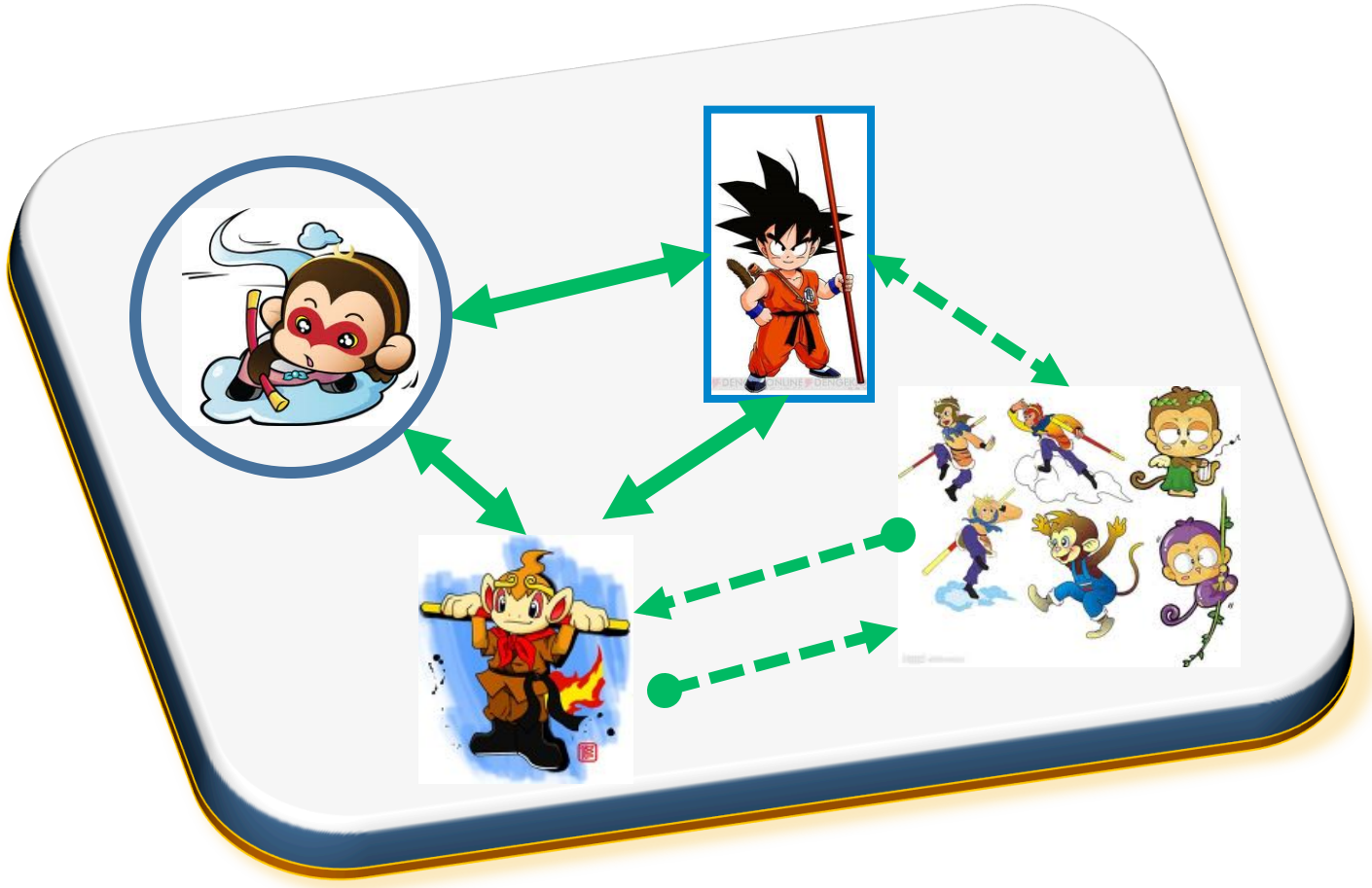
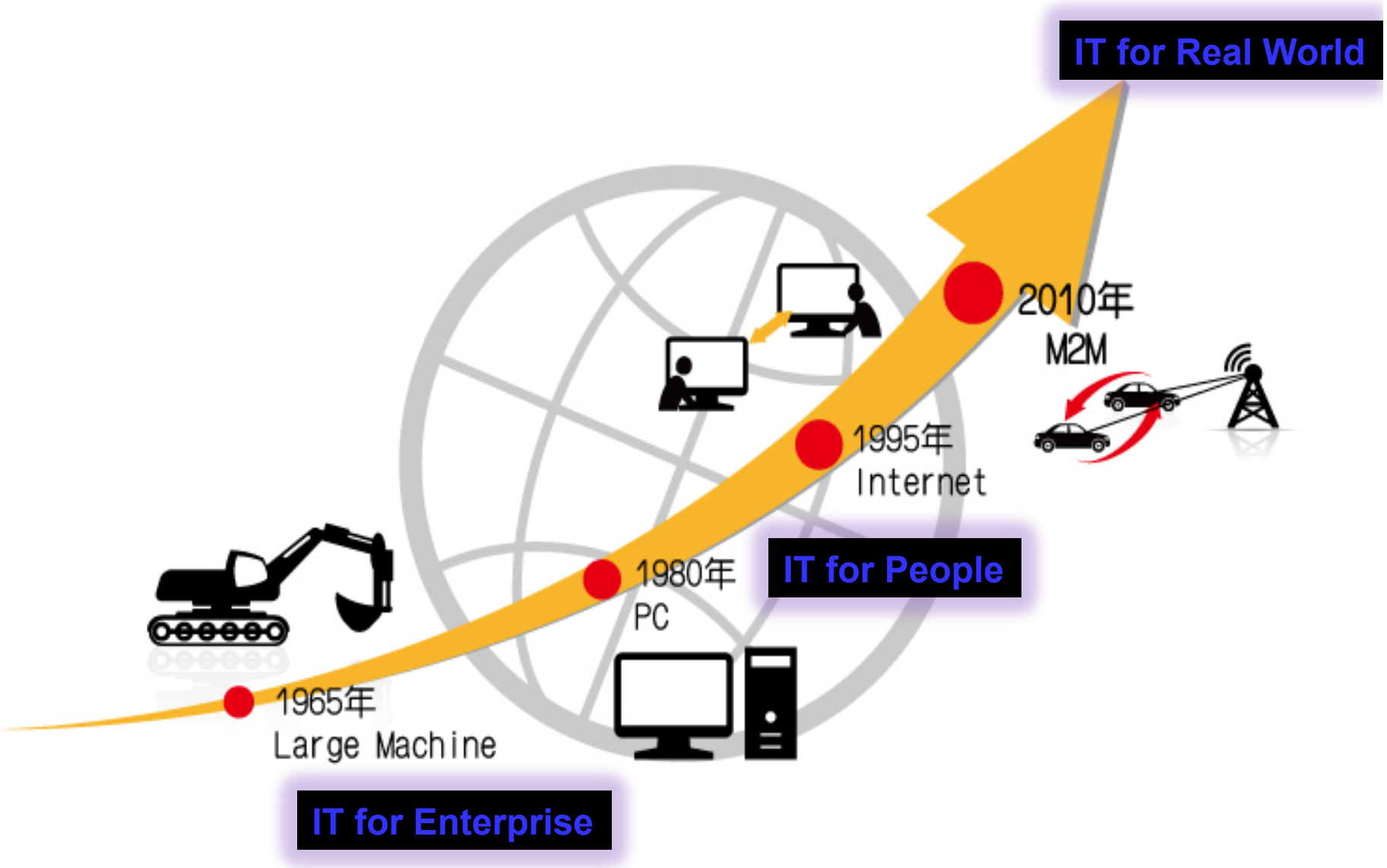
Wu-Kong: WSN Unleashed

An Intelligent M2M Management Project

Kwei-Jay Lin
University of California, Irvine



Largest Growth Opportunity in ICT



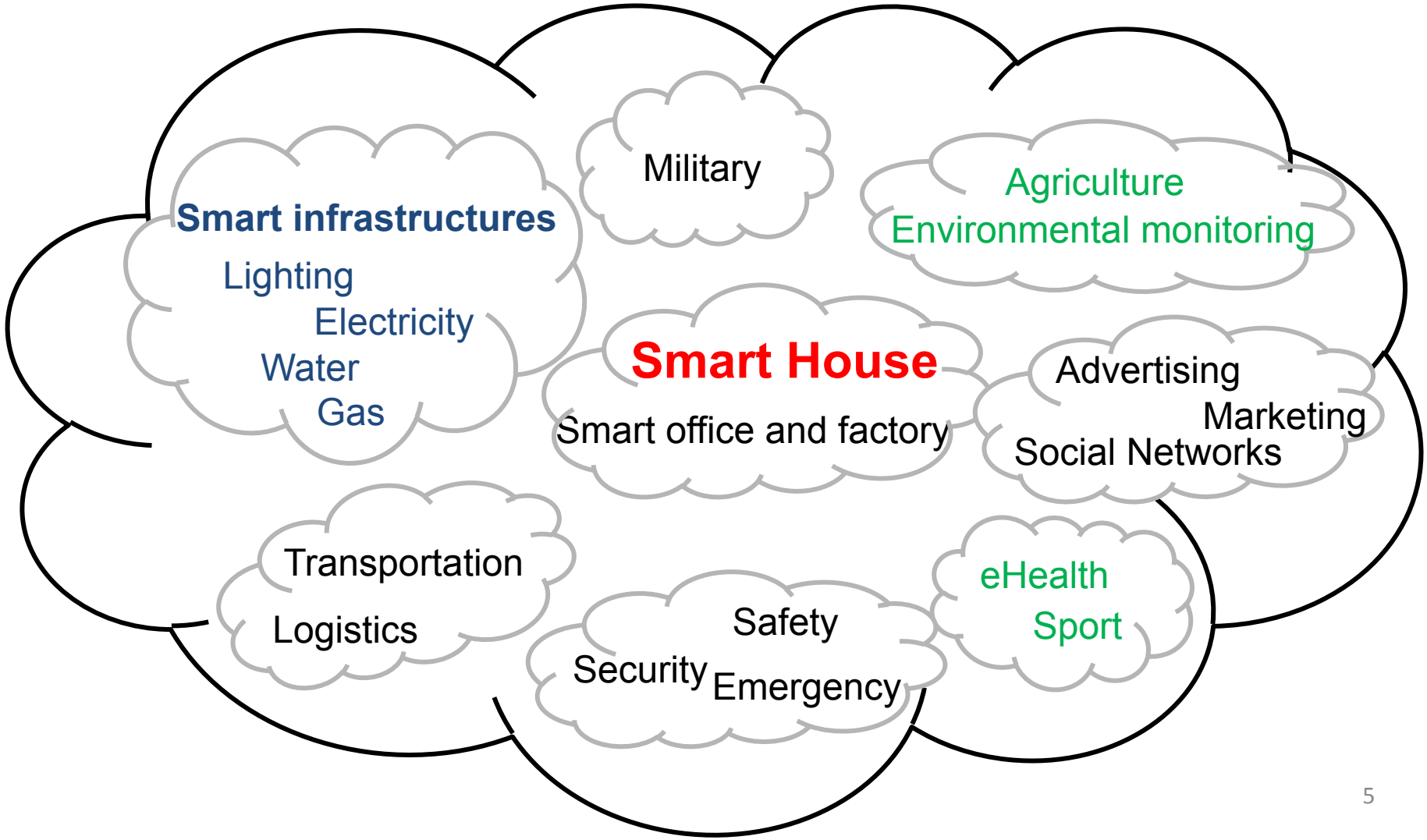
Game Changing Capabilities in Future IT

- Sensing
  - Connected embedded sensors help us “hear/see” things that we could not hear/see in the past
  - Enhance our sensory by environmental and embedded sensors
- Tele-Robotics
  - More things will be connected to Internet than human today
  - Enhance our capability by controlling things through digital devices
- Communication
  - Network traffic will keep increasing by the traffic generated by things
  - Enhance our reach via wireless and high-speed network
- Analysis
  - Connected devices (sensors) can produce a sea of data which can be transformed into knowledge and wisdom
  - Enhance our understanding by cloud computing and data reasoning

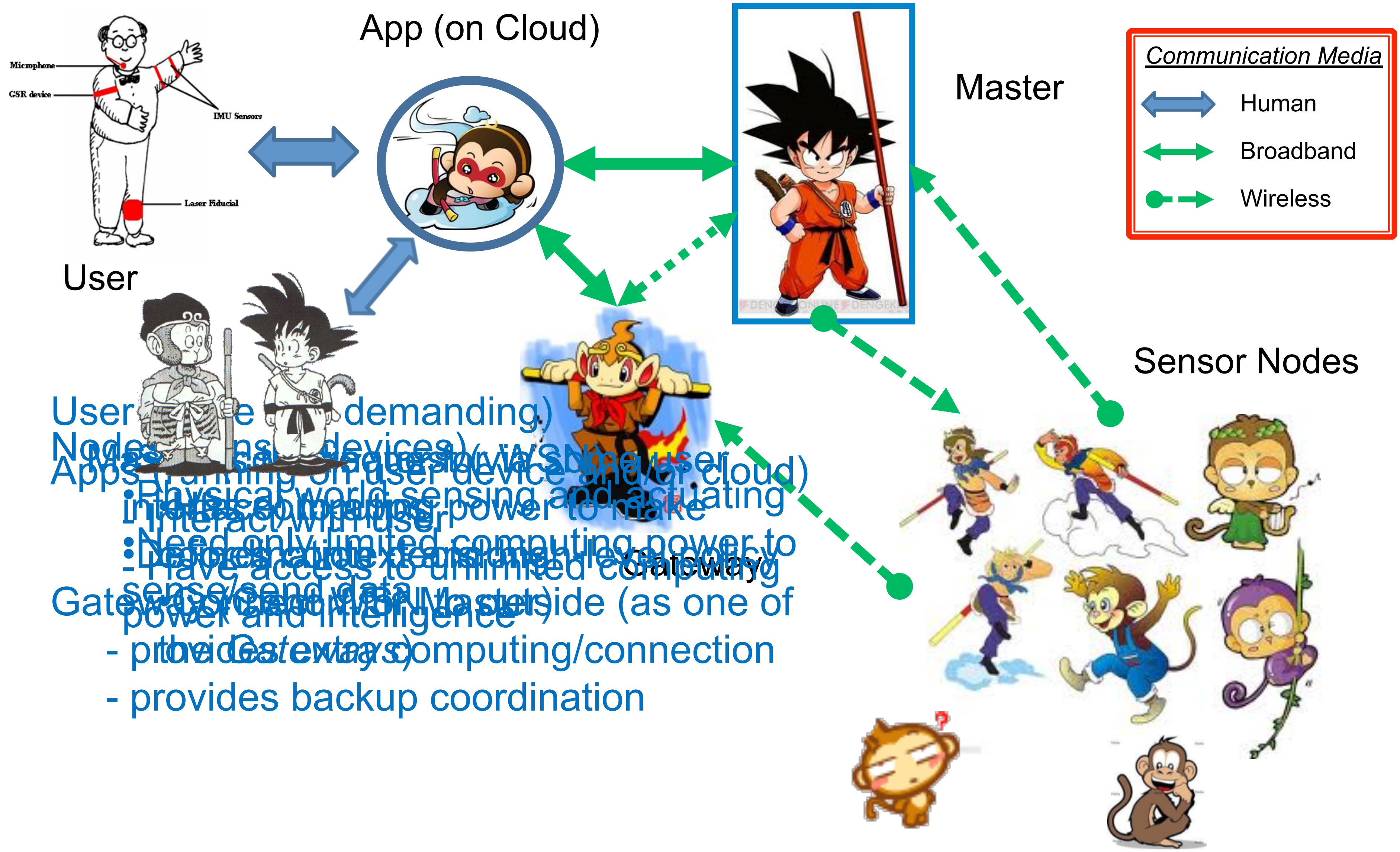
Wu-Kong : Literally, …

- 悟 (Wu): enlightened
  - For our project: *intelligent*
- 空 (Kong): vanity
  - For our project: *virtual middleware*
- The project is to build an *Intelligent Middleware* for M2M, that can
  1. recognize and adapt to context and user demand;
  2. configure or transform devices into service components;
  3. deploy the most powerful yet least expensive solutions;
  4. do all of the above by remotely accessing sensors.

Wu-Kong Application Areas

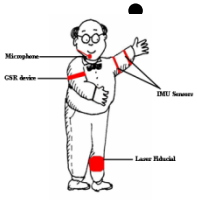


Wu-Kong: User Perspective



- provides backup coordination

Roles, Questions, Challenges

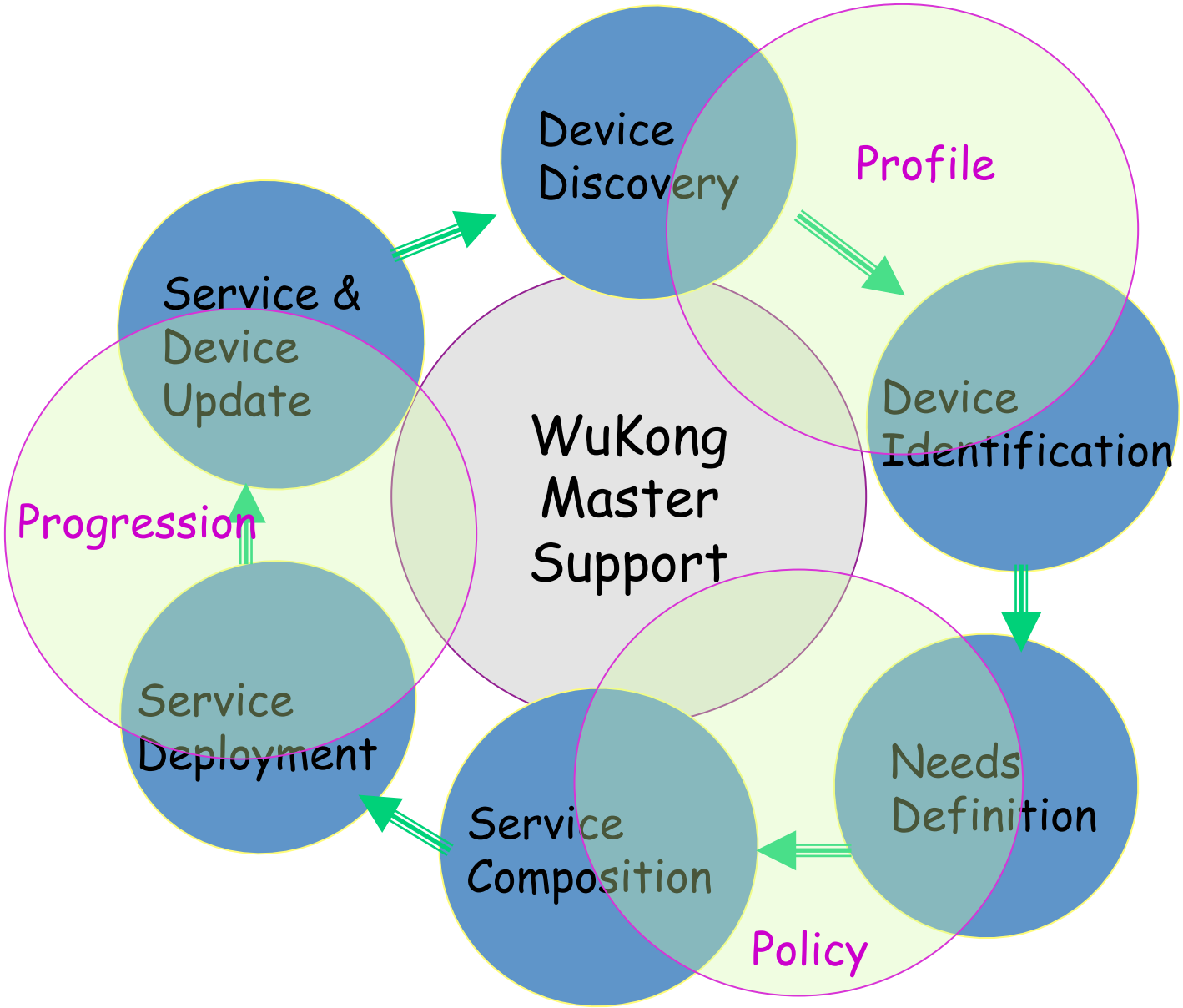


- User (intuitive)
  - How do users define context and high-level policy?
  - Can WSN trust a user and vice versa?

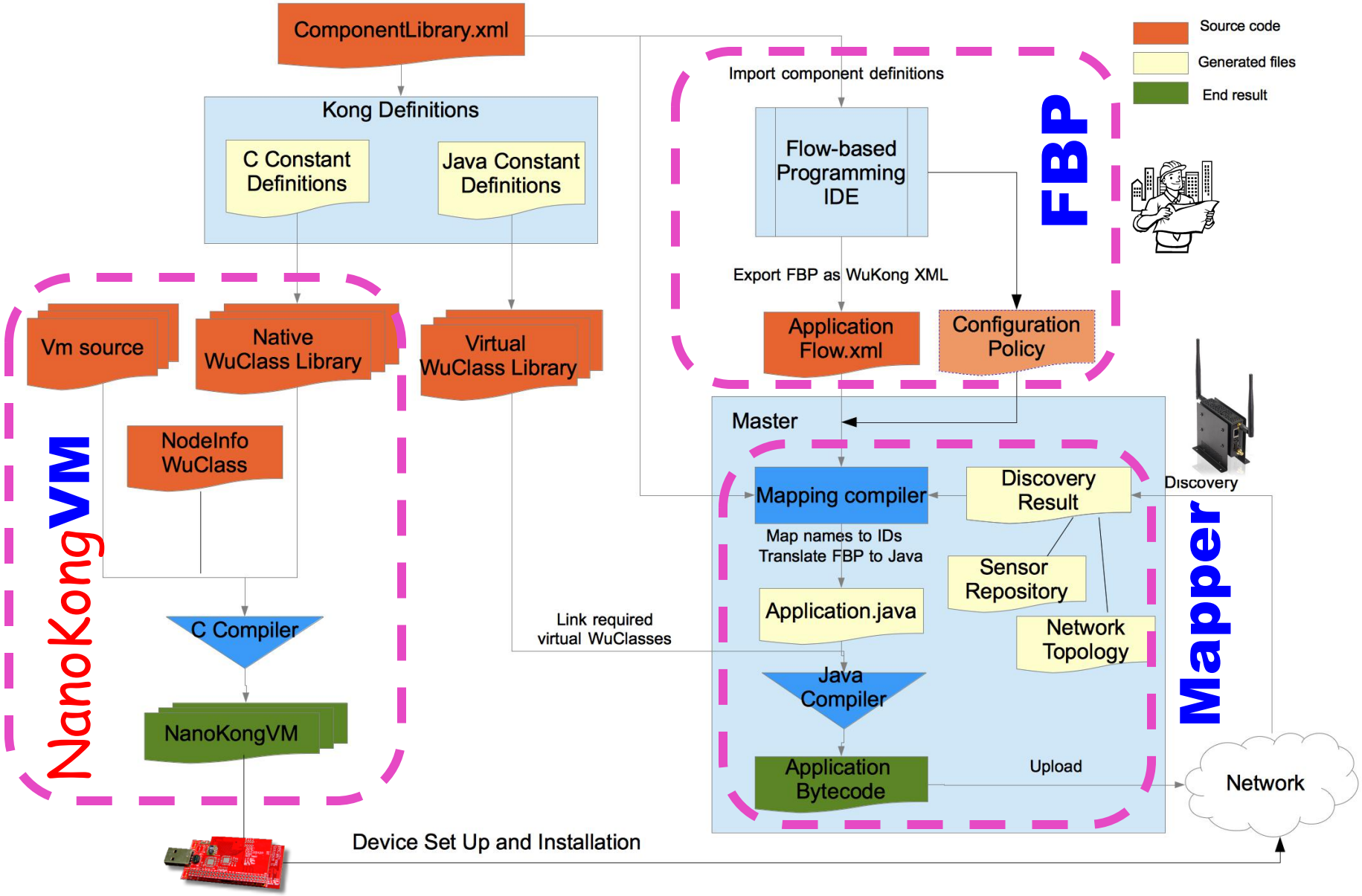
- Apps (concurrent)
  - How do they use, reuse and share sensors?
  - How do users start, pause and kill (if incompatible) apps?

- Master (intelligent)
  - How does it know about & control heterogeneous sensors?
  - What's the tradeoff between complexity and effectiveness?
- Nodes and network (efficiency)
  - Would it be powerful enough to do all that?
  - How do we handle run-away sensors or network blackout?

Wu-Kong : Device's Perspective



Wu-Kong : Students' Perspective



Wu-Kong Profile Framework

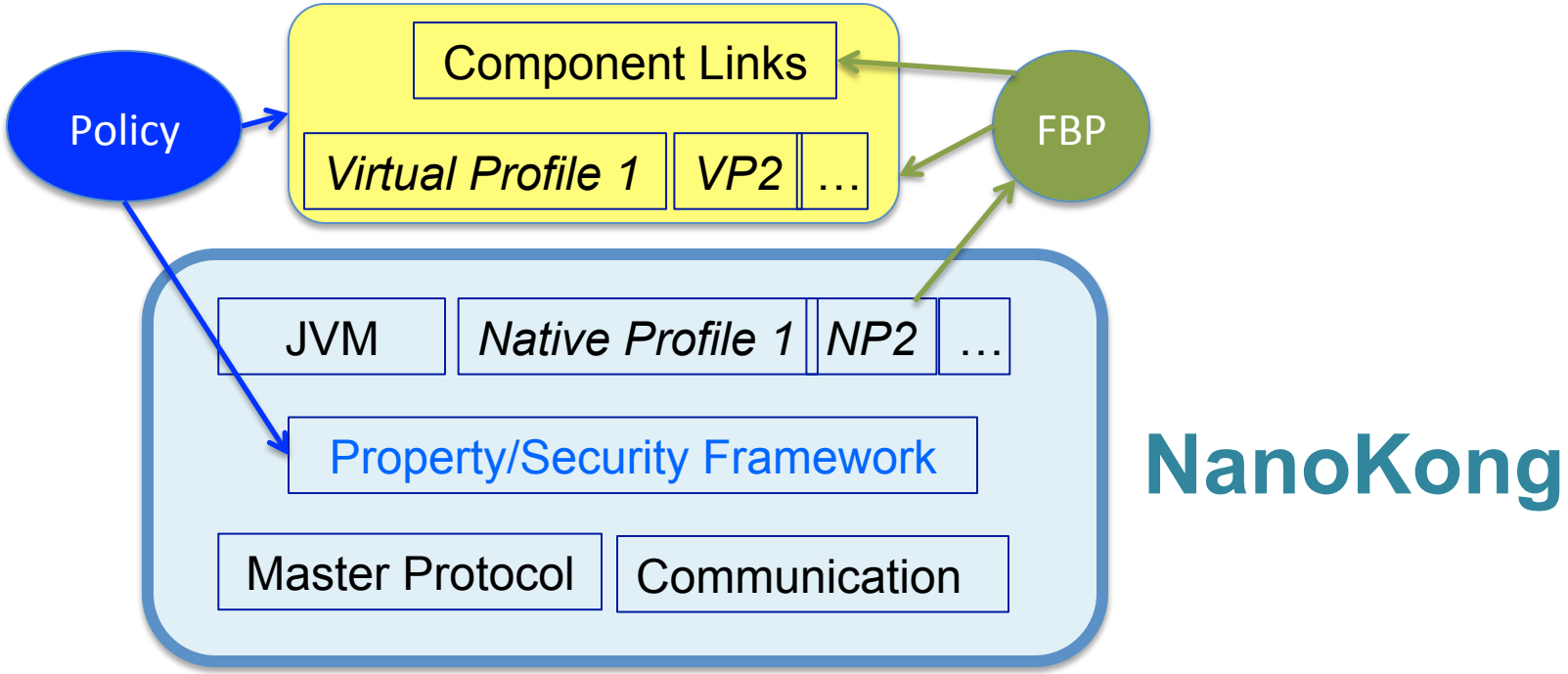
- To configure an M2M network, Master needs to know what sensor resources are available on each sensor device.
- Master-Device protocol has three phases:
  1. Determine what devices are on the network (*discovery*)
  2. Determine what those devices can do (*profile*)
  3. Determine what those devices should do (*policy*)
- After Master has "discovered" devices and queried them, each device reports its native profile that provides:
  - What resources are available on a device
  - How to access those resources

Native vs. Virtual Profile

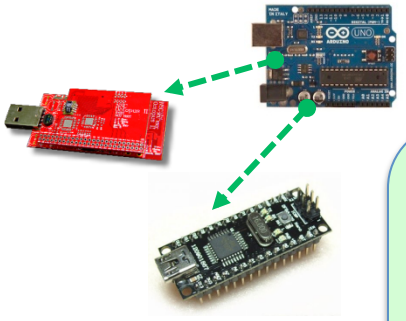
- Most device profiles are defined to access a device's fixed native hardware
- Functionality beyond the device's native hardware design can be added by uploading software function code on a device.
- Such a functionality is referred as a "virtual" profile
  - This profile isn't (directly) tied to the node's hardware
  - It is implemented in software (Java dynamic loadable bytecode)
- In this way virtual sensors can be implemented, e.g. combining data from different sensor sources into a new 'sensor', or reporting the max, min or mean reading in a sliding window.

Building NanoKong Platforms

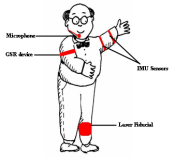
- Every node must be statically pre-loaded with sensor device drivers (native profiles), communication support, and JVM.
  - The pre-loaded code is called *NanoKong* (i.e. tiny WuKong)
- Security and other property framework can also be included.



Wu-Kong : Distributed Perspective



Sensors & Gateways with NanoKong



User Request

Master

Sensor with Native Profile

Discovery & Profile Response

Application Code Upload

Configuration Setup

Application Execution

R1

R2

R3

R4

R5

Sensor Query & Profile Discovery

Sensor & Profile Collection

Context and Policy Identification

Configuration Decision

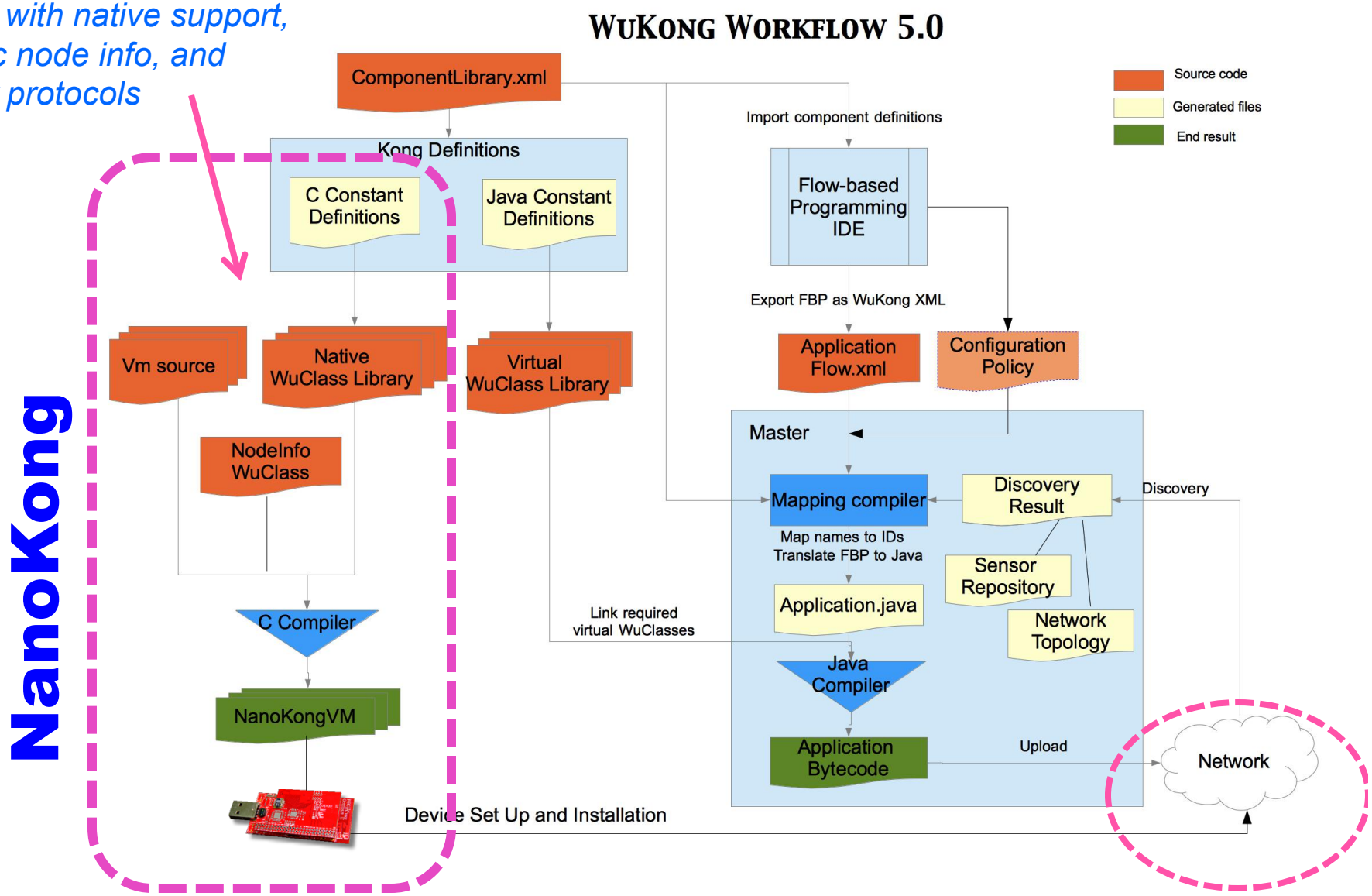
Application Deployment

APP Code

Context

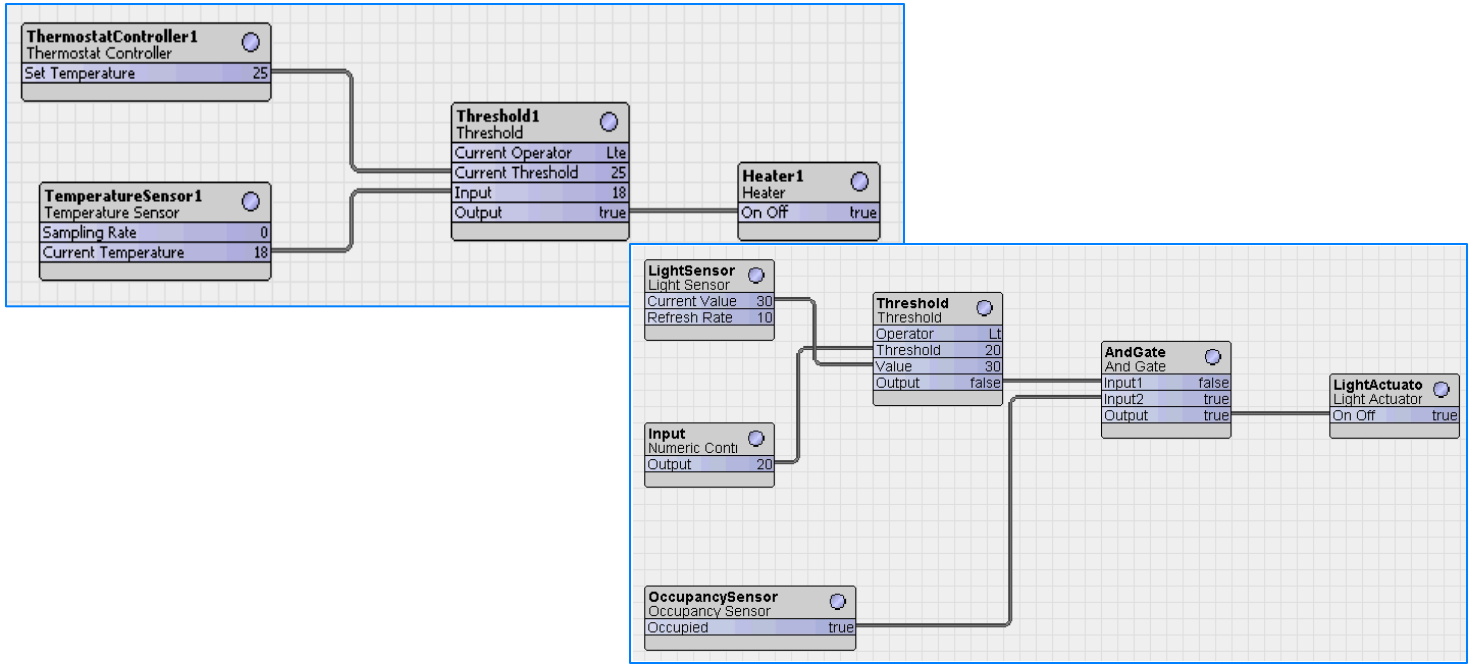
BIG Data

Each sensor device is loaded with native support, specific node info, and Master protocols



Wu-Kong : App Developer's Perspective

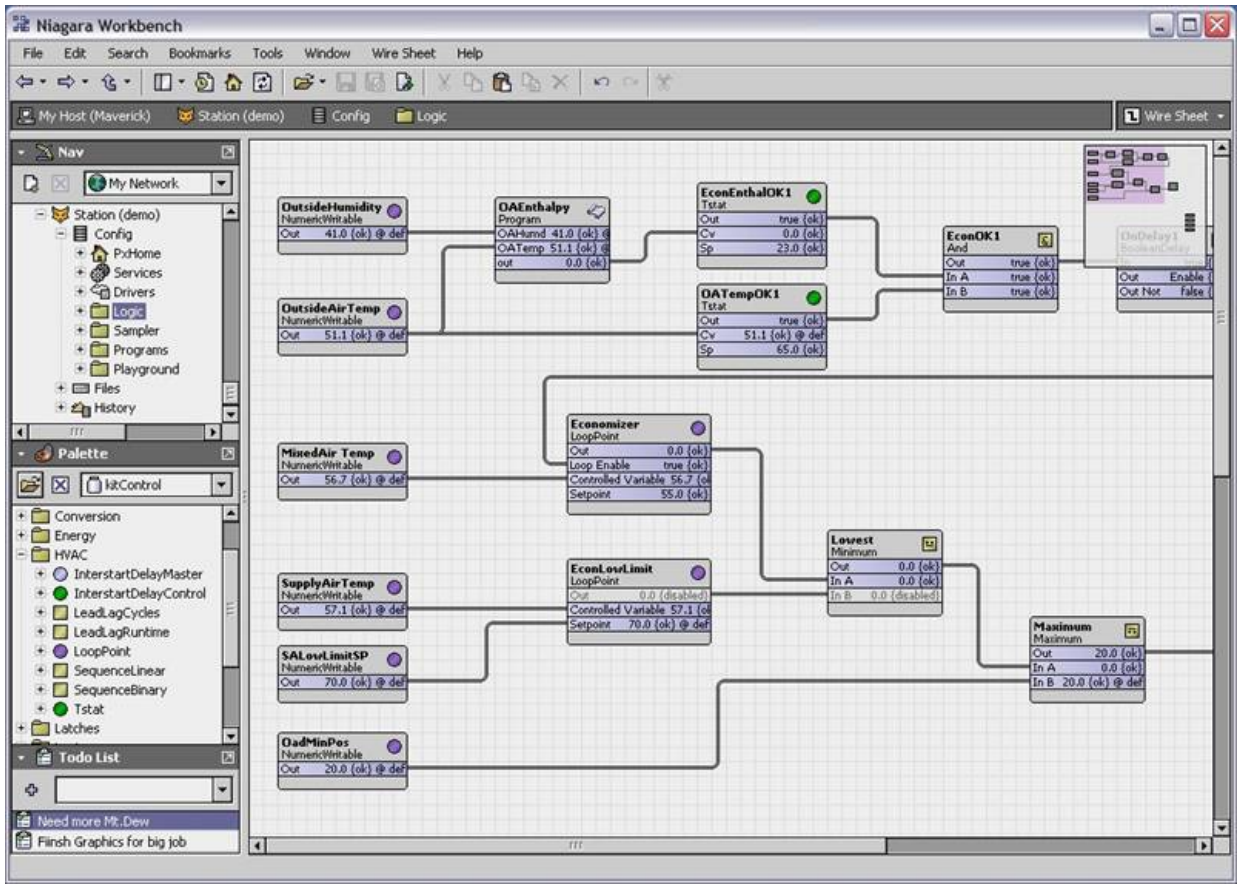
- For various application domains, M2M applications should be easy to develop by domain experts using simple and intuitive programming technologies and without technical device coding knowledge
- The *flow-based programming* (FBP) defines applications as logical networks of "boxes" (i.e., *components*), that exchange data and trigger actions across connections (i.e., *links*) between boxes.



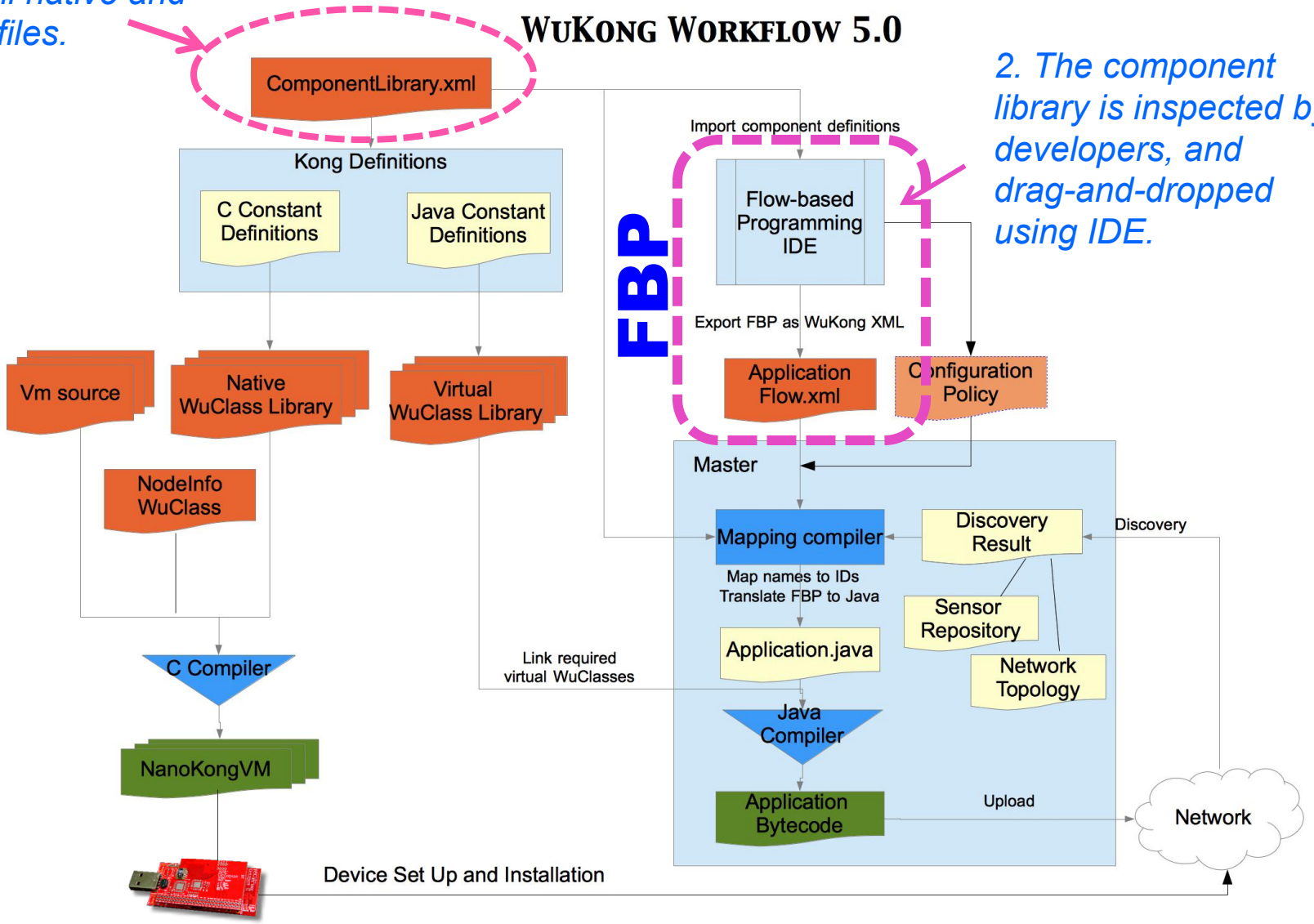
Build Flow Based Programs

Intel-NTU Center

Professional IDE such as *NaigaraAX Workbench* (by *Tridium*) can be used for FBP creation, and for inspecting *WuKong Component Library*.



1. We have created a component library that includes all native and virtual profiles.

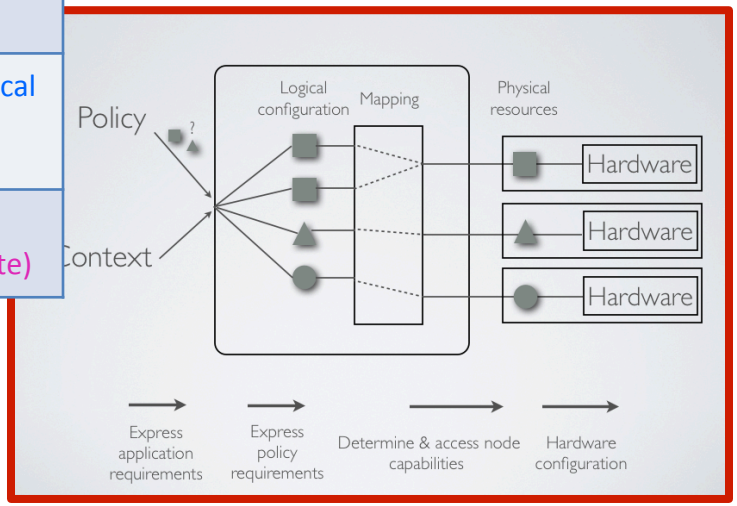


2. The component library is inspected by developers, and drag-and-dropped using IDE.

Wu-Kong : Researcher's Perspective

- Logical service components must be mapped to physical sensor nodes on networks with compatible capabilities
- Mapping may be 1-1, n-1, 1-m, or n-m
- Mapping should be optimized and consider many attributes

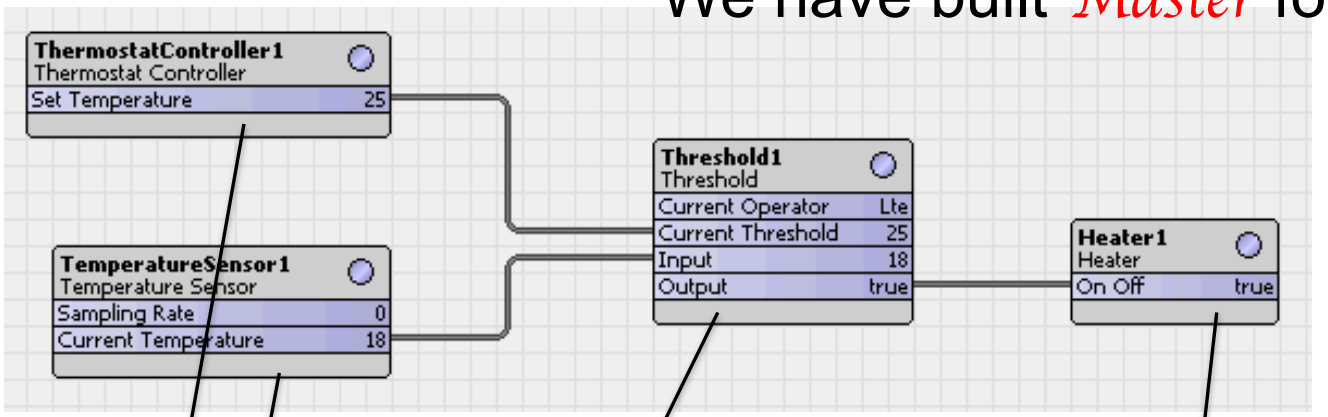
Targets	*Class (definition)*	*object (instantiation)*	*Invocation (trigger)*
Programs (FBP)	**Component**	**Component Instance**	**Link**
Users	Policy	Policy Expression	Periodic, or Critical Component Activation
Sensor Networks	Profile (native, virtual)	Profile Instance (sensor, codes)	Call (local) Message (remote)



Wu-Kong : Master's Perspective

Given logical flows that have been defined, they must be mapped to some physical networks of nodes/sensors for execution.

We have built *Master* for that.



logical

Physical

Candidate1,2

Candidate1,1

WuClass 1

WuClass 2

WuClass 3

WuClass 4

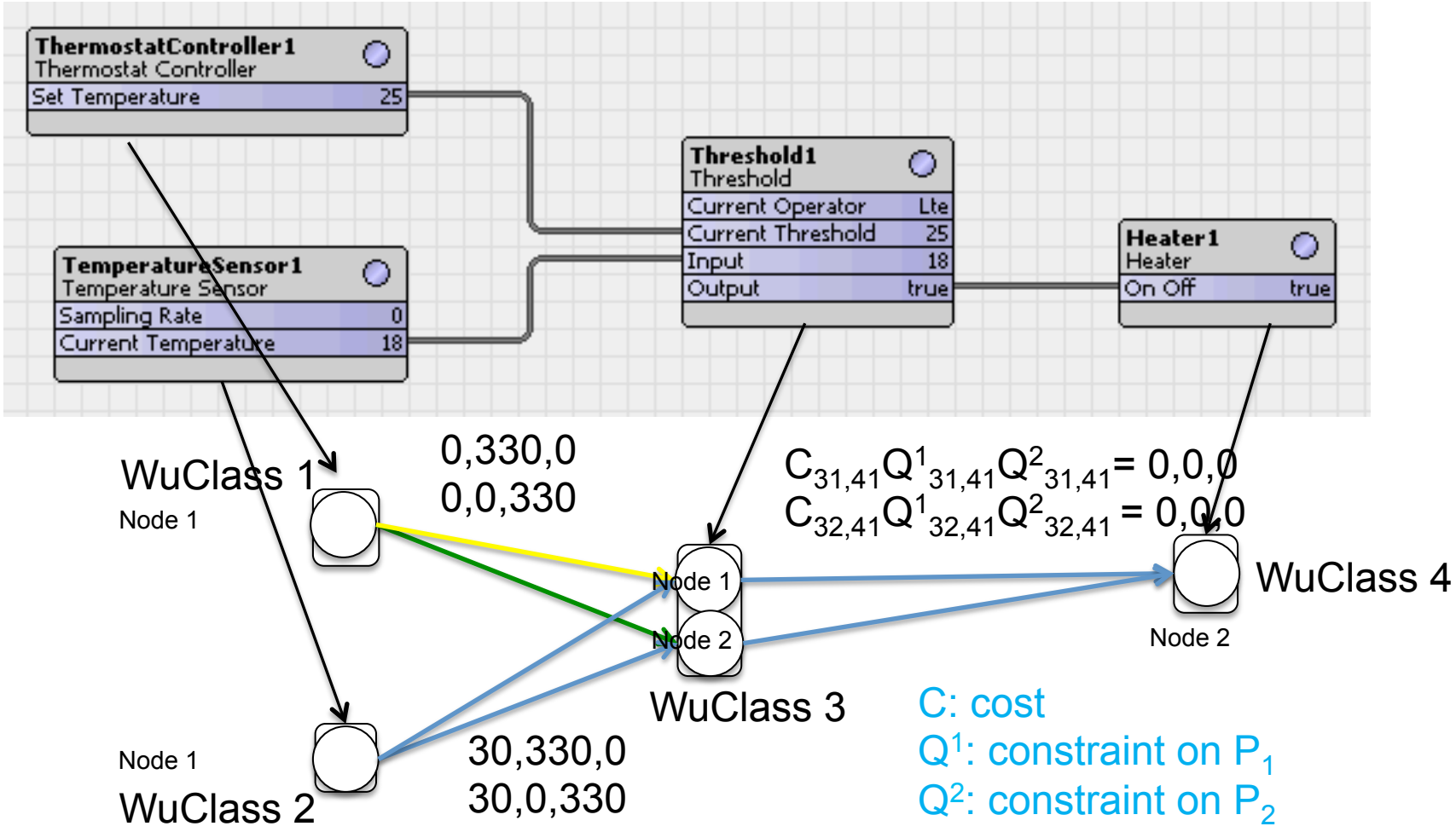
Cost: $C_{31,41}$

Constraints:

$Q^1_{31,41}$, $Q^2_{31,41}$

The Mapping Problem

This can be modeled as *multi-constraints shortest path* (MCSP) problem.



QoS Issues for Mapping

Context (QoS) attributes

- location, distance
- security
- low power
- real time
- reliability
- mutual dependency among context attributes

- Use policy language design
- Hierarchical mapper, distributed mapper
- Reconfiguration, fault-detection

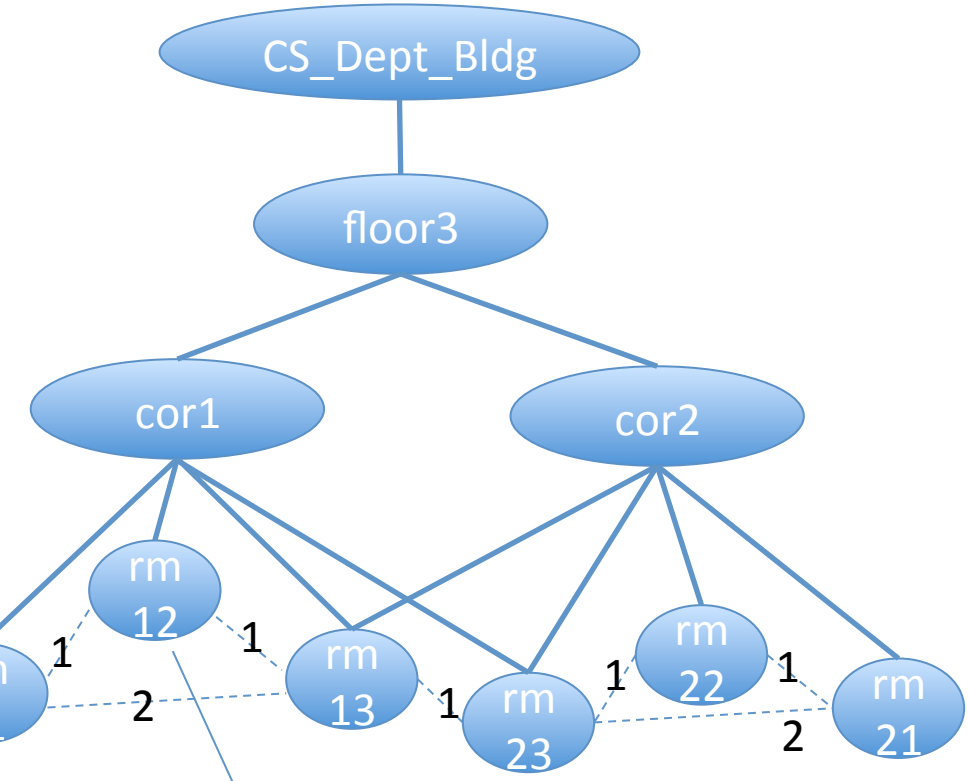
Policy Framework

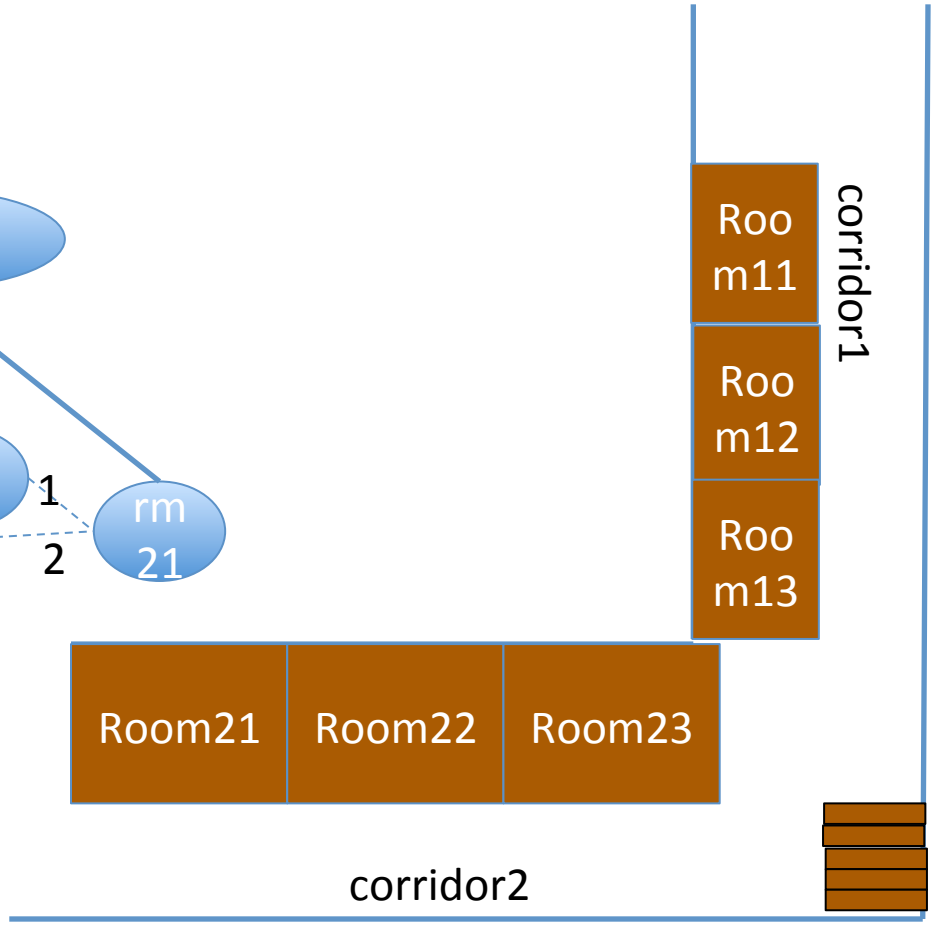
- High level specification for M2M management
  - User or app defines an intuitive objective statement
  - Policy interpreter and configuration engine produce the detailed setup for target systems, enabling higher-level thinking/coding
  - By specifying policies declaratively and independent of actual devices, it is possible to change the behavior on-the-fly for better flexibility.
- Related work: Security (SPF), OS Policy, Ponder/Ponder2 (2001/2006), DSN (SenSys 2007), ADAE (SenSys 2010)
- Configuration and constraint satisfaction engine can take many attributes (e.g. context, fault tolerance, security, trust) into consideration for a better, more optimal performance
- Policy IDE, interpreter and context support are needed

Location Tree

- For modeling objects in physical environment
- Hierarchical Location Tree:
  - Good for imprecise expression of spacial relationship
  - Human readable (efficient for defining human request)
  - Eg. EE-Building/3F/Rm318
- Coordinate Location Tree:
  - Good for global expression of location
  - Machine readable
  - e.g. first-aid @ (10,20,15)
- We can use a hybrid location tree (*CMU 2002*)

Location Example





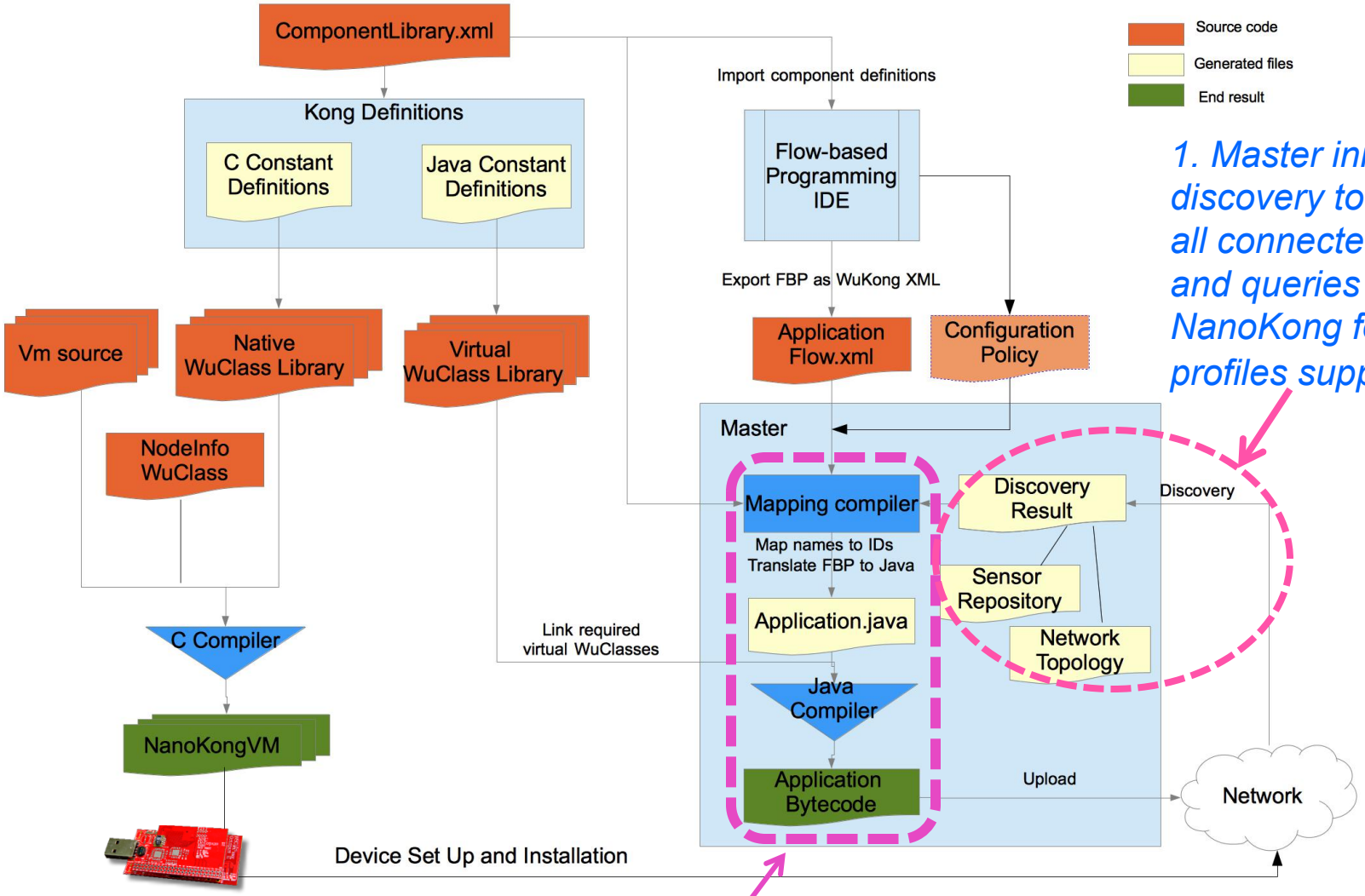
Location Search String

- Describe the conditions for sensor candidate selection
- Example Query String
  - EE_Building/3F/Corridor1/Room318 #near(0,1,2,1)&~near(1,1,3,1)|exact(0)
- First part describes location in hierarchical location tree
- Second part describes range of selection using coordinates
  - Function term:
    - near(0,1,2,1) , near(1,1,3,1), exact(0)
    - A set of sensor nodes is returned for every function
  - Connector:
    - &, ~ ,| with precedence: '~' > '&' > '|'
    - Parsed as disjunctive normal form (DNF), so that many combinations can be expressed

Distance

- From the relationships among nodes in one room, we define the "distance" from one sensor node to the sensor node in another room.
  - we need to model the relationship between rooms with user-defined distance value across rooms and floors
- For hierarchical expression, add edges between nodes in the same layer.
  - For example, in order to express the relationships among the rooms on the same hallway, we add edges between any two rooms on that hallway, whose sensor nodes can connect to each other.
  - Since areas can overlap with one another, a sensor node may belong to more than one areas.

WuKong Workflow 5.0



1. Master initiates the discovery to look for all connected devices and queries their NanoKong for the profiles supported.

2. Master uses mapping algorithms to selects different devices for a FBP, and upload the codes

Wu-Kong : Project's Perspective

1. NanoKong: Build intelligent sensor devices
   1. Sensor classes (*profiles*) are selected for each NanoKong
   2. JVM and device networking support are also included
2. FBP: Build IDE for composing flow-based programs
   1. Program components are selected from a component library and linked together
   2. IDE is used to build an application flow structure and export in XML
3. Mapper: Build an intelligent and working M2M Master
   1. User sends the flow XML to Master
   2. Master discovers sensors remotely, maps them to flow components, and deploys the codes to all sensor nodes

Foundation and Related Areas

- Our research is to design systems that can sense broadly, support proactively and sustain intelligently
- We are studying hardware, software, networking, and computing models for autonomous applications that are deployed on connected sensing and computing devices with interactive context.
- It is built on the foundation developed by many related research areas:
  - WSN provides innovative sensing networks
  - Pervasive computing builds intelligent things and scenario
  - IoT identifies subject status and global perspective
  - CPS seamlessly integrates cyber and physical action models
  - M2M requires intelligent and scalable system management

Final Thought

- Are we trying to simplify the configuration of smart sensor hardware/applications or to build a new future IoT/M2M programming paradigm?
- If former, we need to show its compatibility with existing sensor apps/hardware
  1. The contribution is to generalize/abstract existing designs/protocols
  2. This is like designing the virtualization layer that must show how existing M2M's can run on it.
- If latter, we want to show how powerful it is to build future applications, and compare the pros and cons

How to Measure the Success of Project

- Deployment
  - The flexibility for checking and relocating sensor nodes
- Reprogramming
  - The time taken to physically retrieve, reprogram, and redeploy devices
- Fault-Tolerance
  - The time/cost for identifying and replacing faulty sensors and network
- Security
  - The impact from unauthorized sensor access and reporting fake data
- Ease of Programming
  - Simplicity of application development (coding effort), smaller code size (communication delay and energy saving), ease of re-programming (adaptivity and flexibility), high-level thinking (policy expression)

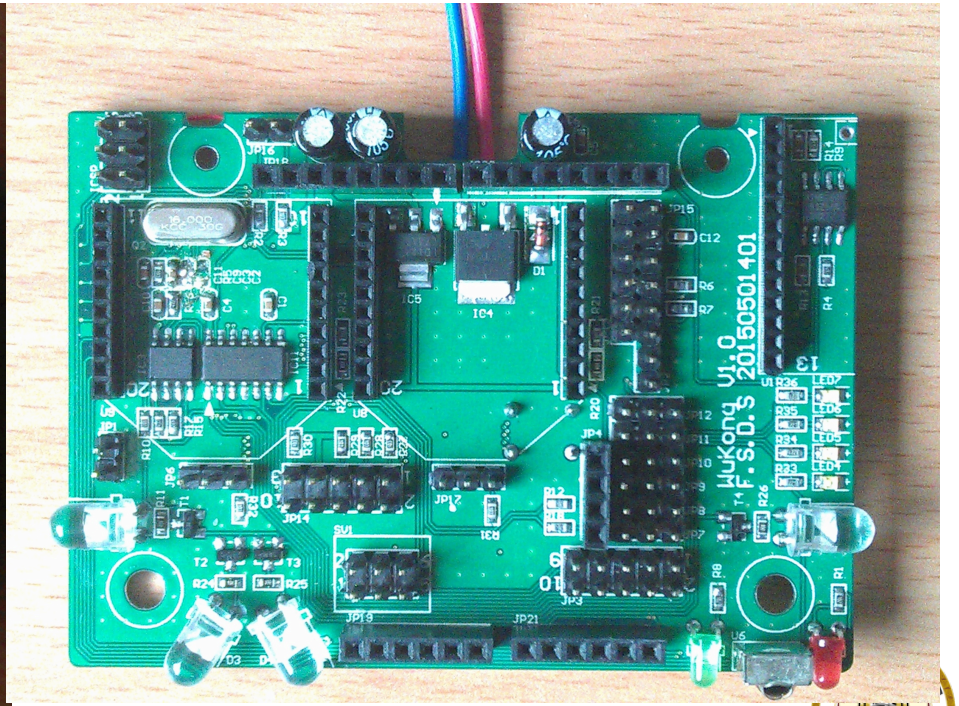
Wu-Kong : Innovation Perspective

- We hope to turn sensor-based system engineering paradigm upside down.
  - Not hardware first, software second;
  - We want software -> hardware

One More Thing …

Intel-NTU Center

- We have built our own sensor platform, WuDevice, with ATMega 2560, 32KB EPROM, Zwave, Zigbee, WiFi, Bluetooth, IR, 3 digital I/O and 3 analog I/O.
- Interested? Let us know.




THANK


YOU