HiDEOS Overview

An Object-Oriented Approach to Low-Level Instrumentation Control and Support

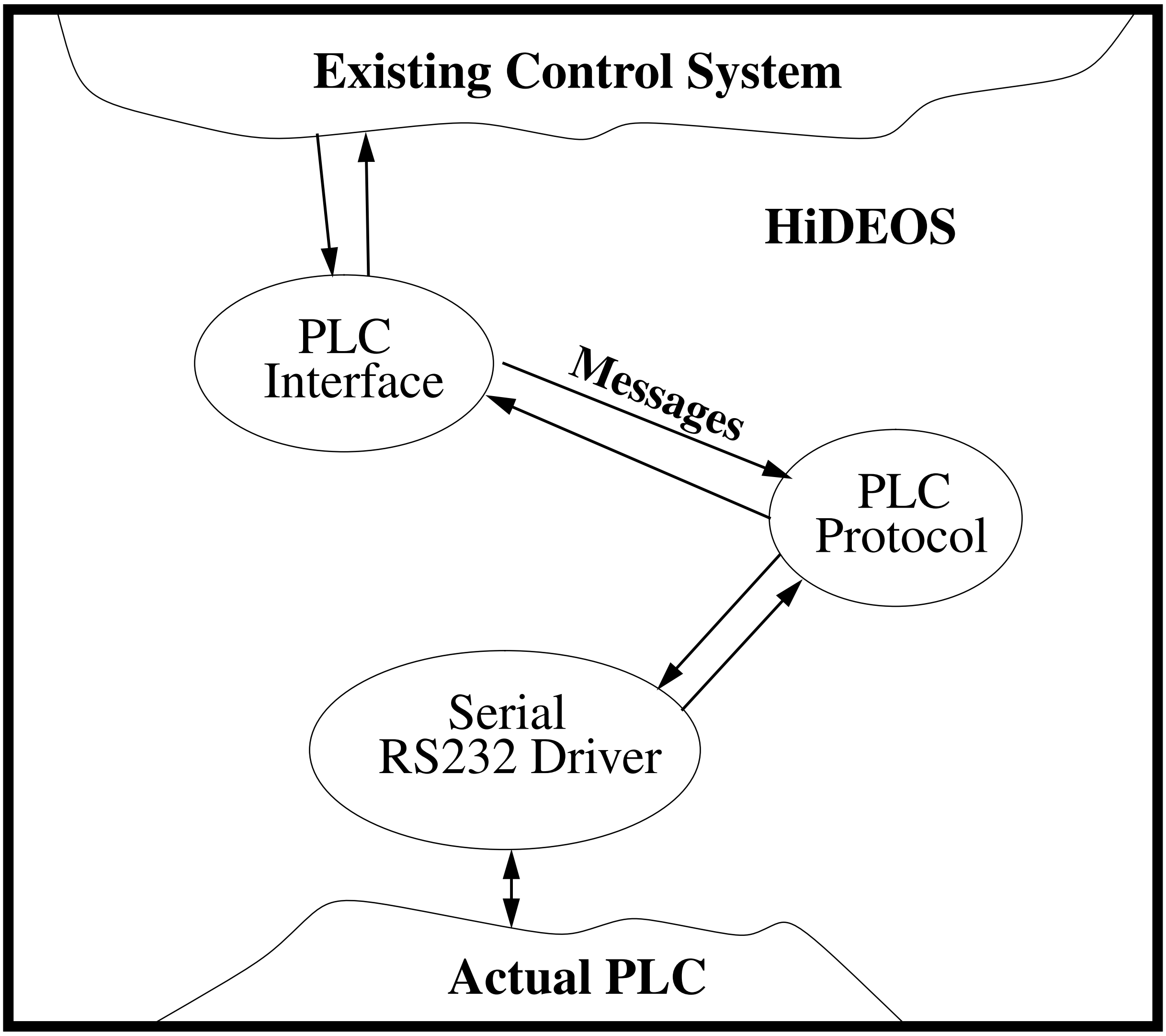
Jim Kowalkowski

Goals

- Create Object-Oriented Operating System Shell
- Embedded System
- Development Environment
- Distributed, Message Driven System
- Multi-Threaded, Multi-Processor Environment
- Implement in C++
- Inexpensively Operate Serial/GPIB

Processing Nodes

- Message or Event Driven
- Nodes Identified by Name
- Can “Hook” Nodes Together
- Communicate Using Messages
- Nodes Present on any Processor



Task Structure

Fundamental Unit From Which Users Develop Code

TaskBase

Stack Control and Context
Sleep, Watchdog Services
Scheduling/Suspension

↓

Task

Name Service - Locate Tasks
Send/Receive Messages
Wait for Message
Message Queuing

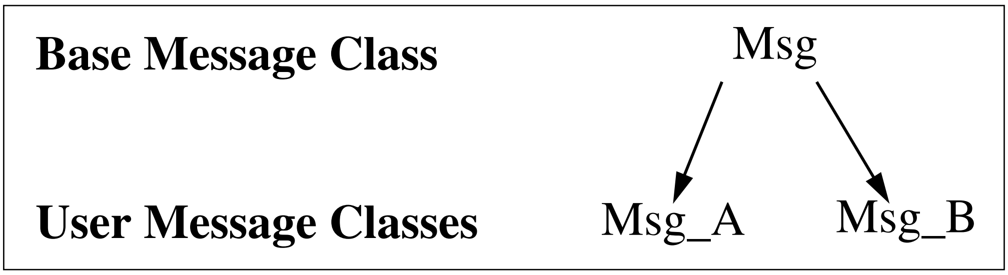
↓

UserTask

Override User Entry Point Receive
Decode Messages
Operate Devices,Protocols
Run User Code

- Each Instance of Task Class is a Process
- All Messages are Queued
- User "Receive" Called for each Message in Queue

Messages



Msg_A = 0 = Msg_A_Type
Msg_B = 1 = Msg_B_Type

- Each Derived Message Type Tagged with Integer value
- Tags Used to Get and Free Messages
- Simple Datagram Protocol with Routing Ability

Board-Level Services as Classes

- IPIC - Industry Pack Bus Controller
- VMECHIP2 - VME Bus Controller

Results

- Current Information
  - Interfaced to EPICS
  - Currently Runs Under vxWorks
  - Operate Secondary Processors in VME Crate Stand-Alone
  - Manages VME Backplane Communications Between Processors
  - Running Serial, GPIB, Analog I/O, Protocols at APS
  - Auto-Configuration and Management of IP Modules and Drivers
- Benefits
  - Quick Development Cycle
  - Maintenance
  - Distribute Load among several processors
  - No Software Licensing Fees
- Documentation Available via WWW

http://www.aps.anl.gov/asd/controls/hideos/intro.html