PARTOUT = PARallélisme parTOUT

PARTOUT = parallélisme parallélisme parallélisme ...

ANR - DEFIS - “Domaines Emergents”

- CNAM : Eric Gressier-Soudan, **Jean-Ferdy Susini**
- **INRIA-Sophia** : **Frédéric Boussinot**, Ilaria Castellani, Manuel Serrano
- LRI : **Louis Mandel**, Marc Pouzet
- + Frédéric Dabrowski (PostDoc Landes)

1/1/2009 - 4 years - 400Keuros

Context

- Parallelism everywhere: multicore, multithreading, multiprocessor, distributed programming, Web
- Preemptive parallelism raises deep semantics issues and, as a consequence, programming issues (atomicity concerns, locks, data-races,...)
- Safety/security concerns everywhere (Web !)

Objective

"The main objective of the project is to design tools for safe and efficient parallel programming, adapted to multicore architectures, to multiprocessors, to distributed architectures, and to the Web."

Background

- Synchronous parallelism (reactive programming variant)
  - SugarCubes (Jean-Ferdy)
  - ReactiveML (Louis)
  - FairThreads & FunLoft (Manuel, Frédéric B&D)
- Formal approach
  - Semantics (everybody!)
  - Language-based security (Ilaria)
  - Type systems (Frédéric D, Ilaria, Marc, Louis)
- Language design || Semantics || Efficient implementation
  - Bigloo, HOP (Manuel)
  - LucidSynchrone (Marc)
  - Games (Eric, Jean-Ferdy, Frédéric B)

Theme 1: Efficient Programming

1. What are the good primitives for multicore programming? Comparison with Software Transactions.
2. GC in presence of instants?
3. JIT techniques in presence of signals?

Theme 2: Distributed Programming

1. How to let distributed synchronous activities interact? Application to the Web (HOP) and to networked multi-players games.
2. Synchronisation of distributed synchronous activities? Synchronised schedulers (FunLoft)? Multi-clock model (LucidSynchrone, SugarCubes)?

Theme 3: Safe Programming

- How to preserve atomicity (i.e. absence of data-races) in a multicore framework? (FunLoft)
- How to preserve reactivity? How to insure the absence of memory leaks? (F. Dabrowski)
- How to preserve (multi-level) security (non-interference) in presence of parallelism, distribution, migration? (I. Castellani)

Theme 4: Dynamic Aspects

- How to introduce dynamic aspects (such as scripts) while preserving safety? (SugarCubes, ReactiveML)

Tasks

- T1 Language Design
  - T1.1 New programming primitives in several directions: distribution, (limited) resource control, safe scripting, migration, dynamic linking
  - T1.2 Information flow security: confidentiality and integrity of sensitive data. Language-based security approach; type (and effect) systems.
- T2 Implementations
  - T2.1 FunLoft: ReactiveGC, distribution, FunLoft → SugarCubes
  - T2.2 ReactiveML: new implementation; multicore, distribution (JoCaml); static analyses in presence of higher-order functions

  - T2.3 SchemeBigloo: extension of the present threading system (several schedulers; unlinked threads)
  - T2.4 SugarCubes: GC with instants; Reactive Virtual Machine; mapping on multi(core/processor) machines; Domain Specific Language on top of Java→Full language

- T3 Applications
  - T3.1 Distributed, synchronised, HOP servers: how to let them communicate and synchronise?
  - T3.2 Networked games on game consoles (and mobile telephones)

- T4 Dissemination (all software under Gnu GPL license)

Relation to Other Work

- Synchronous languages (Esterel, Lustre, ...)
- Preemptive threads + locks (Posix, Java)
- Code parallelisation (Intel's TBB, IBM's X10)
- Software transactions (Haskell, Abadi's AME)
- Safe distribution (Acute, JoCaml)
- Higher-order parallelism (ULM)
- Information flow security (Myers' JIF)
- Dynamic aspects (Scheme, ML)

Lacks?

- Message passing (CML, Erlang)
- Bulk Synchronous Programming

Contacts

- frederic.boussinot@sophia.inria.fr
- ilaria.castellani@sophia.inria.fr
- frederic.dabrowski@irisa.fr
- eric.gressier_soudan@cnam.fr
- louis.mandel@lri.fr
- marc.pouzet@lri.fr
- manuel.serrano@sophia.inria.fr
- jean-ferdinand.susini@cnam.fr

http://partout.gforge.inria.fr