MUMPS ADT and tests automation

Maurice Bremond, INRIA - Rhône Alpes

Toulouse, April 15th, 2010
Outline

INRIA/SED and ADT
  INRIA/SED
  ADT and software engineering tools
  MUMPS ADT

MUMPS tests automation
  PIPOL: a platform dedicated to software porting and testing
  PIPOL: how it works
  remote testing automation on PIPOL
what is SED ? ("Service d’Expérimentation et Développement")

- SED : engineering support for research development and experiments
- SED INRIA Rhône-Alpes : 12 engineers specialized in embedded system, robotics, virtual reality, graphics, network, grid computing, scientific computing

a SED team is present in each INRIA research center

- Bordeaux, Lille, Nancy, Rennes, Rhône-Alpes, Rocquencourt, Saclay, Sophia Antipolis and a national board (D2T)
- some software engineering tools and a link with research teams : ADT
ADT and software engineering tools

ADT : “Action de Développement Technologique”

- time-limited support for technological development granted to research projects

some software engineering tools

- gforge server: cvs, svn, git + project management facilities
- pipol cluster provides root access to various systems
- cdash server collects software tests results
Goal

Ensure the durability and the evolutivity of the MUMPS software.

tasks

- task 1: validation and experimentation tools
- task 2: software engineering
- **task 3: automated tests**
- task 4: performance analysis
- task 5: comparison with other direct solvers
- task 6: documentation
- background tasks
PIPOL ("Plateforme INRIA de POrtage Logiciels"), main features

- administrator ("root") access granted on a temporary machine
- access to real hardware (<10 minutes) or virtual system (<4 minutes)
- processors intel 32, 64, IA64, Intel mac
- about 50 systems: Linux, BSD, Solaris, Macintosh, Windows

Usage

- interactive: [http://pipol.inria.fr](http://pipol.inria.fr)
- automated software testing: “nightly builds”
- continuous integration ... not yet
PIPOL : how it works

On demand computing

- A frontal server holds systems images and access services.
- Limited resources are hardwares or virtual machines slots and system licences.
- Reservations are managed by oar, a batch scheduler.
- Systems installations are done with a self-made tool.

Systems are kept up-to-date!

Systems are automatically deployed, updated, checked and saved every week.
remote tests automation on PIPOL

A pipol command to automate the whole chain

- **pipol-sub** 02:00 snow-leopard /bin/echo Hello World
- this command can be downloaded and executed from anywhere where **ssh** is available

Some options to pipol-sub

- send (rsync) some system configurations (–rc-dir=...)
- send (rsync) sources directory (–export-dir=...)
- use an already deployed system (–reconnect)
MUMPS tests automation (ADT task 3)

for MUMPS developpers mumps-sub hide pipol-sub :

- **mumps-sub** provides uniform access to **pipol** and other servers
- **mumps-sub** may be executed interactively or on a regular basis (i.e. with **cron**)

so system and software configurations are parameters of the test command

main parameters may be:

- for **pipol**, system kind : windows, mac os x, linux, ...
- compiler : Intel fortran, gfortran, ...
- **MUMPS** sequential or parallel
- ordering tool
- ...

Maurice Bremond, Toulouse, April 15th, 2010
MUMPS tests automation (ADT task 3)

what remains to be done

- submit tests outside pipol
- some other configurations on pipol:
  - windows without cygwin?
  - pararellel configurations
- nightly builds
- a results database?
- use test_driver
MUMPS Action of Technological Development

Guillaume Joslin, INRIA-LIP, ENS Lyon

Toulouse, April 15th, 2010
Context

• Three-year project.
• Funded by INRIA.

Objectives:
Ensure the durability and the evolutivity of the MUMPS software.

Means:
• A part-time senior engineer, Maurice Brémond.
• A junior engineer, Guillaume Joslin.
Task 1: Validation and experimentation tools

- Developing a new advanced test driver.

- Rewriting the non-regression testing system.

- Improving the functionalities for adding new test cases.
Task 2: Software engineering

- Cleaning the existing code (compliance with coding rules, removing unused variables, warnings...).

- Managing internal errors.

- Developing tools to improve the portability of the code.

- Improving the code coverage (in relation with task 1).
Task 3: Automated tests (platforms and software)

Presented by Maurice Brémond
Task 4: Performance analysis

- Developing a performance analysis system (in relation with task 1).

- Linking this system with the GRID-TLSE project.

- Documenting the system.
Task 5: Comparison with others direct solvers

• Defining a test environment.

• Producing and analysing the results.

• Automating the test procedure.
Task 6: Documentation

- Modernizing the users’ guide.

- Distribution of the documentation according to the user.
• Participating in user support.

• Validating new functionalities.
Thank you for your attention.

Any questions?