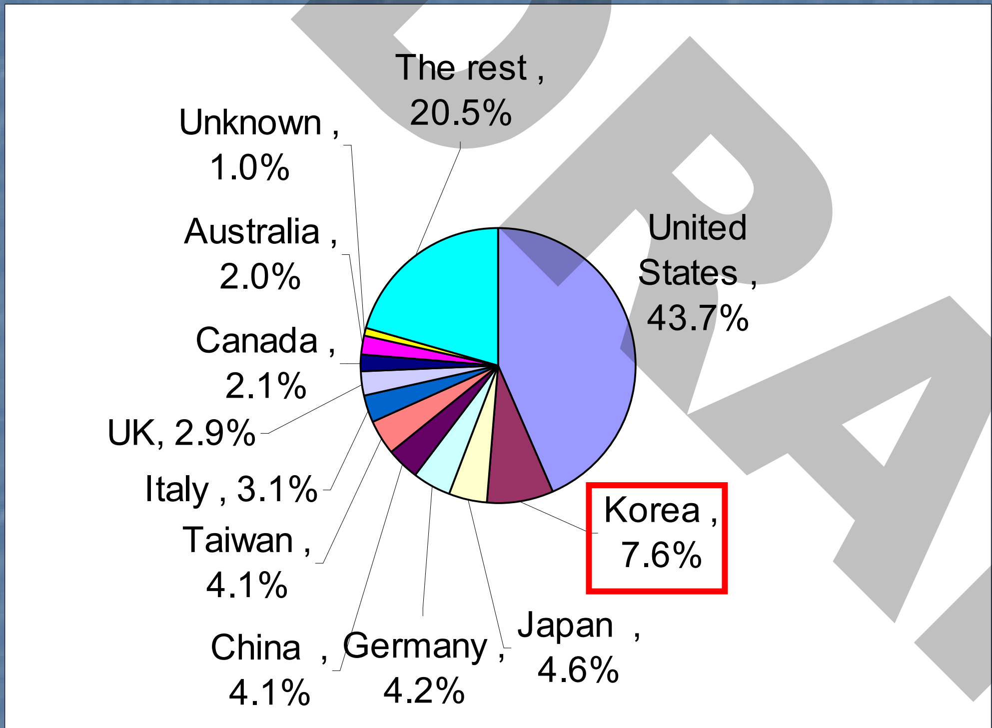




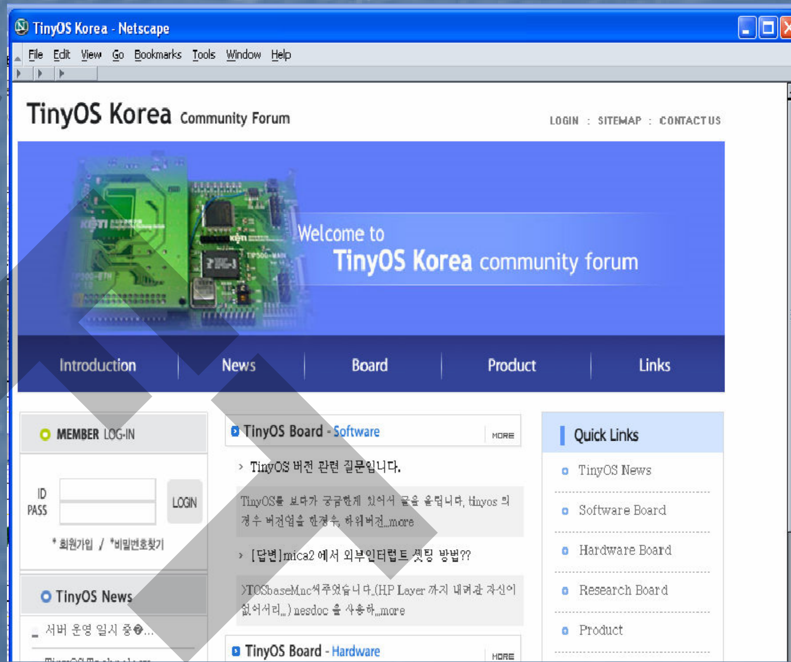


Miniature, low-power connections to the physical world

Interest about TinyOS In Korea



tinyos.net download distribution



Configuration

- Listing components list
  - Specifies the used components
- Wiring
  - Connect specification elements (interfaces, commands, events) together
- E

```
configuration Blink {
}
implementation {
  components Main, BlinkM, SingleTimer, LedsC;
  Main.StdControl -> SingleTimer.StdControl;
  Main.StdControl -> BlinkM.StdControl;
  BlinkM.Timer -> SingleTimer.Timer;
  BlinkM.Leds -> LedsC;
}
```

Components list

Wiring

Example : Networks – Send


```
includes IntMsg;

module IntToRfmM
{
  uses {
    interface StdControl as SubControl;
    interface SendMsg as Send;
  }
  provides {
    interface IntOutput;
    interface StdControl;
  }
}
implementation
{
  bool pending;
  struct TOS_Msg data;

  command result_t StdControl.init() {
    pending = FALSE;
    return call SubControl.init();
  }

  command result_t StdControl.start()
  {
    return call SubControl.start();
  }


    command result_t StdControl.stop()
  {
    return call SubControl.stop();
  }
```

Active Message

```
command result_t IntOutput.output(uint16_t value)
{
  IntMsg *message = (IntMsg *)data.data;

  if (!pending)
    {
      pending = TRUE;

      message->val = value;
      atomic {
        message->src = TOS_LOCAL_ADDRESS;
      }
      if (call Send.send(TOS_BCAST_ADDR, sizeof(IntMsg), &data))
        return SUCCESS;

      pending = FALSE;
    }
  return FAIL;
}
```

Send call

```
event result_t Send.sendDone(TOS_MsgPtr msg, result_t success)
{
  if (pending && msg == &data)
    {
      pending = FALSE;
      signal IntOutput.outputComplete(success);
    }
  return SUCCESS;
}
}
```

sendDone event Handler

Java Applications

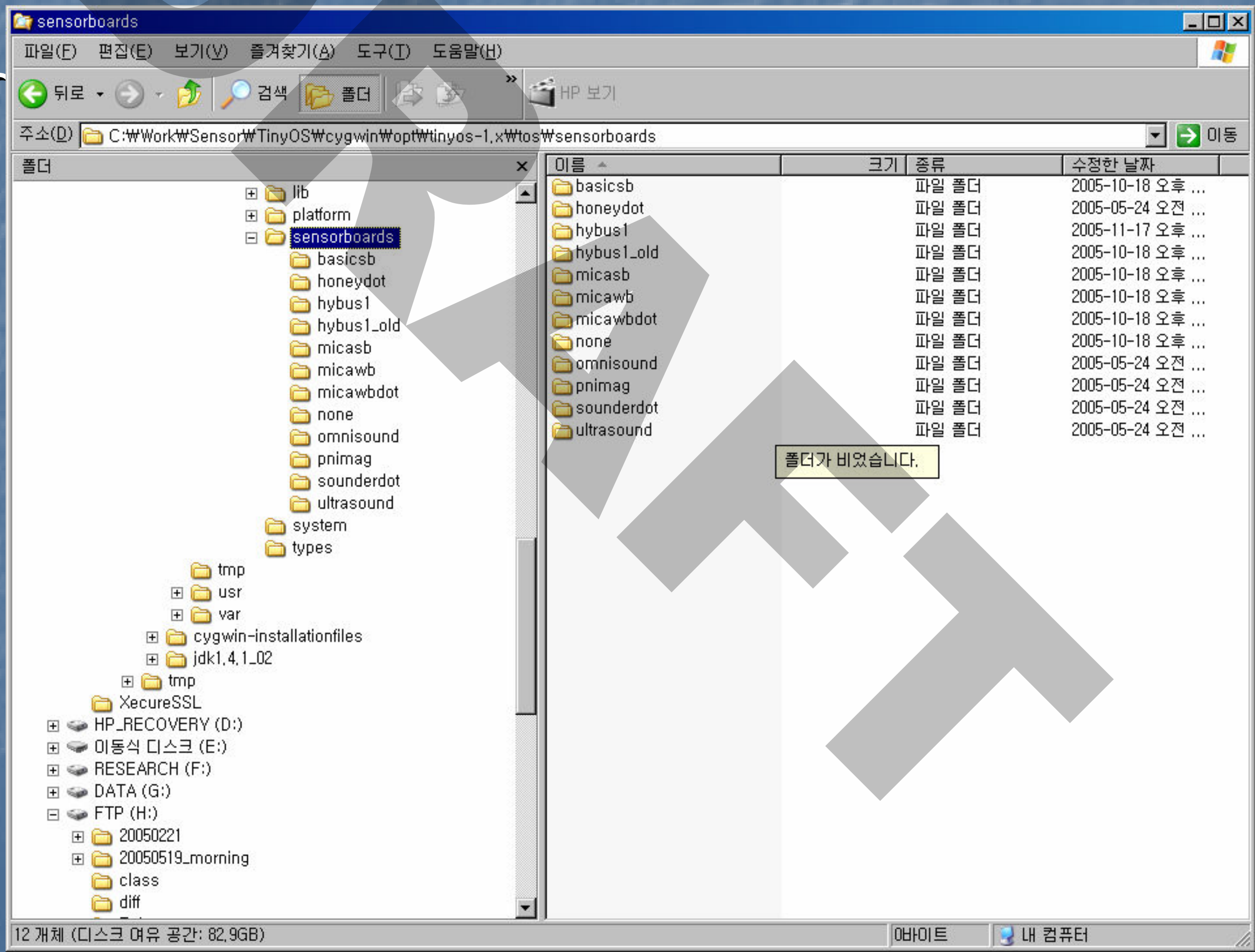
- Class **net.tinyos.message.MoteIF** interfaces with the SerialForwarder's TCP port
  - Provides **net.tinyos.message.Message** objects containing the message data

```
import net.tinyos.message.*;
import net.tinyos.util.*;

public class MyJavaApp {
  int group_id = 1;
  public MyJavaApp() {
    try {
      MoteIF mote = new MoteIF(PrintStreamMessenger.err, group_id);
      mote.send(new OscopeMsg());
    } catch (Exception e) {}
  }
}
```

This must extend net.tinyos.message.Message, which is generated using /usr/local/bin/mig

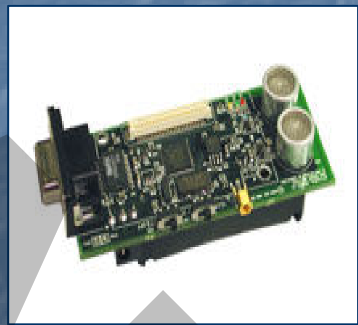
Directory

- Ser

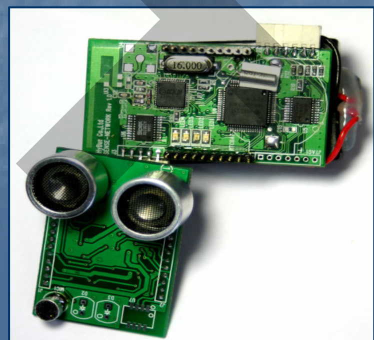


Target Hardware – Network Device View

- CC1000
  - MICA2,

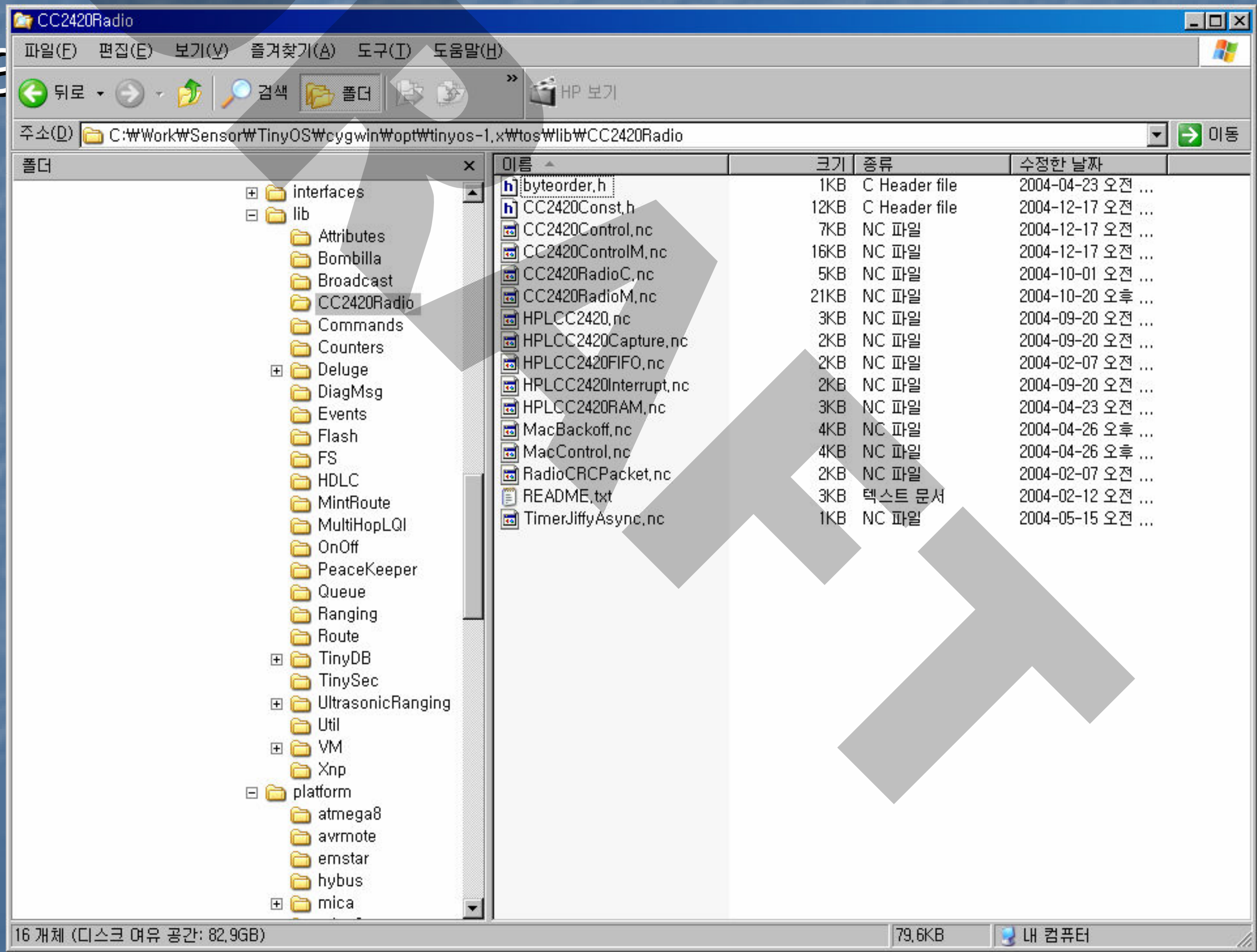


- CC24
  - MIC … eSk … us



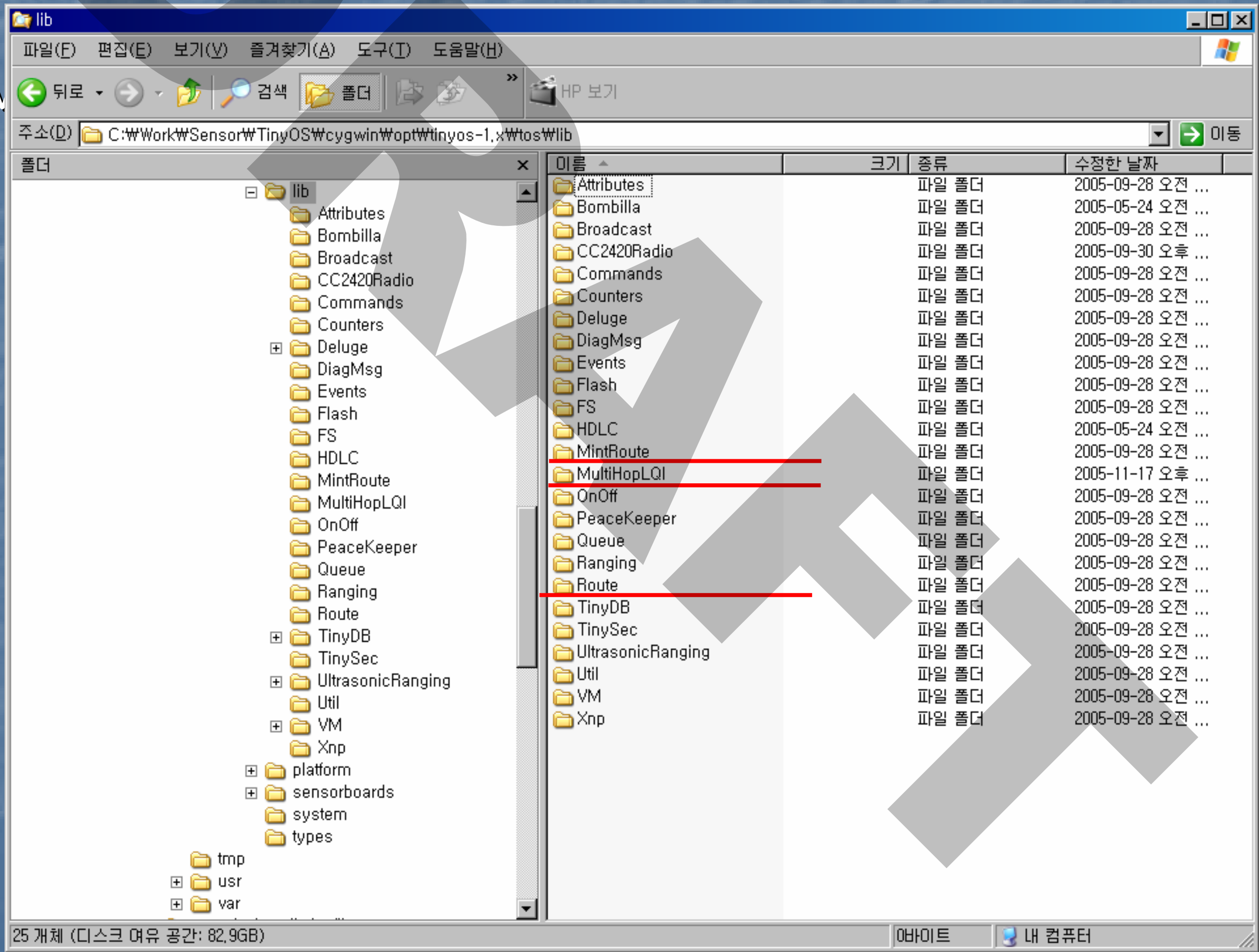
Network Stack

- Rela



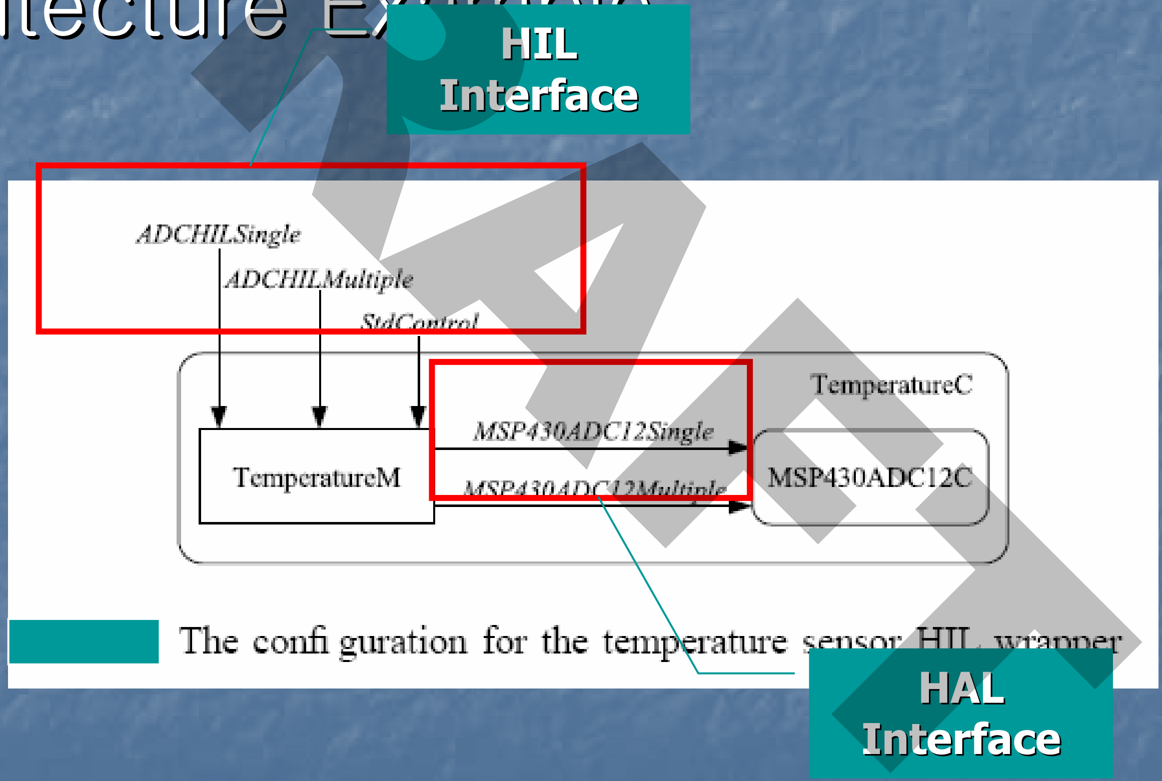
Network Programming

- prov



Architecture

- Architecture Example



HAA Implement

- Data buses
  - Problem
    - Many standard data busses
      - SPI/USART, UART, I2C, and 1-Wire busses.
    - User Clock
    - Power management
  - Solution
    - HPL
      - interface HPLUSARTControl
      - interface BusArbitration

HAA Implement

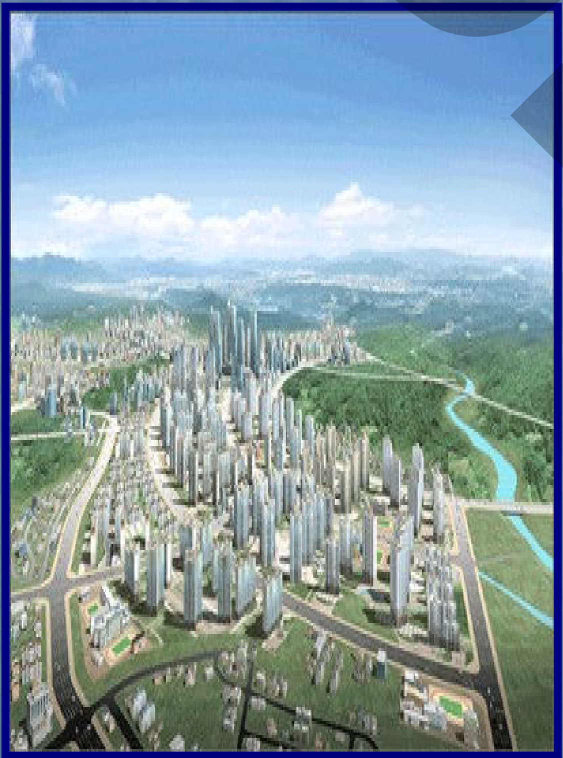
- Radios
  - Problem
    - Radio functionality changes more frequently
    - Physical layer
    - Link layer
  - Solution
    - Physical layer
      - HPL
        - interface HPLXXXBusComm
        - interface HPLXXXRadioControl
        - interface HPLXXXRadioData
        - interface HPLXXXCmd , HPLXXXCapture , HPLXXXFIFO
      - HAL
        - configuration HALXXXRadioC
      - HIL
        - module HILXXXRadioM
    - Link layer
      - HPL
        - interface HPLCC2420, HPLCC2420Capture
        - interface HPLCC2420FIFO
      - HAL
        - Transport, MAC, PHY…
      - HIL
        - interface CSMAControl
        - interface CSMABackoff
        - interface LowPowerListening
        - interface CC1000LowPowerListening
        - interface RadioTimeStamping
        - interface CC1000RadioTimeStamping
        - interface RadioControl

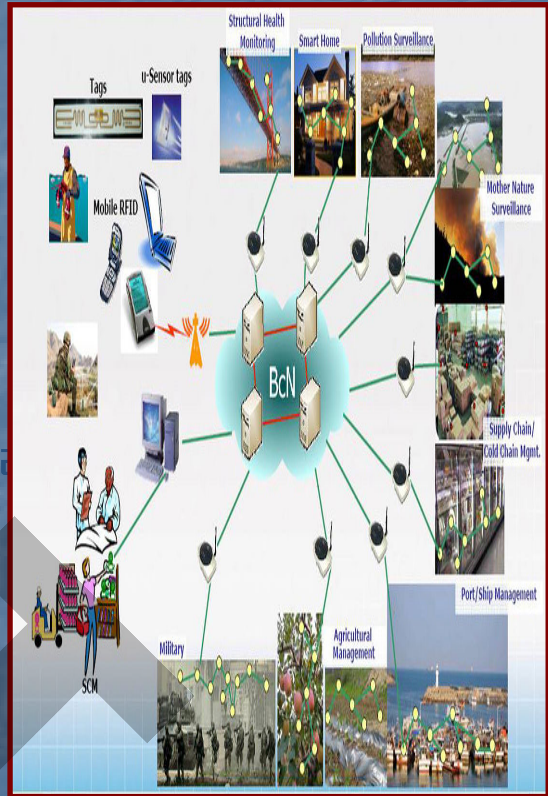
Sensor network and disaster prevention(3)

ChangDuk Jung
Prof. of Korea University
E-mail:jcd1234@korea.ac.kr

Ubiquitous research

1. Ubiquitous City in Disaster prevention





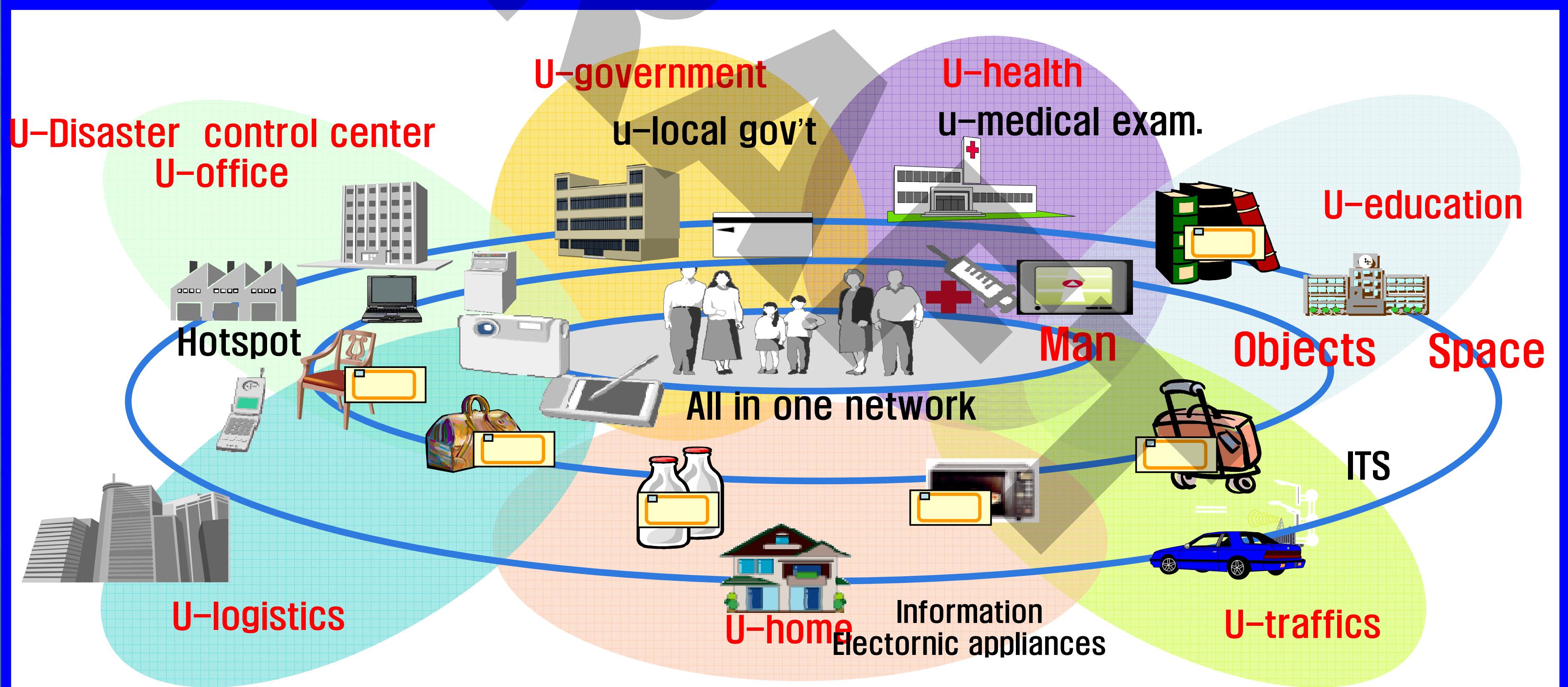
Crisis control-City

3s

3C

2. Ubiquitous City on 3S and 3C

A definition of U-City is that it's a kind of new city in the 21st century using Ubiquitous technologies and high-tech information and telecommunication infra.



Source: 한국 U-City포럼, FUJITSU

. The Present Status of U-City Promotion in Korea

Incheon City

* Hub of East-North Asia

Yongin City

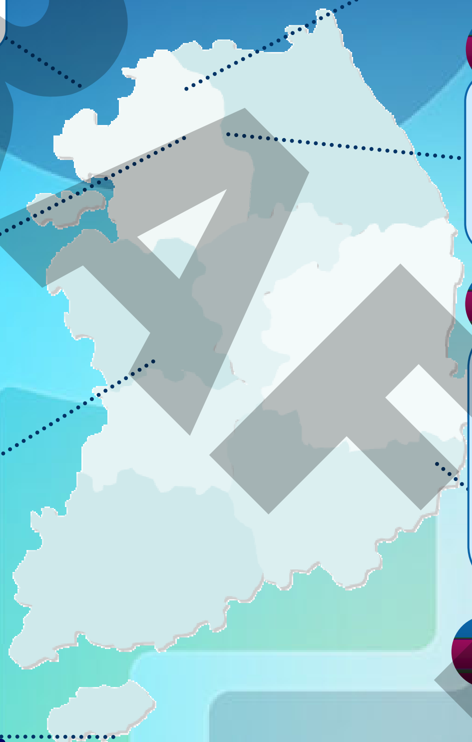
• Construction for Small scaled U-City

Daejeon & Asan City

* Construction of Industry innovative city

Jeju City

* u-Museum, u-Park, and u-Coupon Service



Seoul City

•Construction for Digital Media City

Pajoo City

* High-Tech city

Busan City

* Build for u-Port , u-Convention, and u-Traffic, etc

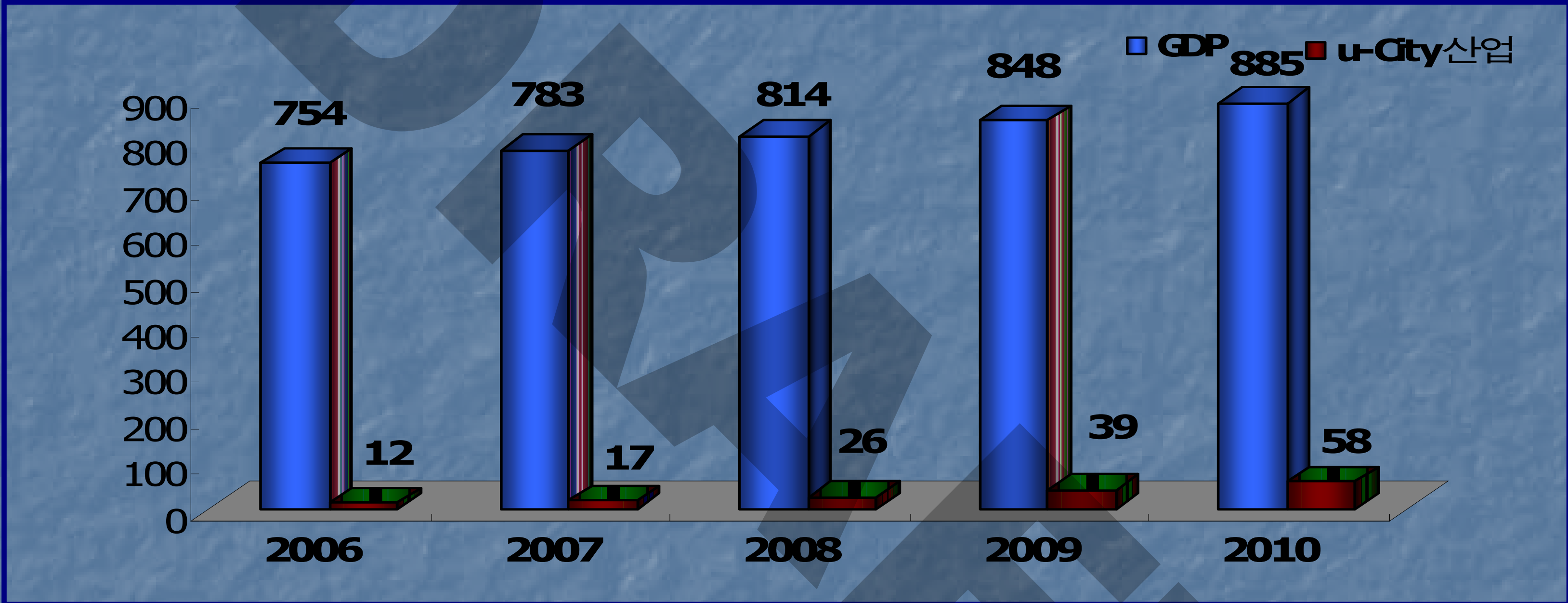
Other Cities

* Kwangjoo City, Suwon City
* Ohsong City, Kwangkyo City

6. A Sizeable Market for U-City

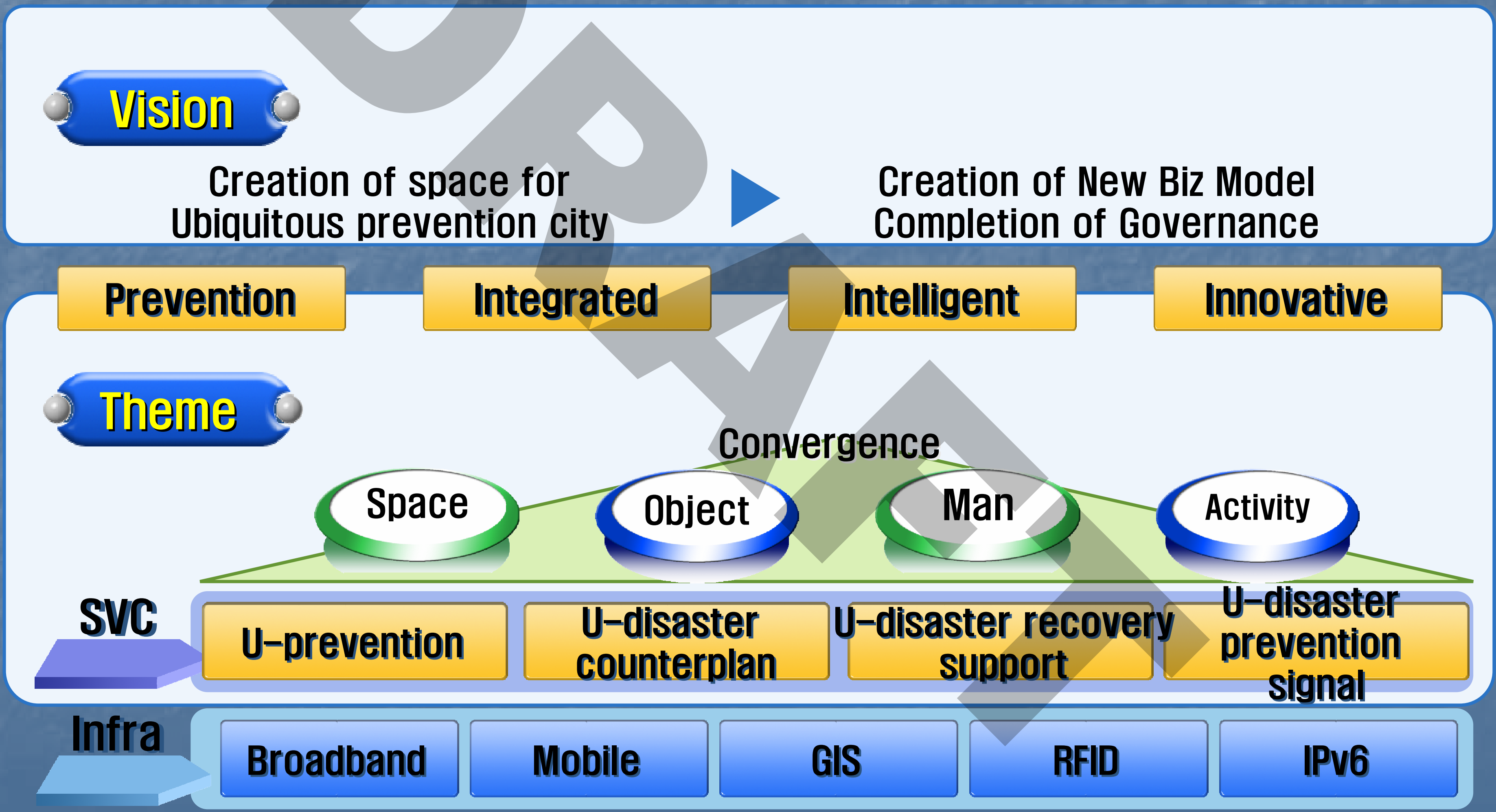
A sizeable market

(단위 :조원)/Korean Won

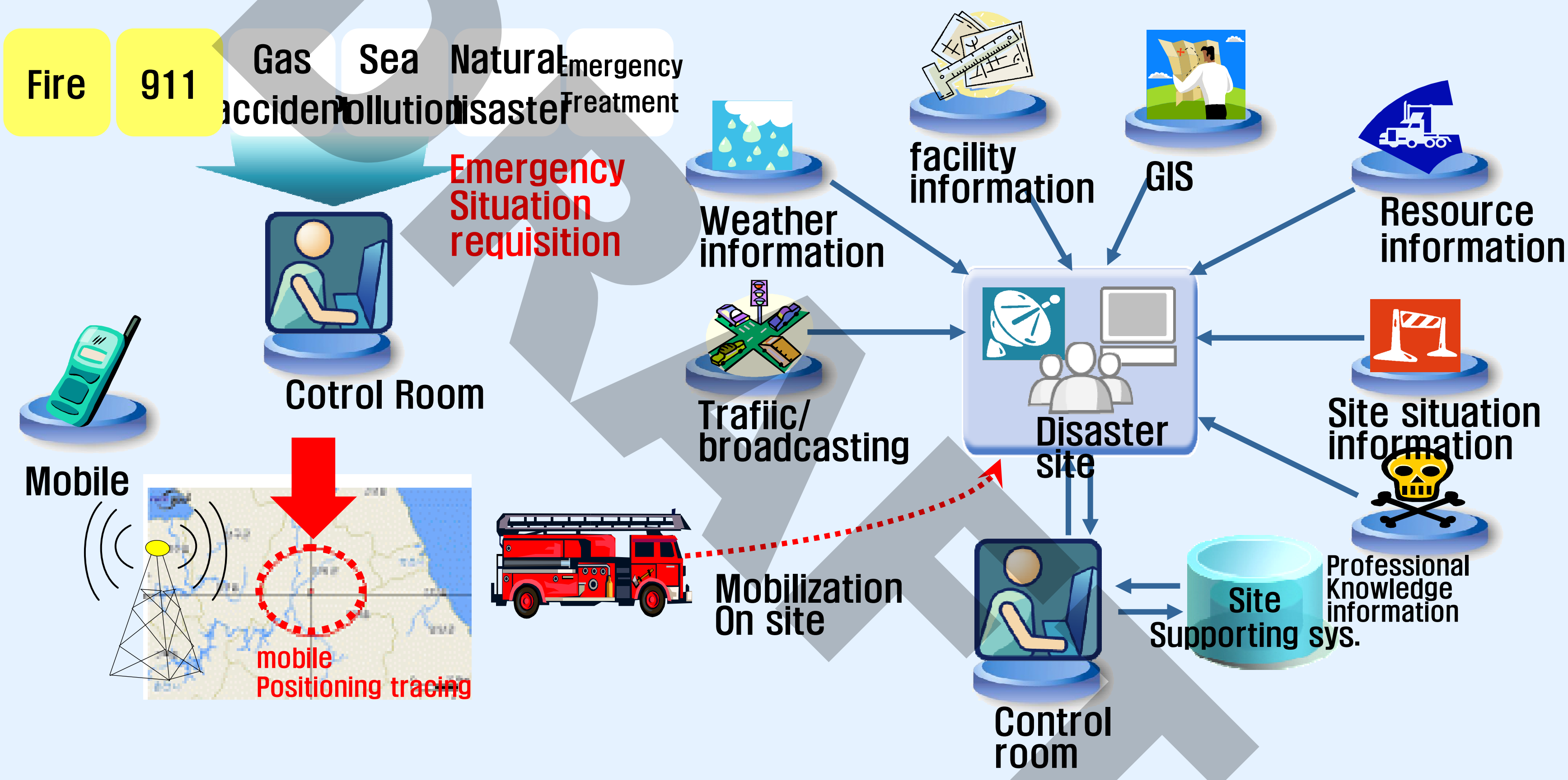


For U-city Industry in Korea
2006: US$ 12,000,000,000
2007: US$ 17,000,000,000
2008: US$ 26,000,000,000
2009: US$ 39,000,000,000
2010: US$ 58,000,000,000

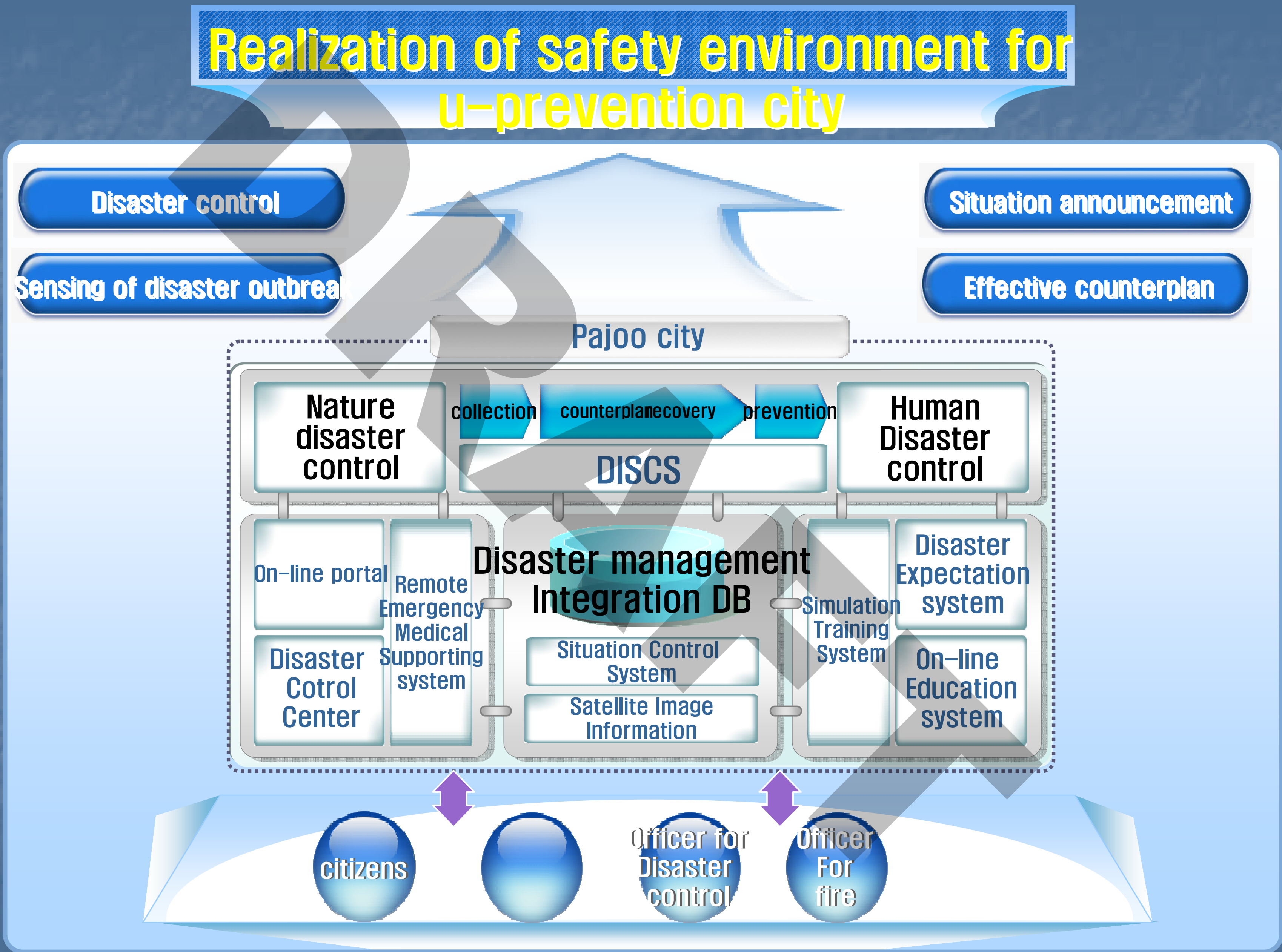
. An Appearance background for U-Prevention



. A Necessity of U-Prevention



. U-Prevention Model in Pajoo City



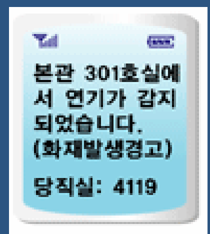
. An Expected Effectiveness

Analysis Item	Result
The person in charge of disaster	① Minimization of Human life injury and property damages ② Increase of ability for person in charge through training ③ Others
Citizens/ The organs concerned	① Increase of citizen's attention & participation ② Offering on-spot service ③ Others

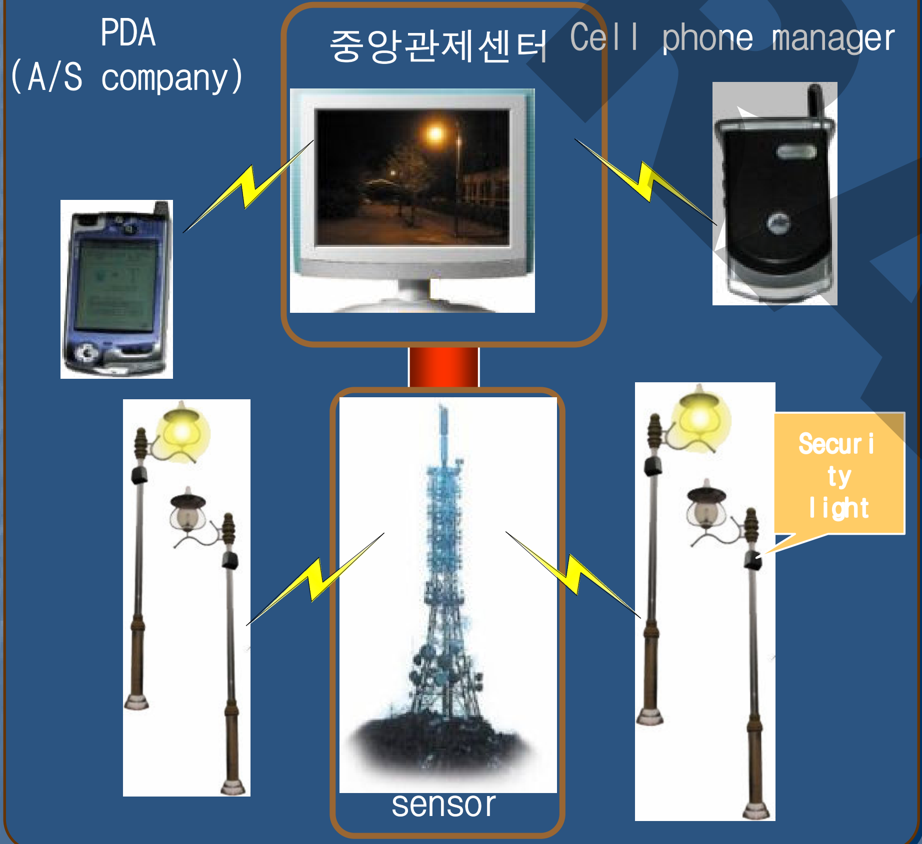
. The Present Status of Possession for Relevant Technologies(2)

Technologies
Network System Reinforcement SW
Regions Weather Broadcasting Editing SW
Real Time Dam & Earthquake Monitoring System
Earthquake Disaster Countermeasure System
Under installation for the seabed seismometer in Ulleun Island, Korea

Crisis control system



Crisis control system

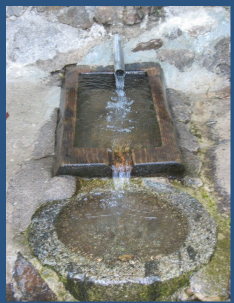


- Wireless sensor network

❑ 재난방지지원 : U-수질 관리 시스템

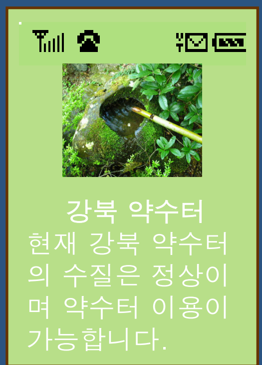
Flood monitoring system

▣ real time flood monitoring



•Sensor
•RFID

▣ display of cellphone



▣

•수

❑ Intelligent War Room for crisis control

Scenario

Decision support system for crisis control system

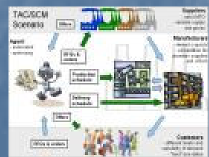
▣ 환경변화 조기 감지

Crisis control system

Crisis control system

Intelligent War Room

▣ Forcast system



Agent

- Device Network supply: Tablet, Wearable,
- Agent
- Intelligent War Room

Thank you!