OROCOS, the open source reference when it comes to real-time and control

Peter Soetens, FMTC

Flanders’ Mechatronics Technology Centre
Leuven

22 March 2006
Embedded Systems in Robotics and Automation
Outline

1. Introduction
   - Problem Domain
   - Orocos’ Solution
   - Orocos History

2. Orocos Framework
   - Building Applications
   - Component API
   - Component Development

3. Demo
   - Application Setup
   - Interfacing a Single Machine Controller
Orocos in one-liners

- *Open Robot Control Software*
  - ⇒ *Open Source* machine control and interfacing
- Real-time Software Toolkit in C++
  - ⇒ Developer’s tool
- Tool for developing components for control
  - ⇒ Real-time, thread-safe, interactive
- Offers common component implementations
  - ⇒ Optional

Freely available on:

http://www.orocos.org
Outline

1. Introduction
   - Problem Domain
     - Orocos’ Solution
     - Orocos History

2. Orocos Framework
   - Building Applications
   - Component API
   - Component Development

3. Demo
   - Application Setup
   - Interfacing a Single Machine Controller

Peter Soetens, FMTC  http://www.Orocos.org
Consider solving...

More products ⇒ Much more software

With monolithic software.

- New devices, same problems to solve
- More software and features
- Device connectivity and networking

'Embedded' Machine Controller

OS

Device
Consider solving…

More threads ⇒ Much more trouble

With bare threads and locks as tools.

- Deadlocks, thread races, data corruption
- Synchronisation between threads?
- Communication between threads?
Consider solving...

More layers $\Rightarrow$ Less control

With closed toolkits.
- 'Solutions' restrict the solution
- Software interaction ?
- Dead vendor products ?
Outline

1. Introduction
   - Problem Domain
   - Orocos’ Solution
   - Orocos History

2. Orocos Framework
   - Building Applications
   - Component API
   - Component Development

3. Demo
   - Application Setup
   - Interfacing a Single Machine Controller
Orocos provides . . .
Middleware for Machine Control
⇒ Software Component deployment *and* interconnection
Safe Software Development

Orocos provides . . .
Tools for Communication ⇒ Thread-safe and Real-Time

Real-Time State Machines

Real-Time Component API
Flexible Software Development

Orocos is . . .
Free Software ⇒ Open Infrastructure with $\infty$ lifetime

Your Component

Orocos Real-Time Toolkit
- Orocos OS Abstraction
  - Threads, Mutexes, Semaphores
  - Operating System
- Hardware Target

Orocos Device Interface
- Orocos Device Interface
- A/D IO, CANOpen, Encoders...
- Operating System Device Drivers
- Hardware Devices

Orocos Application Stack

Peter Soetens, FMTC
http://www.Orocos.org
Outline

1 Introduction
   ● Problem Domain
   ● Orocos’ Solution
   ● Orocos History

2 Orocos Framework
   ● Building Applications
   ● Component API
   ● Component Development

3 Demo
   ● Application Setup
   ● Interfacing a Single Machine Controller

Peter Soetens, FMTC
http://www.Orocos.org
2001: Started as a ‘small’ research project
- Founded by Prof H. Bruynickx, KU Leuven
2001-2005: Developed during the PhD of Peter Soetens
- Sponsored by the EU IST “Orocos”, “Ocean” and “Open Machine Controller” projects and FMTC.
2005-...: Maintained by the FMTC.
- ‘Modular Machines Group’
Outline

1 Introduction
   - Problem Domain
   - Orocos’ Solution
   - Orocos History

2 Orocos Framework
   - Building Applications
     - Component API
     - Component Development

3 Demo
   - Application Setup
   - Interfacing a Single Machine Controller

Peter Soetens, FMTC
http://www.Orocos.org
The Real-Time Toolkit

Components
Self-made or community contributions

Applications
'Templates' select and connect Components

Real-Time Toolkit
C++ Classes

Control Components
Build Applications

Control Applications
Build Components

Real-Time Toolkit
Peter Soetens, FMTC
http://www.Orocos.org
Outline

1. Introduction
   - Problem Domain
   - Orocos’ Solution
   - Orocos History

2. Orocos Framework
   - Building Applications
   - Component API
   - Component Development

3. Demo
   - Application Setup
   - Interfacing a Single Machine Controller
Component Interface

- Properties <XML>
  - Parameters
  - Persistent configuration

- Events <FSM>
  - Alarms
  - Publish state changes

- Methods <Calc>
  - Algorithms
  - Complex configuration

- Commands <Goal>
  - Setpoints
  - Actions taking time

- Data Ports <Control>
  - Data streaming
  - Buffered and unbuffered
Component API Example

"Robot" Component

Properties
- "Kinematic Algorithm"
- "Control Parameters"
- "Tool Type"

Events
- "Position Reached"
- "Object Grasped"
- "Emergency Stop"

Methods
- isMoving()
- writeData("file")
- getError()

Commands
- moveTo(pos, velocity)
- openGripper()
- home()

Joint Setpoints
- Joint Positions
- FeedForward

DataFlow
- <FSM>
- <Calc>
- <Goal>
- <Control>

Config.

Execution

Data

Peter Soetens, FMTC
http://www.Orocos.org
State Controlling_P
{
    double error;
    run {
        set error = Ref.Get() - Ist.Get();
        do Out.Set( task.K * error );
    }
    exit {
        do Out.Set( 0.0 );
    }
    transitions {
        if ( error > task.MaxError )
            select SignalTrackingError
    }
}

"P Controller Component"
Introduction
- Problem Domain
- Orocos’ Solution
- Orocos History

Orocos Framework
- Building Applications
- Component API
- Component Development

Demo
- Application Setup
- Interfacing a Single Machine Controller
Demo Machine Hardware

Diagram showing connections between encoder, end limit, AD, enable, Enc., A.Out, and D.IO.
The basic building blocks ...

- **Joint Level Interpolator Component**
- **PI Controller Component**
- **Hardware Component**

- **Control Kernel Process**
- **User Interface**

Peter Soetens, FMTC

http://www.Orocos.org
Components: Configuration

With properties...

- Joint Level Interpolator Component
- PI Controller Component
- Hardware Component

Control Kernel Process

User Interface
Components: Data Flow

Connecting data ports ...

Joint Level Interpolator Component → SetPoint → PI Controller Component → Output
Position → Hardware Component

Control Kernel Process

S1 → S2 → S3 → S4

User Interface

http://www.Orocos.org
Components: Execution Flow

Executing application logic...

- **Joint Level Interpolator Component**
- **PI Controller Component**
- **Hardware Component**

1. Read Encoders
2. Interpolate Setpoint
3. Control Action
4. Drive Engine

Control Kernel Process

User Interface

Peter Soetens, FMTC

http://www.Orocos.org
Components: Application

Putting it all together...

Joint Level Interpolator Component

PI Controller Component

Hardware Component

1. Read Encoders
2. Interpolate Setpoint
3. Control Action
4. Drive Engine

Control Kernel Process

User Interface
Outline

1. Introduction
   - Problem Domain
   - Orocos’ Solution
   - Orocos History

2. Orocos Framework
   - Building Applications
   - Component API
   - Component Development

3. Demo
   - Application Setup
   - Interfacing a Single Machine Controller
**Demo**

![Diagram of Orocos Framework components: Controllers, Effectors, Estimators, Generators, Commands, States, Programs, Console.](http://www.Orocos.org)
Orocos offers

- a software toolkit for building real-time components
- rich online browsable component interface
- user defined real-time state machines

Further Reference:
http://www.orocos.org