An Embedded JavaVM for a robot development kit
(LEGO Mindstorms™ RCX)

Java User Group Stuttgart (JUGS)
SIG Embedded Java
2002-09-12
Java for the RCX

Jochen Hiller
jochen.hiller@bauer-partner.com
http://www.bauer-partner.com

Principal Consultant
Head of Research
Agenda

• An Introduction to LEGO Mindstorms
• leJOS – LEGO Java Operating System
• Java Robotics Programming
  – Examples
  – Live Demos
• Other Examples / Outlook
LEGO Mindstorms™ – Introduction

- History
- The Brick
- Sensors
- Motors
- LEGO Parts
- The RCX Software Development Kit
LEGO Mindstorms™ – History

- Mindstorms concept: partnership / sponsorship
  - LEGO Group
  - MIT Media Laboratory

- MIT activities
  - Starting 1987: Programmable Brick
  - 4 Versions until 1998, latest big red brick
  - Driven by Fred G. Martin

- LEGO Group
  - Development from ground up, reuse philosophy
  - Brick based software programming for kids
  - 1998: 1st commercial version available, 80,000 kits/200$/3 months
  - Strategic product for LEGO, “Legoland“ activity too
LEGO Mindstorms™ – RIS

• RIS: Robotics Invention System
  – v1.0, v1.5, v2.0

• Family of products
  – computer controlled robots

• Latest product:
  – Vision Command: Camera
    (Logitech)
LEGO Mindstorms™ – The Brick (1)

• RCX
  – Robotic Command Explorer
  – v1.0, v2.0 available
LEGO Mindstorms™ – The Brick (2)

- Components
  - Batteries (9 V, 6 x 1.5V)
  - Microcontroller
  - 4 Buttons
  - Input Ports
  - Output Ports
  - Infrared Serial Communication Port
  - Speaker
  - LCD Display
LEGO Mindstorms™ – CPU

• CPU
  – Hitachi H8/3292 series microcontroller
  – 16 MHz
  – 16 KB ROM
  – 512 B onchip Cache-RAM
  – ROM contains core functionality for hardware access

• RAM
  – 32 kB
  – Contains firmware and user programs
LEGO Mindstorms™ – HMI

- **Buttons**
  - On-Off, Run, Prgm, View

- **LCD Display**
  - Multi functions
  - 4 digits + 1 one digit display

- **Speaker**
  - Variable frequency and duration
LEGO Mindstorms™ – IR Com.

- Infrared based communication
  - PC - RCX, RCX – RCX
  - bidirectional
- IR Tower
  - RS 232, USB versions (from RIS v2.0)
- Firmware and Software download via IR
- PC – RCX
  - communication via IR Tower
- RCX – RCX
  - communication via IR serial communications port
LEGO Mindstorms™ – Sensors

- RCX supports 3 sensors to be connected
- Available sensors from LEGO:
  - Touch sensor
  - Light sensor
  - Rotation sensor
- 3rd party sensors available
  - E.g. compass, temperature, voltage, motion, ...
- Resolution: 10 bit analogue signal (0-1023)
LEGO Mindstorms™ – Motors

- RCX support 3 actuators
- Available actuators from LEGO:
  - Motors
  - Lamp
- Motors
  - can only be controlled via power, internally implemented via PWM
  - Speed not changable, only via gears
LEGO Mindstorms™ – LEGO parts

• 717 parts included
• LEGO basic and LEGO technic parts
• Specific parts for robotics added
LEGO Mindstorms™ – RCX SDK

• LEGO provides Software Development Kit
  – Tutorial
  – Integrated Development Environment (RCX Code)
    • Brick based programming paradigm
  – RCX Firmware (0309)
  – Scripting Language (Mindscript)

• RCX firmware
  – Interpreter, leaves 6 kB code for user programs
  – Limitations
    • No floating point number support
    • 32 variables, no arrays
LEGO Mindstorms™ – Alternatives

- NQC: Not Quite C, Dave Baum
- pbForth: Forth implementation
- VisualBasic
  - Spirit.OCX : Control as part of core LEGO RCX SDK
- legOS: C/C++ based implementation of an OS
leJOS – An Embedded JavaVM

- History
- Overview
- Features
- RCX specific extensions
leJOS – History

• leJOS : LEGO Java Operating System

• How to pronounce: [layhoss]
  – Like Spanish word: „lejos“ which means far

• Initially developed by José Solorzano
  – First version available as TinyVM
  – 09 / 2000: v1.0alpha2 released
  – 09 / 2001: v1.0.2, first stable release
  – 05 / 2002: v1.0.5 released, 3k more memory
  – 07 / 2002: v2.0: communication, smaller footprint, performance optimisations
leJOS – Overview

- leJOS is a JavaVM for very small devices
  - 8 bit microcontroller, <32 kB RAM
- leJOS has a small footprint (~16 kB)
- leJOS replaces the LEGO firmware
- leJOS is the JVM running on RCX
leJOS – Memory Layout

- 4 KB Off Limits used by ROM
- Direct Memory Access
- PersistentMemoryArea
  - Exists after a run
  - Destroyed by new program download
- Recyclable objects
  - Provided through API
- No Garbage Collection
leJOS – Features (1)

• Floating point numbers
  – The only RCX language support FP
  – Enables trigonometric functions
  – 32 bit floats only support
  – Doubles can be used, but will be mapped to floats
  – Trigonometric functions in `java.lang.Math`

• Threads
  – Preemptive scheduling algorithm
  – Supports `synchronize` and interrupting
  – Up to 255 threads supported
leJOS – Features (2)

• Arrays
  – 1-dimensional AND multi-dimensional Arrays supported

• Event Model
  – Supports Java Event Model: Listeners, Event Sources
  – Timer (TimerListener)
  – Button (ButtonListener)
  – Sensor (SensorListener)

• Recursion
  – Max. 10 levels of recursion allowed (Stack limitation)
  – Dependent of number of local variables
leJOS – Features (3)

- Exceptions
  - leJOS supports complete exception handling

```java
try {
    ...
} catch (Exception type) {
    ...
} finally { // optional
    ...
}

- >15 exceptions implemented in `java.lang`
leJOS – Features (4)

• Exceptions
  – Exception display: limited to LCD display
    \( 0052 \times 3 \)
    0052 Method exception raised
    3 Exception class
  – Numbers generated within a signature file
  – RCX beeps for one second
leJOS – Features (5)

• Collections
  – *Hashtable* and *Vector* supported (not all methods)

• Garbage Collection
  – Not yet supported
  – Implementation under progress
  – JVM is prepared to maintain object state

• Reflection
  – Not supported
  – Some classes implemented as a dummy to satisfy tools (e.g. *Class* for javac and jikes)
leJOS – RCX extensions (1)

- **Package** `josx.platform.rcx`
- **Outputs**
  - Support for `Motor`, A/B/C are predefined
- **Inputs**
  - All `Sensor` types with all modes (Raw, Binary, ...) supported
  - `SensorListener` available
- **Button**
  - Can be reprogrammed (except On-Off)
  - `Button`, `ButtonListener` available
leJOS – RCX extensions (2)

• System Time
  – Time resolution in milliseconds
  – `josx.util.Timer, josx.util.TimerListener`

• Battery power:
  – `josx.platform.rcx.Battery`

• LCD support
  – Direct programming of LCD elements
  – Funny `TextLCD` support, emulating alphanumeric characters

• Speaker
  – Complete control over frequency and duration
  – `josx.platform.rcx.Sound`
leJOS – RCX extensions (3)

- Multiple programs
  - Up to 8 programs can be stored, downloading multiple class files
- IR Remote Control
  - Very limited support
leJOS – RCX extensions (4)

- IR Communication
  - PC – RCX / RCX – RCX communication supported

- Multiple protocols
  - LLC (Low Level Communication)
  - F7 (LEGO F7 Opcode protocol)
  - LNP (LEGO Network Protocol), also used by legOS
leJOS – Tools (1)

- leJOS acts as wrapper for standard Java tools
  - Compiler: **lejosc**
  - Disassembler: **lejosp**
  - Class format until 1.3 supported, 1.4 under progress

- **Linker: lejos**
  - Intelligent linker: collects only referred classes
  - Under progress: Static program analysis, collecting on method level
leJOS – Tools (2)

- Loader: lejosrun
- Firmware download: lejosfirmedl
- Emulation support (without UI)
  - Linker: emu-lejos
  - Loader: emu-lejosrun

- Extension for some Java IDE’s available
  - jEdit, Eclipse, special RCX IDEs, others
Java Robotics Programming

• Examples
  – HelloWorld
  – Behaviour Control
  – Navigator

• Live Demos

• Try yourself?
Example – HelloWorld

```java
import josx.rcx.*;

public class HelloWorld {
    public static void main () {
        LCD.clear();
        TextLCD.print("hello");
        Thread.sleep(2000);
        TextLCD.print("world");
        Thread.sleep(2000);
    }
}
```
Example – DisplayException

import josx.rcx.*;

public class DisplayException {
    public static void main(String[] aArg) {
        LCD.clear();
        TextLCD.print("start");
        enforceException();
    }
    public static void enforceException() {
        throw new RuntimeException();
    }
}

© 2002 Jochen Hiller
Robots – Behavior Control (1)

- Pattern from Robotics programming
- Defined by Rodney Brooks, MIT AI Lab
- Influenced by insect world, observing the nature
- Named: Subsumption architecture → Behavior control
Robotics – Behavior Control (2)

Example:

<table>
<thead>
<tr>
<th>Behaviour</th>
<th>Condition</th>
<th>Action</th>
<th>Priorit y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Follow light</td>
<td>Brightness detected</td>
<td>Drive towards lights</td>
<td>Low</td>
</tr>
<tr>
<td>Go home</td>
<td>Battery level low</td>
<td>Drive to starting point</td>
<td>Mid</td>
</tr>
<tr>
<td>Collision</td>
<td>Touch sensor pressed</td>
<td>Back Up and Rotate</td>
<td>High</td>
</tr>
</tbody>
</table>
Robotics – Behavior Control (3)

```java
package josx.robotics;

public interface Behavior {
    boolean takeControl();
    void action();
    void suppress();
}

public class Arbitrator {
    public Arbitrator(Behavior[] behaviors);
    public void start();
}
```
public class DriveForward implements Behavior {
    public boolean takeControl() {
        return true;
    }
    public void action() {
        Motor.A.forward();
        Motor.C.forward();
    }
    public void suppress() {
        Motor.A.stop();
        Motor.C.stop();
    }
}
public class Collision implements Behavior {
    public boolean takeControl() {
        return Sensor.S1.readBooleanValue();
    }
}

class Collision {
    public boolean takeControl() {
        return Sensor.S1.readBooleanValue();
    }
}

class Collision {
    public boolean takeControl() {
        return Sensor.S1.readBooleanValue();
    }
}

public class Collision implements Behavior {
    public boolean takeControl() {
        return Sensor.S1.readBooleanValue();
    }
    public void action() {
        Motor.A.backward(); Motor.C.backward();
        Thread.sleep(500);
        Motor.A.stop(); Thread.sleep(1000);
        Motor.C.stop();
    }
    public void suppress() {
        Motor.A.stop(); Motor.C.stop();
    }
}

Example – Behavior Control (2)
Example – Behavior Control (3)

```java
public class BumperCar {
    public static void main(String[] aArg) {
        Behavior b1 = new DriveForward ();
        Behavior b2 = new Collision ();
        Behavior[] b = {b1, b2};
        Arbitrator arby = new Arbitrator (b);
        arby.start ();
    }
}
```
Robotics – Navigator (1)

- Navigation pattern, based on
  - Direction
  - Distance
- Requires `java.lang.Math`, trigonometric functions
- Multiple Implementations available
  - `TimingNavigator`
  - `RotationNavigator`
Robotics – Navigator (2)

package josx.robotics;

public interface Navigator {
    void forward ();
    void backward ();
    void stop ();
    void travel (int distance);
    void rotate (float angle);
    void gotoAngle (float angle);
    void gotoPoint (float x, float y);
    float getX ();
    float getY ();
    float getAngle ();
}

Robotics – Timing Navigator

public TimingNavigator (Motor left, Motor right, int timeOneMeter, int timeRotate);

• Requires two motors
• Requires timing information for
  – Direction change
  – Distance
• Accuracy not optimal
Outlook - leJOS / VisionCommand

• Prototype from Simon Ritter, Java Evangelist at SUN

• Java API for VisionCommand (Lawrie Griffiths)
  – JMF motion detection software
  – Support for a live video viewer (snapshot, video capture)
  – Support for region definition
  – Support for color and light
  – leJOS remote method execution package

• First implementation with X10XCam2 (wireless) available
Outlook – Other Approaches

• Other platform implementations of leJOS
  – E.g. Prototypes have been implemented for Nintendo Gameboy

• RCX Emulator (Andy Gombos)
  – Complete RCX Brick Emulator

• EmuLegOS
  – legOS emulator
leJOS works in ISS

- LEGO Ultimate Builders Competition (11/2001)
- Winner: Konrad und Bastian Schwarzenbach
- Jitter: „Garbage collector within ISS“
  - Collects pieces
  - Funny Behavior for communication with astronauts

- Sensors, Actuators
  - Light, rotation, IR sensors
  - Impuls control

- Software
  - Based on leJOS (not TinyVM)
leJOS works in ISS – Jitter (1)
leJOS works in ISS – Jitter (2)
Examples - Ballfaenger
Examples - RookerBoogie
Resources (1)

- LEGO Mindstorms
  - http://www.mindstorms.lego.com: The official LEGO Mindstorms homepage
  - http://graphics.stanford.edu/~kekoa/rcx: Kekoa Proudfoot, excellent reengineering of Hardware / Software
Resources (2)

• leJOS
  – http://www.lejos.org: The official homepage of leJOS
  – http://tinyvm.lejos.org: The leJOS predecessor
  – http://www.info.ucl.ac.be/people/chp/projects/javarcx/eclipse/: Eclipse-Plugin, from Christophe Ponsard

• Other RCX languages