Fujitsu Programming Environment

EWOMP 03

Aachen, Germany

24 September 2003

Compilers

- Fujitsu Solaris Fortran 95, C99, and C++
- Fujitsu Linux Fortran 95, C99, and C++
- Full OpenMP 2.0 implementation, including nested parallelism
- Documentation on: http://www.ftcf.net

SPEC OMPM

FUJITSU

- Held the world record from June – August 2003
- On 128-CPU PRIMEPOWER system, with 1.3 GHz SPARC64 V CPUs.
- Using the Fujitsu Solaris compilers (Parallelnavi 2.2)

Compiler switches

- **Fortran switches:**
  -KOMP –Kfast -Kprefetch
  -Kthreadprivate –Kspinwait –X9
- **C & C++ switches:**
  -KOMP –Kprefetch –Kfast

Workbench

FUJITSU

- **Workbench is part of Fujitsu's TAO-W, which combines local and remote computing resources into one virtual resource centre.**
- **This product greatly benefits any organisation with local and remote computing resources and/or a need to provide a single, easy to use interface to computing resources.**

TAO-W Components

Workbench

- Fully Integrated Development Environment
- Ditributed workspa management and r facilities
- Multiple remote ex machines
- Team work basic s

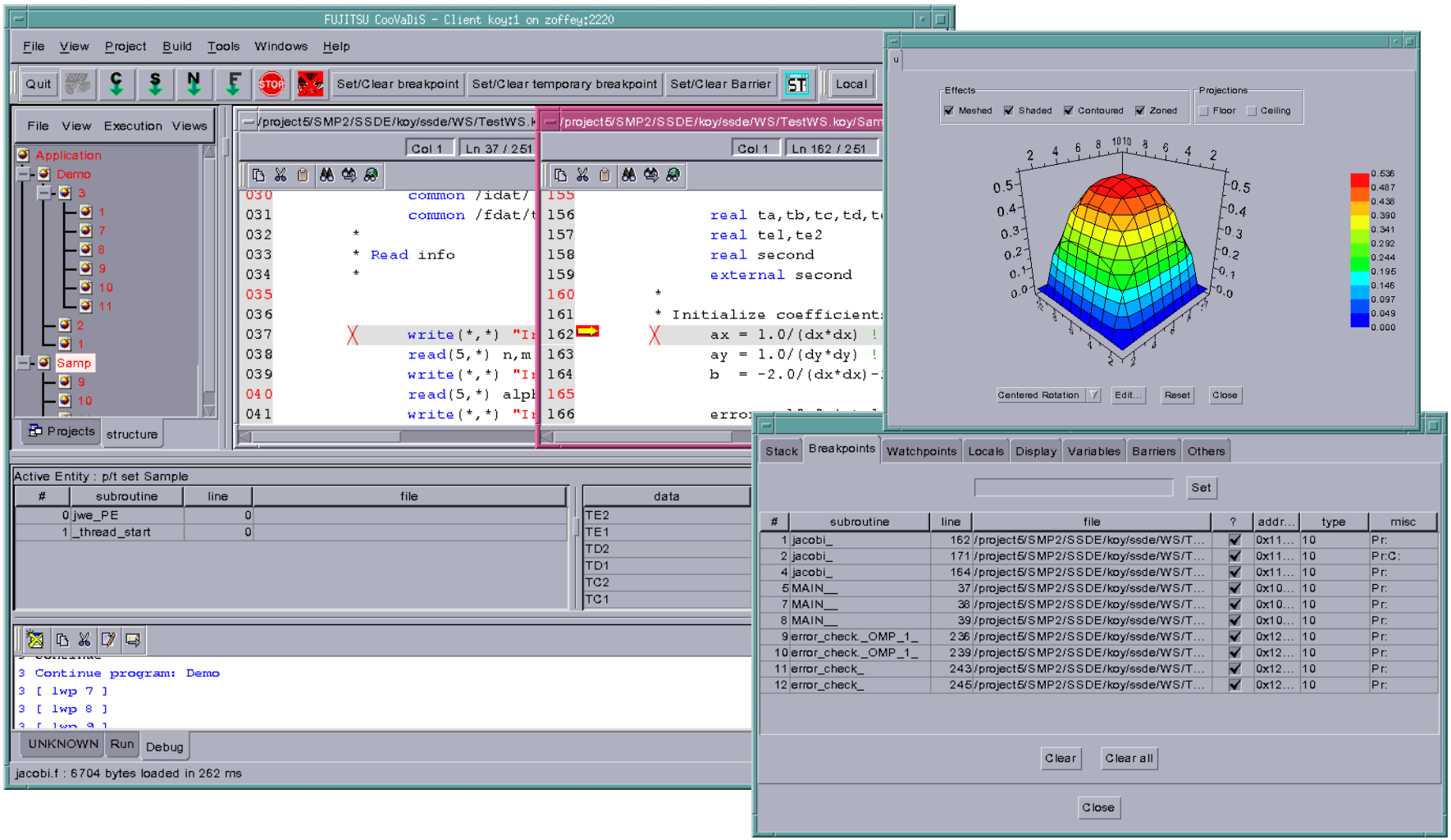
Workshop

- Workflow graphic editor with intuitive GUI
- Heterogeneous netwo access and meta-comp management
- Real time monitoring progress
- Integration of « applet the workflow itself

Workspace

- Integration of Java & XML
- Full control from « data » to « document » process
- Central repository used as shared documentation database
- Database unified access
- Support for XML/3D graphic extension and VR/XML

Workbench Example



Workbench Characteristics (1)

FUJITSU

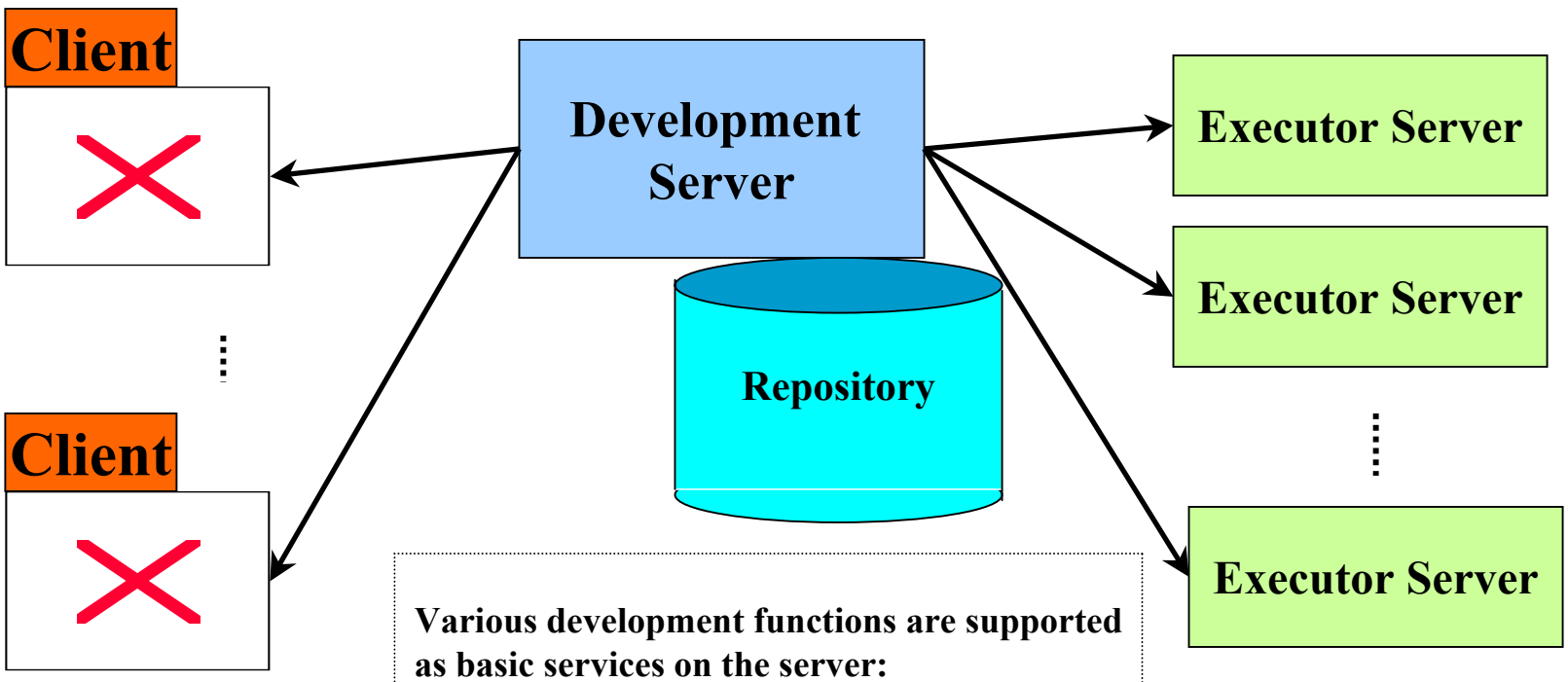
- **Parallel Programming (OpenMP/MPI/XPFortran)**
  - **Support of Mixed Thread-Parallel and Process Parallel Models**
  - **Support of Parallel Programming**
    - ✓ **Template Programming with the editor.Syntax Recognition)**
    - ✓ **Static Analyser for Parallel Programming (Fortran)**
  - **Collection of various tuning information**
  - **Debugging using 3D data visualisation**

Characteristics (2)

FUJITSU

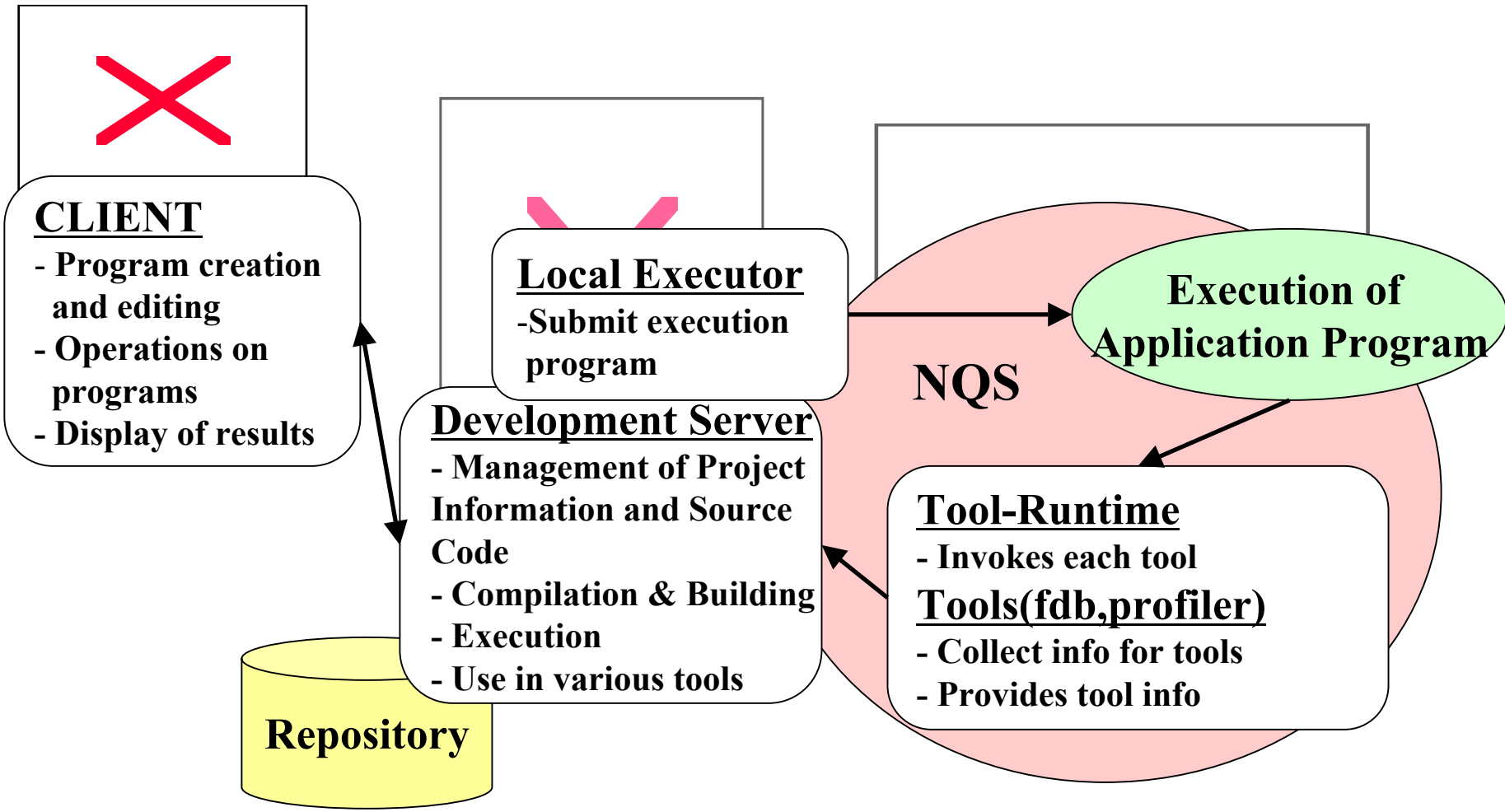
- **Support of network based collaborative development environment**
  - **Project management information and source code are managed and shared on the server repository. This makes collaborative development possible.**
  - **3-tier architecture with client, executor server, and development server enables flexible deployment and brings load balancing.**
  - **Transparent network**
    Working from his client GUI, the user doesn't need to know the complex environment.

Model – Team Usage

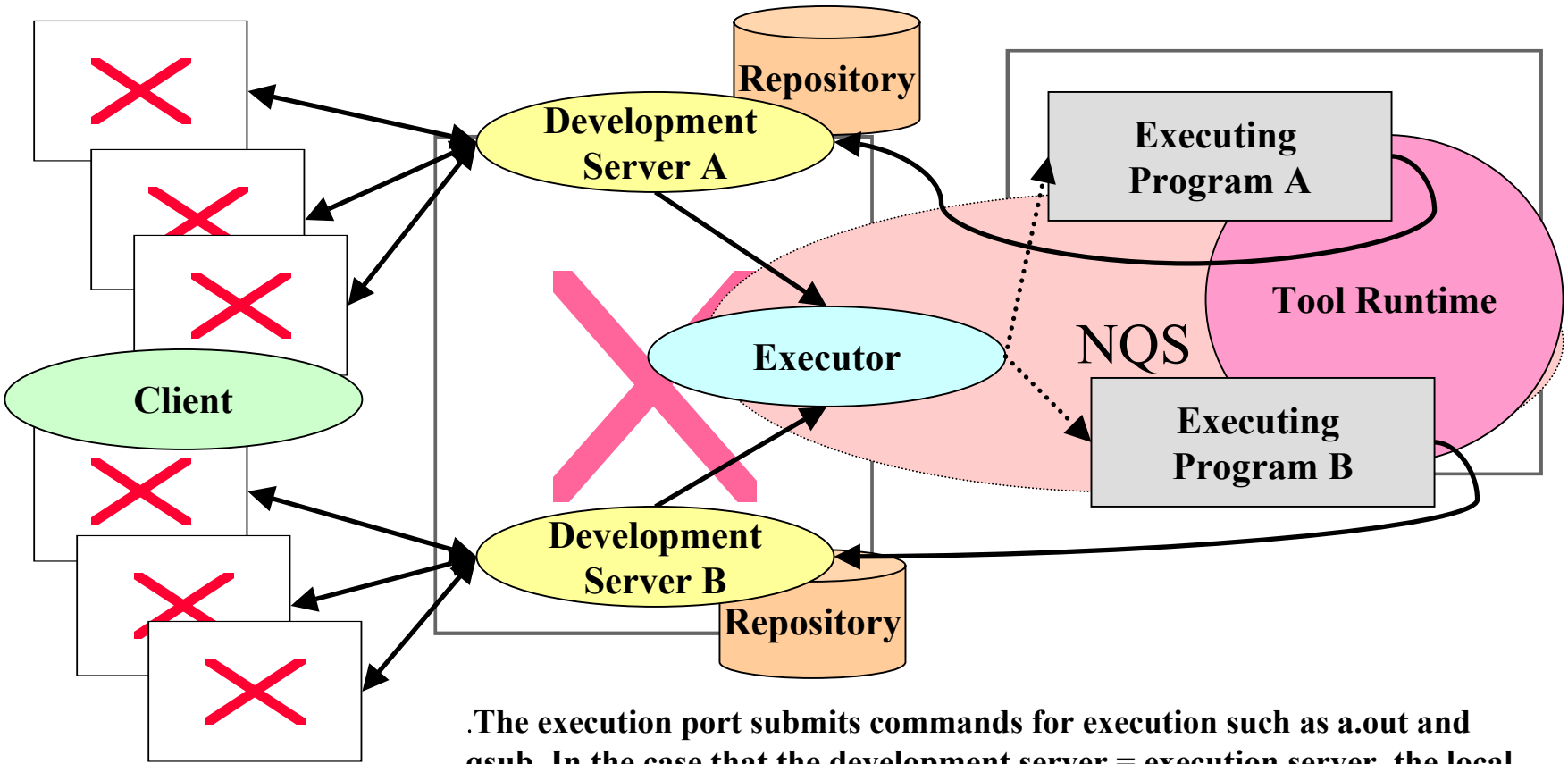


Various development functions are supported as basic services on the server:
.Compilation Service
.Building Service
.Execution Service
.Debugging Service
.Performance Analysis Service
….

Team Development Structure



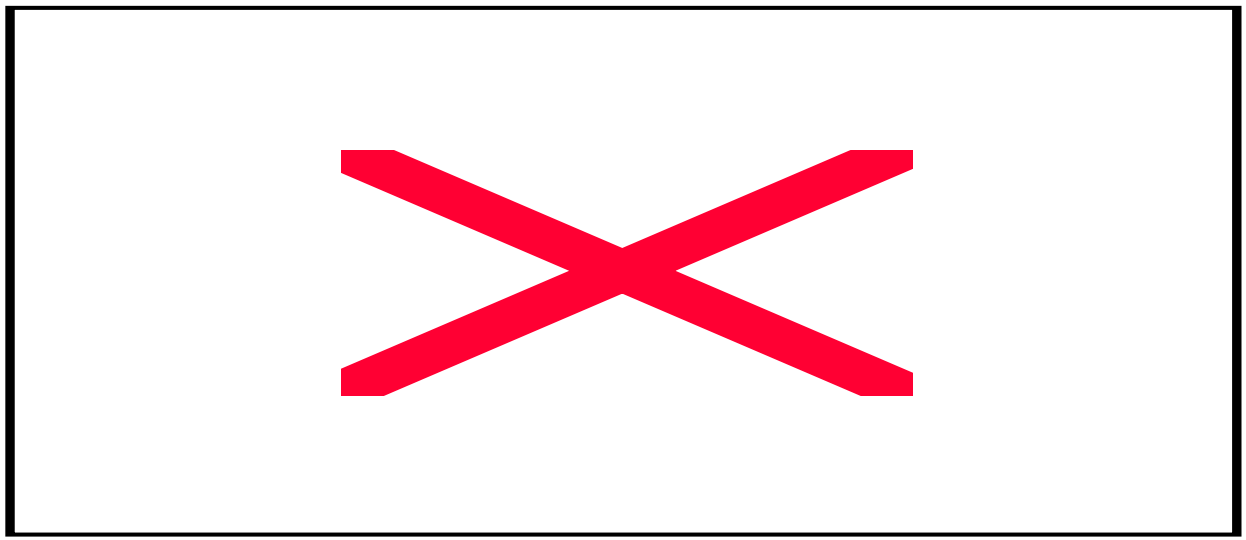
Team Development Model



.The execution port submits commands for execution such as a.out and qsub. In the case that the development server = execution server, the local executor of each developer is used.

.Running a client GUI on each terminal brings good load balancing

Operation Flow –
1. login



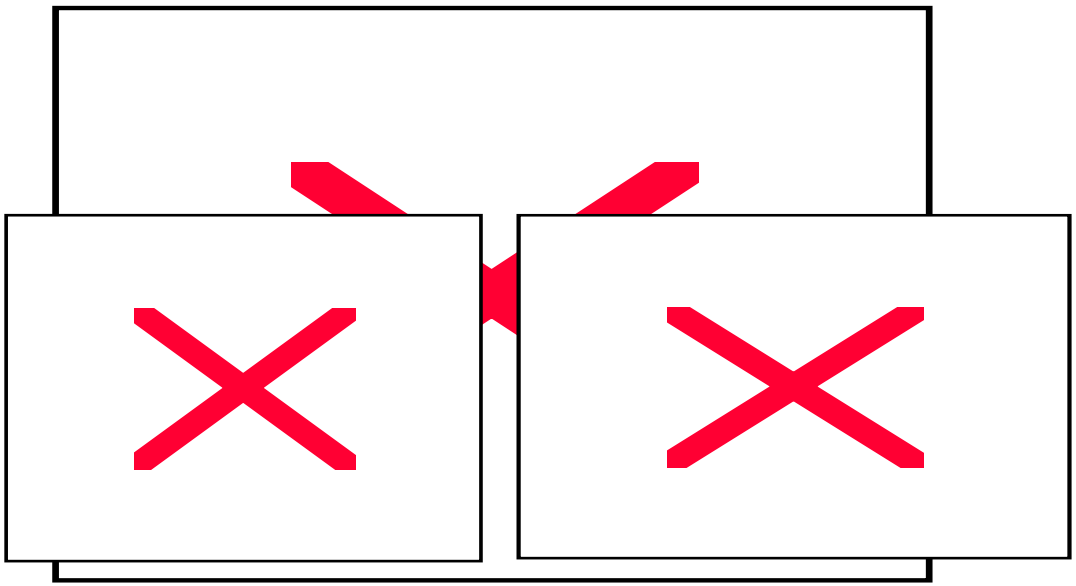
◆Login by invoking the client GUI

➢Before connection, some settings are necessary (setting of server name, port ID, and registration of user account)

➢In the case that more than 1 server is running, the user needs to know which machine to connect to.

Operation Flow –
2. Definition of project information

FUJITSU

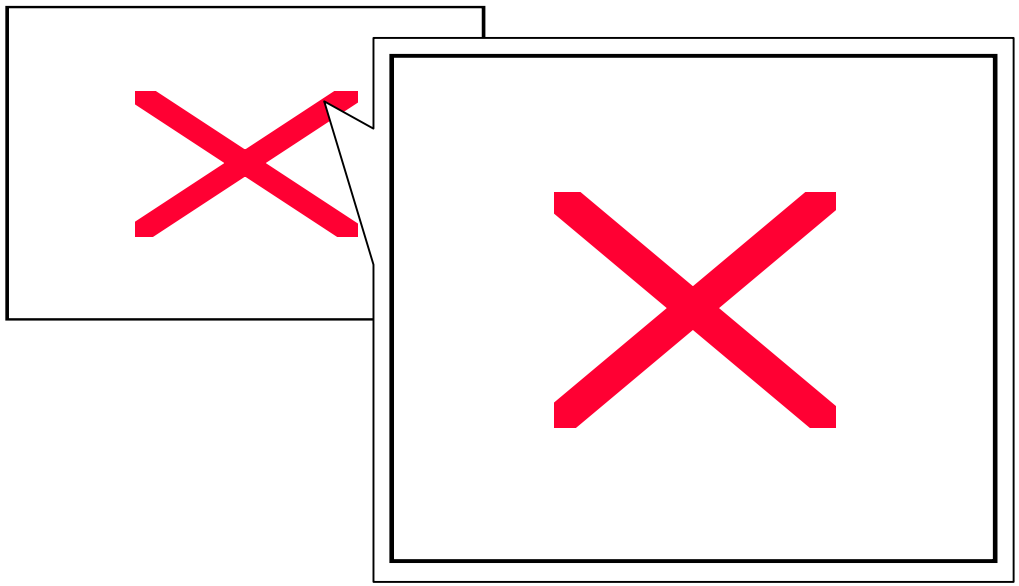


◆Environment settings like definition of project information

- ➢Definition of a workspace for each task unit, and a project for each load module
- ➢Settings for building (compiler, linker, tools, …)
- ➢Add existing source files to the project

Operation Flow –
3. Program Creation and Editing

FUJITSU

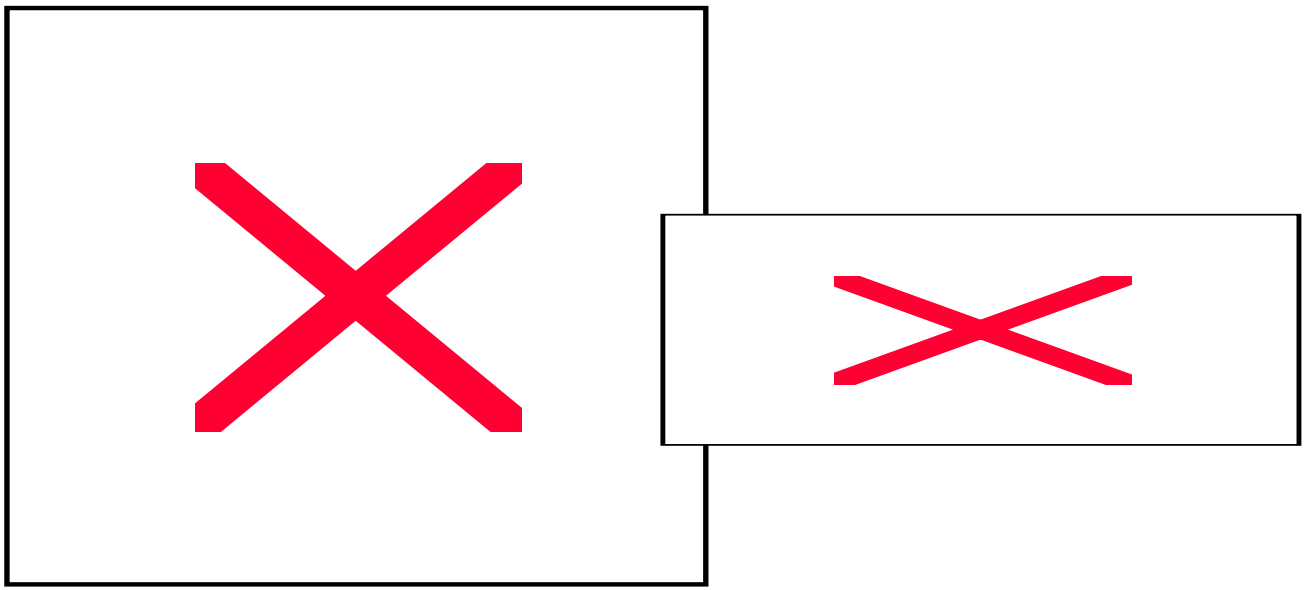


◆Program creation and editing

- ➢Create or copy the source code files on the Repository server
- ➢Attach the created source code files to the suitable project

.parallelising syntax helpers are available.

Operation Flow –
4. Compilation and Building



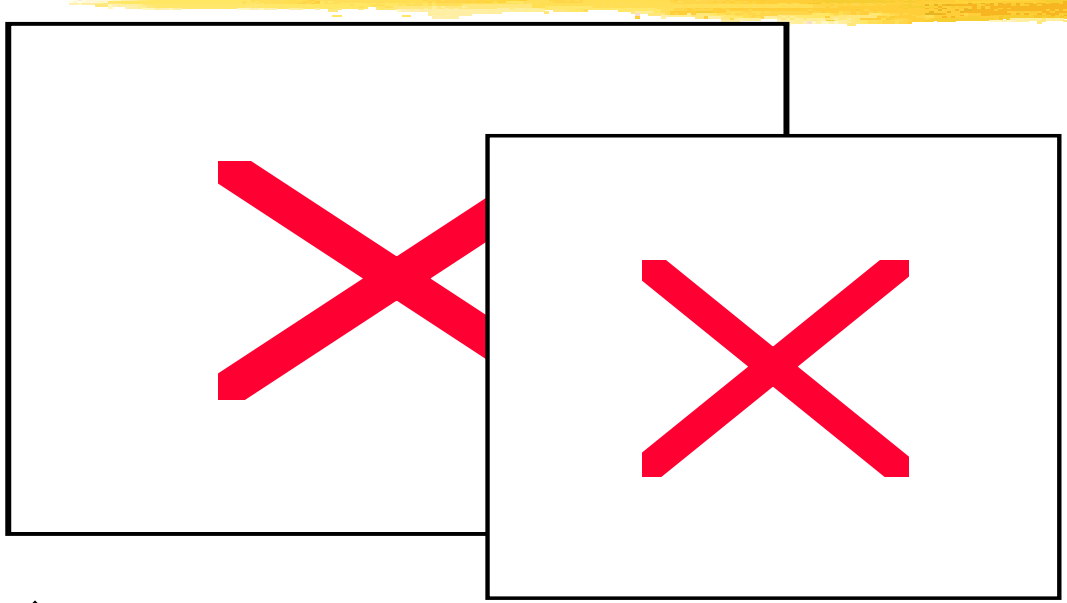
◆Compilation and Building of Application Program

➢Select the desired project, select build on menu, and automatically build the target.

➢In the case of compilation errors, the editor can be invoked at the line of the error.

Operation Flow –
5. Program Execution

FUJITSU



◆Program Execution.Choose desired project, and select “Execution”.

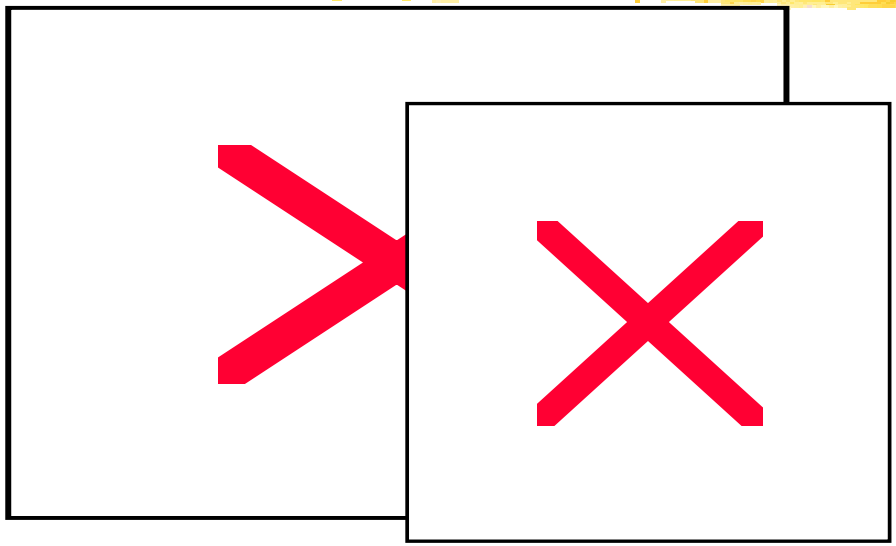
➢Select the server name of the target machine and select the execution type.NQS)

➢In the case of team development mode, user needs to give login info for authentification.

➢Customising of execution shell scripts is also possible.

Operation Flow –
6. Monitoring the Execution

FUJITSU

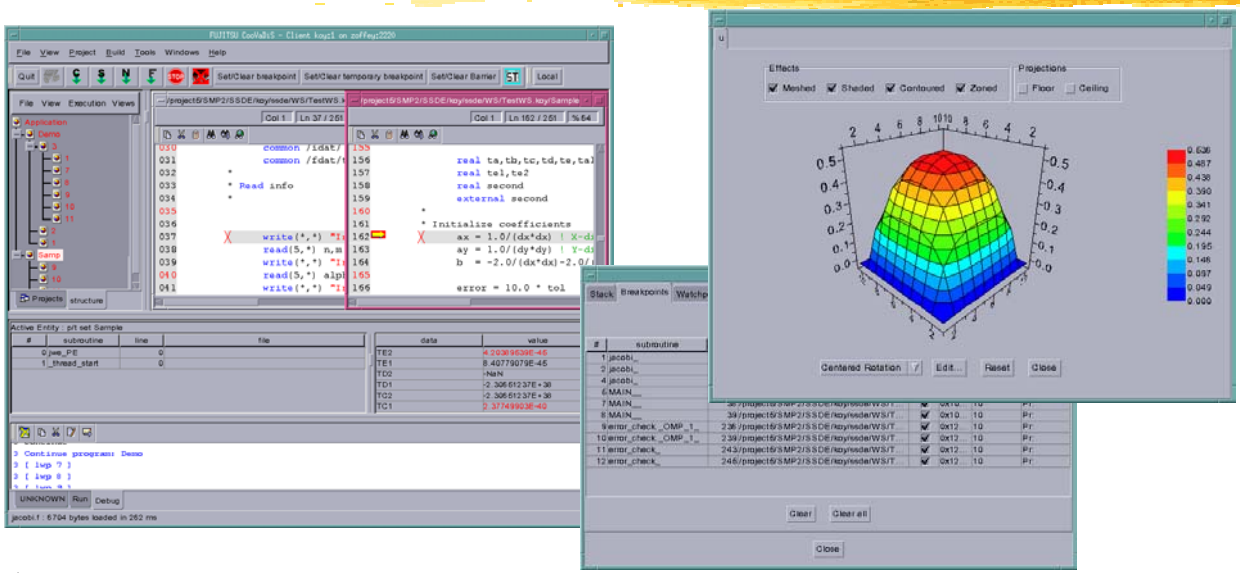


◆Monitoring of Program Execution

➢By selecting the server name of the desired target machine, the monitoring of execution processes is possible.

➢By selecting the desired execution process, the user can invoke debugging of the process (light-weight or full).

Operation Flow –
7. Source Code Debugging



◆Debugging.Choose the desired project and select debug on menu.

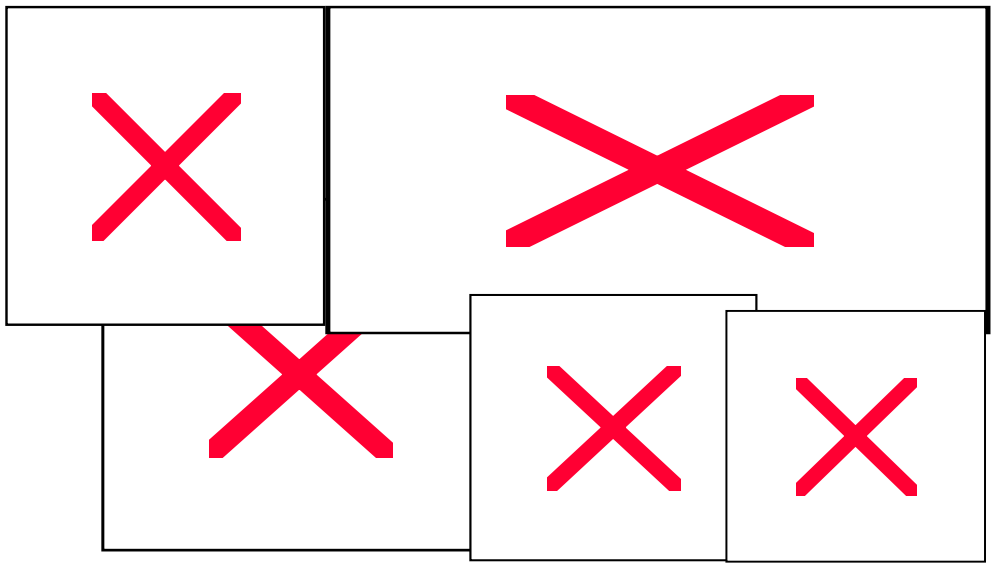
➢Support for OpenMP, MPI, and XPFortran

➢Choose a process or thread from the application structure panel. This shows the dynamic structure of the parallel program. Focus on the desired program to be debugged.

➢3D Visualisation of Array Data

Operation Flow –
8. Profiler

FUJITSU



◆Collection of tuning information.Choose project, and select profile.

- ➢Shows execution time and communication time for each process.
- ➢Finds out bottlenecks
- ➢Analysis of hardware access performance (cache and TLB misses)