Universidade Nova de Lisboa (UNL)
Faculty of Science and Technology
Department of Informatics
CITI – Centre for Informatics and Information Technologies

Parallel and Distributed Processing Group
José C. Cunha, Professor
Overview of Recent Activities
17 January 2008 – CoreGRID WP3 Meeting
**Parallel and Distributed Processing Systems**

- **Mission**: research and education in parallel and distributed computing with a focus on selected themes:

- **Key Researchers**
  - José C. Cunha, PhD
  - Team coordinator
  - Pedro D. Medeiros, PhD
  - João Lourenço, PhD Nov. 04
  - Fernanda Barbosa, PhD Dec. 04
  - Vitor Duarte, PhD May. 05
  - Hervé Paulino, PhD Dec. 06
  - Rui Marques, PhD Nov. 07
  - Cecília Gomes, PhD Nov. 07
  - Carmen Morgado, MSc, PhD in 08
  - Paulo Lopes, MSc, PhD in 08

- **PhDs**
  - # PhD Completed in 2003-2006: 4
  - # PhD Completed in 2003-2007: 6
  - # PhD Submitted in 2007: 1
  - # PhD Ongoing 2 [+4 starting in 2008]
Challenges

Applications

Large volumes of data, requiring
  Efficient and intelligent management and search
  Parallel and distributed processing

Dynamic, distributed, and mobile application entities, requiring management of
  Structure, interaction, and coordination

Integration of distributed, heterogeneous components in dynamic and interactive
  environments

Organisation of small/medium/large scale collections of distributed, intelligent entities

Computing Environments

Increased levels of concurrency, parallelism & distribution

New forms of dynamic behavior
  increasing levels of interaction among system and application components
  more frequent changes in interaction and behavior
  more frequent changes in system and application configurations
  mobility

Increasing scale in terms of system and application components
Main Goal  to address the challenges

- The uptake of **new generations of parallel and distributed computing applications** is being restricted by the availability of suitable abstractions, methodologies, and tools.

- Our goal is to identify:
  - **Abstractions, models and tools** that can be integrated in **parallel and distributed computing environments**, along with problem-domain tools,
  - to enable the deployment of applications.
  - with **improved functionality and efficiency**
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
Software Environments for Grid Computing
Collaboration with Omer Rana, Cardiff U., Distributed Collab. Group (2003-...)

References:
One completed PhD thesis: Cecilia Gomes, 2007
Projects:
- EU AsiaLink Curriculum Development for HPC and Grid Computing (2005-08)
- Multidisciplinary projects at FCT/UNL
Research Context: Application Development and Control in Grid Environments

- Grid environments provide access to diverse heterogeneous resources/services across different organisation boundaries (e.g. high-performance computing; scientific tools; large scientific databases).

- Grid programming is a complex task:
  - Deployment across heterogeneous platforms providing different qualities of service, and with diverse access rights.
  - Orchestration of the diverse resources/services to meet the applications’ requirements.
  - Users with different profiles require diverse levels of transparency to Grid resources’ access and control (e.g. transparent access versus deployment and execution control).
  - Grid environments are dynamic in nature:
    - unavailability/addition of services/resources at some time;
    - the number of users vary in time;
    - cope with the application requirements’ modification on the fly.
Research Problem: Simplification of the Grid Programming Process

- Grid Programming still lack adequate support during the application’s life cycle, namely, in the dimensions of:
  - Higher-level abstractions for structured application configuration and coordination.
  - Structured execution control (e.g. suspend/resume part of an application).
  - Reuse of configurations (including behavioural interactions).
  - Dynamic reconfiguration.

- Different Grid applications present similar *interaction patterns*:
  - a pattern captures common knowledge and experience and describes how a similar set of experiments are to be set-up and managed;
  - it is advantageous to identify those patterns and reuse them.
M. Cecília Gomes, PhD Thesis, November 2007, FCT/UNL

**Goal:**
To abstract common structures and behaviours
To integrate them in software environments

**Approach:**
Model for reuse and flexible composition:

**Structural Patterns:** *Pipeline, Star, Ring, Proxy, Facade, Adapter, ...*

**Behavioural Patterns:** *Master/Slave, Producer/Consumer, Peer-to-peer, ...*

Patterns as *first-class entities* in application life-cycle:
- development
- execution
- reconfiguration

Model integrated in a prototype extending Triana workflow system, in collaboration with Cardiff Un.

**Structural Operators:** *Increase, Decrease, Extend, Reduce, Embed, Extract, Group, Rename/Reshape, ...*

**Behavioural Operator:** *Start, Terminate, Log, Stop, Resume, Restart, Limit, Repeat, ...*

(proof-of-concept)
Approach: Patterns Manipulated by Pattern Operators

- Reuse and flexible composition of typical Patterns which capture common usage configurations and interactions:
  - **Structural Patterns**: Pipeline, Star, Ring, Proxy, Façade, Adapter.
  - **Behavioural Patterns**: Master/Slave, Producer/Consumer, Peer-to-peer, Itinerary (mobile agents), Remote execution, Service Design Patterns for Grids, …

- Patterns as *first-class entities* during all phases of an application life-cycle:
  - development units
  - execution control units
  - reconfiguration units
Pattern Operators

- Invariant on the semantics of the manipulated patterns (e.g. a star will remain a star after being operated), and independently of the development stage (development, execution, reconfiguration).

- Several operator categories provide application manipulation in different dimensions. E.g.:
  - **Structuring and Grouping Operators**: increasing/decreasing the number of elements; build hierarchies; replace patterns;
  - **Execution Operators**: suspend/resume execution; restart execution periodically; repeat execution;

- Amenable to automation through scripts
Complex cycle of user activities in a PSE

1. Problem specification
2. Configuration of the environment:
   - Component selection (simulation, control, visualisation) and configuration
3. Component activation and mapping
4. Initial set up of simulation parameters
5. Start of execution, possibly with monitoring, visualisation and steering
6. Analysis of intermediate / final results
7. Reconfiguration
Allow to build applications in a structured way:

-- select appropriate set of patterns, combine them according to operator semantics,

-- define new patterns and operators found useful and add them to the environment

1. Deploy Structural Patterns,
2. Refine them through Structural Operators,
3. Use Behavioural Patterns to define control/data flow/interactions,
4. Use Behavioural Operators to manage execution
Pattern-based Methodology for Application Development

- Structural Operators
  - refine / inquire / compose
  - Structural Pattern Templates

- Behavioural Operators
  - control execution / change dependencies
  - Behavioural Operators

- Structural Pattern Templates
  - applied to
  - Structural Configuration

- Behavioural Pattern Templates
  - used for the instantiation of
  - Template Configuration

- Components / Tools / Services

- Behavioural Operators
  - produced transformation
  - application direct manipulation
  - Behavioural Operators

- Running Application
  - ownership operations
  - launch execution

- Application Configuration
E.g.: Structuring Operators

Pattern

Result Pattern

Increase(2, pipelinePT)

Extend(facadePT)
Parallel and Distributed Processing

E.g.: Execution Operators

PSE
\{RingSP + StreamingBP\}

Problem Solver

Database System

DataStoring
\{PipelineSP + StreamingBP\}

Monitoring service (adapter)
adaptee (legacy code)

Selected data
source
destination

Steering Interface

SteeringInt
\{ProxyPT + Client/ServerBP\}

Database System
\{StarSP + Master/SlaveBP\}

Database (master)

Database (slave1)

Database (slave2)

Database (slave3)
Open Issues / Future Work

- Coordination and dynamic issues
- Pattern-driven / operator controlled, run-time reconfiguration
  - On-demand (by the User)
  - Automatic (triggered by the middleware)
- Extensive evaluation of our approach on real applications (e.g. concerning expressiveness and scalability)
A GeoScience View: Application of computer algorithms for reservoir and aquifer modelling

Collaboration with Center for Geological Science FCT/UNL

- **Data**
  - Geological units
  - Permeability
  - Porosity
  - Fluid saturation
  - Exploitation and recovery

- **Static model**

- **Dynamic model**

- **Outputs**

  \[ \text{output} = f(i_1, i_2, i_3 \ldots) \]

**Ongoing and future work:**

- Two new PhD theses start Jan’08 at CITI - Scientific Workflows for Parallel and Distributed Computing
- Post-doctorate on HPC and Grid Computing - GeoInfo Project at CITI starting March’08, 5-years contract, Portuguese FCT
- New Project proposal planned with Cardiff Un.
Scientific Workflows Abstractions and Models for Parallel and Distributed Computing

Center for Geological Sciences FCT/UNL:

Numerical simulation of properties with sequential geostatistical algorithms (static model).
Simulation of reservoirs with integration of seismic information (static model).
Upscaling of permeability (interface static-dynamic model) and simulation of fractures.
Stochastic inversion problems (conditioning of static models to dynamic data)

CITI / Parallel and Distributed Processing Stream:

Workflow models are increasingly used to specify scientific applications;

Preliminary experiments with Triana, Kepler and Taverna have shown that many issues are still open;

Expressiveness of Workflow Models: Large-scale parallel and distributed
Event-driven and asynchronous notifications / Sharing state / Data access and transfer / Adaptive and dynamic

Flexible mappings to the execution support environments:
Configuration/Deployment / New requirements for workflow engines
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
Group-oriented Abstractions and Models

Groups at distinct abstraction levels:
- As Application Units
- As Programming Units
- As System Units

GroupLog, an abstract model: agents, groups, forms of interaction

Distinct instances, at distinct abstraction levels:
- Logic-based instance: GroupLog
- Java-based instance: JGroupSpace with cluster implementation

MAGO: distributed interactive applications

- Ongoing M.Sc thesis: 1
- Project: 1 (InStory with IMG CITI Stream)
MAGO - Modeling Applications with a Group-based Approach
Group-based approach for modelling interactive distributed applications

Main goals:

- capture the group dynamics
- manage the interactions between group members
- offer different kinds of communications mechanisms
- share data inside each group
- create explicit and implicit groups
- apply to a diversity of areas
MAGO - Modeling Applications with a Group-based Approach

- MAGO proposes a model to ease the development of interactive distributed applications
- the model encompasses a set of primitives and services specialised for environments supporting group based applications
- allows the organisation of systems in terms of multiple groups, each group being considered as a unit of system composition with a well defined interface
- multiple entities can dynamically enter and leave groups, and distinct forms of communication among group members are allowed
  - **direct**: direct communication between entities
  - **events**: multicast information to a group
  - **shared space**: share information inside each group
- information system maintains the data associated to members and groups
MAGO Model

APPLICATIONS

MAGO MODEL

- Entity management
- Group dynamics
- Communication interactions
- Implicit groups
- Information system access

Information System

Middleware support
(groups + events + shared space)

Infrastructure
(network + operating system)
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
Service-Oriented Computing and Mobility

Seamless combination of service and software mobility paradigms.

The **Mob** programming language:
- High level model for programming distributed applications
- Agents can provide/require services
- Strong mobility of agents on top of Java Virtual Machine
- Bindings for agents based on services provided by agents

**Compiled to the DiTyCO process calculus**
**Run-time system built on top of the one for DiTyCO.**

Protocols for reliable communication in the presence of mobility.

**Results:**
- PhD thesis: **1** (Hervé Paulino, Dec’06)
- Ongoing MSc Thesis: **1**
- Projects: **1** (MIMO, with U.Porto)

**Ongoing directions:**
- To feature the model in a Java framework
- **Service-centric mobility**: extension to express mobility in terms of services rather than hosts.

**Service-oriented middleware for wireless sensor networks** in the Callas national project with U. of Porto and Lisboa.
Service-Oriented Computing and Mobility

Mob compilation process

- `program.mob` to `Compiler` to `name service`
- `program.tyco` to `Libraries`
- `program.ml` to `Run-time`:
  - `JIT compiler` to `bytecode`
  - `bytecode` to `agent layer`
  - `agent layer` to `host layer`
  - `host layer` to `MIL VM`
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
**Multi-threaded Tabling in XSB Prolog**

**Tabled Logic Programming**
- Guarantees termination for datalog logic programs
- Reduces the temporal complexity for many logic programs

**XSB – a Tabled Prolog System**
xsb.sourceforge.net – developed in cooperation:
- SUNY at Stony Brook - USA
- Universidade Nova de Lisboa - Portugal
- Katholieke Universiteit Leuven - Belgium
- Uppsala Universitet – Sweden

Integrates the latest research results in tabling in a Prolog system for real world applications

**Multi-Threaded Tabling extensions**
- Explicit parallelism allowing several queries to be executed concurrently in XSB
- Programming model integrates thread-private and thread-shared tables

**Results:**
- **One PhD thesis**: Rui Marques, Concurrent Tabling: Algorithms and Implementation, November 2007 (supervisors: José C. Cunha and Terrance Swift, SUNY at Stony Brook)
- **Multi-threaded XSB is publicly available and is being used by the community**
Multi-Threaded Tabling in XSB

Strong aspects of the work

- Provided an useful extension to an already useful and intensively used system
- XSB had more than 500 downloads per month in 2007
- Demands continued maintenance and user support

Open Research Directions

- Integrate more Tabling features with the multi-threading extensions – XSB keeps integrating new tabling features
- Extend XSB with distributed programming features (begin with MPI-based extensions)
- Explore Table Parallelism – a new form of implicit logic parallelism, other than the traditional and/or parallelism

Performance Results on a 2x2 Core

Average speedups on an 2x2 core machine for some programs with no data dependencies

![Graph showing speedups for different programs and CPU counts]

Commercial Applications by XSB inc. (www.xsb.com)

- Weapon System Impact Tool, Warwick - Government Business Intelligence, Deductive Spreadsheet, etc
- Rely on CDF (Ontology Classifier program that uses Tabling in XSB) – starting to be Multi-threaded for Parallelism and XSB-Java Interface (already multi-threaded)

Research Applications by the user community

- Flora (Logic Object-oriented Transactional Language) – to be Multi-threaded for Distributed Agent Environments
- XMC (model checker) – to be Multi-threaded for Parallelism
To develop distributed computing models and architectures to support distributed agent systems.

A basis for supporting planning, decision support, and intermediation between user and system levels in distributed grid computing systems (*Semantic Grid*).

How agent abstractions can support reasoning, planning, intelligent decision support and intermediation:

- Between the user and a PSE
- To support reactive and autonomic behaviour at application and system levels.
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming

Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
Software Transactional Memory

[since mid 2006]

- Concurrency control on shared-memory (SMP and multi-core) architectures
  - Optimistic approach
  - Higher level of abstraction and less error-prone

- Current and future research lines
  - Performance optimization
    (1 MSc concluded, 1 undergrad student)
    - Algorithms, monitoring, benchmarking
  - Increase functionality
    (2 MSc ongoing)
    - Development of a Transactional Filesystem
    - Unified model with database transactions
  - Correction testing and debugging
    (1 MSc ongoing)
    - Dynamic instrumentation of STM-based source code
  - Applications of STM
    (1 MSc ongoing)
    - In production environments
    - In speculative execution
    (participation in the Byzantium Project - DSG group)

Results (in 2007):
- MSc Thesis: 1 completed, (4 ongoing)
- Prototypes: 1
- Projects: 1 (with DSG, 2008)
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
**Distributed Monitoring**

- Monitoring of parallel/distributed applications:
  - Application and system behavior
  - Performance and QoS
- Services as basic building blocks
- The system architecture is independent of the application runtime or instrumentation technique used

**Results:**

- PhD thesis: 1 (Vítor Duarte 2005)
- Prototype: 1 (DAMS)

**Projects:** 3
- Ongoing PhD: 1 (started 2007)
- Ongoing MSc: 1

**Ongoing work:** monitoring distributed applications behavior, performance and resource usage

**Case studies:**
- Scientific workflow applications on Grid systems
- Distributed Java applications

---

**Parallel and Distributed Processing**

**CITI**
Flexible Interface for Distributed Debugging – Library and Engine

- Interactive debugging of distributed applications
- Support for multiple debugging tools operating simultaneously over the same target application
  - Allow debugging activity at different abstraction levels
  - Providing multiple distinct views of the target application state

Results:

PhD thesis: 1 (João Lourenço, Nov´4)

Prototypes: 1 (FIDDLE)

[Diagram showing the interaction between clients and the Fiddle debugging engine, with labels for debugging commands and state information.]
Parallel and Distributed Processing Systems

Problem-Solving Environments and Grid Computing

Abstractions & Models
- Group-oriented Abstractions and Models
- Service-oriented Mobile Computations
- Parallel and Distributed Logic Programming
- Software Transactional Memory

Tools for Parallel and Distributed Program Development

Cluster Computing Systems
Cluster Computing Systems

Runtime environments for high-performance computing

Core:
- Lightweight inter-process communication mechanisms
- **File systems for high-performance computing**
- Parallelization of science and engineering codes

Emerging
- File systems for grid systems

**Results** (in the three core topics):

- **M.Sc thesis**: 1 [+2]
- **PhD thesis**: [+1 ]

(Paulo Lopes, to subm. Mar 2008)

**Approach:**
- pCFS, a parallel file system for clusters with shared disk systems;
- uses cooperative cache and all interconnects available to achieve high performance and reliability.

- Cooperation with IRISA and KerLabs; extension to wide area settings
- Participation in CoreGRID WP2 – Knowledge & DataMgt Task since mid-2007
Parallelization of science and engineering codes

**Context:** Using clusters and pools of PCs for executing parallel versions of science and engineering codes

**Application areas:** [Multidisciplinary Collaborations]
- Tomography in Materials Science
- Pollutant dispersion in shallow waters in Environmental Science

**Approach:**
- Shared and distributed memory
- Standard message passing (PVM/MPI) and .NET
- Dedicated frameworks

**Results:**
- MSc Thesis: [+2]
- Ongoing MSc Thesis: 2
- PhD Thesis: 1 [starting in 2008]
**File Systems for High Performance Computing**

**Context:** Efficient and reliable file access in cluster environments

**Approach:**
- **pCFS**, a parallel file system for clusters with shared disk systems;
- Uses cooperative cache and all interconnects available to achieve high performance and reliability.

Cooperation with IRISA-Rennes
  Christine Morin and KerLabs;
  extension to wide area settings

Participation in CoreGRID
  WP2 – Knowledge & DataMgt Task since mid-2007

---

Paulo A. Lopes, Pedro D. Medeiros, Cooperative Caching in the pCFS Parallel Cluster File System, HPDC 2006

Paulo A. Lopes, Pedro D. Medeiros, pCFS: A Parallel Cluster File System, ParCo 2005
Lightweight communication mechanisms

Context: Reducing latency in communication between processes in distributed memory multiprocessors

Approach: Using dedicated hardware (e.g., Infiniband) and/or specialized libraries to reduce the overhead associated with the sending and receiving of messages; influence of communication performance in CPU load

Results:

- MSc thesis 1 (R. Espenica, The PVM System over the VIA architecture, 2003)
- Participation in FCT-funded project TARDE (2002-04)
- Background work: R. Espenica, P. Medeiros, Porting PVM to the VIA architecture using a fast communication library, in EuroPVM/MPI 2002

Some results published in two publications about pCFS
Projects

- **Projects**
  - InStory - Interactive Personalized Mobile Storytelling, 2004-2006, with IMG CITI Stream
    - [Applications of Groups]
    - [Mob Language]
  - TARDE – Tabulation and Revision in a Distributed Prolog Environment, 2002/04, with CENTRIA
    - [Multi-Threaded XSB]
    - [Mob Language]
  - Byzantium - Efficient Byzantine fault-tolerant database replication, 2008-2010 with DSG CITI Stream
    - [Transaction Memory][Monitoring]
  - Portugal-Spain Bilateral Cooperation UNL – UABarcelona, 2007-2008
    - [Performance and Monitoring Tools]
  - IBM SUR Grants – Shared University Research Program for Parallel and Distributed Computing at CITI (2003 and 2007)

- **Related to Grid Computing**
  - Software Environments for Grid Computing (Bilateral Cooperation UNL-U.Cardiff), 2003-...
    - [PSEs and Grid: Patterns & Workflows]
  - Curriculum Development for High-Performance and Grid Computing (EU AsiaLink), 2005-2008
    - with Delft Un. Tech., Tsinghua Un., and GUCAS (China)
    - [Grid Computing] [Education & Research: Joint Curriculum & PhD Thesis co-advising]
Multidisciplinary Projects

- **Ongoing Collaboration Projects at FCT/UNL**
  - Simulation of pollutants in shallow waters, with Environmental Sciences [Cluster & Grid Computing]
  - Tomography, with Material Sciences [Cluster & Grid Computing]
  - Dispersion of contaminants in sub-soil, with Geological Sciences [Cluster & Grid Computing]

- **Joint multidisciplinary Projects with other Research Centers of FCT/UNL**

  4 Post-doctorate positions at CITI (Portuguese FCT Program): [2008-2012]
  - **2**: BioInfo Project: *Concurrency and Parallelism in Bioinformatics*, with PLM CITI Stream [Open]
  - **1**: BioInfo Project: *Grid Computing for Bioinformatics* [Open]
  - **1**: GeoInfo Project: *HPC and Grid Computing*, namely with Geoscience Center of FCT/UNL [Closed]

- **International Colaboratory for Emerging Technologies CoLab Un. Texas at Austin – Portugal**

  Protocol funded by Portuguese Ministry of Science & Technology

  **Joint Research and Advanced Education Programs in Advanced Computing**
  
  **PhD Program Computer Science and Informatics with focus on High Performance, Grid Computing and Computational Science and Engineering**

  involving 8 Portuguese universities and two research associated laboratories
Education
ACM Innovation & Technology
in Computer Science Education
Pattern Operators for Grid Environments

M. Cecília Gomes

Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

November 2007

Supervisor: José C. Cunha
Goal of this Thesis

- To contribute to application development in Problem Solving Environments (PSEs) in the Grid context:
  - through the reuse and flexible composition of typical “idioms”;
  - by providing some level of application control (each “idiom” is directly manipulable, including at execution time).
Motivation

- Grids are complex environments
  - heterogeneous; dynamic

- Build applications (e.g. scientific) that take advantage of the disparate collection of resources and services provided by Grid infrastructures.

- Typical “idioms” are frequently recurrent in Grid applications:
  - an “idiom” captures common knowledge and experience and describes how a similar set of experiments are to be set-up and managed
  - it is advantageous to identify those “idioms” and reuse them
An approach for application development based on:

- Design Patterns as first-class entities
- Operators for pattern manipulation
- Separation of concerns: structural and behavioural patterns and operators
- A methodology supporting pattern-based application development, reconfiguration and execution control (either through a GUI or scripts)
Design Patterns as First-class Entities

- Design Patterns
  - capture common usage “idioms”
  - provide a mechanism to capture common ways to use software within a given application domain

- Patterns as first-class entities during all phases of an application life-cycle:
  - development units
  - execution control units
  - reconfiguration units

- Structural and Behavioural patterns for flexible composition and manipulation
Structural Patterns

- Encode component connectivity. Examples:
  - **Common topologies:**
    - Star
    - Ring
    - Pipeline
  - **Design patterns:**
    - Proxy
    - Facade
    - Adapter
Selected Structural Pattern Templates

- **Pipeline S-PT**
- **Ring S-PT**
- **Star S-PT**

- **Proxy S-PT**

- **Real Subject**
  - **Proxy**
  - **Proxy**

- **Facade S-PT**
  - **Facade**

- **Adapter S-PT**
  - **Adapter**
  - **Adaptee**
Behavioural Patterns

- Capture temporal or (data/control) flow dependencies between components.
- Examples:
  - Master/Slave
  - Producer/Consumer
  - Peer-to-peer
  - Itinerary (mobile agents)
  - Remote execution
  - Client/Server
  - Parameter-Sweep
  - Service Design Patterns for Grids
Pattern Operators

- Invariant on the semantics of the manipulated patterns.
- Perform uniformly at all stages of the application development time.
- Several operator categories provide application manipulation in different dimensions (concerning structure/behaviour)
- Amenable to automation through scripts
Categories of Pattern Operators

- Structuring and Grouping Operators
  - establish/modify the connectivity between components in patterns (e.g. increasing/decreasing the number of elements)
  - build hierarchical patterns

- Inquiry Operators
  - verify structural/behavioural properties (e.g. if a pattern is hierarchical)
  - support pattern comparison on consistency/compatibility (e.g. before replacing a pattern with another)

- Ownership Operators
  - enable/delegate access rights to a pattern (e.g. to define which users may manipulate a pattern)

- Global Coordination Operators
  - establish behavioural dependencies between elements within a pattern (e.g. apply/replace a behavioural pattern)

- Execution Operators
  - provide execution control (e.g. suspend/resume execution)
Examples of Structural Operators

- Increase/Decrease
- Extend/Reduce
- Embed/Extract
- Replicate
- Replace
- Group/Ungroup
Structuring Operators

Pattern

Result Pattern

$\text{Increase}(2, \text{pipelinePT})$

$\text{Extend}(\text{facadePT})$

$\text{facade1}$

$\text{facade2}$
Structuring Operator Sequence

1: Increase(2,proxyPT)

2: Extend(proxyPT)
Hierarchical Patterns Result from Grouping Operators

\[ \text{Embed}( \text{proxyPT, pipelinePT, } \text{“cph3”} ) \]
Execution Operators

- Start/Terminate
- Stop/Resume
- Repeat/TerminateRepeat
- Limit/UndoLimit
- Restart/TerminateRestart
- Log/TerminateLog
- ResumeLog
- SeqLog/TerminateSeqLog
- Steer/SteerComponent
Pattern-based Methodology for Application Development

Structural Operators
- Refine / inquire / compose

Structural Pattern Templates

Behavioural Pattern Templates

Components / Tools / Services

Template Configuration

Behavioural Operators
- Ownership operations

Running Application

Application Configuration

Behavioural Operators
- Control execution / change dependencies

produced
transformation
application
direct
manipulation

Parallel and Distributed Processing
PSE Example

Ring Structural Pattern

Problem Solver

Output data

Monitoring Service

Selected data

Steering Interface

Input data

Database System

Pipeline Structural Pattern

Streaming BP

Streaming BP
Parallel and Distributed Processing

**PSE Example**

- **Star Structural Pattern**
  - Database (master)
  - Database (slave)
  - Database (slave)

- **Adapter Structural Pattern**
  - Adapter
  - Adaptee (legacy code)

- **Master/Slave**
  - User (Proxy)
  - User (Proxy)

- **Client/Server**

- **Steering Interface**

- **Problem Solver**
  - Database System

- **Output data**
  - Selected data
  - Input data

- **Real Subject** (Steering Interface)
  - User
  - Adapter
  - Adaptee (legacy code)
Start(PSE) Limit(time,PSE) Final Configuration

PSE
{RingSP + StreamingBP}

DataStoring
{PipelineSP + StreamingBP}

MonitoringSv
{AdapterSP + ServiceAdapterBP}

SteeringInt
{ProxyPT + Client/ServerBP}

DatabaseSystem
{StarSP + Master/SlaveBP}
Reconfiguration Capabilities of the Approach

- **Pattern-based reconfiguration**
  - independent modification of structure and behaviour
  - first-class manipulation allows reconfiguration automation (pre-defined new dependencies)

- **Reconfiguration at development time**

- **Run-time reconfiguration**
  - without suspending execution
  - with total/partial (pattern-based) execution suspension (based on the Stop/Resume operators)
Reconfiguration: Replacing the Behavioural Pattern of “SteeringInt”
Reconfiguration: Creating a New User of the Steering Interface

PSE
{RingSP + StreamingBP}

Increase( 1, PSE.SteeringInt )
Instantiate(PSE.SteeringInt, "proxy3", "user3")

Problem Solver
Input data
Output data
Database System
Output data
source destination

DataStoring
{PipelineSP + StreamingBP}

proxy
Monitoring Statistics service
subject

MonitStatSys
source destination

Steering Interface
producer
consumer
proxy1 (user1)
proxy2 (user2)
proxy3 (user3)

SteeringInt
{ProxyPT + Producer/ConsumerBP}
The Triana Project (www.trianacode.org) (Extension with Patterns/Operators)

Layer 1

Composition Environment
(Problem Solving Environment / Portal Interface)

Triana

Components
(Units)

Patterns

Operator Library

Distribution Interface

GAP/GAT
(Triana/GridLab)

DRMAA

CoG Core / OGSA

Web Services

Peer-2-Peer
(P2PS/JXTA)

Local Resource Managers
Prototype in Triana
Script for a Galaxy Formation Example

1: Initialize
2: Create Pipeline ImgProjection
3: RunStructuralScript ImgProjection
4: Decrease 1
5: Instantiate DummyUnit /home/mcg/working/toolboxes/Demos/GalaxySim/DataFrameReader.xml
6: SetParameter DataFrameReader fileName /home/mcg/working/triana/old_out.drt
7: Instantiate DummyUnit1 /home/mcg/working/toolboxes/Demos/GalaxySim/ViewPointProjection.xml
8: EndStructuralScript
9: Embed ImgProjection DummyUnit
10: Create Pipeline ImgProcessing
11: RunStructuralScript ImgProcessing
12: Decrease 1
13: Instantiate DummyUnit /home/mcg/working/toolboxes/ImageProc/Processing/Effects/EnhContrast.xml
14: Instantiate DummyUnit1 /home/mcg/working/toolboxes/ImageProc/Output/ImageView.xml
15: Activate EnhContrast
16: EndStructuralScript
17: Embed ImgProcessing DummyUnit1
18: Activate ImgProcessing
19: Create Pipeline ImgAnalysis
20: RunStructuralScript ImgAnalysis
21: Decrease 1
22: Instantiate DummyUnit /home/mcg/working/toolboxes/ImageProc/Processing/Detection/CountBlobs.xml
23: Instantiate DummyUnit1 /home/mcg/working/toolboxes/Common/Const/ConstView.xml
24: Activate CountBlobs
25: EndStructuralScript
26: Embed ImgAnalysis DummyUnit2
27: Activate ImgAnalysis
28: TriggerUnit ImgProcessing
29: TriggerUnit ImgAnalysis
30: Start
Galaxy Formation Example
Open Issues / Future Work

- Coordination issues related to Hierarchical Patterns
- Run-time reconfiguration
  - On-demand (by the User)
  - Automatic (triggered by the middleware)
- More extensive evaluation of our approach on real applications (e.g. concerning expressiveness and scalability)
Results

- Cooperation with Omer Rana and the **Triana Group** (Cardiff University) resulting on a (simplified) prototype over Triana

References:

What is our expected role and contribution until August 2008?

What we plan to do:

- Further explore patterns/operators
  - integrate into a really working prototype and relate to workflow tools that can be used for applications
  - explore use in support of dynamic reconfiguration
- Further explore group concepts
- Get comments from CoreGRID members
  - identify possible collaborations
- Effort on education: curriculum and tools