Using the ACE Framework and Patterns to Develop OO Communication Software

Douglas C. Schmidt

schmidt@cs.wustl.edu
Washington University, St. Louis
http://www.cs.wustl.edu/~schmidt/

Sponsors
NSF, DARPA, Bellcore, Boeing, CDI/GDIS,
Kodak, Lockheed, Lucent, Microsoft, Motorola, OTI,
SAIC, Siemens SCR, Siemens MED, Siemens ZT, Sprint
Motivation: the Distributed Software Crisis

- **Symptoms**
  - Hardware gets smaller, faster, cheaper
  - Software gets larger, slower, more expensive

- **Culprits**
  - *Accidental* and *inherent* complexity

- **Solutions**
  - Frameworks, components, and patterns
Techniques for Improving Software Quality and Productivity

- **Proven solutions**
  - **Components**
    * Self-contained, “pluggable” ADTs
  - **Frameworks**
    * Reusable, “semi-complete” applications
  - **Patterns**
    * Problem/solution pairs in a context
  - **Architecture**
    * Families of related patterns and components
Why We Need Communication Middleware

- **System call-level programming is wrong abstraction for application developers**
  - *Too low-level* → error codes, endless reinvention
  - *Error-prone* → HANDLEs lack type-safety, thread cancellation woes
  - *Mechanisms do not scale* → Win32 TLS
  - *Steep learning curve* → Win32 Named Pipes
  - *Non-portable* → socket bugs
  - *Inefficient* → *i.e., tedious for humans*

- **GUI frameworks are inadequate for communication software**
  - *Inefficient* → excessive use of virtual methods
  - *Lack of features* → minimal threading and synchronization mechanisms, no network services
The ADAPTIVE Communication Environment (ACE)

- **ACE Overview**
  - A concurrent OO networking framework
  - Available in C++ and Java
  - Ported to VxWorks, POSIX, and Win32

- **Related work**
  - x-Kernel
  - SysV STREAMS

http://www.cs.wustl.edu/~schmidt/ACE.html
ACE Statistics

- ACE contain > 135,000 lines of C++
  - Over 15 person-years of effort
- Ported to UNIX, Win32, MVS, and embedded platforms
  - e.g., VxWorks, LynxOS, pSoS
- Large user community
  - www.cs.wustl.edu/~schmidt/ACE-users.html
- Currently used by dozens of companies
  - Bellcore, Boeing, Ericsson, Kodak, Lockheed, Lucent, Motorola, SAIC, Siemens, StorTek, etc.
- Supported commercially
  - www.riverace.com
Patterns for Communication Middleware

- **Observation**
  - Failures rarely result from unknown scientific principles, but from failing to apply proven engineering practices and patterns

- **Benefits of Patterns**
  - Facilitate design reuse
  - Preserve crucial design information
  - Guide design choices
Active Objects with ACE Tasks

- ACE Task Features
  - Queueing
  - Event demultiplexing
  - Concurrency
  - Dynamic linking
The ACE Stream Class Category

- ACE Stream Features
  - Layered service composition
  - Synchronous and asynchronous messaging
  - Dynamic configuration
Alternative Concurrency Models

- **Message-based Evaluation**
  - Low overhead
  - Harder to program

- **Task-based Features**
  - Higher overhead
  - Easier to program
Use-cases for ACE

- Domains
  - Medical imaging
  - Network management
  - Wireless communications
  - Real-time avionics
  - Multimedia services
Applying ACE to Medical Imaging

- Domain Challenges
  - Large volume of “Blob” data
    * e.g., 10 to 40 Mbps
  - “Lossy compression” isn’t viable
  - Prioritization of requests

- \(~schmidt/COOTS-96.ps.gz\)
- \(~schmidt/av.ps.gz\)
Applying ACE to Network Management

- Domain Challenges
  - Low latency
  - Multi-platform
  - Family of related services
  - ~schmidt/DSEJ-94.ps.gz
Applying ACE to Global PCS

- **Domain Challenges**
  - Long latency satellite links
  - High reliability
  - Prioritization
  - ~schmidt/TAPOS-95.ps.gz
Applying ACE to Real-time Avionics

- **Domain Challenges**
  - Real-time periodic processing
  - Complex dependencies
  - Very low latency
  - ~schmidt/JSAC-98.ps.gz
• TAO Overview
  - High-performance, real-time ORB
    * Networking and avionics focus
  - Leverages ACE
    * Runs on VxWorks, POSIX, and Win32

• ~/schmidt/TAO.html
JAWS Adaptive Web Server

- JAWS Overview
  - A high-performance Web server
    - Flexible concurrency and event dispatching mechanisms
    - Full HTTP 1.0 and CGI support
  - Leverages the ACE framework
    - Ported to most OS platforms

---

Washington University, St. Louis
Java ACE

- Java ACE Overview
  - A version of ACE written in Java
  - Used for medical imaging prototype

Java ACE Overview

- A version of ACE written in Java
- Used for medical imaging prototype

~schmidt/JACE.html
~schmidt/MedJava.ps.gz
~schmidt/C++2java.html
Lessons Learned Building ACE

• Good components, frameworks, and software architectures take time to develop

• Reuse-in-the-large works best when:
  – The marketplace is competitive
  – The domain is complex
  – Building middleware in-house costs too much
  – Corporate culture is supportive

• Produce reusable components by generalizing from working applications
  – *i.e.*, don’t build components in isolation

• The best components (and systems research) come from solving real problems
Concluding Remarks

- Developers of communication software confront recurring challenges that are largely application-independent
  - e.g., service initialization and distribution, error handling, flow control, event demultiplexing, concurrency control
- Successful developers resolve these challenges by applying appropriate design patterns to create communication frameworks
- Application frameworks are an effective way to achieve broad reuse of software
Obtaining ACE

- ACE is an OO framework that reifies key communication software patterns
- All source code for ACE is freely available
  - www.cs.wustl.edu/~schmidt/ACE.html
- Mailing lists
  - ace-users@cs.wustl.edu
  - ace-users-request@cs.wustl.edu
  - ace-announce@cs.wustl.edu
  - ace-announce-request@cs.wustl.edu
- Newsgroup
  - comp.soft-sys.ace