Generic component-based middleware for a peer-to-peer flexible robot architecture

• What is MT Robot AG?
• System description
• Middleware Description
• Generic Components
• Communication Patterns
• Preliminary Results
• Conclusions
What is MT Robot AG?

- AGV manufacturer for health care and production/manufacture
- Autonomous multi-robot navigation in existing logistic systems
- Our system’s characteristics are:
  - **Flexibility** and **easy-to-install** products according to customer needs
  - **Scalability** for extending software and hardware modules
  - **Automatic response** to system or environment changes
System Description

- Mission planning and execution for logistic operations
- Multiple-resource allocation
  - Passive: resource reacts mechanically
  - Active: resource provides feedback (state)
- Multi-purpose middleware for dynamic and inter-connected (generic) components
- Measuring performance of a complex and changing structure
- Near to real-time user interaction (GUI)
- **Customer extenuating environment conditions**
System Description
Middleware Description

• Basic functionality
  • Interfacing hardware and software components
  • Tasking processes (POSIX)
  • Managing middleware system commands

• Component communication:
  • Inter-process communication
  • TCP/IP

• Manage scalability and easy-to-deploy
Generic Components

- Message passing framework for inter-process and networked processes
- Peer-to-peer like system
- Minimalistic components as base of all programs
  - **Efficient** (how good?) vs **Effective** (how useful?)
- Run-time configuration and data sharing
- Defacto with continuously attachable capabilities
Generic Components

- Generic component characteristics:
  - Logging data remotely or locally (DB, files, etc.)
  - Brokerless process and node communication interface
  - Configuration based in data-dictionaries (meta-data)
  - Managing and reporting system performance (middleware commands)
Communication Patterns

• Multi-threaded mesh of interconnected processes and network distributed software components
  • E. g. algorithms, interfaces and drivers
• Patterns organise a continuous flux of data streaming
• Components compete for same processor or share same physical network
• Architecture based in messaging patterns
Communication Patterns

- **Request/response**
  - Synchronous or asynchronous two-way communication
  - Multi-process „Handshake“
  - E. g. Client/Server
Communication Patterns

- **Publish/Subscribe**
  - Asynchronous and filtered (topic) communication
  - Message queue
Communication Patterns

• **Pipeline**
  
  • Synchronising chains of processes with distinct execution time
  
  • Allocate messages from parallel executed workers
Communication Patterns

• Exclusive pair
  • Synchronously sent and reception of message passing
  • Single connection
Preliminary Results

- Experimental platform: GenuineIntel 1.60GHz, 2065440k RAM and a network frequency 5.2 GHz.

- Quantify performance of architecture
  - QoS like:
    - Connection state
    - Latency
    - Throughput
    - CPU usage and memory map
Preliminary Results

1. Acceptable latency for this network
Preliminary Results

Latency (10 packets of 10 bytes) every 4s

1. Acceptable latency for this network
2. Robot navigating in remote area
Preliminary Results

1. Acceptable latency for this network

2. Robot navigating in remote area

3. Network traffic peaks

Latency (10 packets of 10 bytes) every 4s

Iterations every 4s [2013-05-03 04:00:00-18:00:00]
Conclusions

• Interconnection of *generic components* of UnitR middleware

• **Multi-pattern messaging** system

• Adaptable components according to *customer requirements*

• Provide *quantitative measurements* of architecture style
Questions?

- Renato Samperio, PhD
- Senior Software Engineer

- MT Robot AG,
- Riedstrasse 16,
- CH-4222 Zwingen

- Tel: +41 (0) 61 775 20 23
- Fax: +41 (0) 61 775 20 21