Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

May 95 TC presentation: DIMMA Progress Report

Youcef Laribi

Abstract

This talk gives an update of the progress made by the "Distributed Multimedia" group since the last TC in the areas of :

1- DIMMA Nucleus engineering: The nucleus engineering work has been concentrated on the design and implementation of an object-oriented framework to cope with the increasing complexity of advanced OS features and diverse functional requirements of different object adaptors. A version of the Nucleus is targeted for release by the next TC.

2- The runtime : The runtime offers an API for applications (and stubs and will use the underlying engineering engine features to implement them. The current API state supports objects, signatures, interfaces, invocation references, terminations, basic types and local garbage collection. A handcrafted trader server stub has also been developed to reuse the ANSAware trader. By the next TC, we expect supporting structured types concurrency and service withdrawal. The resulting runtime will use the DIMMA nucleus as its only engineering engine in a first stage.

3- Stub generation: In the DIMMA environment, service interfaces can be described in different IDL languages, as long as the semantics of these IDL languages can be mapped into common semantics. Similarly the stubs generated from these IDL descriptions can target different computational models, expressed probably in different languages.

APM.1496.01.02 **Draft** 1st June 1995

Project Management (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

DIMMA Engineering Update

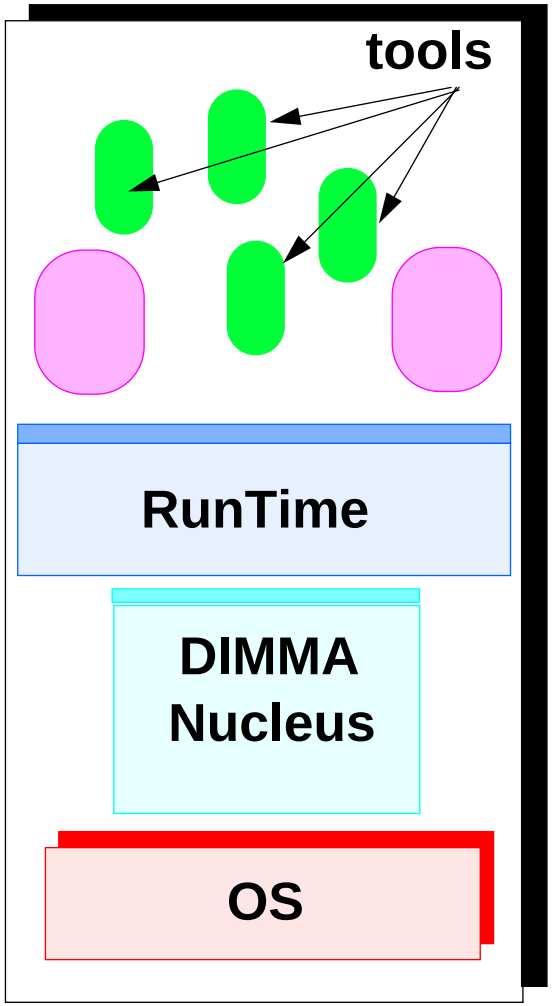
Youcef Laribi (yl@ansa.co.uk)

DIMMA Objectives

- **Developing a platform suitable for distributed multimedia and real-time programming.**
- **Using standard APIs to ensure stable and portable applications.**
- **Combining different computational models within the same platform (ODP, CORBA, TINA).**
- **Supporting new models of distributed interaction: Streams & Signals.**
- **Allowing flexible and user-controlled resource management (e.g explicit binding).**

The First Stage...

- Designing and implementing a programming interface and the associated runtime for which programmers can write stable applications.
- Develop the set of tools necessary to generate stubs for the above runtime from different IDL service descriptions.
- Design and implement an engineering engine (DIMMA Nucleus) which will handle distributed multimedia and real-time requirements and interface the upper runtime and the underlying operating systems.



DIMMA Engineering Practice

- C++ as the programming language.
- Self-contained software.
- Ensure portability on different OS platforms.
- Manage the building process and the source tree using GUIs.

The Runtime Component.

- Provide applications with standard APIs (CORBA, TINA, ODP).
- Support several engineering engines with possibly different capabilities.
- Client/Server Stubs are completely independent from the protocols used for marshalling/unmarshalling.
- Using a uniform engineering API for stubs, DII and DSI.

Runtime progress status

- **Defined and implemented interfaces and semantics for a set of computational abstractions:**

  - Objects - Invocation References - Interfaces

  - Signatures - Exceptions (Terminations) - Basic Types

- **Supported utilities:**

  - Local garbage collection. - Tracing/debugging.

- **Tools:**

  - A capsule manager. - A handcrafted trader server stub.

AST-based tools for stub generation

- **Objectives:**

  1- Capture the semantics of an interface into an Abstract Syntax Tree (AST).

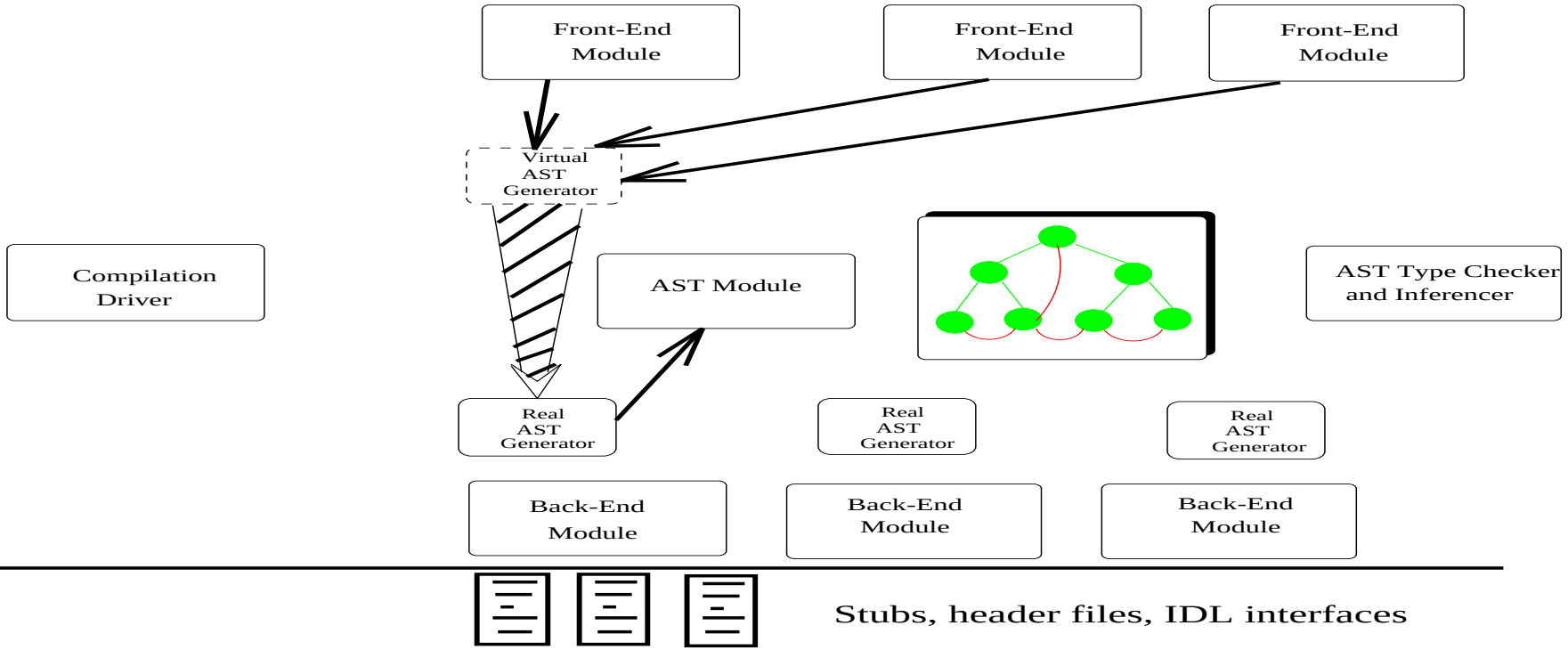
  2- Perform the semantic analysis (type inferencing and checking) on an AST tree independently of the IDL originally used to describe that interface.

  3- Construct tools that generate ASTs from IDL descriptions or from header files (front-ends).

  4- Construct tools that generate stubs or alternative IDL descriptions, from an AST (back-ends).

AST Tools architecture

Input IDL



Stubs, header files, IDL interfaces

Stub Generation Progress Status

- Developed a C++ back-end targetting the runtime API and experimenting with the description of multimedia and real-time requirements in an IDL.
- Fixed stable interfaces between different module types.
- Integrated with the Type Checker and Inferencer work done within the ANSA C4a workpackage (AST type checking and inferencing).
- Working on the integration of the CORBA IDL front-end into our tools platform.

DIMMA Nucleus

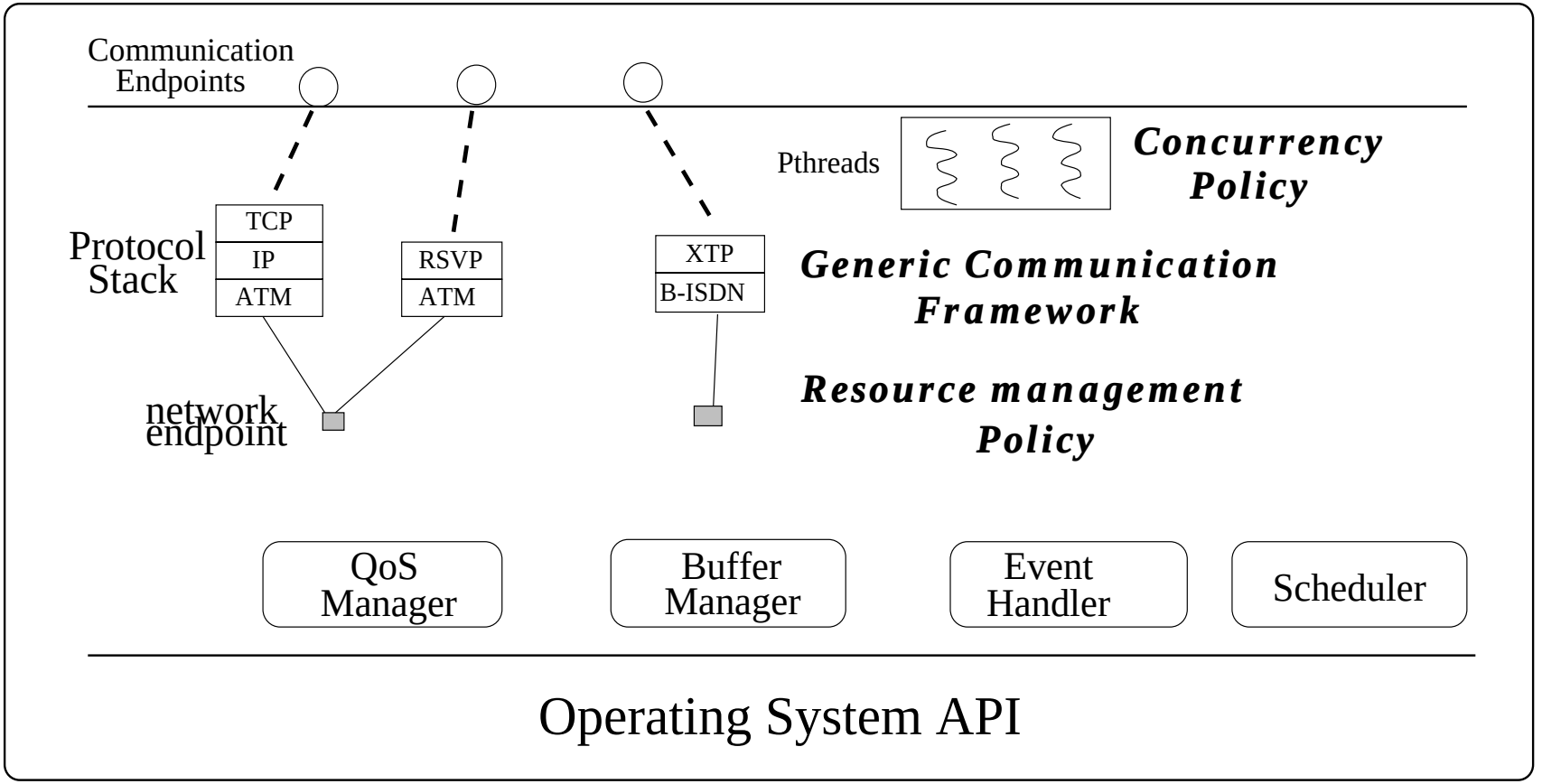
Motivation:

- A desire to understand multimedia and real-time engineering issues.

Objectives:

- Target a lightweight, highly modular and efficient engine.
- Interwork with other ORBs (using IIOP).
- Support several personalities (CORBA, ODP, TINA).
- Simultaneous support of several, dynamically configured protocol stacks.
- Support of QoS processing.

DIMMA Nucleus Architecture



Current Progress Status

- Generic Communication framework.
  - Protocol module drivers for TCP and UDP.
  - A simple RPC protocol (doesn't understand concurrency yet).
- Scalable data management scheme (Uses an i-node like indexing algorithm for locating interface references from handles).
- C++ wrapper patterns for the Posix threads package.
- Timing management.

The Next Steps

- **Finish the runtime component by adding the necessary features and mapping them to the nucleus counterparts.**
- **Understand the translation between CORBA and other computational models (e.g TINA and ODP) and incorporate the CORBA IDL front-end into the stub generator.**
- **Working on a nucleus release by the next TC.**