

MicroQoS CORBA

A QoS-Enabled, Reflective, and
Configurable Middleware Framework
for Embedded Systems

A. David McKinnon, Tarana R. Damania, David E. Bakken,
Kevin E. Dorow, Wesley E. Lawrence

July 2002

MicroQoS CORBA Objectives

- Go beyond Minimum & Real-Time CORBA
- Be Small
 - Distributed sensor networks, home appliances
- Have QoS
 - Run time performance, security, fault tolerance
- Maintain interoperability
- Create S/W Eng. tools that aid the developer

MicroQoS CORBA Overview

- Lifecycle choices
 - Exploit limiting choices from the initial idea until the end of execution
- Quality of Service choices
 - Decide what to support (e.g., Security, Fault Tolerance, Real-Time performance)
- Development tools
 - Design Patterns
 - Profiling metrics & heuristics

Lifecycle Time Epochs

Lifecycle Epoch	Constraint Bound	Representative Examples
Design	HW Heterogeneity	Symmetric, asymmetric
	HW Choice	X86, TINI, ColdFire
	Communications HW	Ethernet, Serial, Infrared
	Processing Capability	50 Mhz, 1 Ghz, 8bit, 32bit
	System size	small, medium, large (e.g., transducers to jets)
	Power Usage	line, battery, and/or parasitic power
IDL Compilation	Communication Style	Passive, Pro-active, Push, Pull
	Stub/Proxy Generation	Inline vs. library usage
	Message Lengths	Fixed, variable length messages
	Parameter Marshalling	Fixed Formats
Application Compilation	Space/Time Optimizations	Loop unrolling, code migration, function and proxy inlining
	Library Usage	Static vs. dynamic library linkage
System /Application Startup	Device Initialization	
	Network Startup	Bootp, dhcp
	Major QoS adaptation	Select between QoS modules
Run Time	Minor QoS adaptation	Adjust QoS parameters

1. Initial Design

Embedded Hardware	Roles			SW I/O	IDL Subsetting
	Connection Setup	Data Flow	Interaction Style		
<p>System Comp.</p> <ul style="list-style-type: none"> • Homogenous • Asymmetric <p>HW I/O Support</p> <ul style="list-style-type: none"> • Serial, 1-Wire, Parallel, Digital, Ethernet, IrDA, Bluetooth, GSM, GPRS <p>Resources</p> <ul style="list-style-type: none"> • Memory • Power <p>Processing Capabilities</p> <ul style="list-style-type: none"> • 8-bit, 16-bit, ... 	<p>Initiates Conn. Setup</p> <p>Receive Conn. Requests</p>	<p>Bits In</p> <p>Bits Out</p> <p>Bits In/Out</p>	<p>Sync (Send/Recv)</p> <p>Async (One-Way Msgs)</p> <p>Msg Push</p> <p>Msg Pull</p> <p>Passive</p> <p>Pro-Active</p>	<p>Data Representation</p> <ul style="list-style-type: none"> • CORBA CDR • MQC CDR • ... <p>Protocols</p> <ul style="list-style-type: none"> • TCP/IP • UDP • PPP • 1-wire <p>Direct (i.e., IIOP)</p> <p>Indirect (i.e., IIOP Gateway)</p>	<p>Message Types</p> <p>Parameter Types</p> <p>Data Types</p> <p>Exceptions</p> <p>Message Payload</p>

2. Interface Compilation

- IDL Subsets
 - Verify conformance with design choices
- Application specific code generation
 - Leverage IDL subset meta-data
 - Optimize client stub/server skeleton code
- Representation Optimizations
 - Optimize static string references
 - Object references

3. Application Compilation/Linkage

- Space/Time optimizations
 - Use existing compiler techniques
- Library usage
 - Static
 - Dynamic

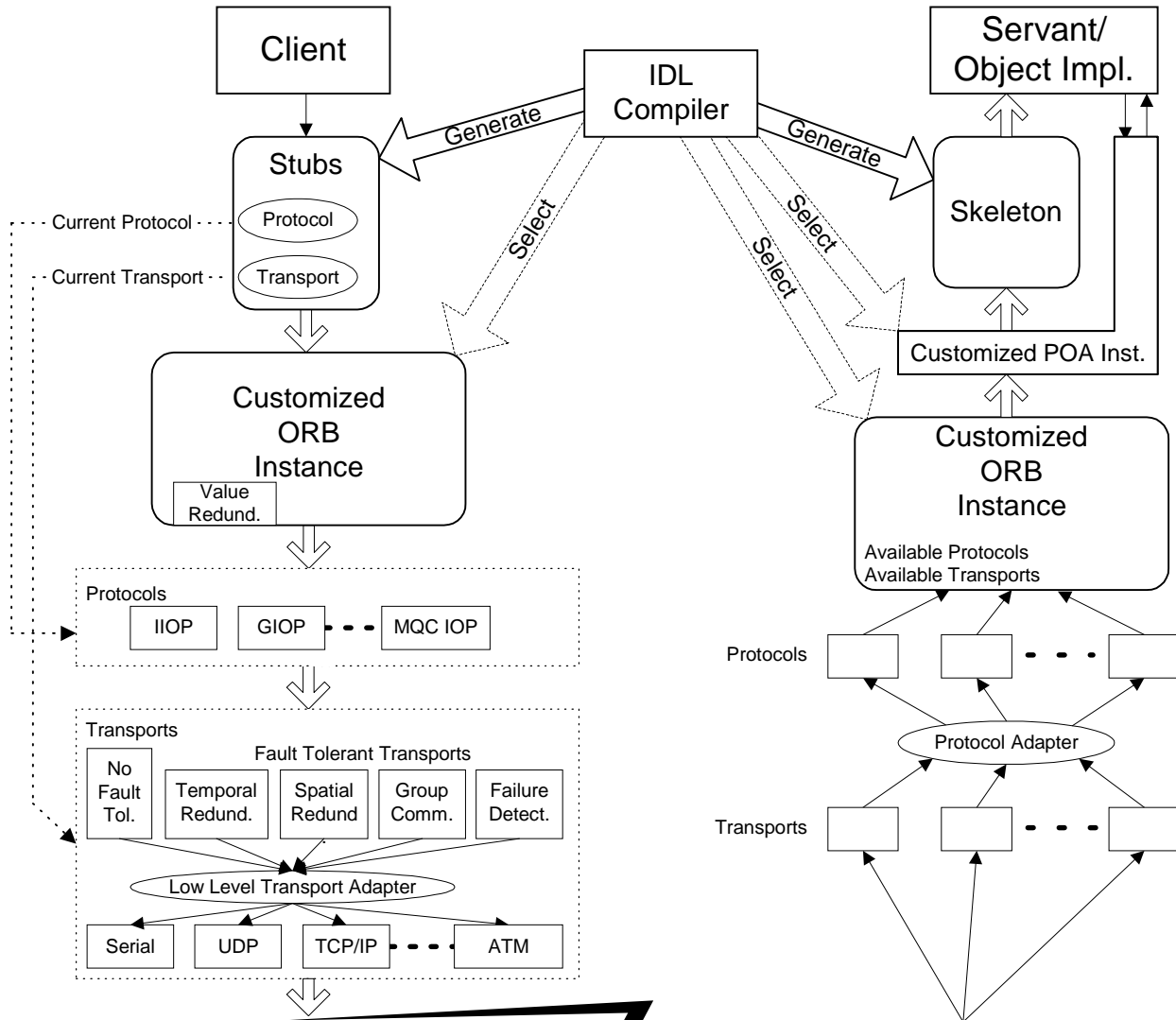
4. System Initialization/Startup

- Device initialization
 - Use reflection to load appropriate modules
 - Initialize QoS subsystems
- Network initialization
 - Locate other key nodes on the network

5. Run Time Execution

- Fixed functionality
 - Most common case
- Dynamic functionality
 - Increased Resource Usage

Architecture Overview



Quality of Service

- Security
 - Multiple strengths/Algorithms
- Fault Tolerance
 - Quantity and types of faults tolerated
- Real-time Behavior
 - Scheduling Algorithms, Network performance
- Resource Issues
 - Memory footprint
 - Power awareness

Fault Tolerance Taxonomy

Redundancy	Reliability	Ordering
<ul style="list-style-type: none"> • Temporal <ul style="list-style-type: none"> – Multiple transmits • Spatial <ul style="list-style-type: none"> – Multiple Channels – Replicated Servers • Value <ul style="list-style-type: none"> – Checksums, CRC 	<ul style="list-style-type: none"> • Group Communication <ul style="list-style-type: none"> – Best Effort – Reliable – Atomic – Uniform • Failure Detection 	<ul style="list-style-type: none"> • Sender FIFO • Causal <ul style="list-style-type: none"> – Logical Timestamping • Total <ul style="list-style-type: none"> – Sequencer / Token based

Configurability

- Each embedded system is slightly different
- Not everyone will need the kitchen sink
- Let the developer pick and choose
- Configure QoS properties, too
 - Multiple implementations/strengths for one property to choose from
 - Choose at startup with reflection/introspection

CASE Tools

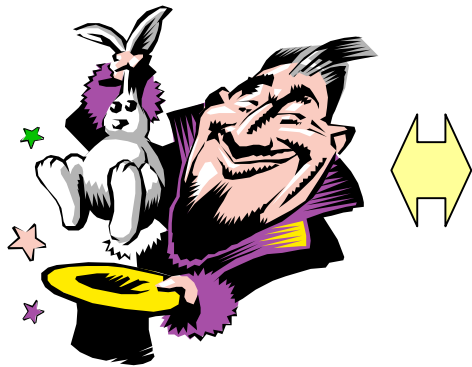
- Not all developers are created equal
- Make it easy for the casual programmer
 - Domain expert, but QoS novice
 - Lifecycle support personnel
 - Temporary/contract employees
- Let the tools choose compatible components based upon
 - QoS requirements
 - Resource configuration



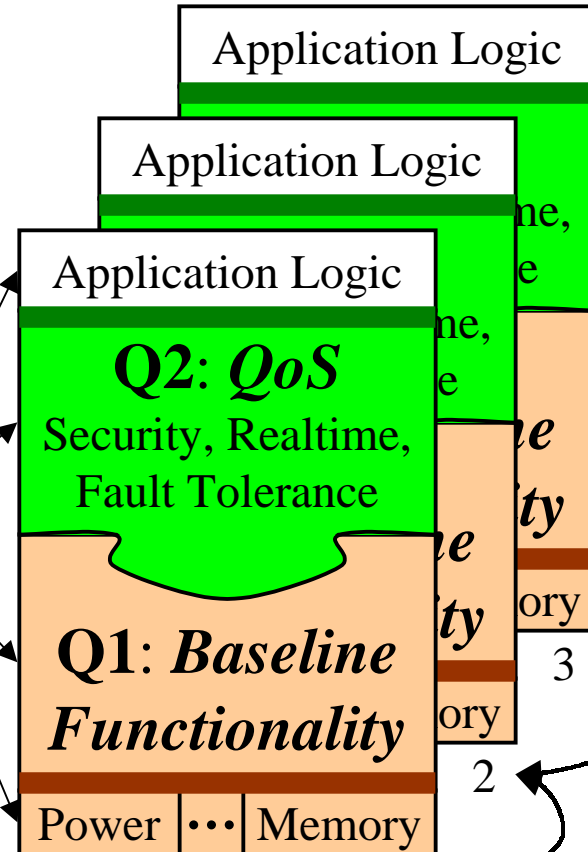
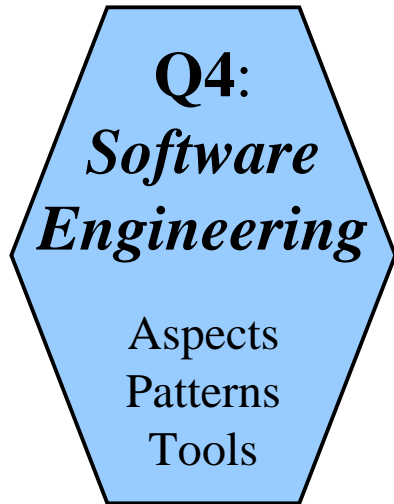
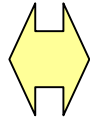
Key Questions

- **Q1: Baseline Constraints:**
 - What constraints must be imposed on middleware to achieve a small footprint (ignoring QoS for now)?
- **Q2: QoS Properties:**
 - What strength or “flavor” of QoS properties (security, fault tolerance, real-time) can reasonably be supported in resource-starved embedded systems?
- **Q3: Network-Wide Composition:**
 - What issues, optimizations, and tradeoffs must be made when a network of QoS-enabled devices integrated? How can we support global properties, metrics, and goals involving the entire distributed embedded system?
- **Q4: Software Engineering Issues:**
 - How can we use QoS aspects to make the middleware more manageable and quantifiable? How can we use patterns and tools to help the developer to be more productive?

Key Questions in Context



Embedded Systems
Application Developer



Node 1

Q3: Local-Global Interactions

MicroQoS CORBA Bottom Line

- MicroQoS CORBA the only framework that
 - Is a “bottom-up” rethinking from the device level of what should be configurable, and in what ways
 - Is tailorable for a given application to the wide range of
 - Device constraints, *and*
 - Application-dictated constraints
 - with a fine granularity of configuration constraints
 - Will support both “functional” and QoS properties
 - Not just realtime, but security and fault tolerance are also key QoS properties

Follow-up

- Please visit our poster for more details and a chance to ask questions!
 - Poster Session: Wednesday, 1500 – 1600
 - Memory footprint results
 - Performance profiling results
- MicroQoSCORBA website
 - <http://microqoscorba.eecs.wsu.edu>