The Ocarina tool suite
http://ocarina.enst.fr

Thomas Vergnaud, ENST
ENST is doing research on middlewares for distributed real-time applications

- PolyORB, a generic middleware
  - Highly tailorable
  - PolyORB instances from the application requirement
- Need support for configuration & deployment

To generate a complete distributed application from its AADL description

- Generate code from the software components of the description
- Extract configuration information from the description
- Deploy the whole application

Need for lightweight AADL tools

- To be the PolyORB configuration toolchain
- To be associated with other tools
- To be integrated within existing applications
Set of tools to describe and generate a distributed application
  - Core library
  - Clients for this library
Comparable to a compiler
  - No IDE
  - External editors
Aims at generating an application
  - Architectural concerns

OADL workshop Thomas Vergnaud

OADL workshop

OADL (XML)
OADL (Text)
OADL (graphic)
Core library
Specific Analyzer
Code Generator
Model Transformation
Ada, C
AADL
Run-Time
Lightweight library to manipulate AADL descriptions

3 main functionalities:
- Build AADL descriptions
- Verifications
- Instantiation

Allows tree manipulation

Several I/O libraries:
- AADL (text) parser/printer
- AAXL (XML) parser/printer
- Dia (graphical) (XML) parser/printer
- 3ADL (assert project) (XML) parser/printer

I/O libraries are clients of the Ocarina core
- Allow multiple files & multiple syntaxes simultaneously
- Emacs mode
- Plug-in for Dia (a diagram editor for the GNOME graphical environment)
Several nodes, distributed over a network

Each node has 3 parts:
- User application
- Communication layer
- Local operating system

Need for a formalism
- Perform verification on the global system
- Enable components reuse
- Facilitate configuration & deployment description

AADL as an implementation description language
- Clear semantics, meant for system generation
- Last step before the actual implementation
Distribution models in AADL

- Message passing
- Remote procedure calls
- Shared memory
- Distributed objects

- The runtime should support them all
  - Distribution middleware: communications
  - Local execution support: thread scheduling
- ... and should be adapted to the application requirements
PolyORB: main principles

- A three-layer architecture
  - Neutral core: common generic middleware components
  - Personalities: CORBA, MOMA, DSA, AWS, SOAP, GIOP, MIOP
  - Profiled personalities: RT CORBA, FT CORBA

- Neutral core
  - Canonical middleware functions
  - Configurability

- Personalities
  - Adaptation layers
  - Interoperability
  - Genericity

- Eases middleware prototyping
  - Only a new personality to create
  - The neutral core remains unchanged => facilitates verification

- A PolyORB instance is a middleware
Define canonical middleware functions
- Addressing, Binding, Representation, Protocol, Transport, Activation, Execution
- Each building block (function implementation) includes:
  ✓ Interface (independent from distribution model)
  ✓ One or more implementations
  ✓ Behavioural (formal) description (when needed)

μBroker at the core of the middleware behaviour
- Allocate task to handle I/Os, requests
- Schedule tasks, dispatch requests
- Manage middleware state
The middleware is a software part of each local application node
- Collection of subprograms integrated in each process
- Node configuration
  - Components of the personalities
  - Configuration of the implementation of some services
  - Number of threads to use
  - Scheduling policy
- The core runtime is provided by
  - the µBroker
  - the lower layers (sockets, …)

Operating system

### AADL Application

### AADL Application personality

### Protocol personality (e.g. GIOP)

- binding
- protocol
- transport
- representation
- execution
- activation
- addressing
Process overview

Appl. Arch. (AADL) → Deployment Tool → Deployment Data

Configuration file for MW instance → Code Generator

PolyORB generic AADL description → PolyORB sources repository

PolyORB formal descriptions repository → Code Generator

Logical node formal description → Configured sources for PolyORB

Process for each application node
Gaia: overview

- Gaia is an application generator that processes AADL descriptions
  - Relies on the Ocarina core library
- Identification of the description elements:
  - AADL process
    - Application node
  - AADL thread
    - Hook in the runtime
    - Not necessarily an OS thread
  - Subprograms & data
    - Procedures and data types
- 2 main parts
  - Analyzer / processor
    - Semantic verification
    - Architecture configuration
  - Code generator
    - Application part: encapsulation of the component application code
    - Runtime part: tailored support for the application execution
Translate AADL components into source code
  - Generate Ada
  - Can be adapted to other languages

Instantiate a tailored runtime based on PolyORB
  - Execution runtime: thread creation
  - Distribution runtime: communication management

Generate source code for middleware specific components
  - AADL personality
  - Neutral layer services

Generate PolyORB configuration files
Example: Bulletin Board System

- Classic BBS application
  - Clients send messages to servers
  - Servers broadcast received messages to all clients

- Relies on a publish/subscribe architecture
  - Common in real-time distributed architectures

- Servers
  - An aperiodic thread dispatched by incoming messages

- Clients
  - A periodic thread that sends messages
  - An aperiodic thread that receives messages

- All threads call subprograms
  - Associated with source code implementation
The Ocarina tool suite

PolyORB
- Free Software available at http://polyorb.objectweb.org
- Support provided by AdaCore at http://www.adacore.fr/addons_polyorb.php

Automatic application code generation
- From AADL description
- + implementation files (source code or formal method)

Use of a runtime
- Providing full range of distribution models
- Based on a customized middleware architecture
- With respect to the application requirements

Ongoing work in the context of IST-FP6 ASSERT
- Ocarina and Gaia provide an experimentation support
  - Modelling of real-time systems
  - Modelling of the middleware (virtual machine)