Groovy Goes RFID With Smart Sensors for Real-World Control

Jim Clarke

Chief RFID Architect
Sun Microsystems

Jim Wright

Senior Staff Engineer
Sun Microsystems

Bruce Boyes

Principal
Systronix

TS-5386

Groovy Goes RFID with Smart Sensors

For real world control

Learn how to use Groovy scripting to integrate identity and sensor data through the Sun Java™ System RFID middleware software and intelligently secure a room and its contents.

Agenda

What Will We Build?

Edge Computing and Smart Sensors

Architecture for Networked RFID

Smart Devices

Groovy Script Walkthrough

What It All Means

Demo

What Will We Build?

Groovy-scripted response to humans removing controlled objects from a room

- Controlled objects will have unique identities
  - RFID-tagged
- Human intrusion will be detected
  - Detecting body heat
- Humans will be credentialed
  - ID badge
- Action taken if object is removed illicitly
  - Strobe light

Edge Computing and Smart Sensors

Sensing and controlling things in the real world

> “In designing a pervasive sensor network to interact with the physical world, the heterogeneity of the transducers and the physical environment pose significant challenges to the system architect.”
>
> \- http://jddac.dev.java.net

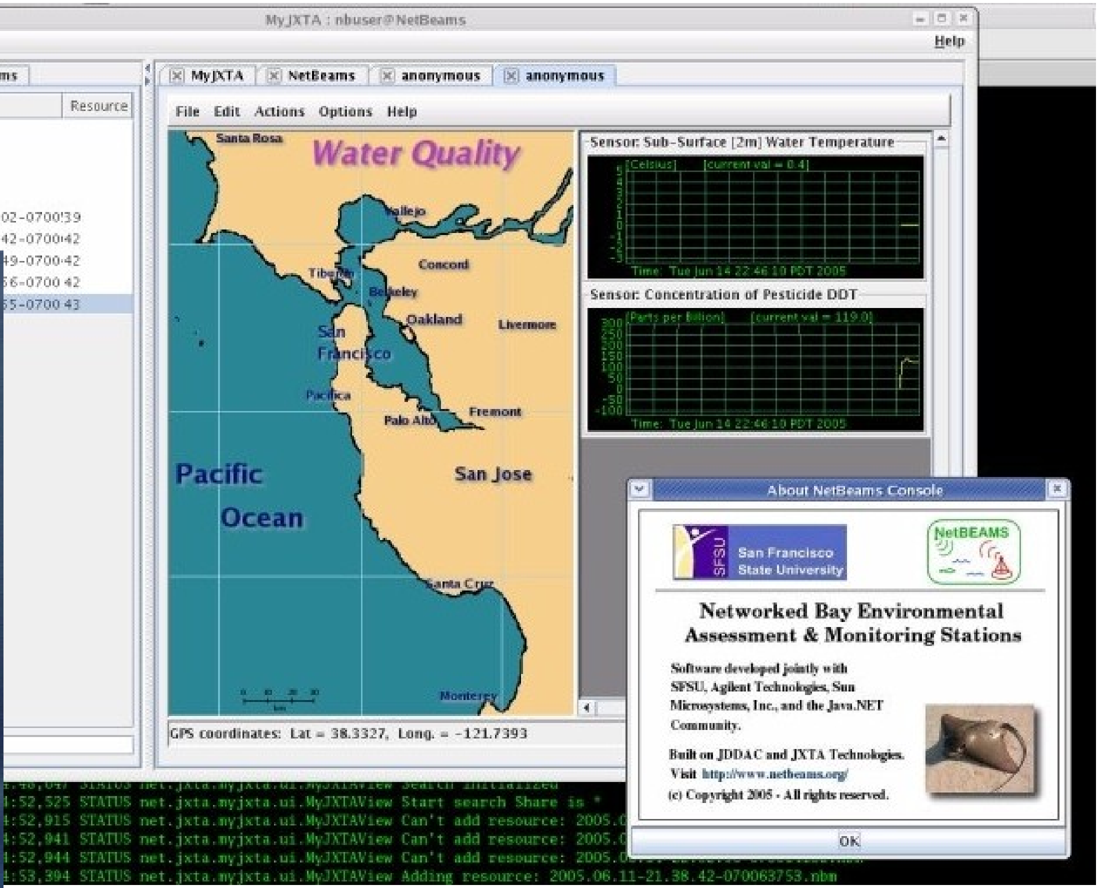
JDDAC—Making Sensors Smart

Distributed Data Acquisition Control for the Java^{TM} platform (JDDAC)

- Open source Java network transducer software
    - Sensors and actuators
    - Self-describing devices and measurements
    - Plug and Play sensor integration
- Based on IEEE 1451 and IEEE 1588 standards
    - NIST-supported
- Brings real-world control to Java^{TM} EE
    - Via TCP/IP, Cellular

Sensing Over Any Network

e.g., Oceanographic data over cellular wireless



Sensor: Sub-Surface [2m] Water Temperature

[Celsius] [current val = 0.4]

Time: Tue Jun 14 22:46:10 PDT 2005

Sensor: Concentration of Pesticide DDT

[Parts per Billion] [current val = 119.0]

Time: Tue Jun 14 22:46:10 PDT 2005

GPS coordinates: Lat = 38.3327, Long. = -121.7393

Networked Bay Environmental
Assessment & Monitoring Stations

Software developed jointly with SFSU, Agilent Technologies, Sun Microsystems, Inc., and the Java.NET Community.

Built on JDDAC and JXTA Technologies.
Visit http://www.netbeams.org/
(c) Copyright 2005 - All rights reserved.

```
1:52,525 STATUS net.jxta.myjxta.ui.MyJXTAView Start search Share is *
4:52,915 STATUS net.jxta.myjxta.ui.MyJXTAView Can't add resource: 2005.0
4:52,941 STATUS net.jxta.myjxta.ui.MyJXTAView Can't add resource: 2005.0
1:52,944 STATUS net.jxta.myjxta.ui.MyJXTAView Can't add resource: 2005.0
4:53,394 STATUS net.jxta.myjxta.ui.MyJXTAView Adding resource: 2005.06.11-21.38.42-070063753.nbm
```

Self-describing Transducers

Abstraction via machine-readable metadata

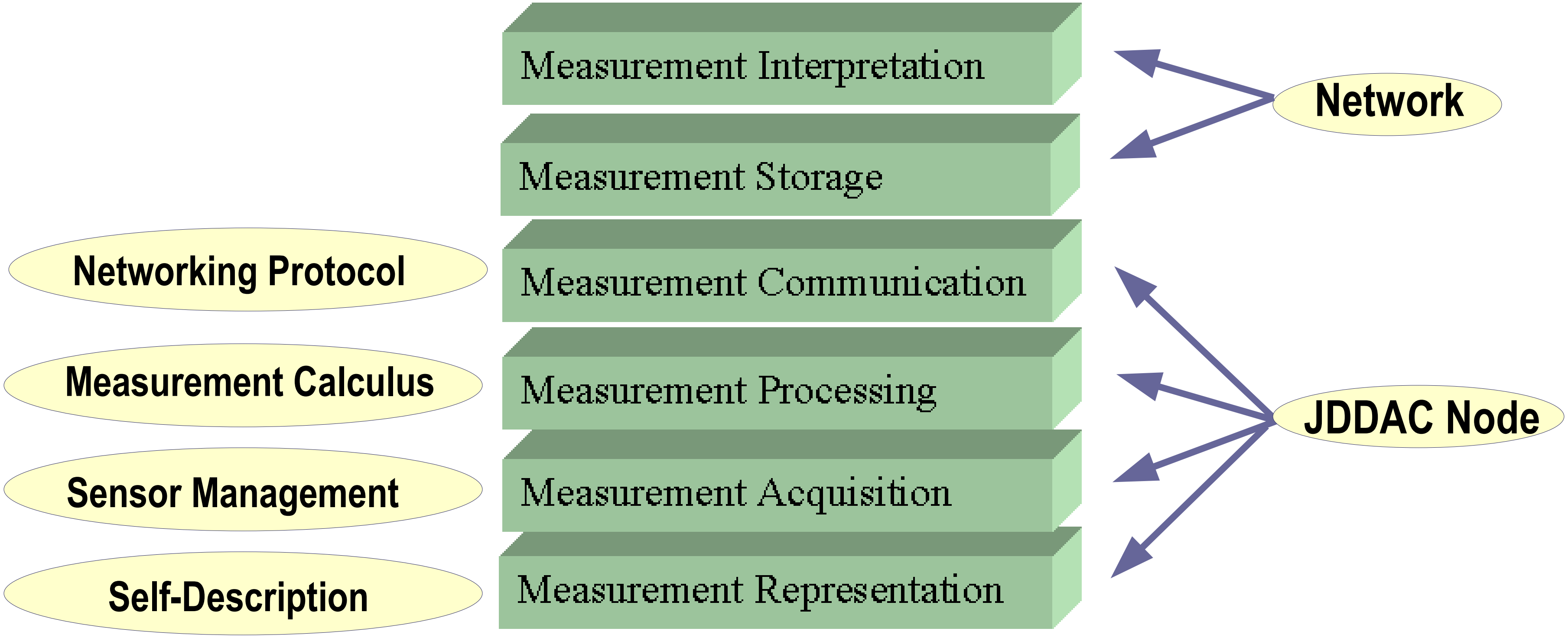
- “Plug and play” for analog transducers and their data
- Information needed to identify, characterize, interface, and use the signal
- Embedded in the sensor or hosting device

Basic Area	Manufacturer ID	Sensotec
	Model Number	41
	Serial Number	462992
	Version Letter	53e
Standard and Extended Areas	Calibration Date	04/22/02
	Temperature effect on span	0
	Temperature effect on offset	0
	Min Operating Temperature	-53
	Max Operating Temperature	121
	Response Time	0
	Min Electrical Output	-2
	Max Electrical Output	2
	Sensitivity	2
	Bridge Impedance	350
	Excitation Nominal	10
	Excitation Maximum	15
	Excitation Minimum	3
	Max Current Draw	30
User Area	Sensor Location	23 right dyno
	Calibration Due Date	04/21/03
Templates	Special Calibration Data	0.04
	Wiring Code	Wiring Code #15

Defined by IEEE 1451

Anatomy of the Measurement Process

Representation, acquisition, processing, communication



Need a data flow-oriented computation and communication framework…

JDDAC Components

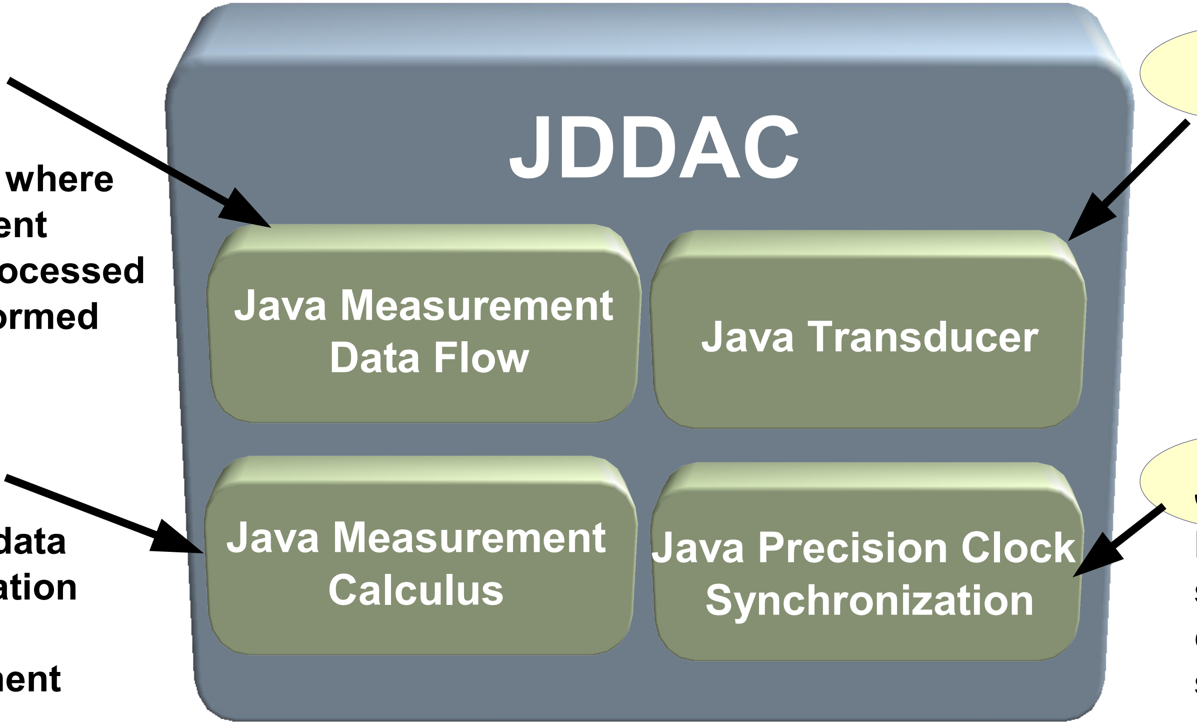
Java components to build transducer nodes

JMDI

Data flow framework where measurement data are processed and transformed

JMCI

Common data representation and a measurement calculus



JDDAC

Java Measurement Data Flow

Java Transducer

Java Measurement Calculus

Java Precision Clock Synchronization

JTI

Transducer objects and an electronic data sheet to characterize transducers

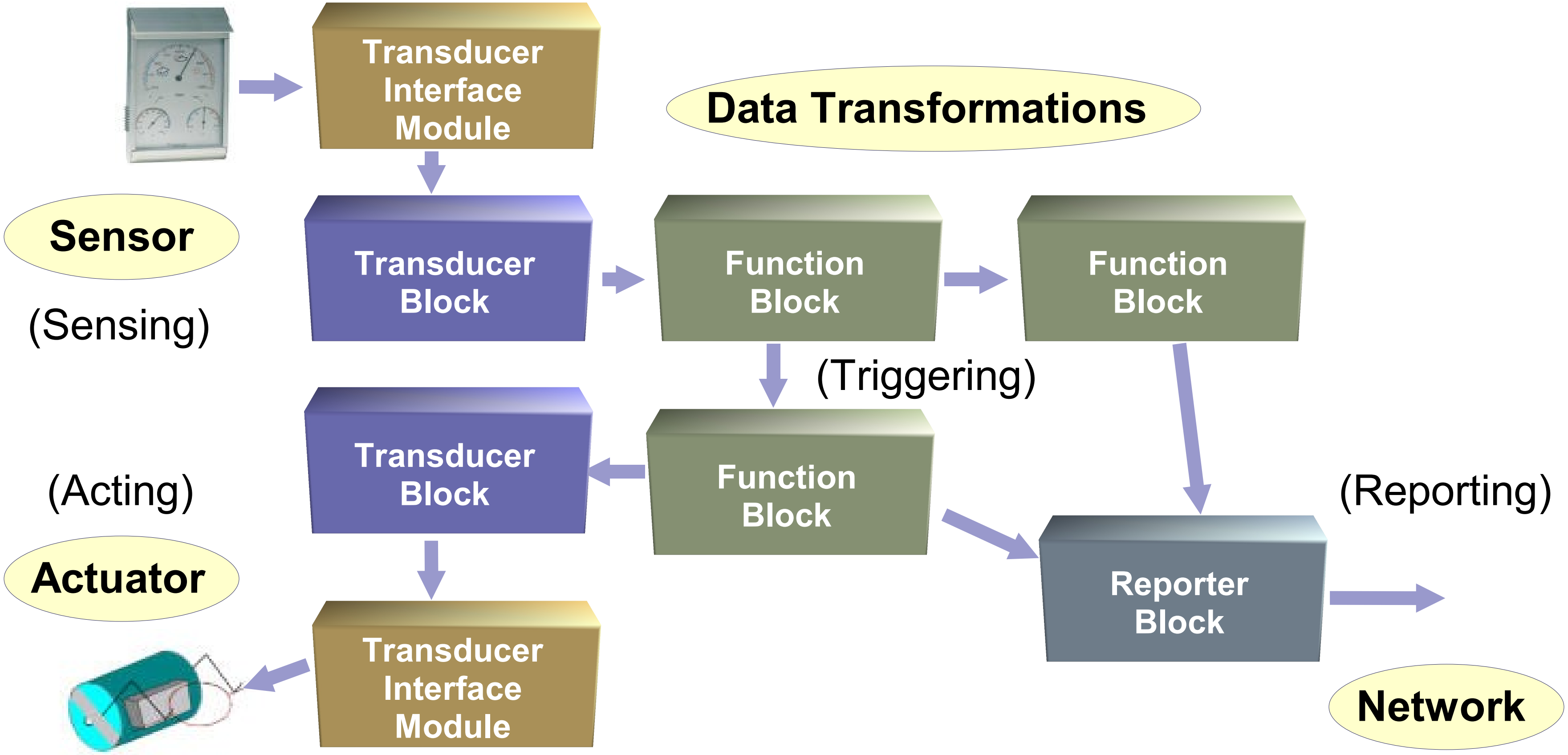
JCPSI

Manage and utilize synchronized clocks in a distributed system (IEEE 1588)

Runs in Java™ ME CLDC and Java™ SE

Putting It Together at the JDDAC Probe

Data flow, transformations and reporting



Architecture for Networked RFID

Knowing the identity of things

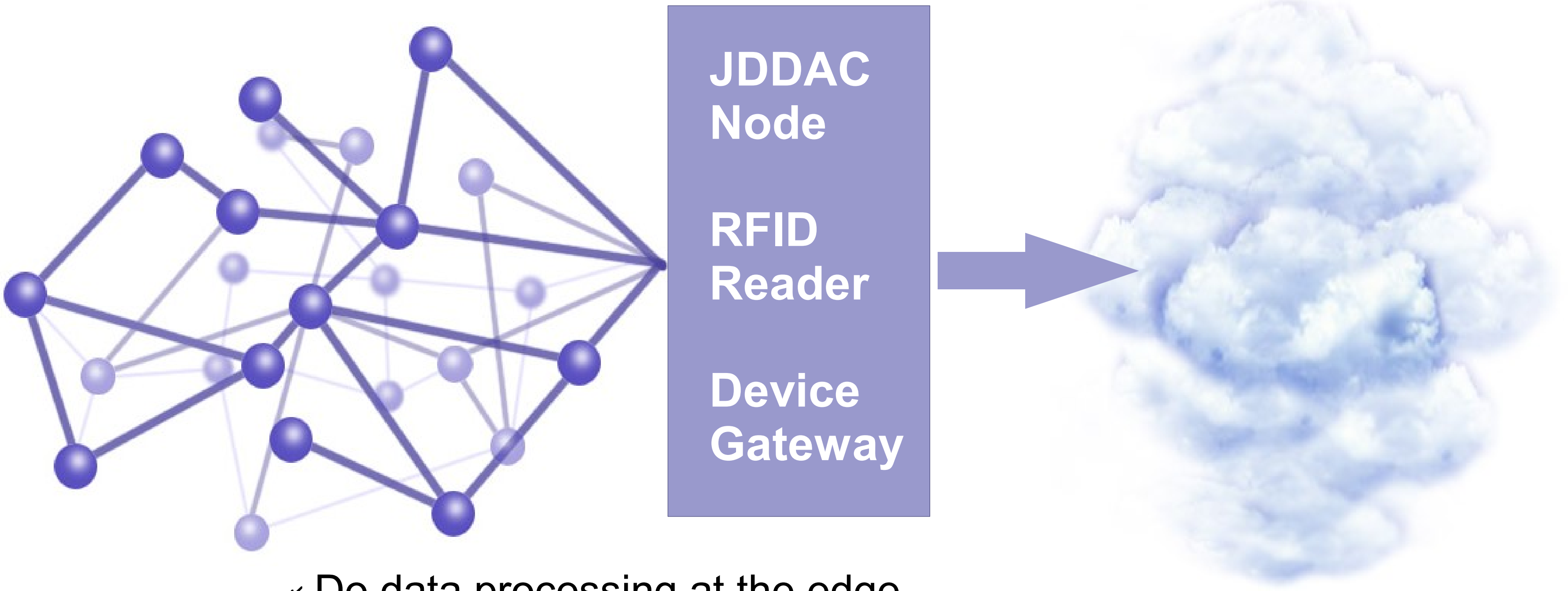
Sensor networks tell you about the properties of the world and enable you to influence them. RFID helps keep track of the unique identity of objects so that you know which ones you sense and manipulate. There are a ***lot*** of things in the world...

Building Scalable Device Networks

Things (Sub-IP)
RFID tag, sensor, actuator

Devices (IP)

The Network



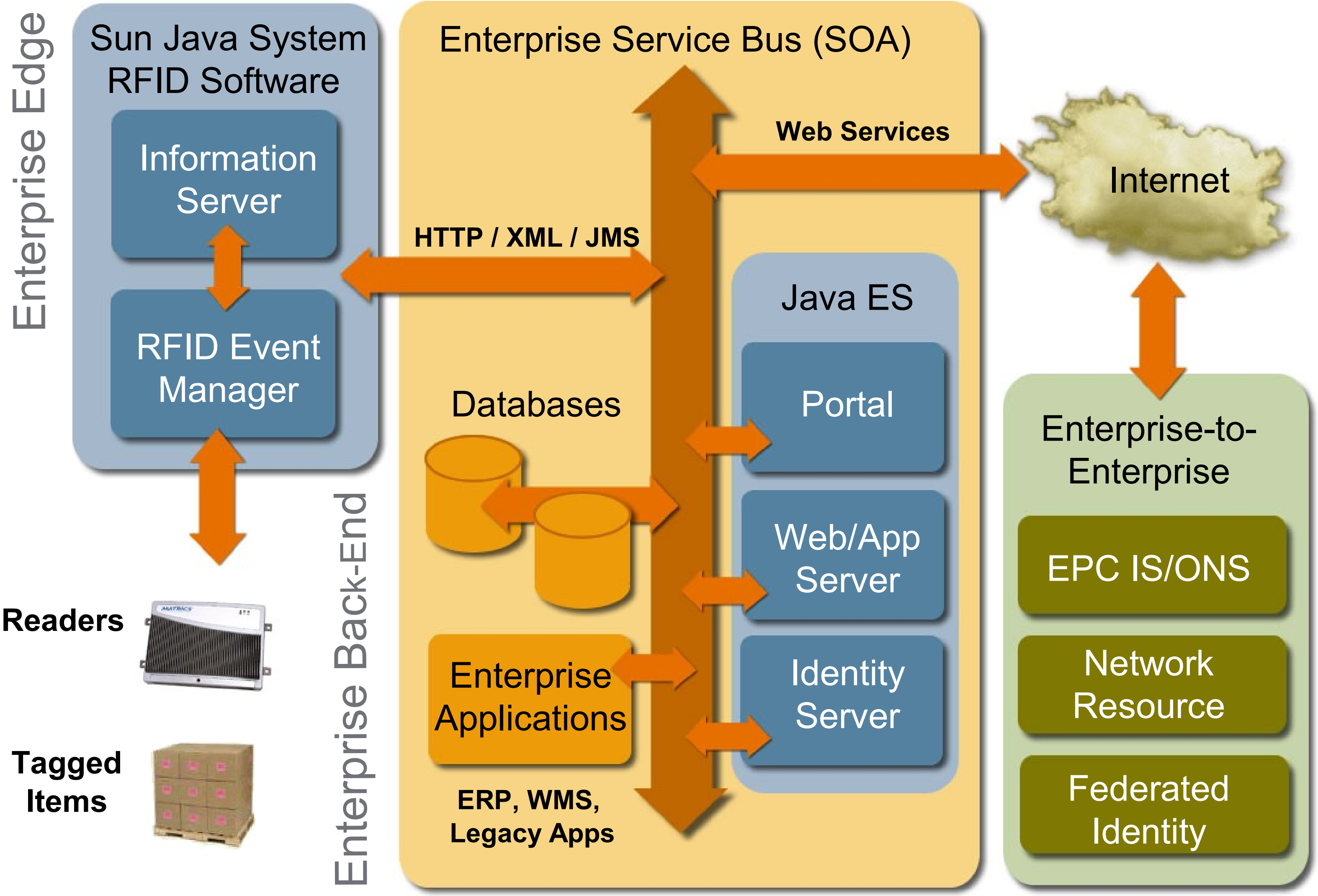
JDDAC Node

RFID Reader

Device Gateway

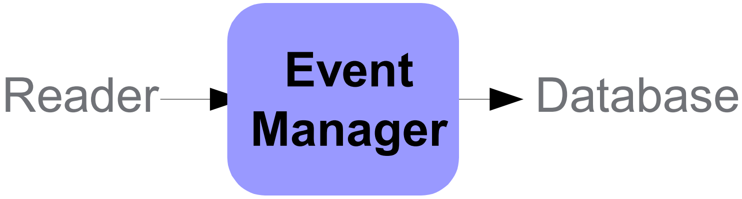
- Do data processing at the edge
- Keep data locally and sync when required
- Use self-assembling networks
- Use self-describing or standards-based data types
- Support redundancy and avoid single points of failure

Networking RFID Edge Devices

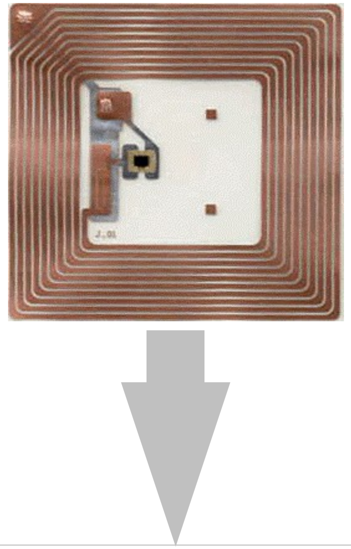


Managing RFID Events

Sun Java System RFID Software Event Manager



- Purpose: event data collector
- Supports EPC Gen 2, ISO, other tags (active/passive) and sensors
- Self-healing, fault tolerant, automatic
- Central monitoring console, remotely administrable (JMXTM, SNMP)

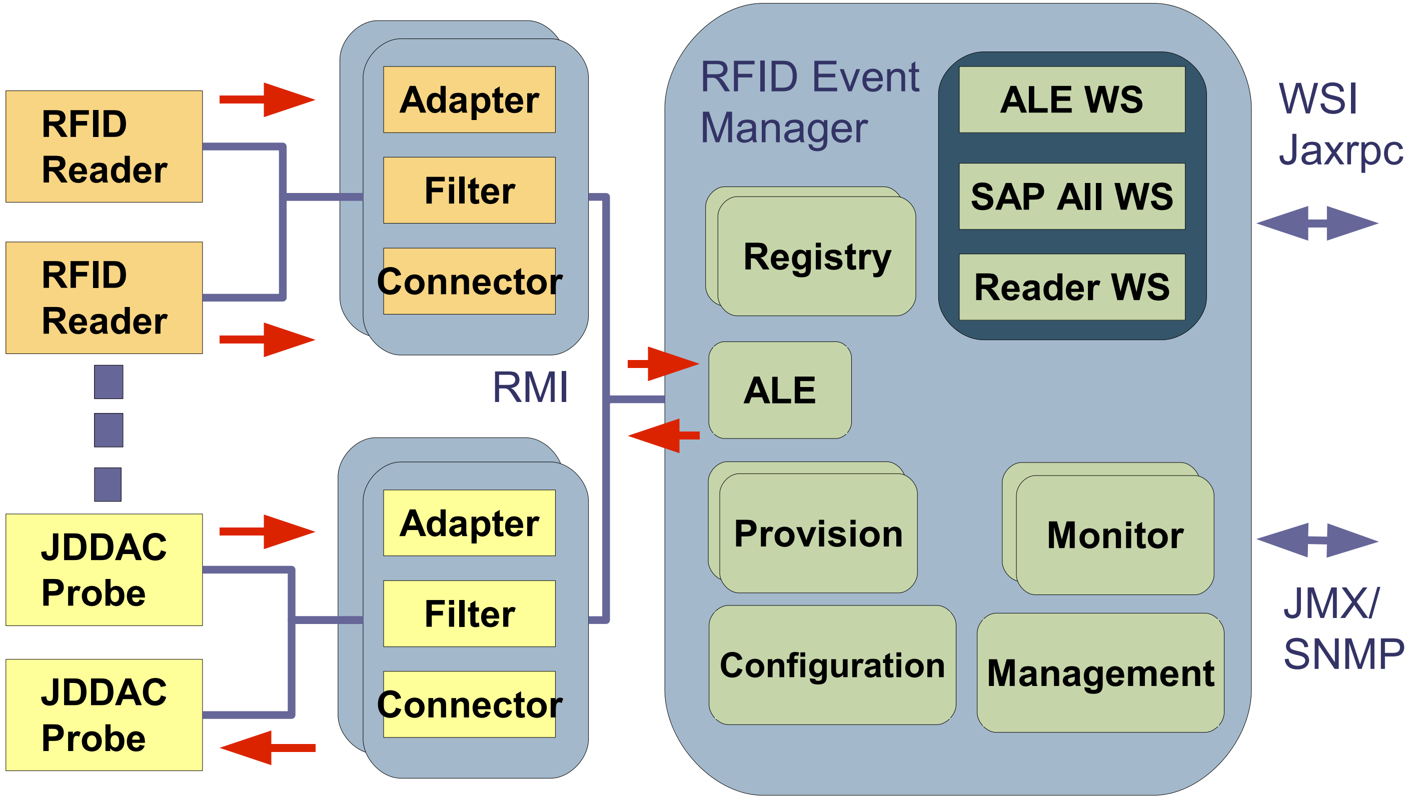


01.0000A89.00016F.000169DC0

Numbering Schemes (EPC)

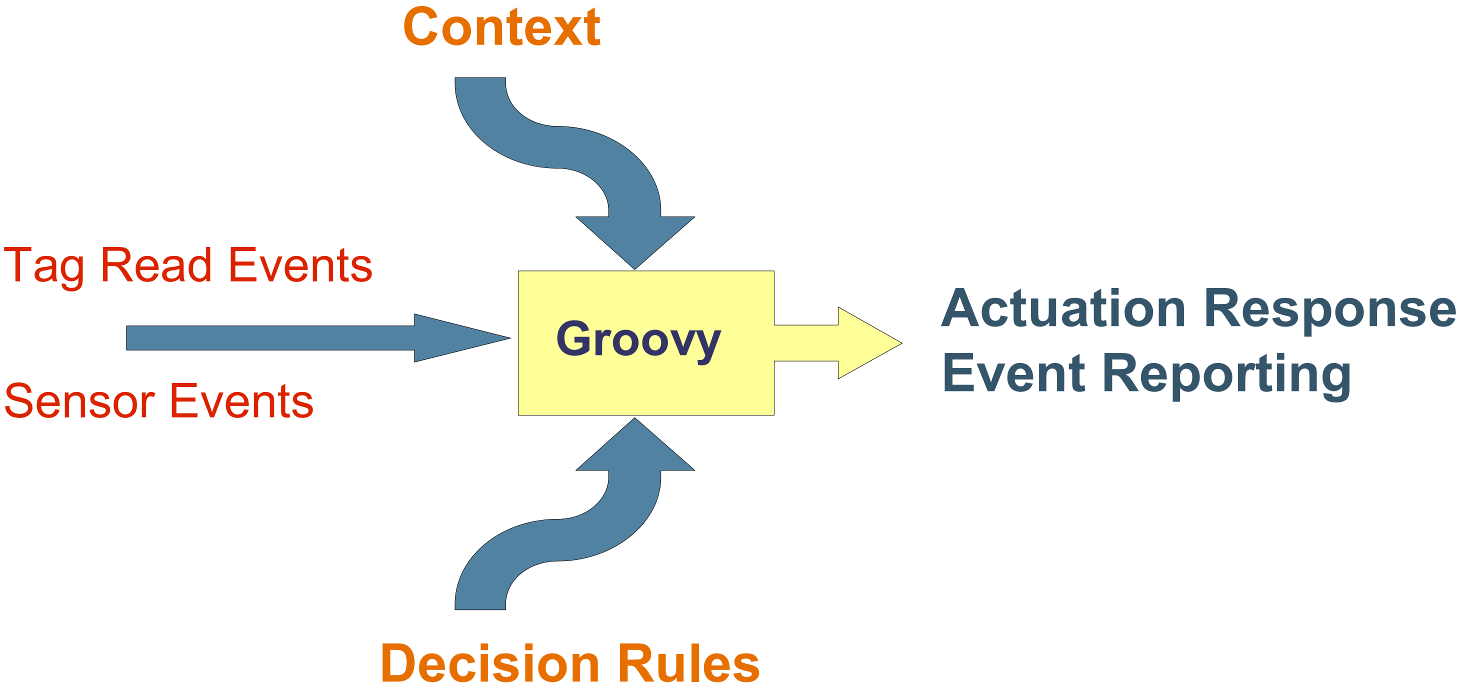
Integrating JDDAC Sensors with RFID

Managing RFID readers and transducers



Needed: Flexible Decision Making

Scriptable, rule-governed actions to manipulate things



Context

Tag Read Events

Sensor Events

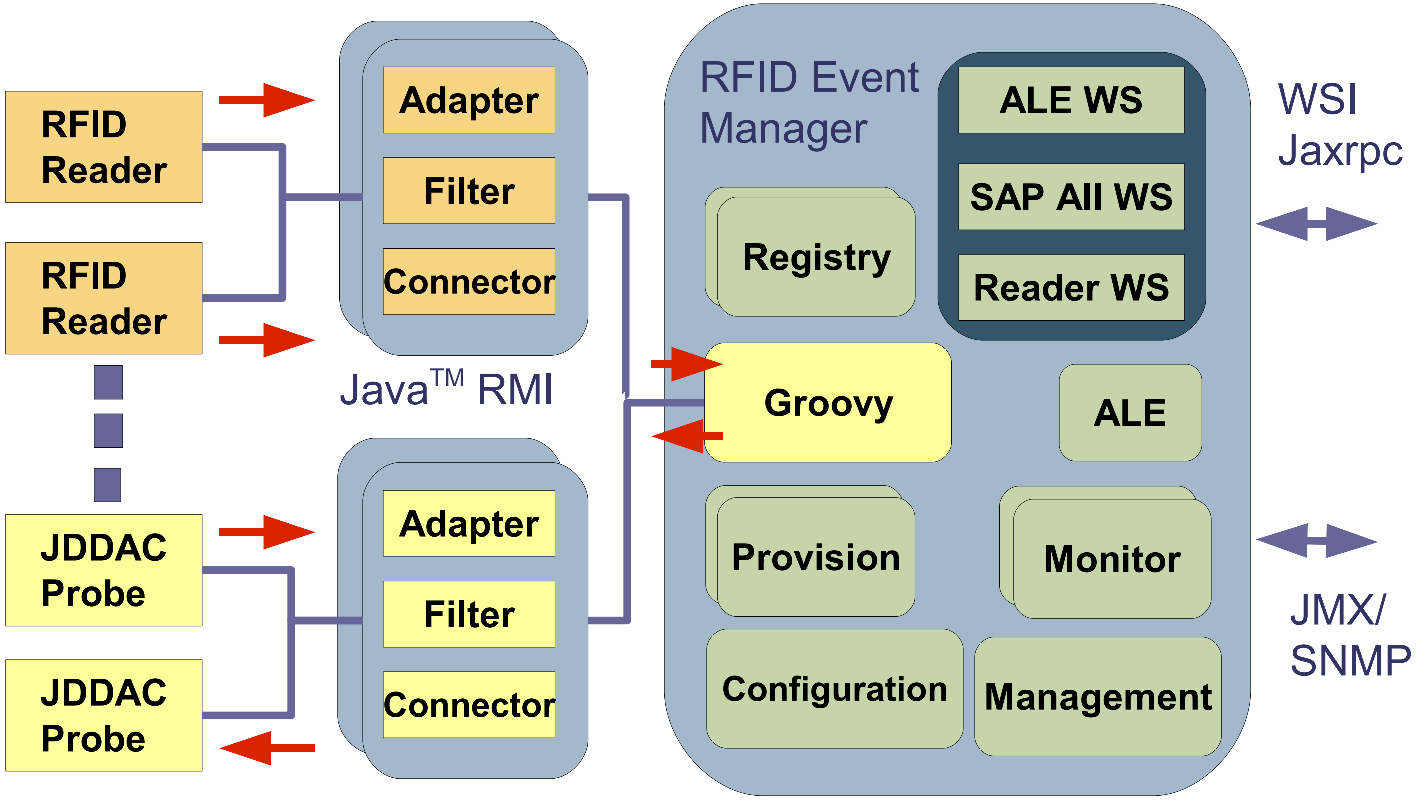
Groovy

Actuation Response
Event Reporting

Decision Rules

Integrated Scripting with Groovy

Rules for interacting with the world



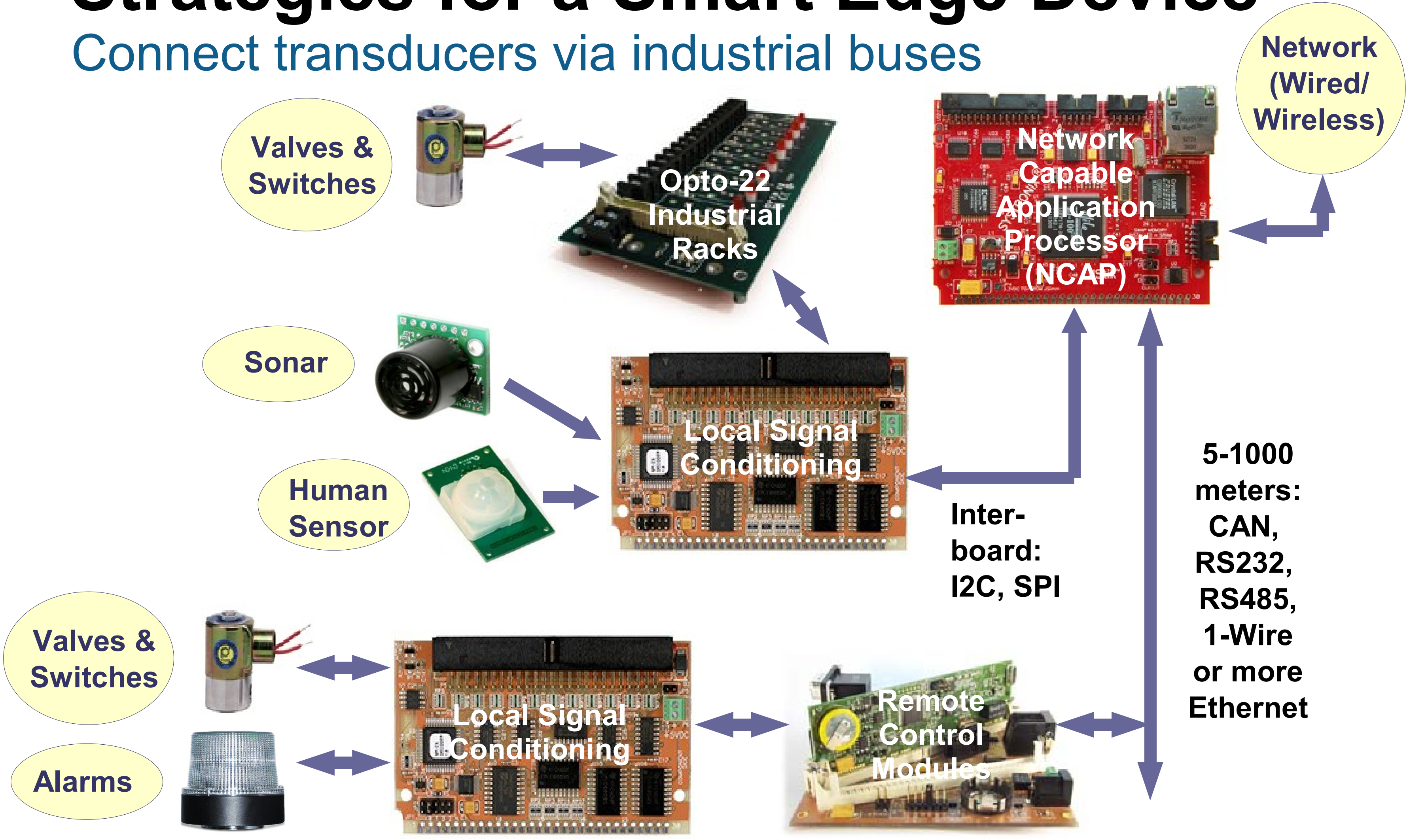
Smart Edge Devices

Where monitoring and control meet the “real world”

Embedded Java based controllers make it practical to carry the benefits of Java technology into physically tiny, power-efficient, low-cost systems. The embedded modules presented here are about an order of magnitude more power efficient than typical “embedded PCs”

Strategies for a Smart Edge Device

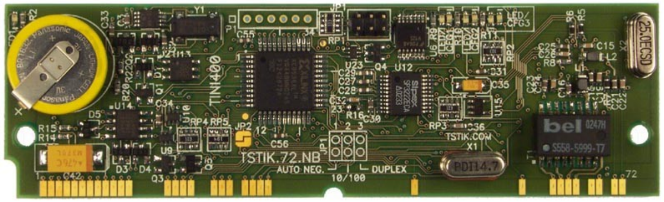
Connect transducers via industrial buses

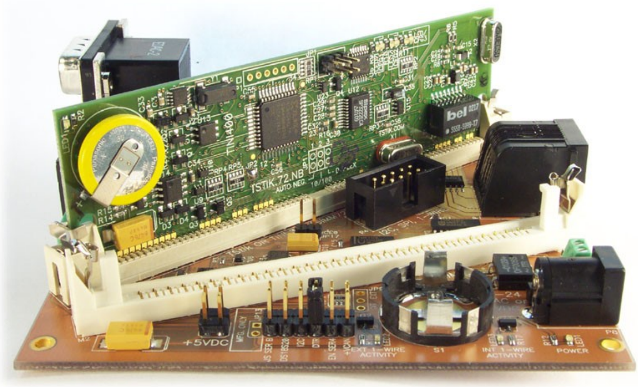

Valves & Switches
Opto-22 Industrial Racks
Network Capable Application Processor (NCAP)
Network (Wired/ Wireless)
Sonar
Human Sensor
Local Signal Conditioning
Inter-board: I2C, SPI
5-1000 meters: CAN, RS232, RS485, 1-Wire or more Ethernet
Valves & Switches
Alarms
Local Signal Conditioning
Remote Control Modules

JStik—NCAP

- J2ME™/CLDC 1.0
- 10BaseT Ethernet
- Native execution of 20x106 Java byte codes/sec
- 2 MB SRAM, 4 MB flash (8MB in near future)
- HSIO bus bursts to 50 MB/sec, variable speed
- 2 x RS232 serial to 115200 baud
- RTOS and Java VM in silicon, highly deterministic
- 2.65 x 3.00 inches, 265 mA at 3.3V, ~US$300

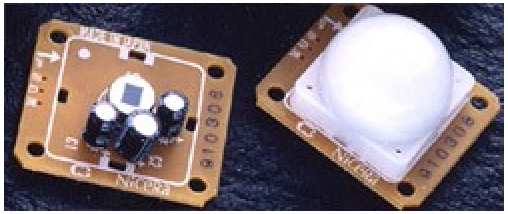
TStik—NCAP

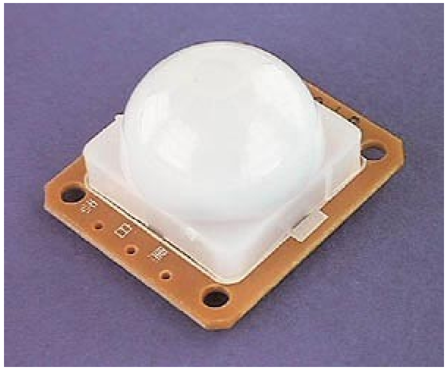




- Low cost 10/100 Ethernet connectivity, based on JDK™ 1.1.8 software
- Dallas DS80C400 running a firmware Java VM
- 1.25” x 4.00”, 5 VDC 200 mA
- 1 MB SRAM, 2 MB Flash
- 3 x UARTs, CAN, I^{2}C, SPI, 1-Wire
- About US$100

Human Sensor





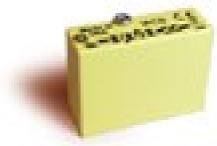
- Nippon Ceramic SGM-5915-1
- 5 volts, 40 uA typical
- Stabilizing time 15 seconds typical
- Open-collector NPN output
- Includes fresnel lens for 5 meter detection +/- 5 degrees, 3 meter +/- 25 degrees
- Will not false trigger on pets, but will false trigger on any large, transient IR signal
- Integrates out ambient signature, so only detects changes in IR signal

Sonar Rangefinders

- Devantech SRF04
  - 33 samples/sec
  - Ranges 5 cm to 3 meters
  - 30 mA at 5V
  - TTL signal width proportional to distance to target

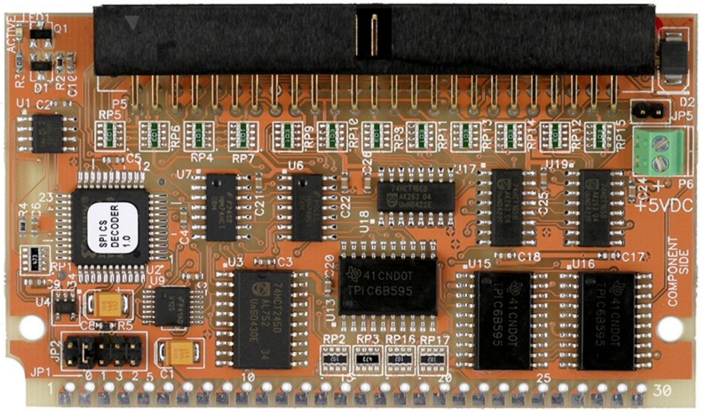
- Maxbotix EZ-1
  - 20 samples/sec
  - Range 15 cm to 6.4 meters
  - 2 mA at 5V
  - Analog, pulse or serial outputs

Opto-22 Compatible I/O Racks



- Several versions and vendors: Dataforth, Opto-22, Grayhill, Potter-Brumfield
- Rugged, opto-isolated modules plug into a backplane with large screw-terminal field wiring
- 50-conductor 0.100” header for I/O
- 8/16/24 channels. 16: Opto-22 G4PB16H, 24: Opto-22 G4PB24
- AC and DC input/output to 240V, 3A
- Plug into Systronix JCX.DIO and other boards

JCX DIO



- 24 open-drain outputs capable of 350 mA at 35V
- Each output is also an input with TTL levels but tolerating at least 35 VDC
- 2.125 x 3.00 inches
- Tagging memory and Java API support
- Opto-22 compatible 25x2 100-mil header
- Wire directly to switches, relays, human sensor

Smart LED Alarm Strobe

- SAE Class I (red)/II—bright!
- Extremely rugged
- 6 Luxeon LEDs, each individually controlled
- 12-24 VDC, 0.2 to 1 A
- Microcontroller for multiple flash patterns, ambient light detection, vehicle reverse detection
- Red, green, blue, white or amber
- Future versions with 802.15.4

Why Groovy for Sensor Networks

Dynamic control to adapt to a changing world

Groovy allows for a robust and flexible way to apply Business Rules closer to the edge.
It enables an adaptive architecture that can quickly respond to new business process requirements

Scripting and the Java Application Environment

Bean Scripting Framework

- BSF—Set of Java classes which provides scripting language support within Java applications
  - http://jakarta.apache.org/bsf/index.html
    - bsf.jar
- Groovy is an agile dynamic language for the Java 2 Platform
  - http://groovy.codehaus.org/
  - groovy-1.0-jsr-05.jar

Enabling Bean Scripting Framework

```
public ScriptController(Properties properties) {
  super(...);
  bsfManager = new BSFManager();
  BSFManager.registerScriptingEngine(
      "groovy",
      "org.codehaus.groovy.bsf.GroovyEngine",
      new String [] {"groovy","gy"});

  bsfManager.setClassLoader(
       Thread.currentThread().getContextClassLoader());

  bsfManager.registerBean("service", this);
  bsfManager.registerBean("devices", this.deviceMap);
```

Enabling Bean Scripting Framework

```
Reader in = new
      InputStreamReader(scriptURL.openStream());
script = IOUtils.getStringFromReader(in);
scriptLanguage =
         BSFManager.getLangFromFilename(
               scriptURL.toString());

public void startup() {
    engine = new AsyncEngine(
               bsfManager,
               scriptLanguage,
               script,
               scriptURL.toString());
    engine.start();
    super.startup();
}
```

Running Bean Scripting Framework

```
public class AsyncEngine extends Thread {
    public AsyncEngine(BSFManager bsfManager, String
language, String script, String source) {
        this.bsfManager = bsfManager;
        ...
    }
    public void run() {
       try {
            bsfManager.exec(language, source, 0, 0,
script);
        }catch(Exception ex) {
           ....
        }
    }
    public void shutdown() {
        bsfManager.terminate();
    }
```

Accessing JavaBeans™ from the Script

```
  bsfManager.registerBean("service", this);
  bsfManager.registerBean("devices", this.deviceMap);

=======================

   // “service” getNextEvent property
   public Event getNextEvent() throws
                            InterruptedException  {
           Event event =  (Event)eventQueue.take();
           logger.info("Take event");
           return event;
       }
```

Accessing JavaBeans from Groovy

```
service = this.bsf.lookupBean("service");
devices = this.bsf.lookupBean("devices");

while ( true ) {
    event = service.nextEvent;
    source = event.source;
    if(source == "Reader") {
           status = devices["Reader"].status;
           println status;
    }
}
```

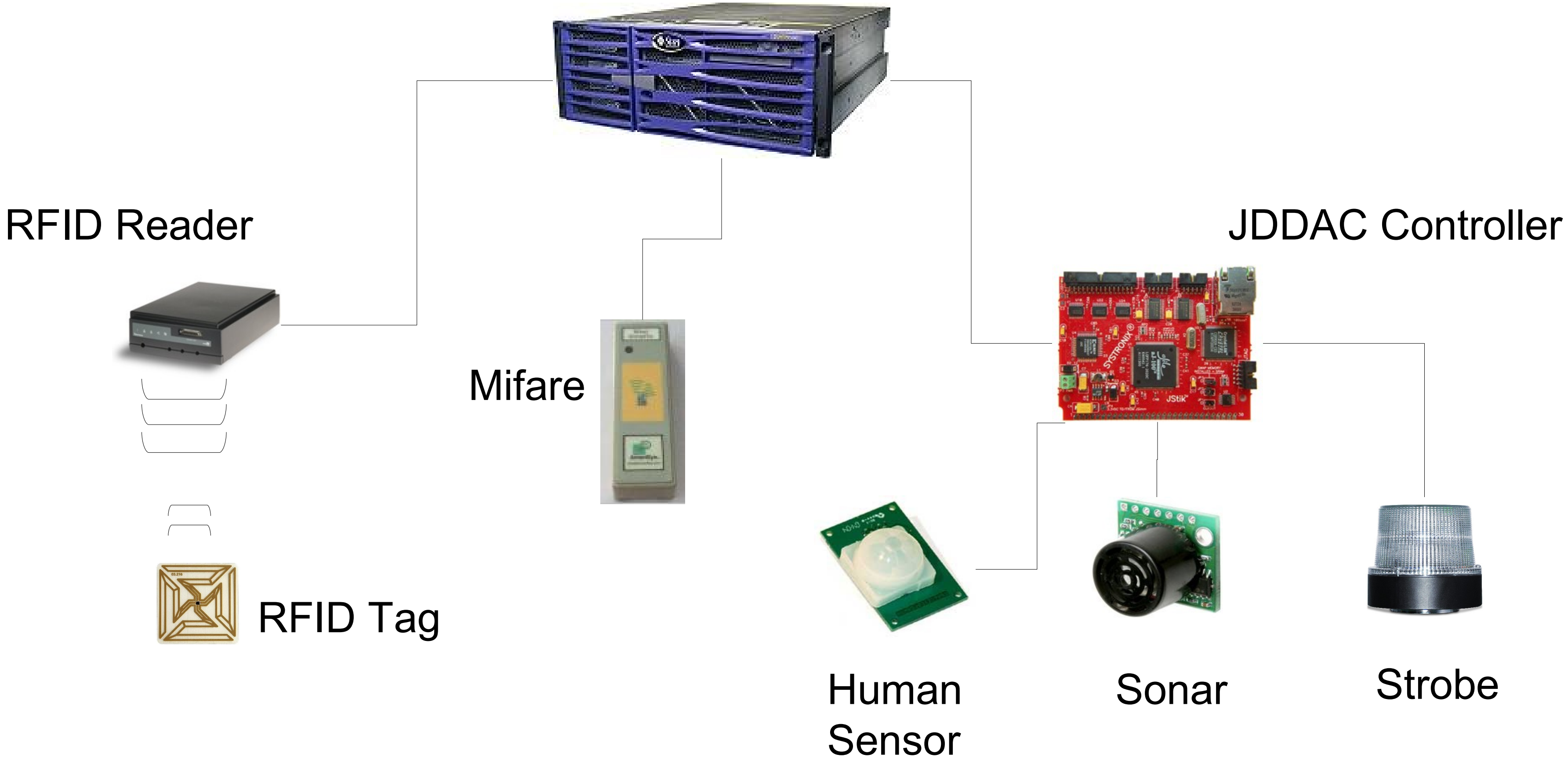
Security with Sensors, RFID and Groovy

USE cases

- #1–Authorized Access
  - Read Badge
  - Sense Human Presence
  - Allow item retrieval
- # 2–Unauthorized Access
  - Sense Human Presence
  - Alarm if item removed

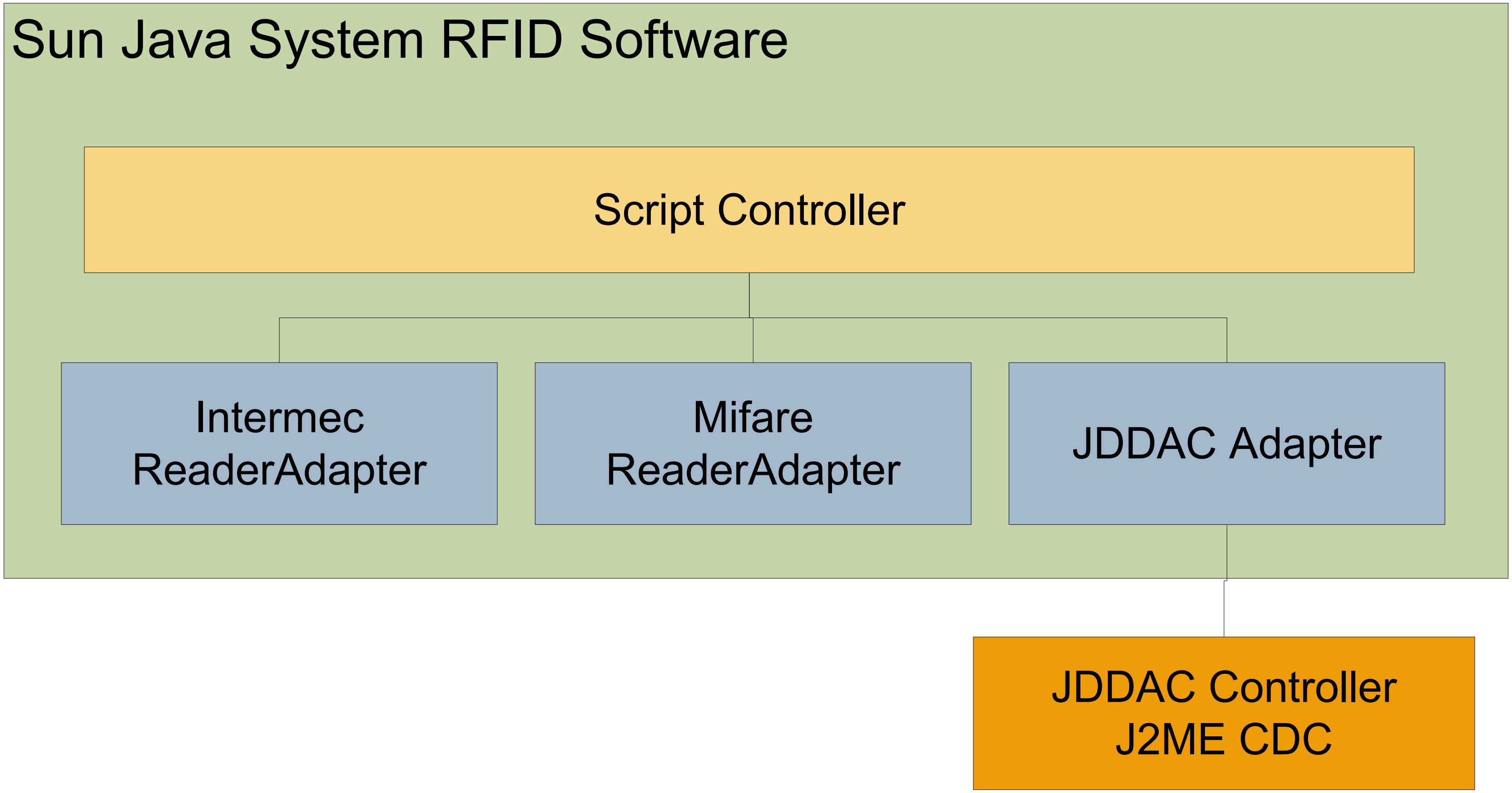
Security with Sensors, RFID and Groovy

Physical architecture



Security with Sensors, RFID and Groovy

Software architecture



What It All Means...

Out there in the real world

Groovy provides a scripting capability that enables flexible control of networked sensors, RFID readers and actuators in a task requiring coordination and decision-making

For More Information

Software and hardware

- Groovy
  - http://groovy.codehaus.org
- Sun Java RFID System network middleware
  - http://sun-rfid.dev.java.net
- Distributed Data Acquisition Control for the Java™ Platform
  - http://jddac.dev.java.net
- Smart edge devices
  - http://www.systronix.com
- Related sessions
  - BOF-0554, BOF-2521, TS-1246, TS-3273, TS-3714

DEMO

Security using Sensors, RFID and Groovy

Q&A

Jim Clarke, Sun Microsystems
Jim Wright, Sun Microsystems
Bruce Boyes, Systronix

Groovy Goes RFID With Smart Sensors for Real-World Control

Jim Clarke

Chief RFID Architect
Sun Microsystems

Jim Wright

Senior Staff Engineer
Sun Microsystems

Bruce Boyes

Principal
Systronix

TS-5386