Applications of OSGi to Embedded Systems

…or How I Think We May Finally Starting to Practice "Engineering" in Software Engineering

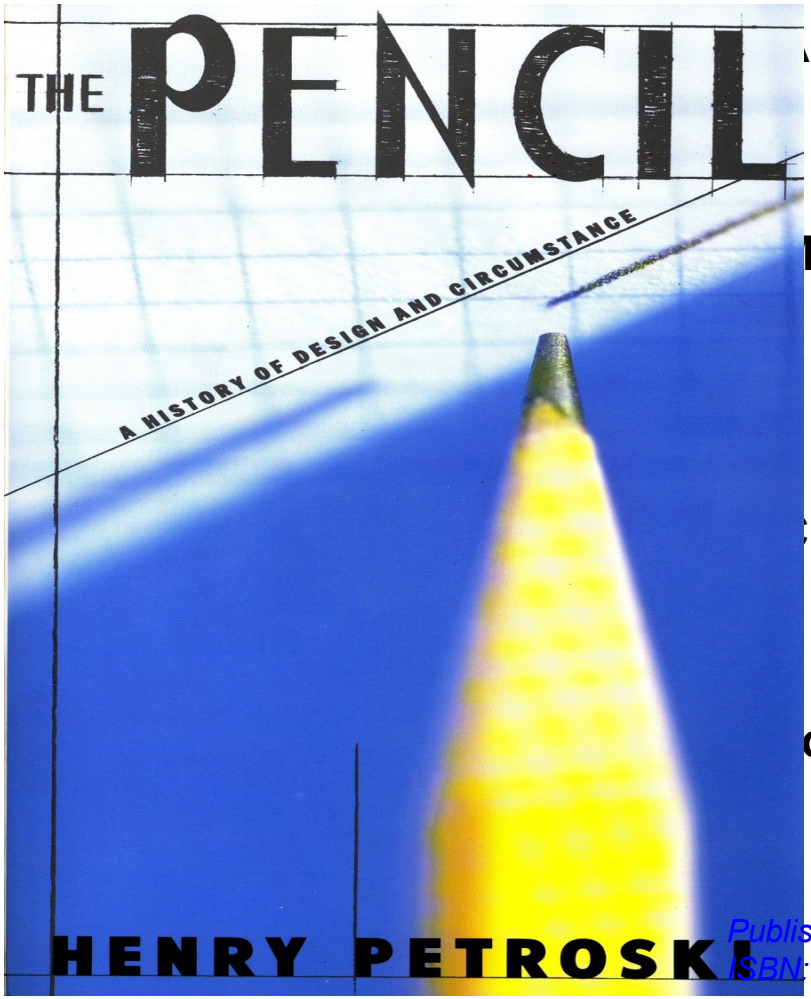
www.bandxi.com

Overview

- **Engineering Is More Than Technology**
  - *Setting a context for the value proposition of components*
- **Open Services Gateway Initiative (OSGi)**
  - *Framework for Software Engineering*
- **Component Engineering with OSGi**
  - *Eclipse Equinox & Service Oriented Bundle Architecture (SOBA)*
- **OSGi Applied in the Embedded Problem Space**
  - *Diagnostics/Prognostics Component Architecture*

Engineering Is More Than Technology

The Pencil: A Story of Engineering Complexity



pplied Science
- *Basic Science & Technology*
- *Applied to Problem Space*

larket Economics
- *Resources*
- *Motivations*
- *Constraints*

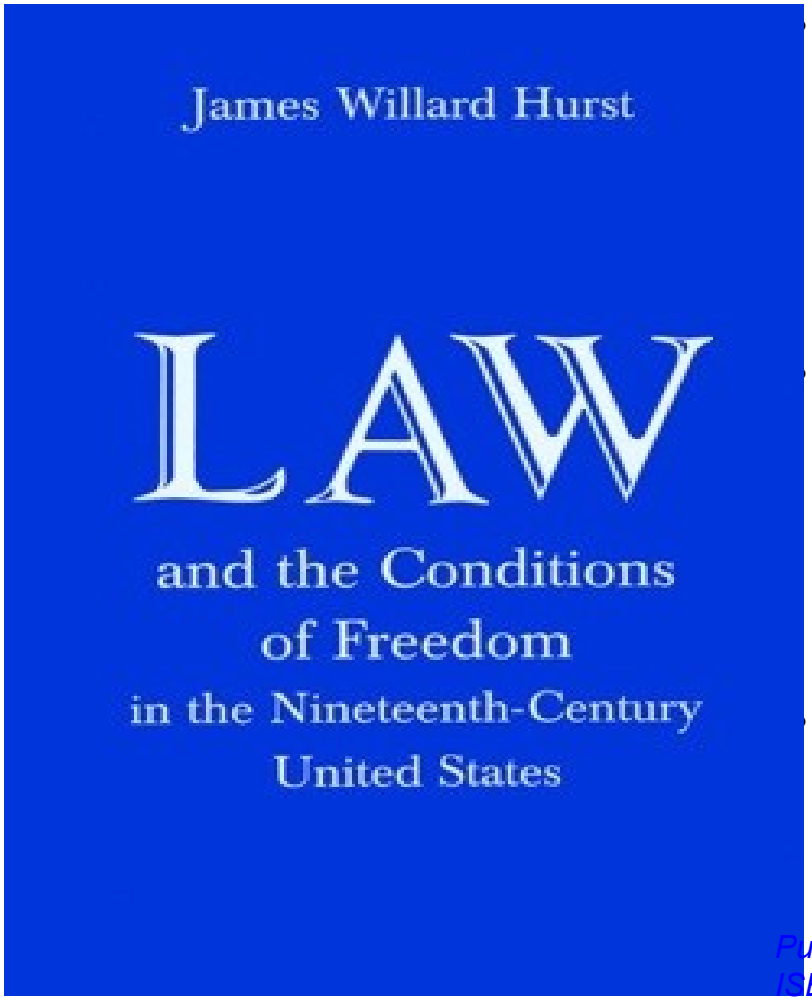
ulture & Politics
- *Opportunities, Priorities & Choices*
- *Modes of Interaction & Behavior*

ooling
- *Innovation Byproducts*
- *Elevated Concerns & Applications*

Publisher: Knopf (November 10, 1992)
ISBN: 978-0679734154

Engineering Is More Than Technology

The Framework: Law & The Conditions of Freedom

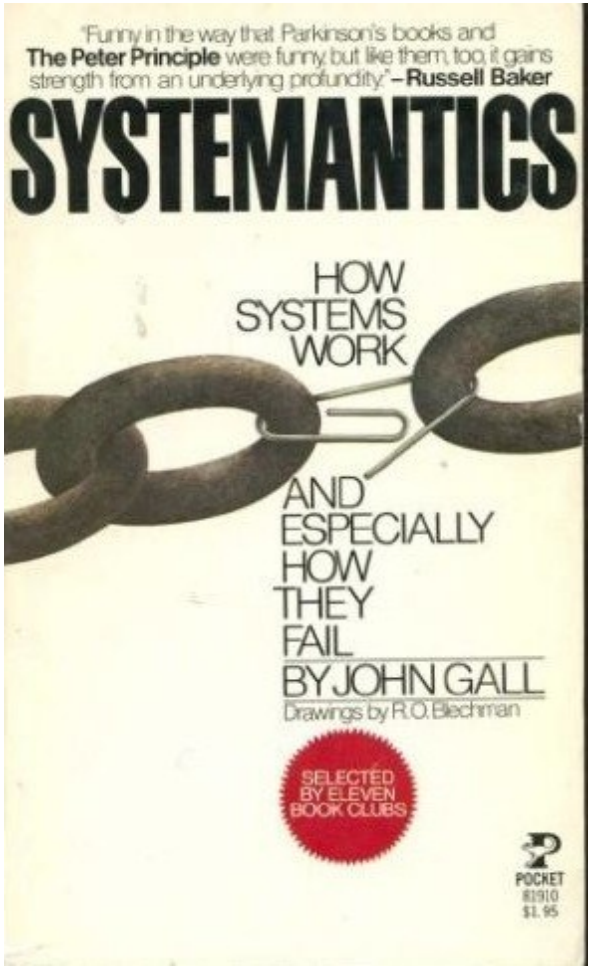


- **Law**
  - *Contracts*
  - *Property & Intellectual Property*
  - *Precedent, Predictability*
- **Freedom**
  - *Opportunity & Risk Management*
  - *Growth & Prosperity*
  - *Innovation & Scaling*
- **Transformation**
  - *Craftsman to Corporation*
  - *Small Volume to Mass Production*

Publisher: University of Wisconsin Press (June 15, 1964)
ISBN: 978-0299013639

Engineering Is More Than Technology

Be Humble When Building Large, Complex Systems



- **Every working large system was once a working small system**
- **A complex system cannot be "made" to work. It either works or it doesn't**
- **Loose systems last longer and work better**
- **The larger the system, the greater the probability of unexpected failure**

Because evolution is the only system known to produce intelligent behaviour, it is to be preferred

Publisher: General Systemantics Press (1978)
ISBN: 978-0671819101

Engineering Is More Than Technology
Software…Engineering?

- **Software development practice to date**
  - *Has not yet become an engineering discipline, but we are better*
  - *Sharing of best practices, designs, and code helps learning*

- **Software Shamans, Hacks, & Pretenders**
  - *10% of developers do 90% of the work*
  - *The other 90% impair 50% of the work of the 10% being productive*
  - *In the end, only half the work ever gets done!*

- **The Holy Grail**
  - *The Right Mechanisms for Construction*
  - *The Right Model for Delivery*
  - *The Right Tools & Technologies for Enablement*
  - *The Right Market for Innovation, Growth, & Prosperity*

Engineering Is More Than Technology

Components Design By Contract

- **Highly Cohesive, Loosely Coupled**
  - *Efficient interactions, minimized transaction costs*
  - *Eliminate rigid dependencies, commoditized implementations*

- **The Contract or API – Establish Law & Order**
  - *Advertises the supported services provided*
  - *Conforms to the enabling framework's laws*
  - *Behaves predictably and responsibly*

- **Design-by-Contract**
  - *Bertrand Meyer's OOSC (1988, 2000)*
  - *Enforced in the Eiffel programming language*
  - *Enforces structured duties & obligations*
  - *Too rigid implementation, but valuable illustration of concepts*

What Is Engineering
The Software IC

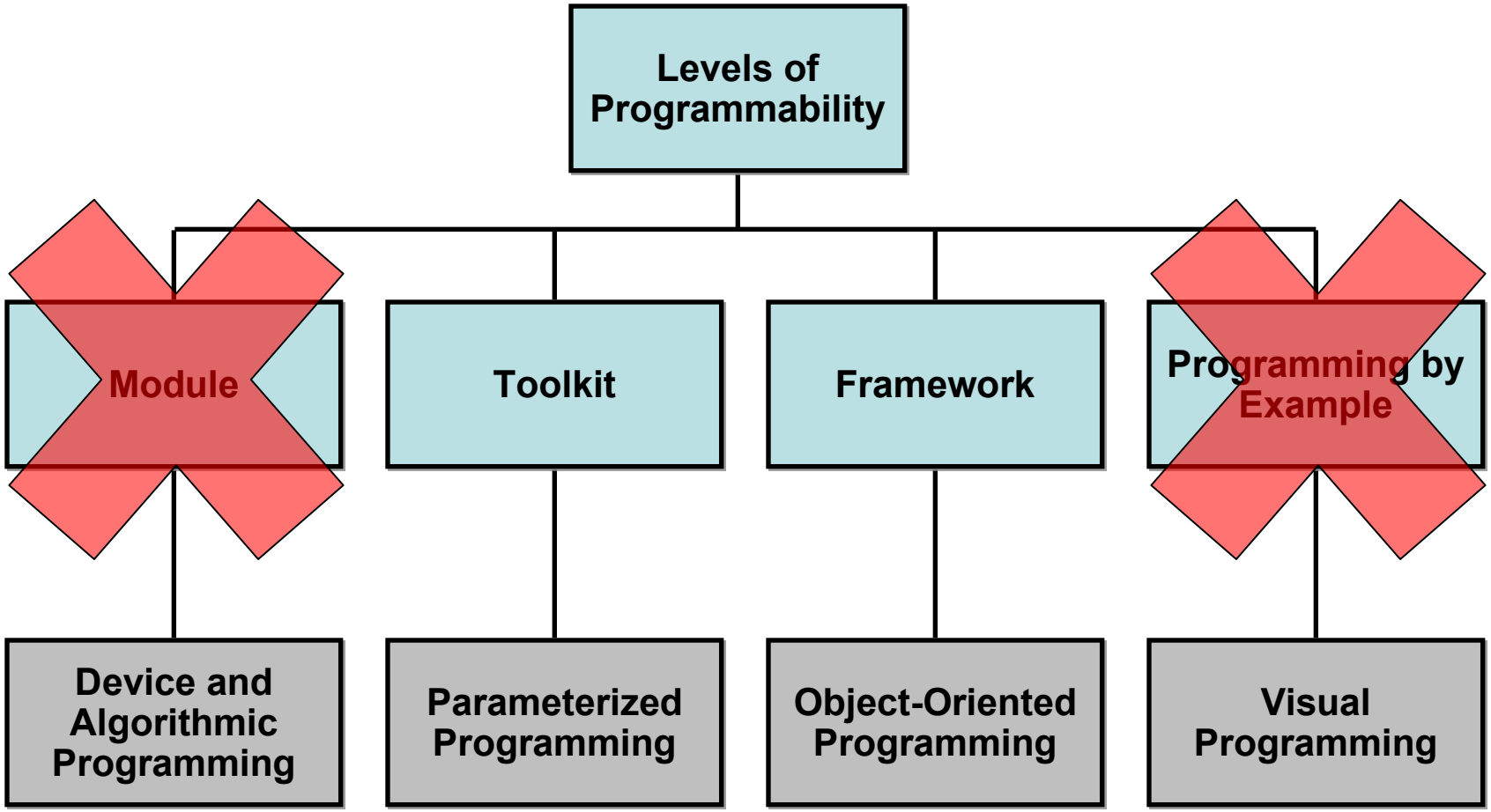
- **Highly Cohesive, Loosely Coupled**
  - *Efficient interactions, minimized transaction costs*
  - *Eliminate rigid dependencies, commoditized implementations*

- **The Software Integrated Circuit (IC)**
  - *Conforms to a Contract, or API*
  - *Implementations could be supplied by many vendors*

- **Software IC's In Practice**
  - *Brad Cox (1988)*
  - *Prototyped in Objective-C programming language*
  - *Reuse of code vastly over promised and under delivered*
  - *May never see a 'software industrial revolution', but don't discount the value of the concept of components*

What Is Engineering

Component Programming Models

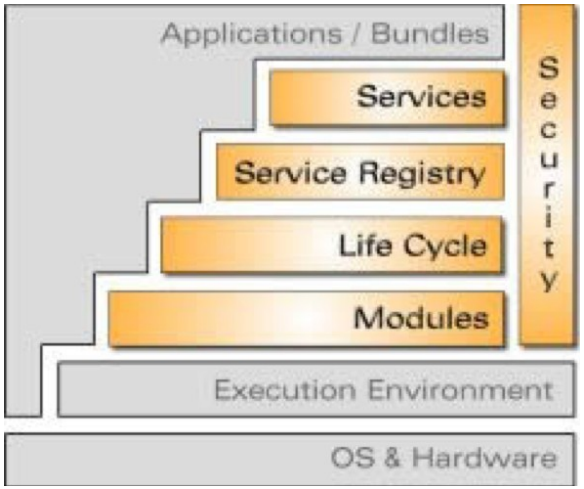


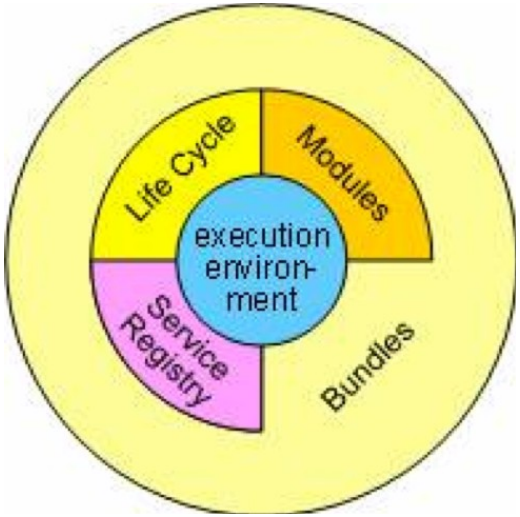
Open Services Gateway Initiative (OSGi)

- Open standards organization focused on Java component models
- Original vision was for remotely managed home gateways
- Established 1999, initial implementations done by our OTI/IBM lab
- We adopted it immediately for embedded Java & telematics work
- Nearly 10 years of experience building and deploying OSGi apps
- Growing international community, huge presence in Europe

Open Services Gateway Initiative (OSGi)

The Big Picture



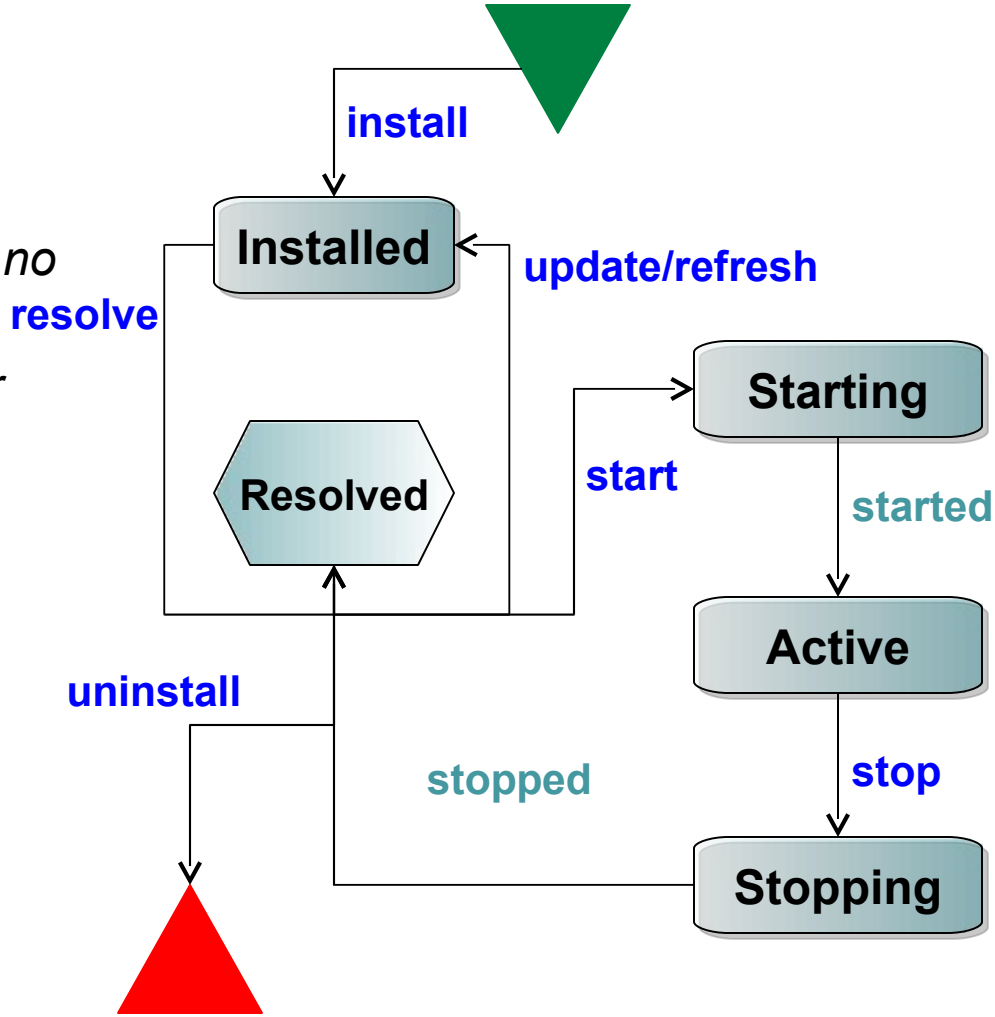


- **Execution Environment**
  - *Your Java Platform*
- **Service Registry**
  - *The Wiring Breadboard*
- **Life Cycle Management**
  - *Keeping Things Live*
- **Modules**
  - *Slicing & Dicing*
- **Security**
  - *Control & Safety*
- **Services & Bundles**
  - *The Really Good Stuff*
  - *What People Pay Us to Build*

Open Services Gateway Initiative (OSGi)

Bundle Life Cycle Management – Activator Managed

- **Benefits**
  - *Programmatic access to runtime configuration*
  - *Hot swappable bundles, no restart required!*
  - *Real components, rather than de facto monolith*
- **Bundle States**
  - *Installed*
  - *Resolved*
  - *Started*
  - *Active*
  - *Stopped*
  - *Uninstalled*



Open Services Gateway Initiative (OSGi)
Service Management Approaches

- **Service Activator Toolkit (SAT)**
  - *Encodes bundle wiring in the bundle Activator*
- **Declarative Services (DS)**
  - *Encodes bundle wiring in XML files*
- **Service Tracker (ST)**
  - *Programmatic and transparent, revealing all the plumbing*
- **SpringSource Application Platform (AP)**
  - *Targets enterprise dependency injection*

…but they all simply wrap the Service Registry

Component Engineering with OSGi
The Component Platform: OSGi & Eclipse Equinox

- **Implementation of OSGi R4 Specification in Java**
  - *Implementation of all aspects of the OSGi specification (including the EEG, MEG and VEG work)*
  - *Investigation and research related to future versions of OSGi specifications and related runtime issues*

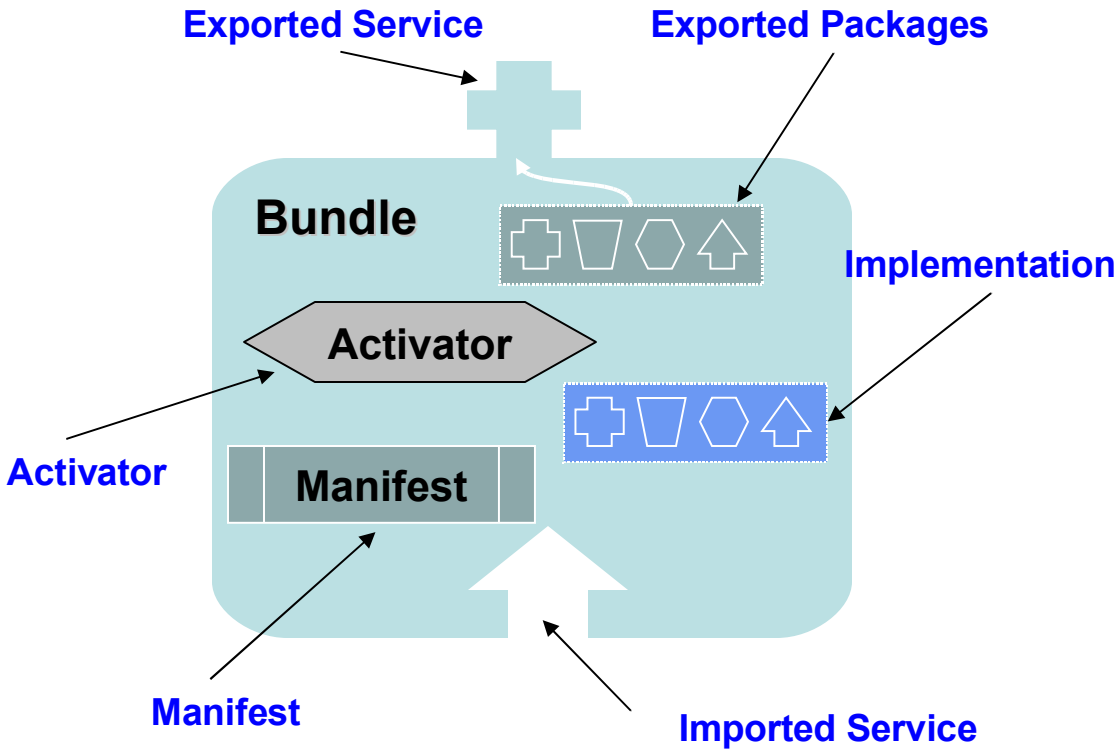
…and…

  - *Development of non-standard infrastructure deemed to be essential to the running and management of OSGi-based systems*
  - *Implementation of key framework services and extensions needed for running Eclipse (e.g., the Eclipse Adaptor, Extension registry) and deemed generally useful to people using OSGi.*

Component Engineering with OSGi

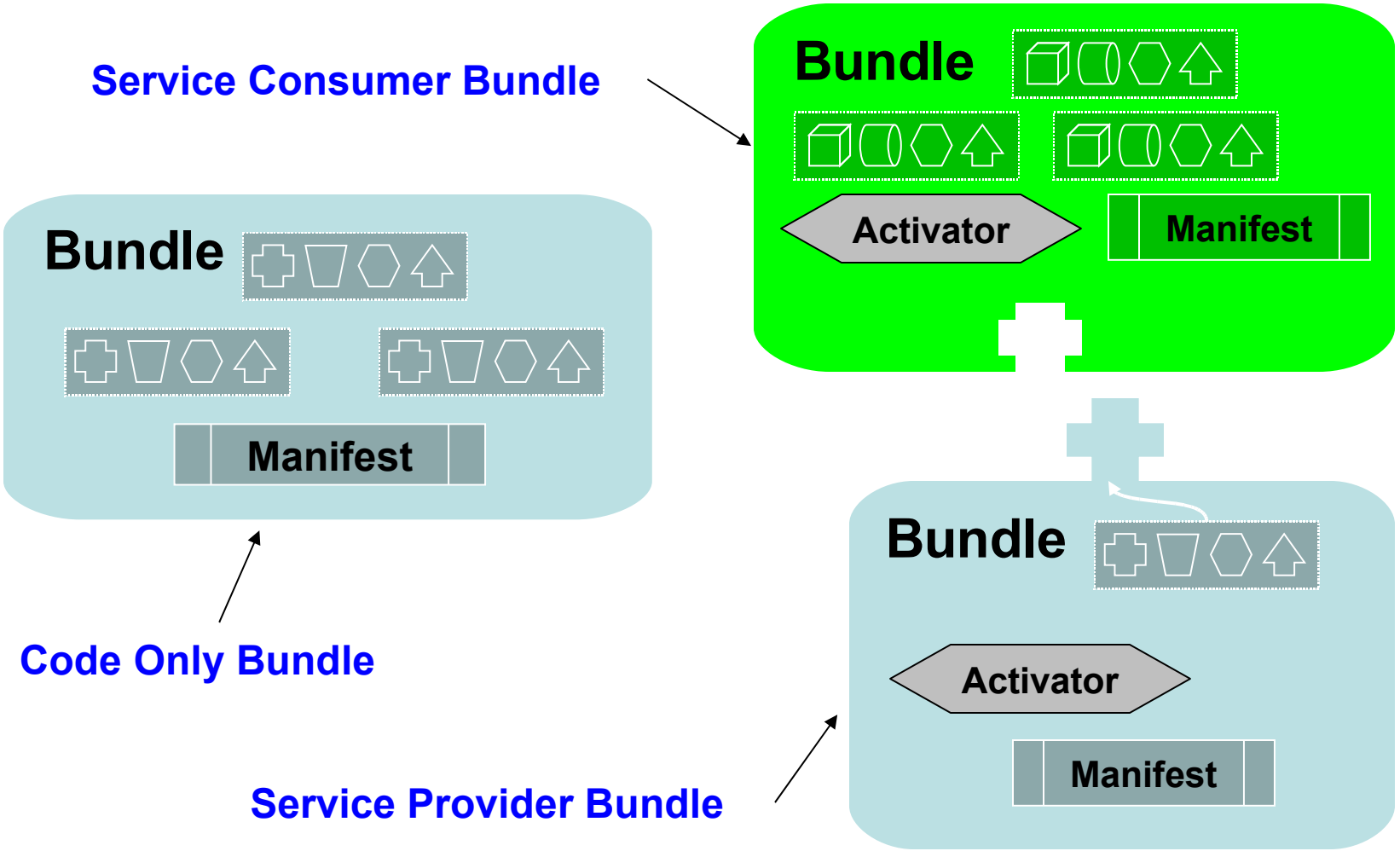
Bundles: Units of Implementation & Distribution

- **Contents of a Bundle**
  - *MANIFEST*
  - *Activator*
  - *Service*
  - *Package*
  - *Implementation*
- **Imports**
  - *Services*
  - *Packages*
- **Exports**
  - *Services*
  - *Packages*



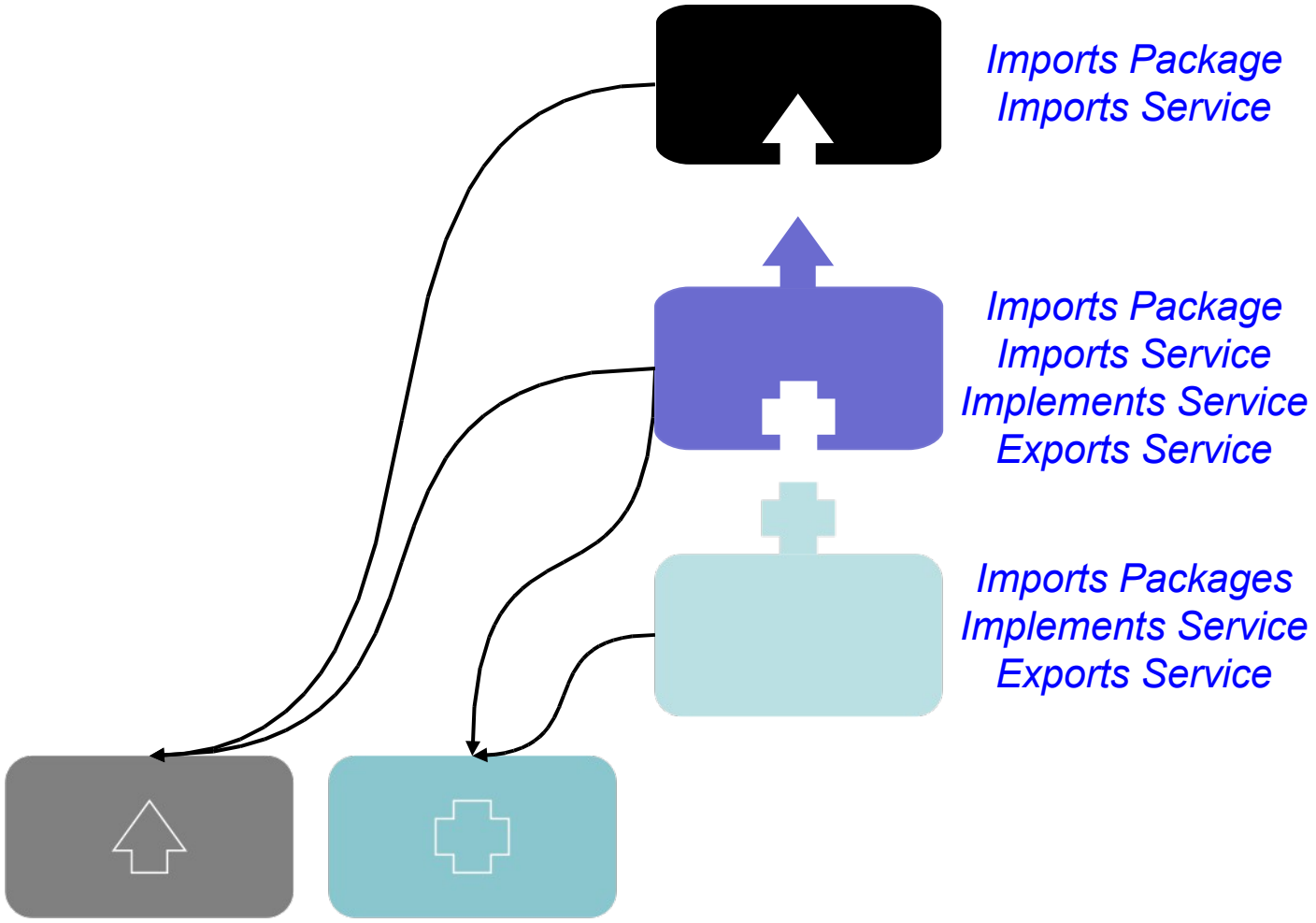
Component Engineering with OSGi

Bundle Flavors: Code Only, Service Providers, Service Consumers



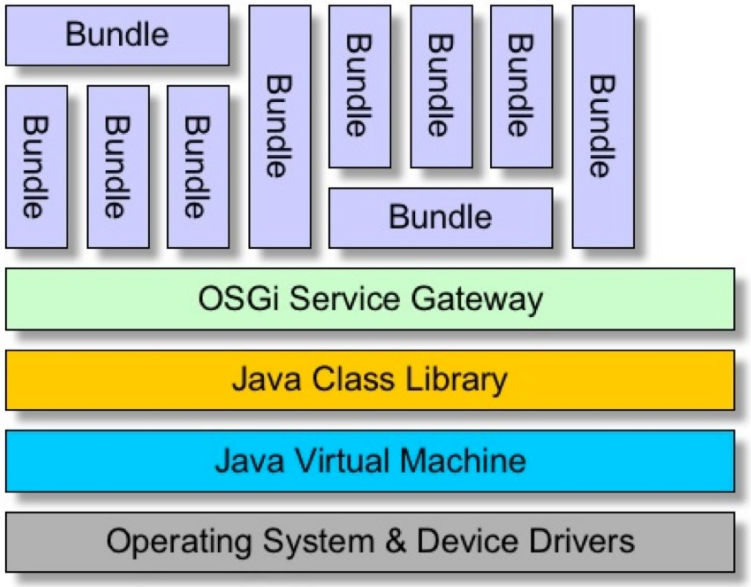
Component Engineering with OSGi

Component Assembly



Component Engineering with OSGi

Service Oriented Bundle Architecture (SOBA)



- **EclipseCon 2007/2008 Tutorials**
  - **Building SOBA**
  - **Embedding Equinox**
  - **Device Interfacing**
  - **P2 Provisioning**

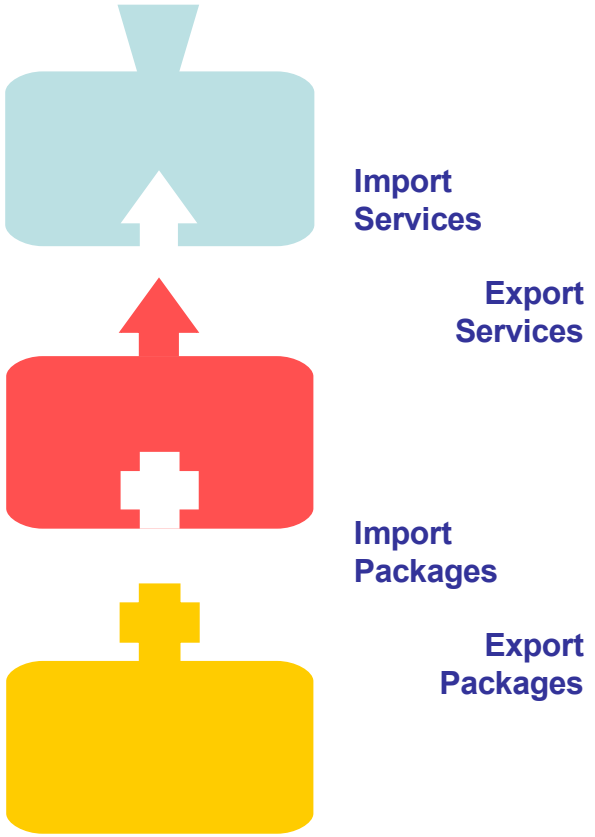
http://www.bandxi.com/soba

- **Emphasizes pure services approach to wiring component**
- **Employs the Service Activator Toolkit for bundle wiring**
- **Sits on the Eclipse Equinox Runtime**

Component Engineering with OSGi

Simple Construction Process – Build TDD POJO's First!

- **Create a POJO based implementation using Junit/JMock and Test Driven Development**
- **Extract the service level interfaces as contracts between components**
- **Build Bundle Activators and craft Bundle Manifests that wire the services together**
- **Run and test them in your workspace locally first**
- **Run them remotely on the embedded target**

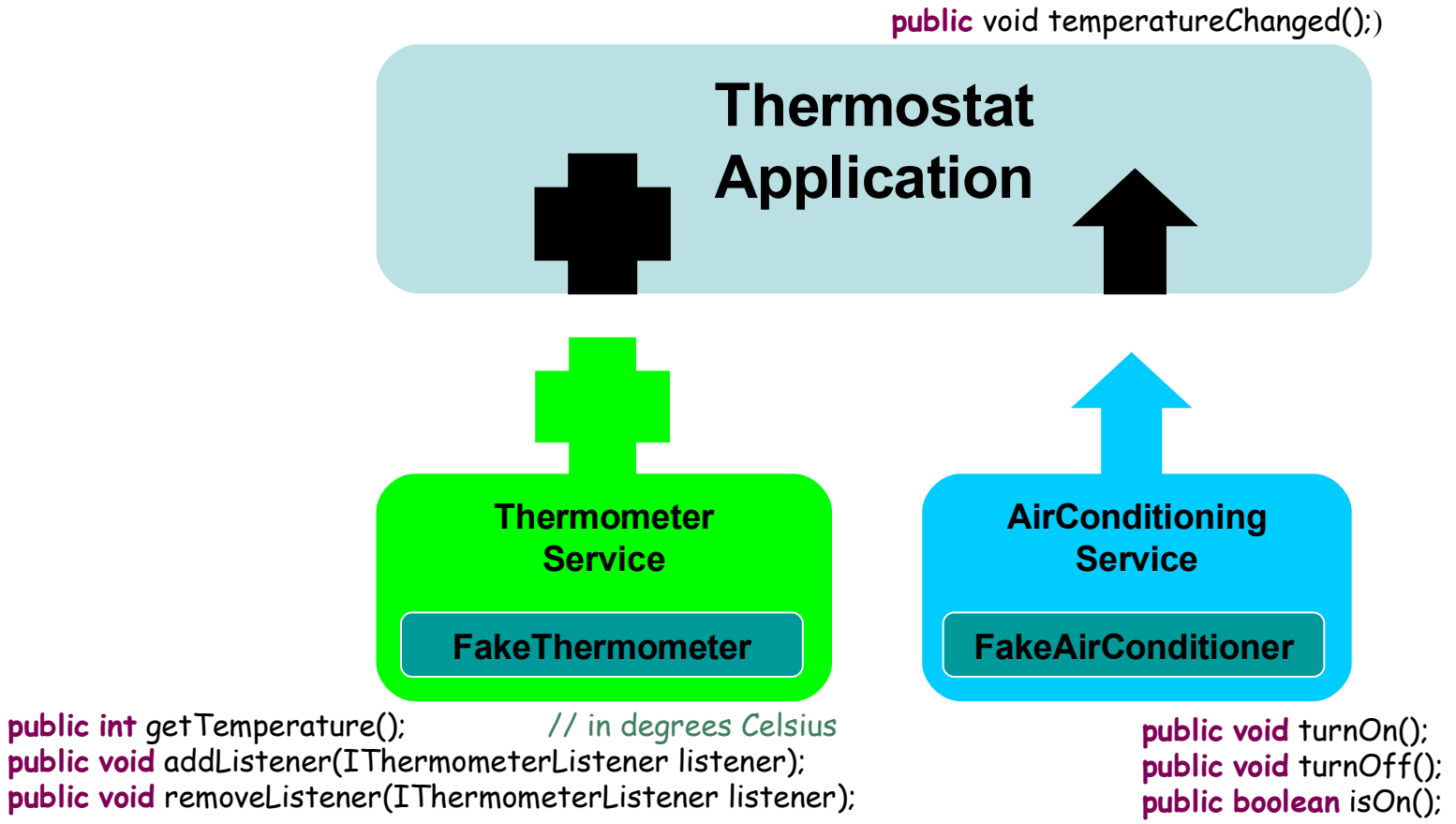


Component Engineering with OSGi
The Bundle Activator

- **At bundle activation time:**
  - *Instantiate the business logic*
  - *Fetch any imported services from OSGi registry*
  - *Bind the business logic to the imported services*
  - *Export services*
- **At bundle deactivation time:**
  - *(SAT automatically unregisters exported services)*
  - *Unbind the business logic from the imported services*
  - *Destroy the business logic*
- **What NOT to do:**
  - *Do this in random order*
  - *Hang on to imported services*
  - *Reach into the business logic*
  - *Do business-specific stuff*

Component Engineering with OSGi

The Thermostat Example



Component Engineering with OSGi

Service Activator Toolkit (SAT) Bundle Activator

```
public class Activator extends BaseBundleActivator {
    private ThermostatApplication thermostat;
    protected void activate() {
        thermostat = new ThermostatApplication();
        thermostat.bind(getIThermometerService(), getIAirConditioningService());
    }
    protected void deactivate() {
        thermostat.unbind();
        thermostat = null;
    }
    private IAirConditioningService getIAirConditioningService() {
        return (IAirConditioningService)
  getImportedService(IAirConditioningService.SERVICE_NAME);
    }
    private IThermometerService getIThermometerService() {
        return (IThermometerService)
  getImportedService(IThermometerService.SERVICE_NAME);
    }
    protected String[] getImportedServiceNames() {
        return new String[] { IAirConditioningService.SERVICE_NAME,
                    IThermometerService.SERVICE_NAME };
    }
}
```

Component Engineering with OSGi

Declarative Services Bundle Activator

```
<?xml version="1.0" encoding="UTF-8"?>
<component name="emergency">
  <implementation class="com.bandxi.jaoo.thermostat.internal.bundle.Component"/>
  <reference name="ac"
    interface="com.bandxi.jaoo.dev.ac.IAirConditioning"/>
  <reference name="thermo"
    interface="com.bandxi.jaoo.toast.dev.thermometer.IThermometer"/>
</component>
```

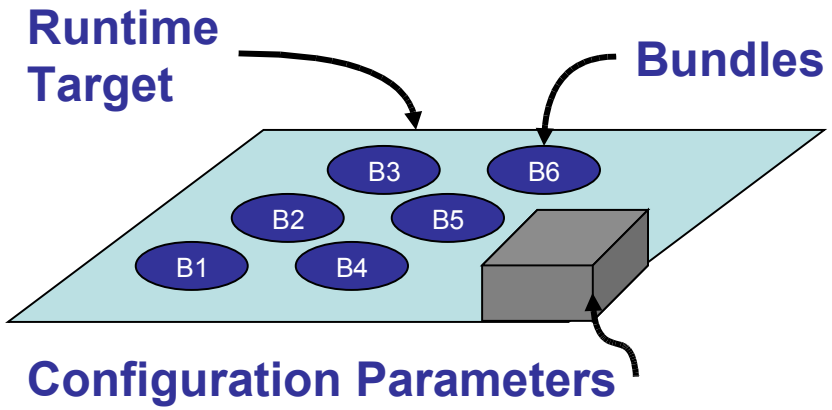
```
public class Component {
    private Thermostat thermostat;

    protected void activate(ComponentContext context) {
        IThermometer thermometer = (IThermometer) context.locateService("thermo");
        IAirConditioning ac = (IAirConditioning) context.locateService("ac");
        thermostat = new Thermostat();
        thermostat.bind(thermometer, ac);
    }

    protected void deactivate(ComponentContext context) {
        monitor.unbind();
        monitor = null;
    }
}
```

Component Engineering with OSGi
Defining the Target

- **Defines**
  - *List of bundles to include*
  - *Application arguments*
  - *VM arguments*
  - *Java Runtime & Execution Environment*
  - *Environment Variables*
- **Expressed as XML file**
  - *Extruded from Eclipse*
  - *Very, very ugly*
  - *Accessible with editor/tooling*



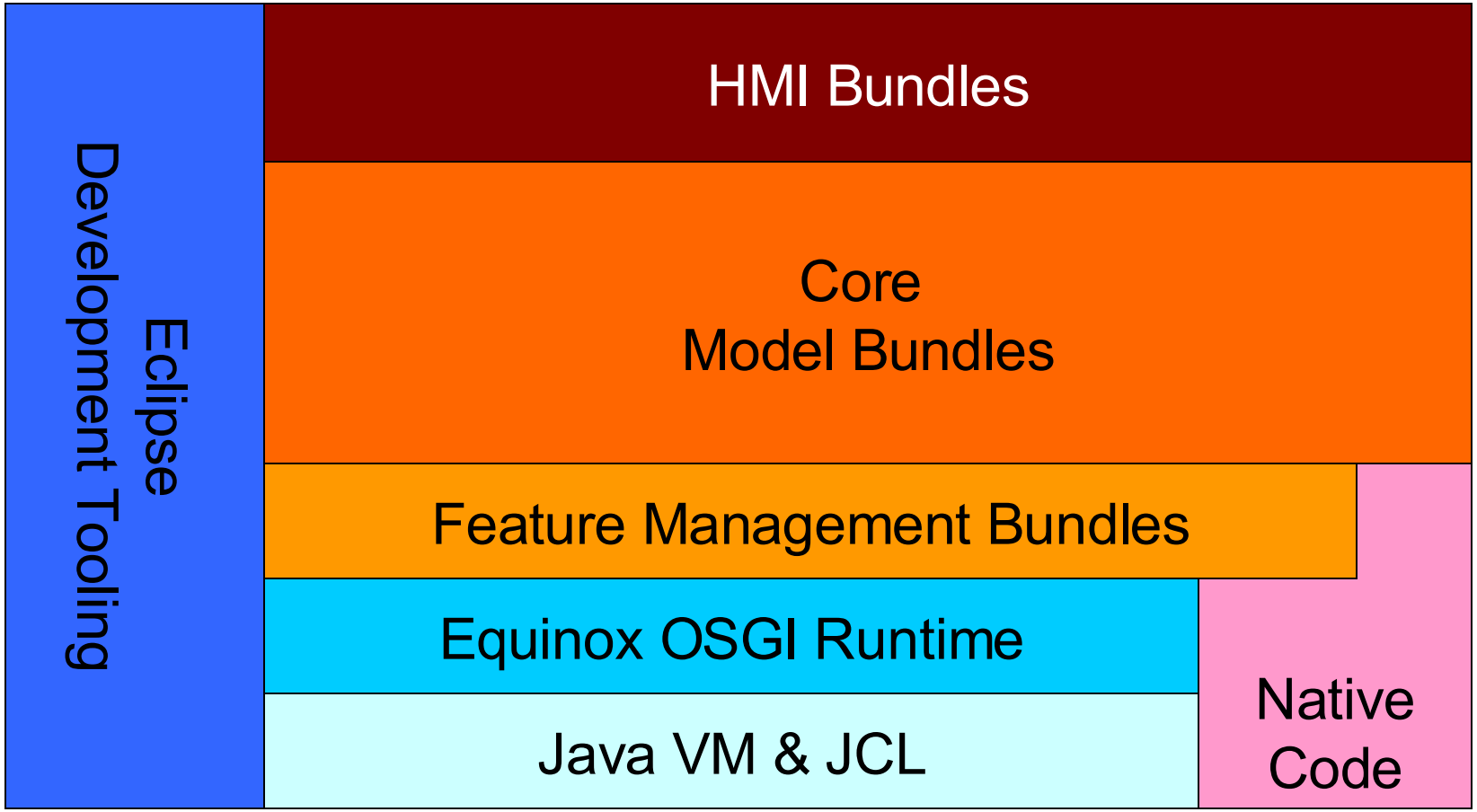
OSGi Applied in the Embedded Space

Applied to Many Kinds of Applications

- **Automotive Telematics**
  - *In vehicle computing, car bus integration & entertainment systems*
- **RFID Supply Chain Management**
  - *Warehouse shipping & delivery, inventory management*
- **Safety & Security Systems**
  - *Monitoring & reporting on CBRNE hazardous materials sensors*
- **Industrial Systems**
  - *Construction equipment head unit*

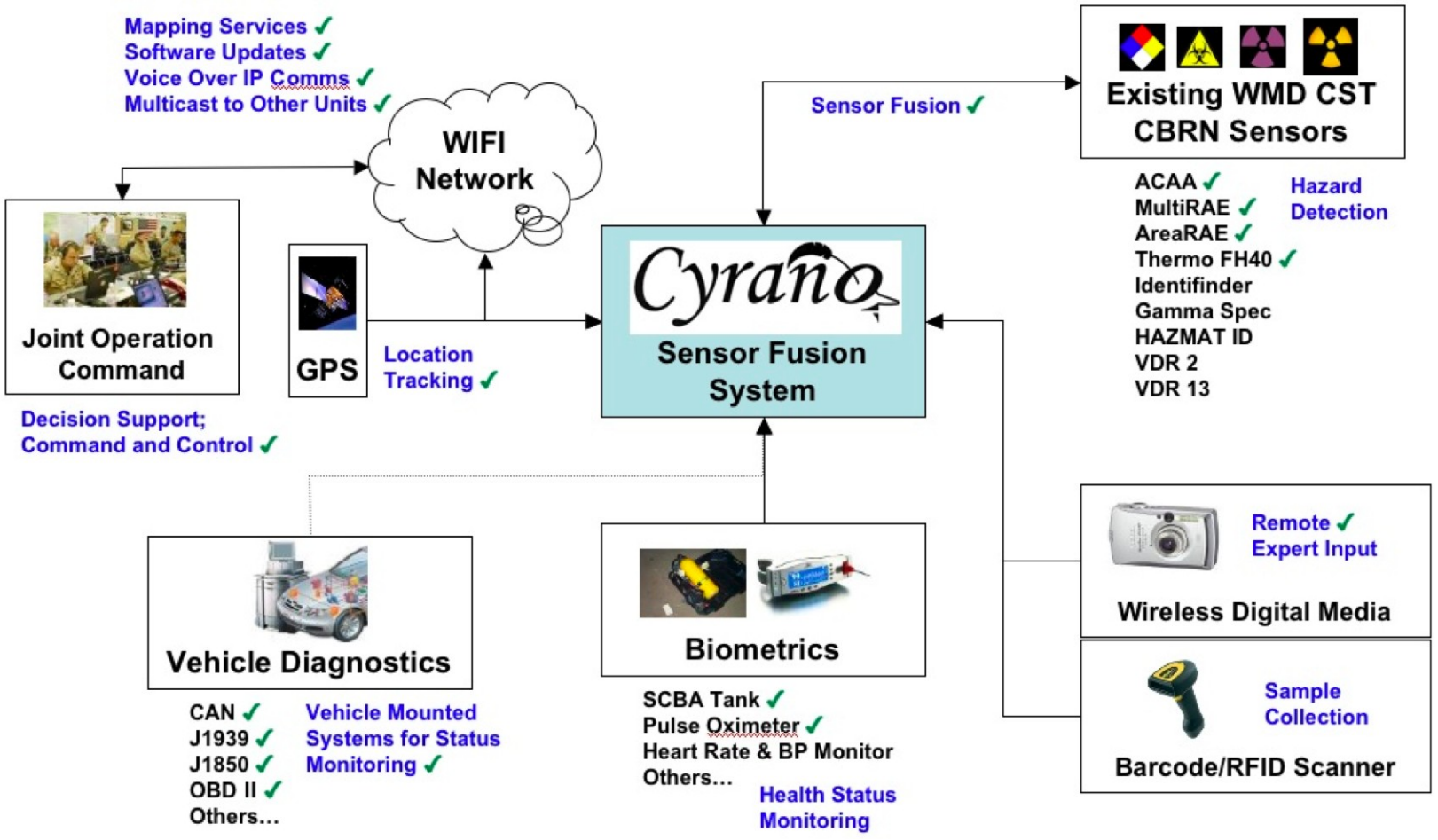
OSGi Applied in the Embedded Space

Obligatory Cake Chart



OSGi Applied in the Embedded Space

Cyrano Sensor Survey System



OSGi Applied in the Embedded Space

Cyrano Sensor Survey System Video

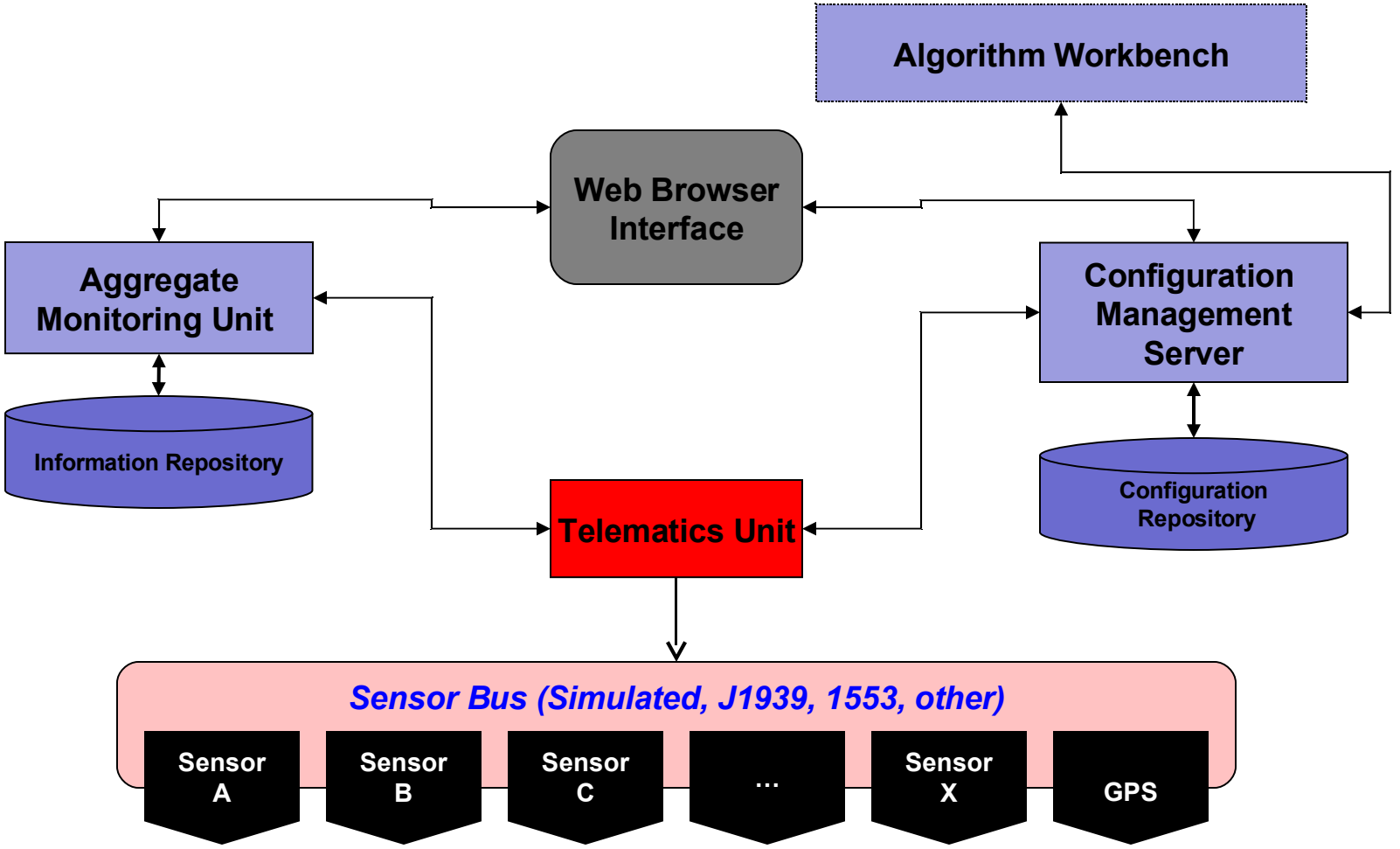
OSGi Applied in the Embedded Space

Diagnostics/Prognostics Component Architecture Project

- **Platform and Software Architecture**
- **Algorithm Configuration, Packaging and Deployment Workbench**
- **Basic Reference Implementation**
  - *Sensor Simulators and Device Models*
  - *User Interface for Telematics Unit*
  - *Diagnostic/Prognostic Algorithm Integration, Testing, Hot Swapping*
  - *Data Collection, Transmission, and the Bandwidth Gatekeeper*
  - *Load Testing and Platform Porting*
- **J1939 Bus Implementation & Support**
  - *Hardware Test Bench and Real Sensor Integration*
- **1553 Bus Implementation & Support**
  - *Hardware Test Bench and Real Sensor Integration*
- **Training and Documentation**

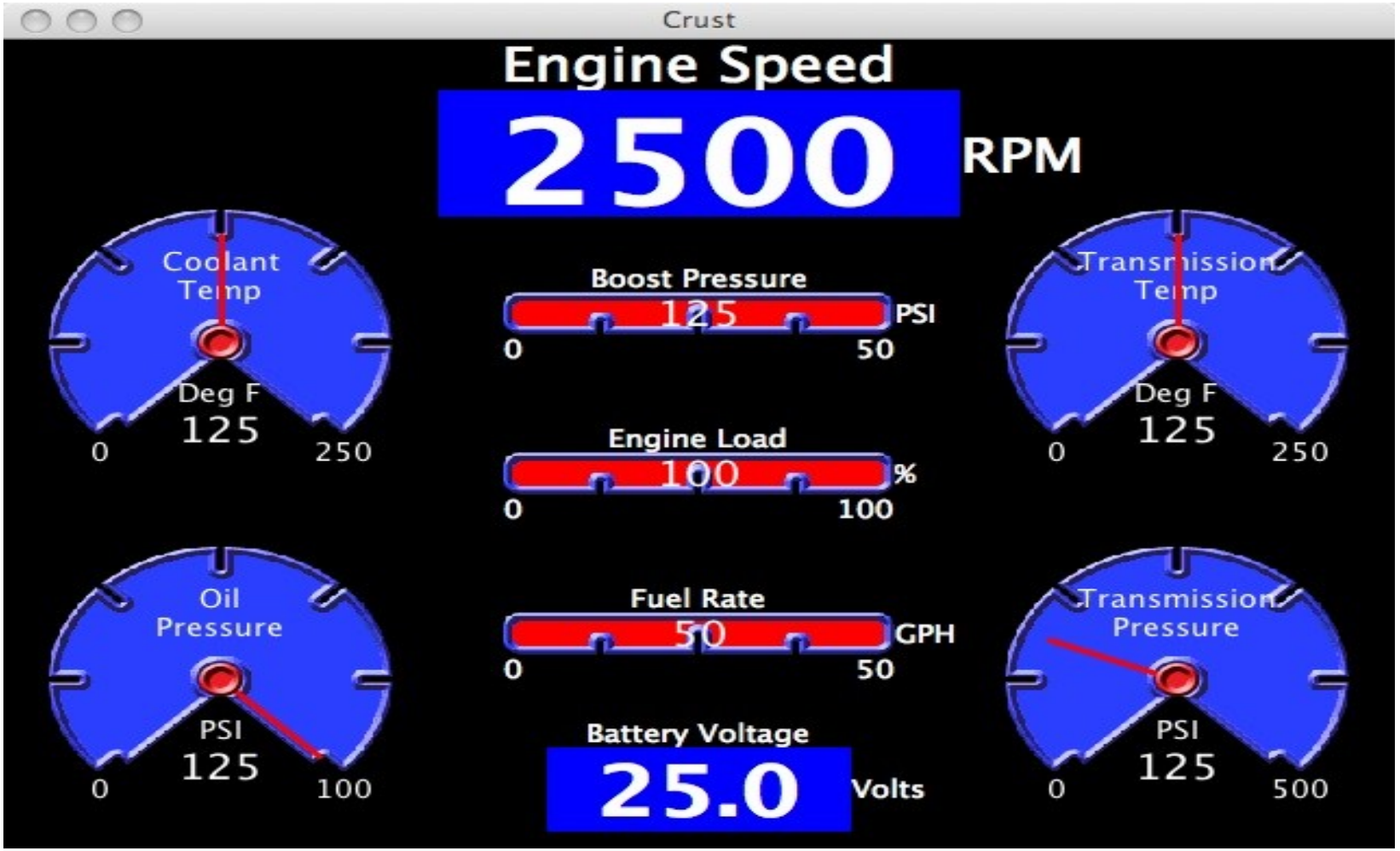
OSGi Applied in the Embedded Space

System Level View



OSGi Applied in the Embedded Space

The Carabas Demo – Industrial Controls



OSGi Applied in the Embedded Problem Space

Talking to the Sensor Bus

J1939 2-wire CAN Bus

Ethernet to J1939

Protocol Bridge

RS232 to J1939

Protocol Bridge

J1939 Sensors

Embedded Linux/WinCE Reference Platform

w/ on-board CAN transceiver

Closing

- **Questions?**

- **For more information…**
  - *http://www.bandxi.com/soba*
  - *http://www.osgi.org*
  - *http://www.eclipse.com/equinox*