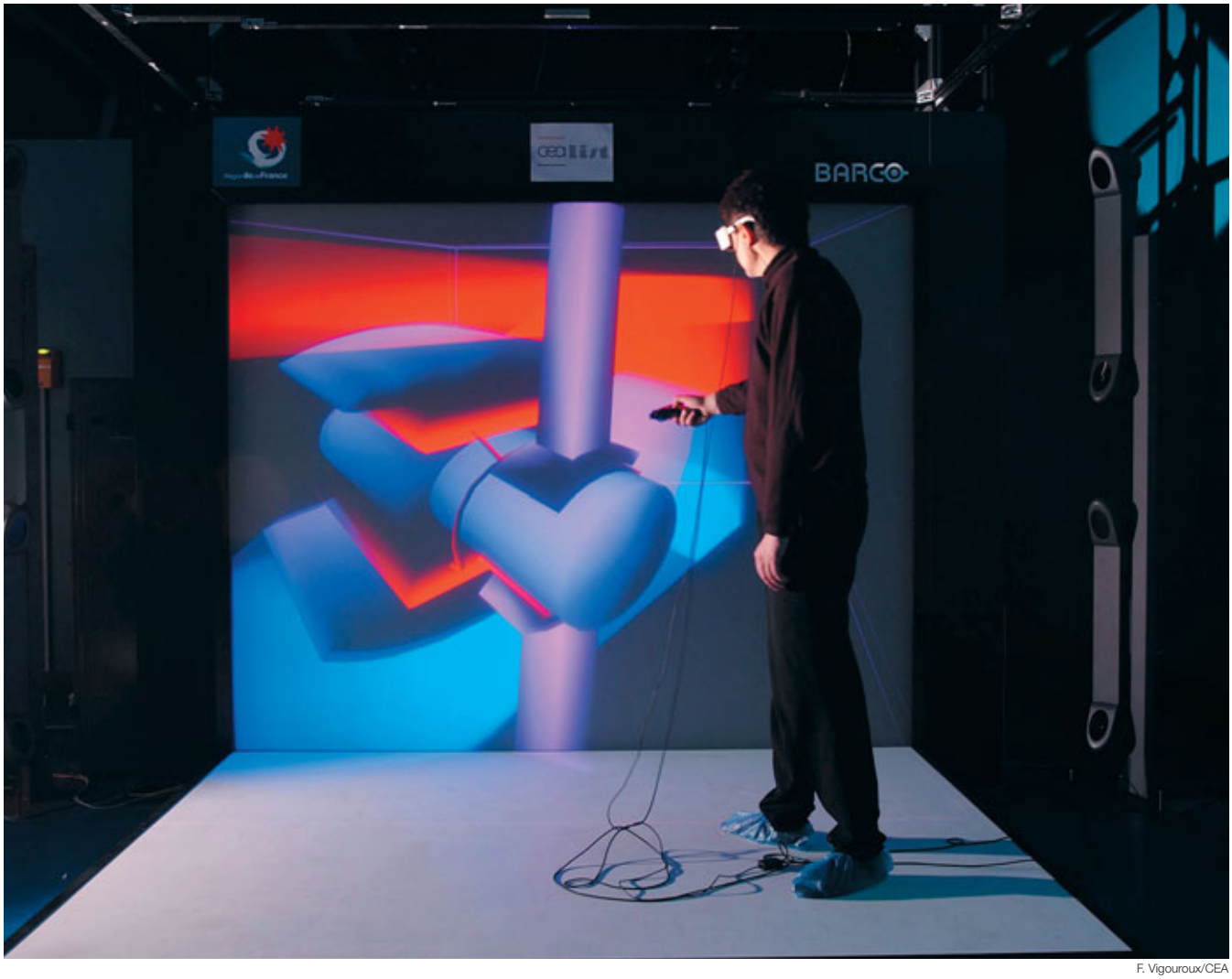
III. SIMULATION FOR ACTION

As shown in the chapter on Simulation for design, understanding and modeling of processes make it possible to predict, within the limits for interpolation of models, the evolution of systems, by means of the modern conceptual and computational tools. These new tools have increased the ability of research workers and engineers to anticipate, and to that extent the ability of users to react when confronted with evolving or even unexpected situations. They are an aid to decision and thus for action.

Thus, the Scar simulator, whose "numerical engine" is the Cathare code, which simulates the thermalhydraulic evolution of a nuclear reactor, can place the operator in a simulated accident situation and thus prepare him to have the right reactions under such circumstances. The intervention of operators in a hostile environment (under radiation, for example) may be prepared in order to reduce exposure. Or the behavior of a container holding nuclear materials may be adjusted to control the consequences of a fire. And virtual prototyping, based on direct human interaction with a digital mockup, enables optimization of complex systems governed by many parameters.

Finally, modeling and simulation of defects in metal parts or other components, starting from non-destructive test findings, allow detection of incipient damage to systems, and make possible an informed choice as to the requisite corrective measures.



F. Vigouroux/CEA

Work simulated in a 3D environment rendered by the Phare virtual-reality platform created by CEA/List at the Fontenay-aux-Roses Center. The image is displayed to the operator by a stereoscopic visualization system comprising two work surfaces (wall and floor).

NUCLEAR REACTORS: FROM SIMULATION TO SIMULATORS

The design, and safety demonstration of nuclear installations, in particular pressurized-water reactors, relies heavily on numerical simulation, itself corroborated by experiments, whether fundamental and analytical or of an overall nature, or integral tests, able in turn to qualify the physical models used as a basis for the simulation. A thermalhydraulics simulation software such as Cathare, developed by CEA and its partners, is also integrated – this being the purpose of the Scar project – into simulators used in particular, to confront plant operators with accident situations, even the most improbable ones

R. de Seynes /P. Berenger/EDF photo library.

The control room of the EDF's Golfech 2 nuclear power plant (Tarn-et-Garonne département). "Full-scope" simulators are authentic replicas of actual control rooms. Inset : control simulator at EDF's Bugey power station at the end of the 1980s.

Why simulate? What is to be simulated?

The safety of nuclear installations, and particularly that of pressurized-water reactors (**PWRs**) imposes consideration, from the design stage, of all types of accident situations, even the most improbable ones. Such a demonstration of safety cannot be based, obviously, on full-scale experiments and requires **numerical simulation tools** (see Box A, ***What is a numerical simulation?***). The most commonly considered accident, for dimensioning safety systems, is that of a sudden failure of a cooling loop pipe in the primary circuit. The ensuing violent depressurization would cause water in the circuit to vaporize and the formation of two-phase steam-water flows, whose characteristics must be predicted accurately, if the installation's behavior is to be determined sufficiently reliably.

What tools for simulation?

Computation of such complex situations requires appropriate softwares, based on physical **models** (systems of equations) yielding the best possible description of the phenomenology of two-phase flows, and validated on the broadest experimental data base possible. This simulation approach was evidenced at CEA's Nuclear Energy Division by the design and operational use of the Cathare software, which has been used since the mid-1980s by the industry and by safety organizations, in France and abroad.

From simulation to simulator… there is but a step – requiring, however, full modeling of the process (including the interface with the operator, and control systems), and computation *in real time* of the most diverse scenarios. To achieve greater realism, simulators rely on the very same software, combining the interactivity of the simulation with a display of flow configurations of training value.

A suitable physical model

The prime requirement is thus the availability of a physical model that will discriminate between the two forms in which cooling-circuit water may appear (water, and steam). The respective role of these two phases, particularly in cooling the core, is indeed anything but similar, since the liquid takes up the energy generated more readily than steam. When both are present in the pipes, of course, they interact with one another, and with metal structures. The physical models must thus take into account a variety of transfers, modeled in **closure laws**, such as mass and energy exchanges at the liquid–vapor interface (boiling, condensation), and exchanges of momentum between liquid and vapor (interface friction) and exchanges of energy between each phase and the walls (convection exchanges, local boiling or condensation, etc.). The relevance of these laws will determine how realistic the software's response will be, when simulating the various phases of an accidental transient.(1)

A number of physical phenomena (as shown in Figure 1) play a particularly important role during a loss-of-coolant accident (LOCA), caused by breaching of the primary circuit.

Implementation in a software program: the Cathare and Scar projects

Once the physical model has been specified, the task to be addressed is to program (to "code") it in a **computation software** program so as to be in a position to validate it, and ultimately use it in application calculations. Thus, the Cathare project was initiated in 1979, on the basis of specifications drawn up by EDF, Framatome and IRSN (Institut de radioprotection et de sûreté nucléaire – French Radiological Protection and Nuclear Safety Institute, known as IPSN at the time) to meet their requirements for demonstrations, and analyses, of the safety of the various accidental transients liable to occur on a PWR.

The outcome of this project was the gradual development of a software program with the ability to describe all the **thermalhydraulic**

(1) A transient is a slow or fast, programmed or accidental evolution of an installation's operational state. In the case of a nuclear reactor, a distinction is made between normal transients during which the values of the physical parameters remain within technical operating specifications, and accidental transient that trigger the action of protection systems and then of safeguard systems.

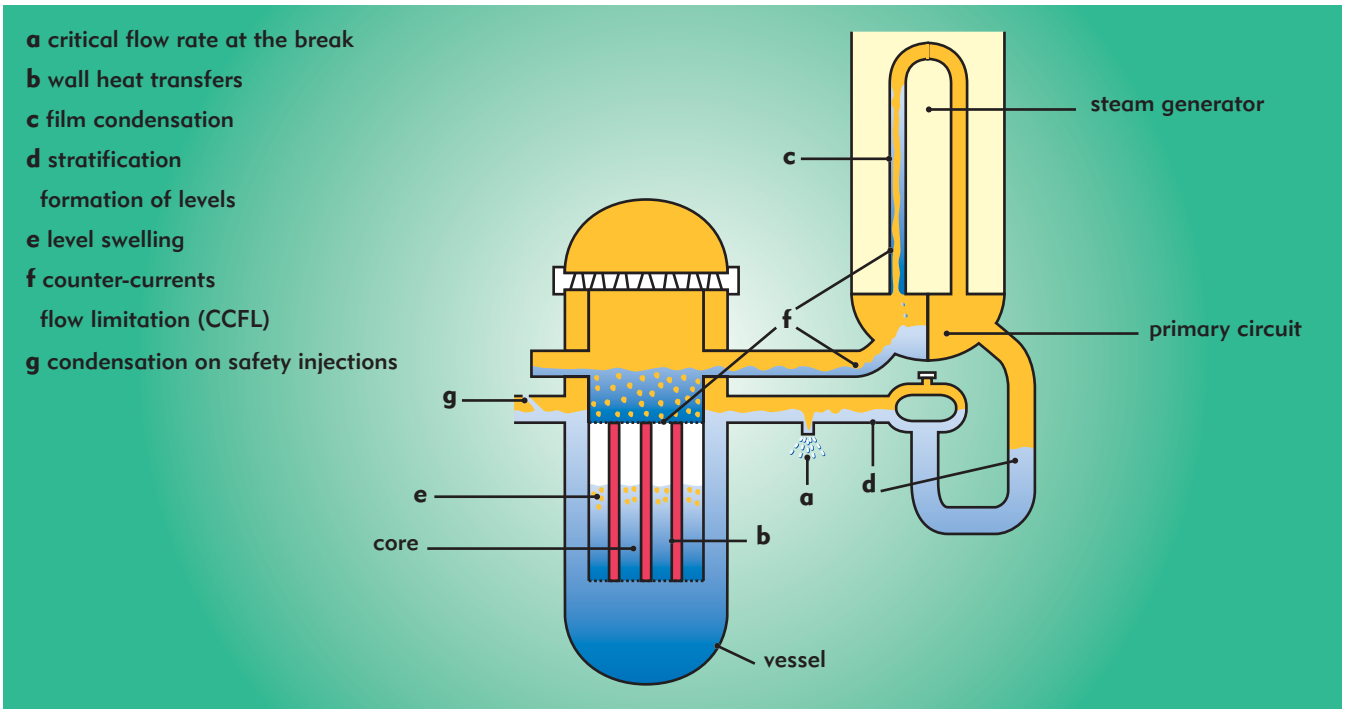


Figure 1. Main physical processes encountered in a PWR during a loss of coolant accident (LOCA).

1

A modular-structure software program revalidated at regular intervals

Cathare's main characteristics are as follows:
• modular IT structure allowing modeling of a simple, analytical-type, or complex "system"-type installation (see Box D, ***Analytical experiments and integral experiments***), or even a complete nuclear reactor;
• various modules, configurable as required: point "0D" (to describe components such as large volumes or pumps), one-dimensional "1D" (for pipes or for a simplified approach to reactor-vessel behavior), three-dimensional "3D" (for a better description of the geometrically complex distribution of flows in the vessel, for example);
• a basic two-fluid, six-equation **model** (one equation for each net balance – mass, momentum, energy – for each phase), with the ability to take into account, themechanical non-equilibrium (differences in velocity between vapor and liquid in co-current or counter-current flow) or thermal non-equilibrium (overheating or undercooling of one phase);
• a consistent and documented set of **closure laws**, subject to a rigorous validation procedure over the entire range of available experiments;
• clear identification of the limits of utilization of the physical laws, and an integrated method for the evaluation of their uncertainties;
• a **numerical** method (see Box A, ***What is a numerical simulation?***), **implicit** in 0D and 1D, **semi-implicit** in 3D, robust and effective, allowing a good tradeoff between precision and computation cost;
• publication of a user's guide integrating operational feedback from the validation phase, to minimize the "user effect."

Since the beginning of the 1980s, out of the fifteen or so versions of the software delivered, three have been subjected to a complete validation program.
In 1987, Cathare 1 V1.3 (revision 4 of the physical laws) was the first version validated for "small break"-LOCAs, and was used as reference for the development of accident procedures and completion of the Sipa simulators.
In 1996, Cathare 2 V1.3L (revision 5) was the first version validated for "large break"- LOCAs, which are those considered for reactor size specification and design.
In 1999, Cathare 2 V1.5 (revision 6) added new possibilities of three-dimensional modeling of the reactor vessel and **parallel** multi-processor computation (see Box B, ***Computational resources for high-performance simulation***). Validation of this version is ongoing into 2003.

CEA

Partial view of the Bethsy installation, operated between 1988 and 1998 at CEA's Grenoble Center, representing on a smaller scale (1/100 in volume, 1/1 in height) all of the systems in a 900 MWe PWR.

phenomena that may arise during an accident, and covering all two-phase flow regimes (bubbly, slug, mist, stratified, annular, etc.), for a wide range of physical parameters (pressures from 1 kPa to 26 MPa, gas temperatures up to 2,300 K, velocities up to the speed of sound) and for a broad variety of geometric configurations (see Box 1).

A two-stage validation

A software program used for demonstrations of safety must be developed under clear quality-assurance rules, and subjected to rigorous validation. The Cathare validation methodology is applied, for a duly referenced version, to a consistent set of closure laws known (as is also each successive modification) as a *revision*, forming an integral part of the version concerned. This process is marked by two main stages: qualification and verification.

The qualification stage

The first stage consists in comparing the results obtained with the software to the measurements effected during **analytical experiments** (see Box D, ***Analytical experiments and integral experiments***). Typically, the qualification matrix for a revision of Cathare's physical laws involves about a thousand tests carried out on 40 different installations, belonging to CEA or other organizations, based in France or abroad.
This amounts to a "stock-taking" and inventory, so to speak, of the quality of the closure laws developed for that revision. This stage also allows a determination, by means of an adjoint sensitivity method (ASM) and a dedicated statistical tool (CIRCE), of the uncertainty for the main parameters of the closure laws. It further allows implementation, if required, of specific improvements to a particular physical model.

The verification stage

The second stage consists in comparing the results obtained with the software to measurements carried out on **integral** experimental installations, i.e. smaller-scale replicas of a nuclear reactor (Box 2) in which accident situations are simulated. The aim here is to validate all of the model's laws on accidental transient scenarios, in the course of which a strong coupling occurs between the various elementary processes.
The Cathare verification dossier includes about thirty "integral" tests carried out on eight "system" test facilities, commissioned since the beginning of the 1980s in France (Bethsy) or in other countries. Here again, each verification calculation may be seen as a "stock-taking" (ou "evaluation"?), of the software's ability to simulate this or that transient. For that reason, any modification of the version of the closure laws undergoing validation must be excluded when carrying out the "basic calculation."
Figure 2 illustrates one such calculation: the evolution over time of one of the parameters (pressure in the primary circuit) is perfectly calculated by Cathare, whereas the mass of

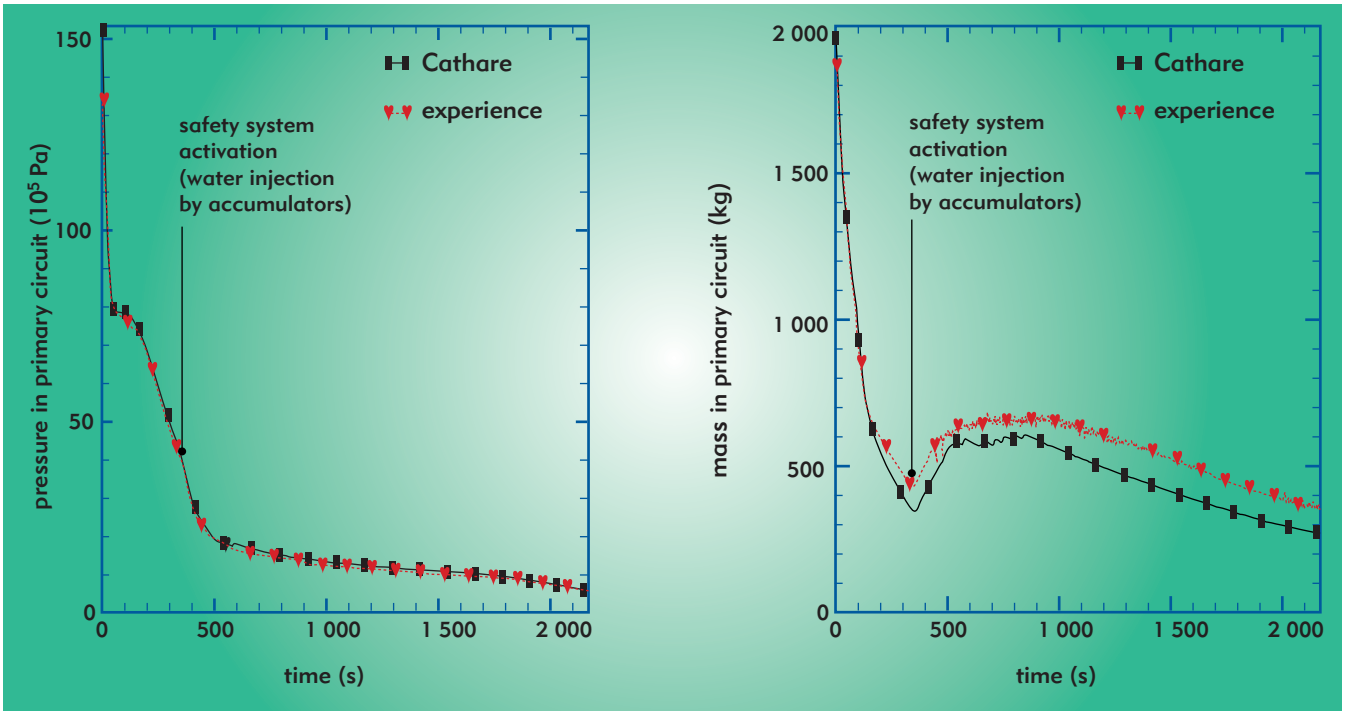


Figure 2. Verification of Cathare computation results for the consequences of a break in the cold leg of a PWR, compared with the findings of a test carried out on the Bethsy test facility.

the fluid present in the circuit is quite clearly underestimated.

It may however be useful, depending on the discrepancies observed, as the case may arise, also to carry out, during this stage, *sensitivity calculations*, in which a limited number of parameters in the physical laws are made to vary, to determine which one appears to be responsible for these discrepancies. This approach allows identification of those models that would require improvement in a subsequent revision, and for new qualifications experiments to be derived from this.

The validation dossier

As the outcome of the validation process, a synoptic document sums up the main conclusions from qualification and verification, specifies the software's utilization limitations, according to the accident sequences considered, and delivers the "quality seal" for the version. One major byproduct of this approach is the writing of a guide, giving recommendations, in the light of the calculation results, as to utilization of the various modeling options.

Experimental thermalhydraulics installations

2

Knowledge of the physics of two-phase flows has involved, for almost forty years, carrying out experiments to support **modeling** work (see Box A, ***What is a numerical simulation?*** and box D, **Analytical experiments and integral experiments**). All contribute to the same goal: improving the validity of the physical models used in the simulation software programs.

Installations of a *fundamental* character, sometimes using simulation fluids (e.g. CFCs) and fitted with very precise instrumentation for local phenomena (behavior of liquid–gas interfaces, for example) have made it possible to lay the groundwork for the current models, and are serving yet, thanks to advances in measurement techniques, to develop those of tomorrow.

Installations of an **analytical** character are designed to represent, sometimes on a smaller scale, the geometry of a component (steam generator) or part of a component (**fuel** assembly) in a reactor. They frequently operate under representative physical conditions (steam-water and possibly high pressure), and allow – through measurements of a more general character (differences in pressure, fluid density and velocity, fluid and structure temperatures, etc.) but in great numbers – information to be gained about a particular process (boiling, condensation, etc.) and correlations to be derived from it, that will be fed into the closure laws for the numerical models.

Installations of a **integral** character, also known as *system loops*, are intended to investigate the interactions, frequently very strong, between the various processes. In order to be representative of a reactor's behavior in an accident situation, they must respect a certain similarity in terms of geometry and scale. Thus Bethsy, operated at CEA's Grenoble Center from 1988 to 1998, is a mockup at full scale (1/1) in height, and 1/100 in volume of a 900?MWe PWR. The core, comprising over 400 electrically heated rods, is cooled, just as the real reactor, by circulation of pressurized water in the three primary loops, each fitted with a pump and steam generator. Some 80 tests covered, in very complete fashion, the various types of accidental transient, and enabled, thanks to highly sophisticated instrumentation (over 1,200 measurement channels), a large database to be set up for the validation of software programs like Cathare.

Software implementation

Cathare is provided not only to partner organizations (EDF, Framatome–ANP, IRSN and CEA), but also to thirty-four organizations abroad (electricity producers, safety institutes, research organizations, universities), in twenty-one countries. Each new version comes with a complete set of documentation and test cases enabling the user to validate installation (Box 1).

Goals assigned to simulators

Initially designed for the training and instruction of plant operating personnel, nuclear power-station simulators were required, first and foremost, to provide a realistic environment (control room) and no less realistic a response to control actions for normal operations, or even in incident conditions. To meet this requirement, relatively simple physical models – restricted as they were to the utilization domain – often proved adequate. This concept was implemented in the early 1990s with the Sipa simulator produced by Thales for EDF and IRSN, together with a simplified version of Cathare that enabled real time with the then-available tools. Sipa simulators have been widely disseminated, since that time, in EDF training centers.

Another use of simulators is for teaching: through visualization of the characteristics of two-phase flows in the various circuits, they allow an understanding to be gained of the phenomenology of simulated accidents.

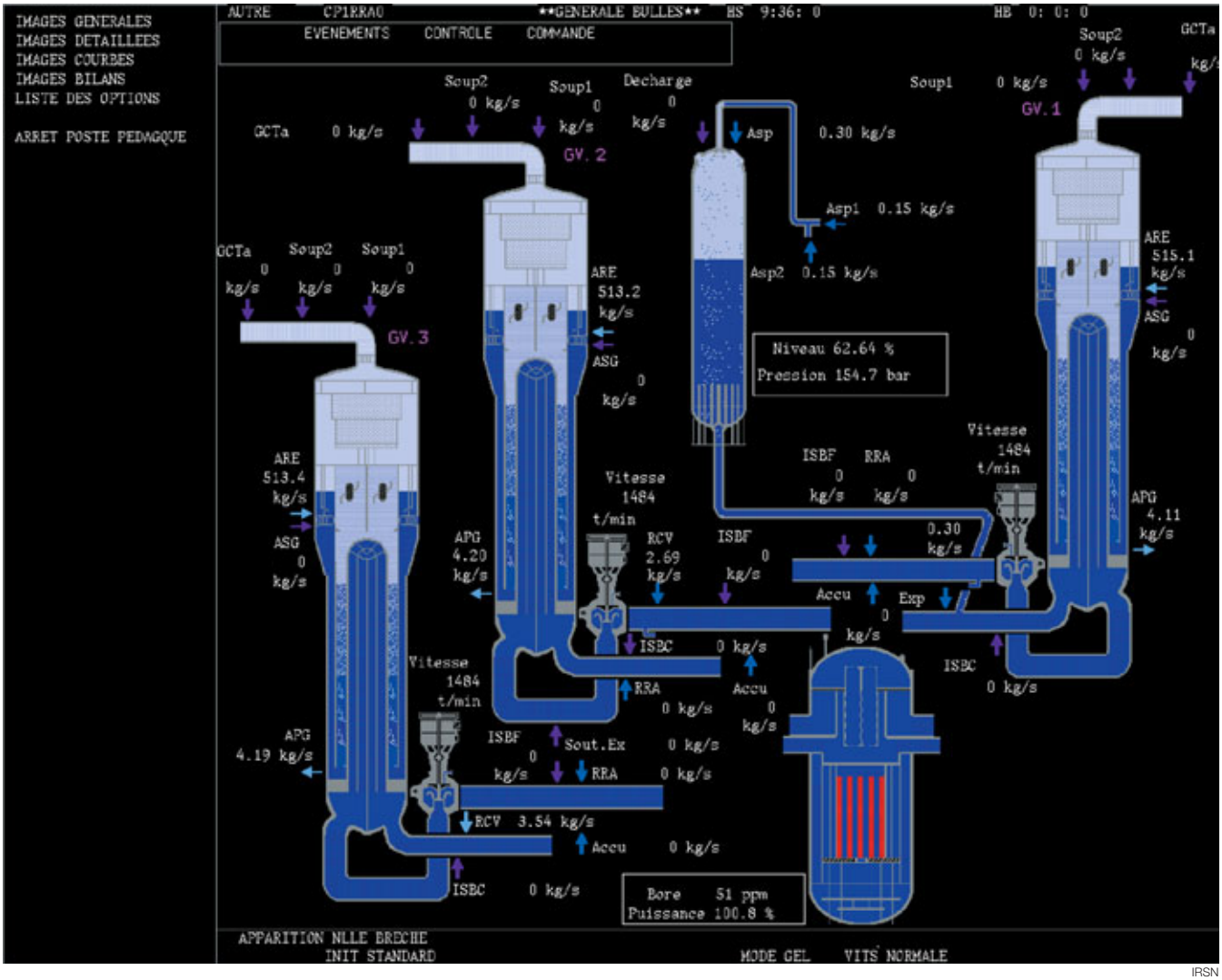
On the other hand, simulators are required to be effective tools for safety investigations. For this purpose, it was necessary to extend their simulation domain, especially to cover reactor shutdown states in normal, incident and accident operating conditions, and other accident situations for which it is particularly important that teams be trained: the main goal of the Scar (Simulator Cathare Release) project was to integrate standard code into Sipa simulators, with all the constraints involved in terms of real time.

Scar project methodology

Integrating Cathare into a simulator is primarily an interface problem. Cathare models the reactor's primary and secondary circuits and the residual heat removal (RHR) circuit,(2) and exchanges data with the hundred or so other components (containment, other circuits, control monitoring, etc.), whose operation is provided by simple models already implemented in the initial simulators. The ability for Cathare to model the RHR circuit in addition to the primary and secondary circuits represented one stage in the Scar project.

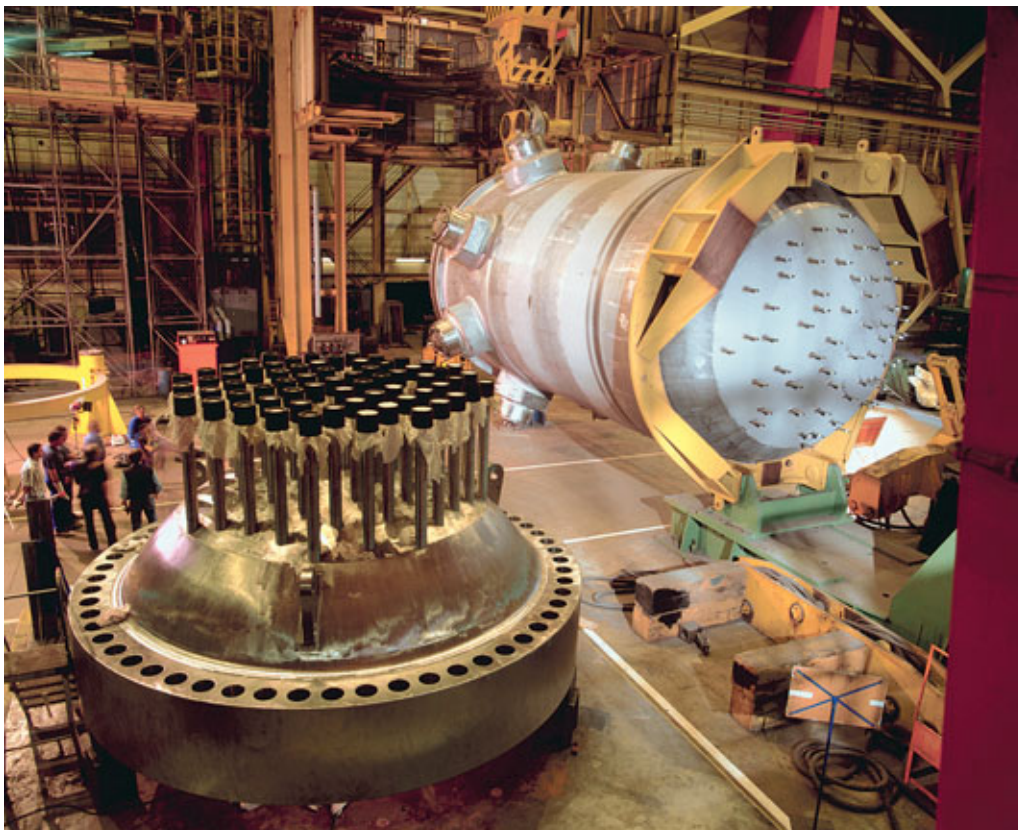
The basic task is to specify clearly, on paper, the boundaries of what is to be computed by Cathare, along with the physical variables that will have to be exchanged with the other systems. The mechanism and cycle rate of these exchanges are included in the Sipa simulators' formal framework. Schematically, the aim is for Cathare to sync temporally with the

(2) The RHR circuit's role is to cool a water-cooled reactor when the normal circuit cannot be used. This system is used mainly to dissipate residual heat released by the core after shutdown of the chain reaction.



The Sipa2 simulator teaching station, driven by the Cathare software.

IRSN



Framatome ANP/J.-P. Salomon

Preparing for hydraulic testing of the vessel (at right) intended for EDF's Civaux 1 power station, at Framatome–ANP's Chalon-Saint-Marcel plant (Saône-et-Loire département, central France). At left, the vessel cover, with control-rod crossings.

simulator's cycle rate, acquire for each cycle the *boundary conditions* that will control the calculation and then yield the variables required by the other systems. For that purpose, Cathare has been provided with all the interfaces involved in the exchange of variables (fluid takeoff branches, actuators, failure awareness, etc.), and it has been seen to it that this exchange be "codable" easily and automatically. The three circuits were then **discretized** and modeled in accordance with recommendations given in the user's guide.

Another aspect of this project is ensuring a good level of interactivity for the simulator, and hence increasing the code's computation speed, with real time being sought. Achieving this goal involves, on the one hand, implementing **parallel** computation techniques (see Box B, ***Computational resources for high-performance numerical simulation***) on multi-processor machines (speed increases by a factor of 50 have been achieved over the past four years), and, on the other hand, ensuring reliability of the numerical resolution method for the physical model. Well prepared, integration of Cathare into the simulator was a successful operation, currently validated by 23 control and accidental transients. As the outcome of this project, it is thus a fully validated version of the software which drives the simulator resulting from the Scar project. In addition to conventional interfaces (synoptics, operating diagrams, etc.), this tool features a "teaching station" which allows real-time visualization of the two-phase flows in the various circuits.

A general overview of accident sequences

Numerical simulation of accident situations in nuclear reactors is an essential means to a better knowledge, control and hence advancement of installation safety. The software programs used for these simulations rely on complex physical models and thus require a major validation effort on a very broad experimental basis.

Integration into research or training simulators of a software program used for safety analysis such as Cathare appears as a very important breakthrough, inasmuch as it allows combining high-level modeling of accident thermalhydraulics with the simulator's complete environment and graphic interface. As a training tool and instrument for the analysis of physical phenomena, the simulator offers operators and engineers charged with safety studies a general overview of accident sequences, while conserving the validity of the physical phenomena simulated.

The advances required, in terms of simulation realism (Box E, ***Advances in software engineering***), will lead to further improvements to the physical models, numerical methods and software architecture. Combined with virtual imagery techniques, these new tools will be at the core of tomorrow's simulators!

Bernard Faydide
Nuclear Energy Division
CEA Grenoble Center

What is a numerical simulation? A

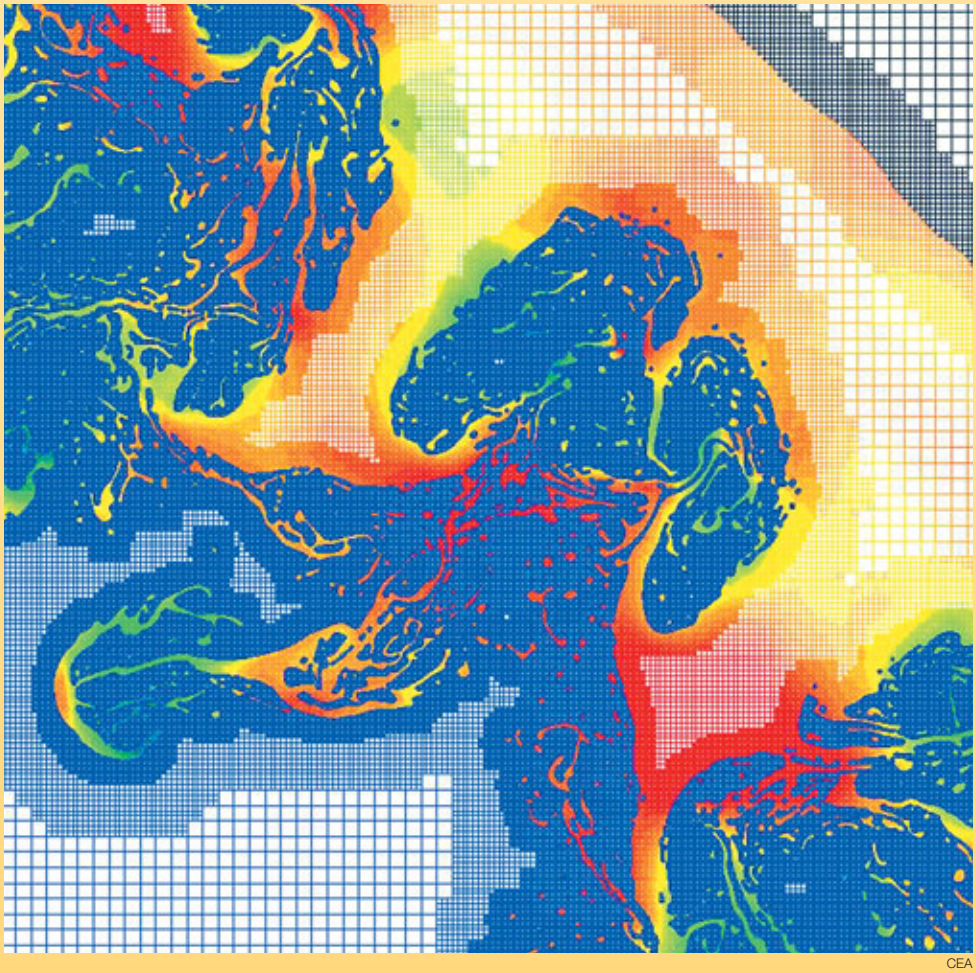
Numerical simulation consists in reproducing, through computation, a system's operation, described at a prior stage by an ensemble of **models**. It relies on specific mathematical and computational methods. The main stages involved in carrying out an investigation by means of numerical simulation are practices common to many sectors of research and industry, in particular nuclear engineering, aerospace or automotive.

At every point of the "object" considered, a number of physical quantities (velocity, temperature…) describe the state and evolution of the system being investigated. These are not independent, being linked and governed by **equations**, generally **partial differential** equations. These equations are the expression in mathematical terms of the physical laws modeling the object's behavior. Simulating the latter's state is to determine – at every point, ideally – the numerical values for its parameters. As there is an infinite number of points, and thus an infinite number of values to be calculated, this goal is unattainable (except in some very special cases, where the initial equations may be solved by analytical formulae). A natural approximation hence consists in considering only a finite number of points. The parameter values to be computed are thus finite in number, and the operations required become manageable, thanks to the computer. The actual number of points processed will depend, of course, on computational power: the greater the number, the better the object's description will ultimately be. The basis of parameter computation, as of numerical simulation, is thus the reduction of the infinite to the finite: **discretization**.

How exactly does one operate, starting from the model's mathematical equations? Two methods are very commonly used, being representative, respectively, of **deterministic computation** methods, resolving the equations governing the processes investigated after discretization of the variables, and methods of **statistical** or **probabilistic calculus**.

The principle of the former, known as the **finite-volume method**, dates from before the time of computer utilization. Each of the object's points is simply assimilated to a small elementary volume (a cube, for instance), hence the *finite-volume* tag. Plasma is thus considered as a set or lattice of contiguous volumes, which, by analogy to the makeup of netting, will be referred to as a **mesh**. The parameters for the object's state are now defined in each mesh cell. For each one of these, by reformulating the model's mathematical equations in terms of volume averages, it will then be possible to build up *algebraic relations* between the parameters for one cell and those of its neighbors. In total, there will be as many relations as there are unknown parameters, and it will be up to the computer to resolve the *system* of relations obtained. For that purpose, it will be necessary to turn to the techniques of **numerical analysis**, and to program specific **algorithms**.

The rising power of computers has allowed an increasing fineness of discretization, making it possible to go from a few tens of cells in the 1960s to several tens of thousands in the 1980s, through to millions in the 1990s, and up to some ten billion cells nowadays (Tera machine at CEA's Military Applications Division), a figure that should increase tenfold by the end of the decade.



CEA

Example of an image from a 2D simulation of instabilities, carried out with CEA's Tera supercomputer. Computation involved adaptive meshing, featuring finer resolution in the areas where processes are at their most complex.

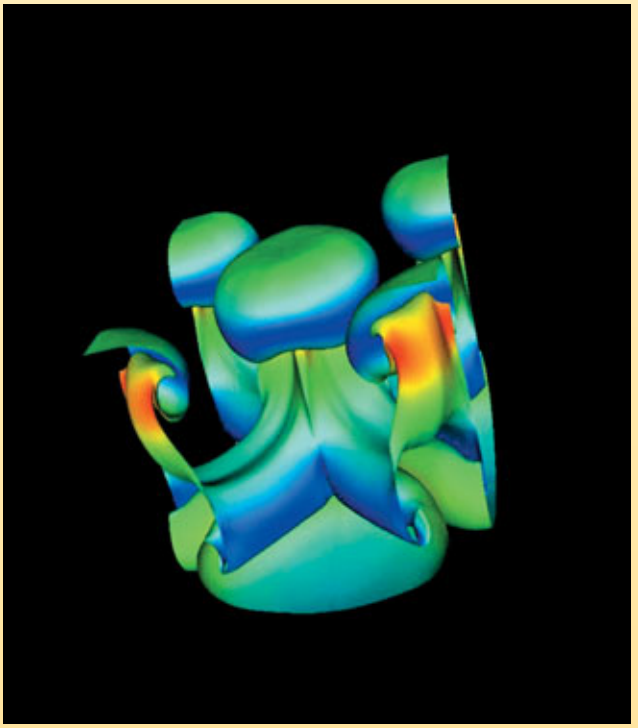
A refinement of meshing, **adaptive remeshing**, consists in adjusting cell size according to conditions, for example by making them smaller and more densely packed at the interfaces between two environments, where physical processes are most complex, or where variations are greatest.

The finite-volume method can be applied to highly diverse physical and mathematical situations. It allows any shape of mesh cell (cube, hexahedron, tetrahedron…), and the mesh may be altered in the course of computation, according to geometric or physical criteria. Finally, it is easy to implement in the context of **parallel computers** (see Box B, ***Computational resources for high-performance numerical computation***), as the mesh may be subjected to partitioning for the purposes of computation on this type of machine (example: Figure B).

Also included in this same group are the **finite-difference method**, a special case of the finite-volume method where cell walls are orthogonal, and the **finite-element method**, where a variety of cell types may be juxtaposed.

The second major method, the so-called **Monte Carlo** method, is particularly suited to the simulation of *particle transport*, for example of neutrons or photons in a **plasma** (see *Simulations in particle physics*). This kind of transport is in fact characterized by a succession of stages, where each particle may be subject to a variety of events (diffusion, absorption, emission…) that are possible *a priori*. Elementary probabilities for each of these events are known individually, for each particle.

It is then a natural move to assimilate a point in the plasma to a particle. A set of particles, finite in number, will form a representative sample of the infinity of particles in the plasma, as for a statistical survey. From one stage to the next, the sample's evolution will be determined by random draws (hence the method's name). The effectiveness of the method, implemented in Los Alamos as early as the 1940s, is of course dependent on the statistical quality of the random draws. There are, for just this purpose, *random-number* methods available, well suited to computer processing.



CEA

3D simulation carried out with the Tera supercomputer, set up at the end of 2001 at CEA's DAM-Île de France Center, at Bruyères-le-Châtel (Essonne* département*).

Finite-volume and Monte Carlo methods have been, and still are, the occasion for many mathematical investigations. These studies are devoted, in particular, to narrowing down these methods' convergence, i.e. the manner in which approximation precision varies with cell or particle number. This issue arises naturally, when confronting results from numerical simulation to experimental findings.

How does a numerical simulation proceed?

Reference is often made to *numerical experiments*, to emphasize the analogy between performing a numerical simulation and carrying out a physical experiment.

In short, the latter makes use of an experimental setup, configured in accordance with initial conditions (for temperature, pressure…) and control parameters (duration of the experiment, of measurements…). In the course of the experiment, the setup yields measurement points, which are recorded. These records are then analyzed and interpreted.

In a numerical simulation, the experimental setup consists in an ensemble of computational programs, run on computers. The **computation codes**, or **software** programs, are the expression, via numerical algorithms, of the mathematical formulations of the physical models being investigated. Prior to computation, and subsequent to it, *environment software* programs manage a number of complex operations for the preparation of computations and analysis of the results.

The initial data for the simulation will comprise, first of all, the delineation of the computation domain – on the basis of an approximate representation of the geometric shapes (produced by means of drafting and CAD [computer-assisted design] software) –, followed by discretization of this computation domain over a mesh, as well as the values for the physical parameters over that mesh, and the control parameters to ensure proper running of the programs… All these data (produced and managed by the environment software programs) will be taken up and verified by the codes. The actual results from the computations, i.e. the numerical values for the physical parameters, will be saved on the fly. In fact, a specific protocol will structure the computer-generated information, to form it into a numerical database.

A complete protocol organizes the electronic exchange of required information (dimensions, in particular) in accordance with predefined formats: modeler,(1) mesher,(2) mesh partitioner, com-

(1) The modeler is a tool enabling the generation and manipulation of points, curves and surfaces, for the purposes, for example, of mesh generation.
(2) The geometric shapes of a mesh are described by sets of points connected by curves and surfaces (Bézier curves and surfaces, for instance), representing its boundaries.

putation codes, visualization and analysis software programs. *Sensitivity* studies regarding the results (sensitivity to meshes and models) form part of the numerical "experiments."
On completion of computation (numerical resolution of the equations describing the physical processes occurring in each cell), analysis of the results by specialists will rely on use of the numerical database. This will involve a number of stages: selective extraction of data (according to the physical parameter of interest) and visualization, and data extraction and transfer for the purposes of computing and visualizing diagnostics.
This parallel between performing a computation case for a numerical experiment and carrying out a physical experiment does not end there: the numerical results will be compared to the experimental findings. This comparative analysis, carried out on the basis of standardized quantitative criteria, will make demands on both the experience and skill of engineers, physicists, and mathematicians. Its will result in further improvements to physical models and simulation software programs.

Bruno Scheurer
Military Applications Division
CEA DAM-Ile de France Center

Frederic Ducros and **Ulrich Bieder**
Nuclear Energy Division
CEA Grenoble Center

The example of a thermalhydraulics computation

Implementation of a numerical simulation protocol may be illustrated by the work carried out by the team developing the **thermalhydraulics** computation software Trio U. This work was carried out in the context of a study conducted in collaboration with the French Radiological Protection and Nuclear Safety Institute (IRSN: Institut de radioprotection et de sûreté nucléaire). The aim was to obtain very accurate data to provide engineers with wall heat-stress values for the components of a pressurized-water reactor in case of a major accident involving turbulent natural circulation of hot gases. This investigation requires simultaneous modeling of large-scale "system" effects and of small-scale **turbulent** processes (see Box F, ***Modeling and simulation of turbulent flows***).
This begins with specification of the overall computation model (Figure A), followed by production of the CAD model and corresponding mesh with commercial software programs (Figure B). Meshes of over five million cells require use of powerful graphics stations. In this example, the mesh for a steam generator (Figures C and D) has been partitioned to parcel out computation over eight processors on one of CEA's parallel computers: each color stands for a zone assigned to a specific processor. The computations, whose boundary conditions are provided by way of a "system" computation (Icare–Cathare), yield results which it is up to the specialists to interpret. In this case, visualization on graphics stations of the instantaneous values of the velocity field show the impact of a hot plume on the steam generator's tube-plate (section of the velocity field, at left on Figure E), and instantaneous temperature in the water box (at right).

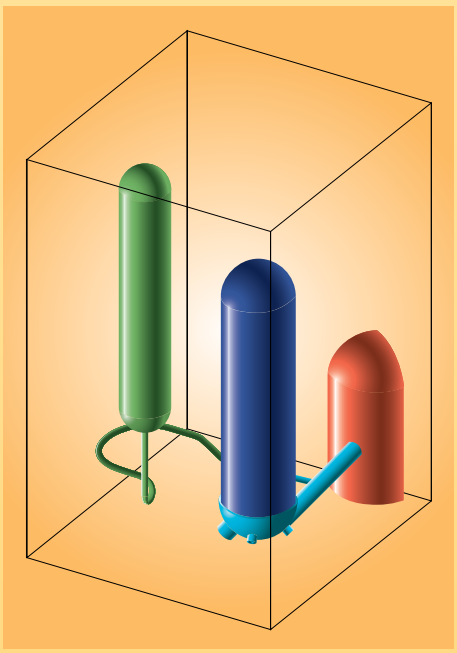


Figure A. Overall computation domain, including part of the reactor vessel (shown in red), the outlet pipe (hot leg, in light blue), steam generator (dark blue), and pressurizer (green).

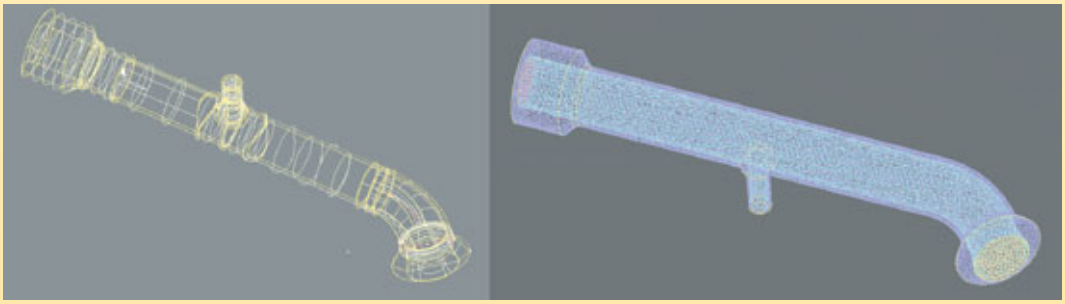
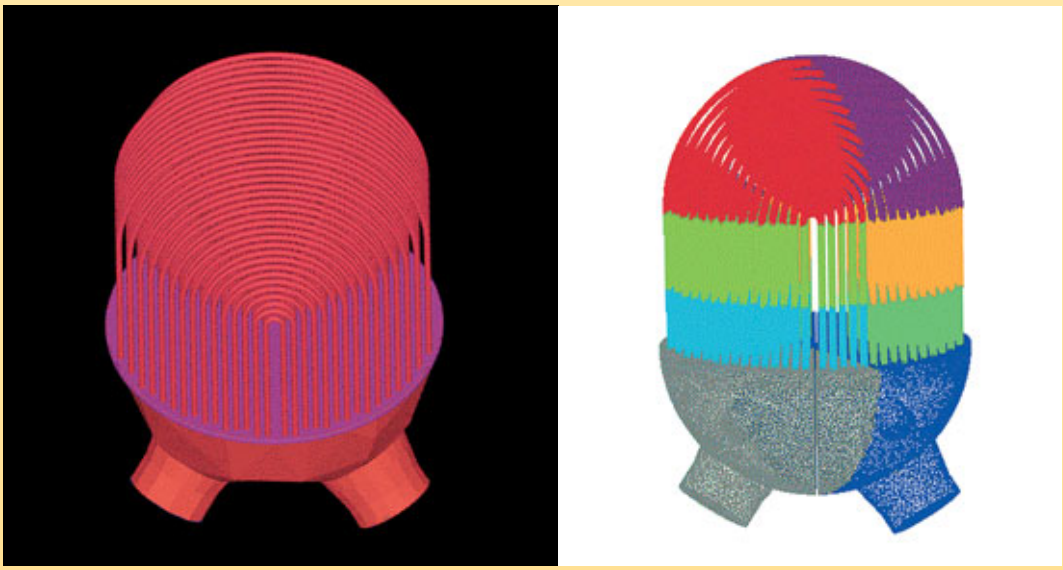


Figure B. CAD model of the hot leg of the reactor vessel outlet (left) and unstructured mesh for it (right).



Figures C and D.

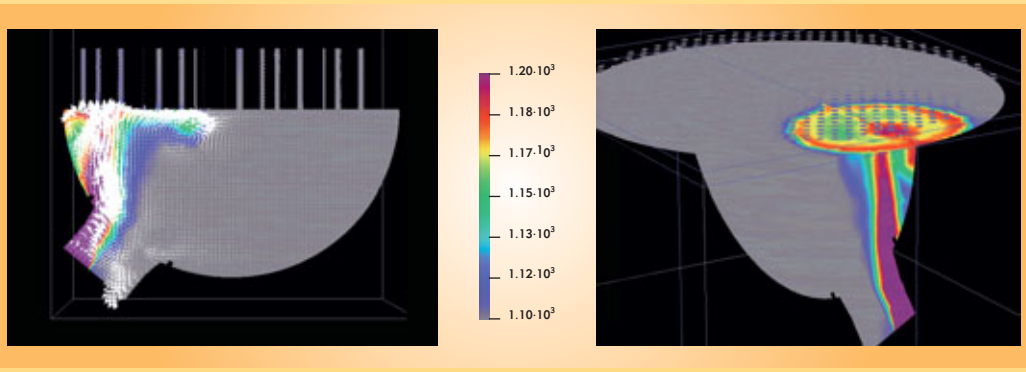


Figure E.

Computational resources for high-performance numerical simulation

B

Carrying out more accurate **numerical simulations** requires the use of more complex physical and numerical **models** applied to more detailed descriptions of the simulated objects (see Box A, ***What is a numerical simulation?***). All this requires advances in the area of simulation software but also a considerable increase in the capacity of the computer systems on which the software runs.

Scalar and vector processors

The key element of the computer is the processor, which is the basic unit that executes a program to carry out a computation. There are two main types of processors, **scalar processors** and **vector processors**. The former type carries out operations on elementary (scalar) numbers, for instance the addition of two numbers. The second type carries out operations on arrays of numbers (vectors), for example adding elementwise the numbers belonging to two sets of 500 elements. For this reason, they are particularly well suited to numerical simulation: when executing an operation of this type, a vector processor can operate at a rate close to its maximum (peak) performance. The same operation with a scalar processor requires many independent operations (operating one vector element at a time) executed at a rate well below its peak rate. The main advantage of scalar processors is their price: these are general-purpose microprocessors whose design and production costs can be written-down across broad markets.

Strengths and constraints of parallelism

Recent computers allow high performances partly by using a higher operating frequency, partly by trying to carry out several operations simultaneously: this is a first level of **parallelism**. The speeding up in frequency is bounded by developments in microelectronics technology, whereas interdependency between the instructions to be carried out by the processor limits the amount of parallelism that is possible. Simultaneous use of several processors is a second level of parallelism allowing better performance, provided programs able to take advantage of this are available. Whereas parallelism at processor level is automatic, parallelism *between processors* in a parallel computer must be taken into account by the programmer, who has to split his program into independent parts and make provisions for the necessary communication between them. Often, this is done by partitioning the domain on which the computation is done. Each processor simulates the behavior of one domain and regular communications between processors ensure consistency for the overall computation. To achieve an efficient parallel program, a balanced share of the workload must be ensured among the individual processors and efforts must be made to limit communications costs.

The various architectures

A variety of equipment types are used for numerical simulation. From their desktop computer where they prepare computations and analyze the results, users access shared computation, storage and visualization resources far more powerful than their own. All of these machines are connected by networks, enabling information to circulate between them at rates compatible with the volume of data produced, which can be as much as 1 **terabyte** (1 TB = 10^{12} bytes) of data for one single simulation.

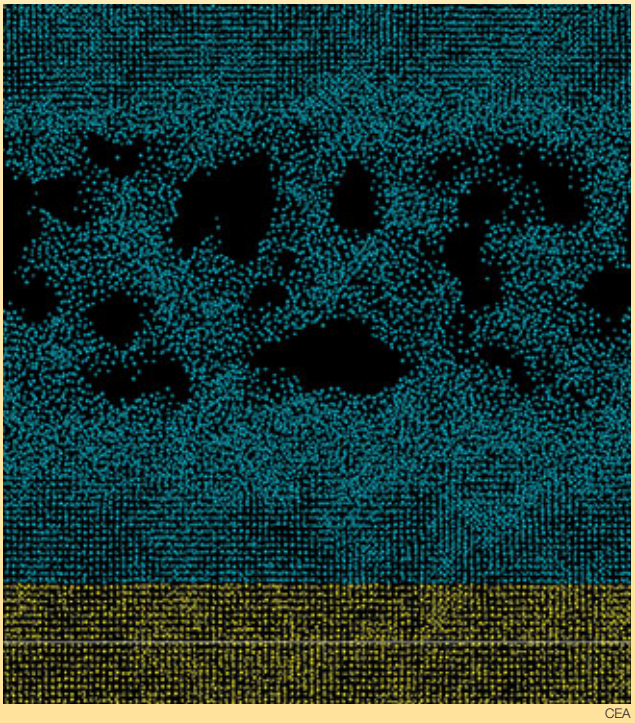
The most powerful computers are generally referred to as **supercomputers**. They currently attain capabilities counted in **teraflops** (1 Tflops = 10^{12} floating-point operations per second).

Currently, there are three main types of supercomputers: vector supercomputers, clusters of mini-computers with shared memory, and clusters of PCs (standard home computers). The choice between these architectures largely depends on the intended applications and uses. Vector supercomputers have very-high-performance processors but it is difficult to increase their computing performance by adding processors. PC clusters are inexpensive but poorly suited to environments where many users perform numerous large-scale computations (in terms of memory and input/output).

It is mainly for these reasons that CEA's Military Applications Division (DAM) has choosen for its Simulation Program (see *The Simulation Program: weapons assurance without nuclear testing*) architectures of the shared-memory mini-computer cluster type, also known as **clusters of SMPs** (symmetric multiprocessing). Such a system uses as a basic building block a mini-computer featuring several microprocessors sharing a common memory (see Figure). As these mini-computers are in widespread use in a variety of fields, ranging from banks to web servers through design offices, they offer an excellent performance/price ratio. These basic "blocks" (also known as *nodes*) are connected by a high-per-

Installed at CEA (DAM-Ile de France Center) in December 2001, the TERA machine designed by Compaq (now HP) has for its basic element a mini-computer with 4 x 1-GHz processors sharing 4 GB of memory and giving a total performance of 8 Gflops. These basic elements are interconnected through a fast network designed by Quadrics Ltd. A synchronization operation across all 2,560 processors is completed in under 25 microseconds. The overall file system offers 50 terabytes of storage space for input/output with an aggregate bandwidth of 7.5 GB/s.

Computational resources for high-performance numerical simulation (cont'd)



CEA

Parallel computers are well suited to numerical methods based on meshing (see Box A, What is a numerical simulation?) but equally to processing ab-initio calculations such as this molecular-dynamics simulation of impact damage to two copper plates moving at 1 km/s (see Simulation of materials). The system under consideration includes 100,000 atoms of copper representing a square-section (0.02 µm square) parallelogram of normal density. The atoms interact in accordance with an embedded atom potential over approximately 4–6 picoseconds. The calculation, performed on 18 processors of the Tera supercomputer at Bruyères-le-Châtel using the CEA-developed Stamp software, accounted for some ten minutes of "user" time (calculation carried out by B. Magne). Tests involving up to 64 million atoms have been carried out, requiring 256 processors over some one hundred hours.

formance network: the cumulated power of several hundreds of these "blocks" can reach several Tflops. One then speaks of a **massively parallel computer**.

Such power can be made available for one single parallel application using all the supercomputer's resources, but also for many independent applications, whether parallel or not, each using part of the resources.

While the characteristic emphasized to describe a supercomputer is usually its computational power, the input/output aspect should not be ignored. These machines, capable of running large-scale simulations, must have storage systems with suitable capacities and performance. In clusters of SMPs, each mini-computer has a local disk space. However, it is not advisable to use this space for the user files because it would require the user to move explicitly his data between each distinct stage of his calculation. For this reason, it is important to have disk space accessible by all of the mini-computers making up the supercomputer. This space generally consists in sets of disk drives connected to nodes whose main function is to manage them. Just as for computation, parallelism of input/output allows high performance to be obtained. For such purposes, parallel overall file systems must be implemented, enabling rapid and unrestricted access to the shared disk space.

While they offer considerable computational power, clusters of SMPs nevertheless pose a number of challenges. Among the most important, in addition to programming simulation software capable of using efficiently a large number of processors, is the development of operating systems and associated software tools compatible with such configurations, and fault-tolerant.

François Robin
Military Applications Division
CEA, DAM–Ile de France Center

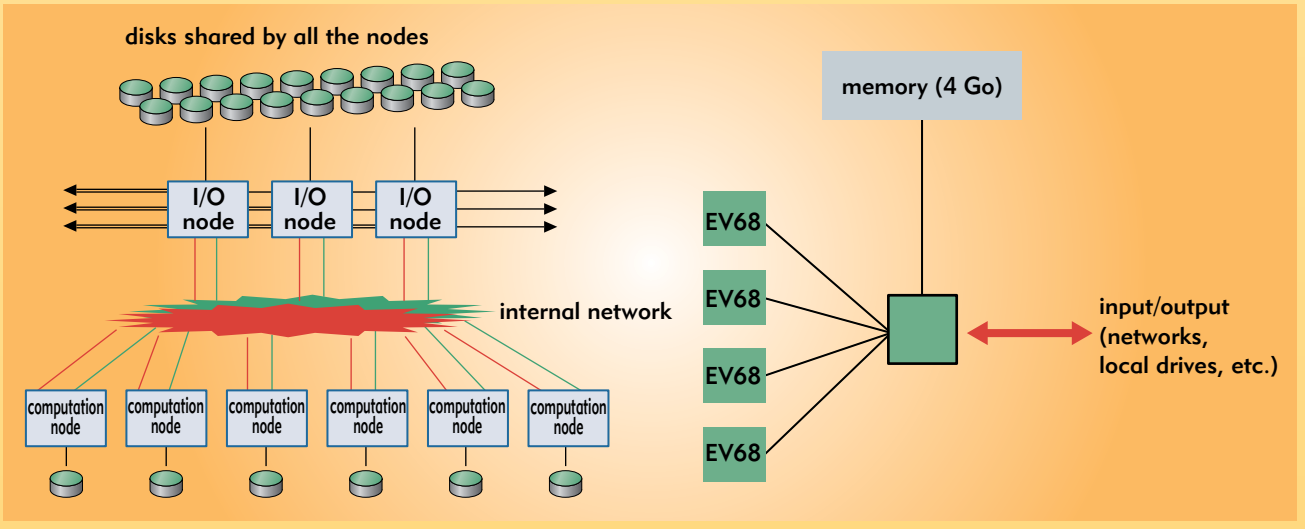


Figure. Architecture of an "SMP-cluster" type machine. At left, the general architecture (I/O = input/output), on the right, that of a node with four Alpha EV68 processors, clocked at 1 GHz.

Modeling and simulation of turbulent flows F

Turbulence, or disturbance in so-called turbulent flow, develops in most of the flows that condition our immediate environment (rivers, ocean, atmosphere). It also turns out to be one, if not the, dimensioning parameter in a large number of industrial flows (related to energy generation or conversion, aerodynamics, etc.). Thus, it is not surprising that a drive is being launched to achieve prediction for the process – albeit in approximate fashion as yet – especially when it combines with complicating processes (stratification, combustion, presence of several phases, etc.). This is because, paraxodically, even though it is possible to predict the turbulent nature of a flow and even, from a theoretical standpoint, to highlight certain common – and apparently universal – characteristics of turbulent flows,(1) their prediction, in specific cases, remains tricky. Indeed, it must take into account the considerable range of space and time scales(2) involved in any flow of this type.

Researchers, however, are not without resources, nowadays, when approaching this problem. First, the equations governing the evolution of turbulent flows over space and time (Navier–Stokes equations(3)) are known. Their complete solution, in highly favorable cases, has led to predictive descriptions. However, systematic use of this method of resolution comes up against two major difficulties: on the one hand, it would require complete, simultaneous knowledge of all variables attached to the flow, and of the forced-flow conditions imposed on it,(4) and, on the other hand, it would mobilize computational resources that will remain unrealistic for decades yet.

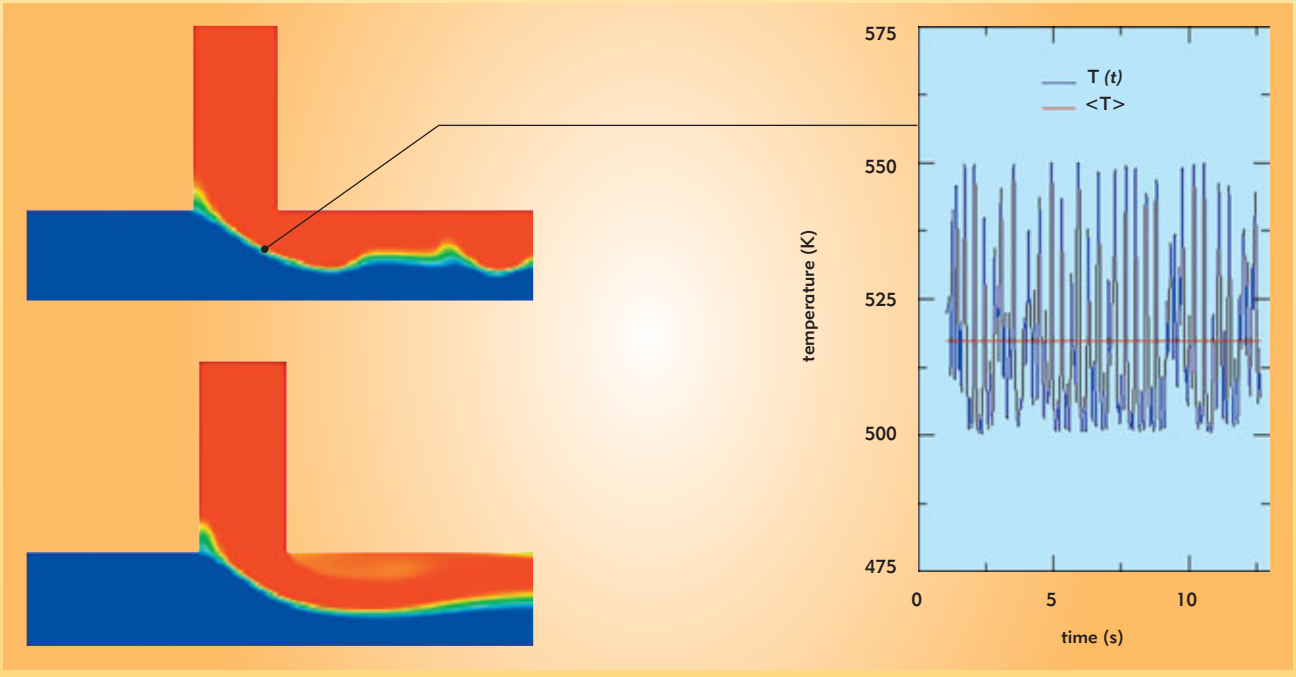


Figure. Instantaneous (top) and averaged (bottom) temperature field in a mixing situation. The curve shows the history of temperature at one point: fluctuating instantaneous value in blue and mean in red (according to Alexandre Chatelain, doctoral dissertation) (DEN/DTP/SMTH/LDTA).

The sole option, based on the fluctuating character of the flow due to turbulent agitation, must thus be to define and use average values. One of the most widely adopted approaches consists in looking at the problem from a statistical angle. The mean overall values for velocity, pressure, temperature… whose distribution characterizes the turbulent flow, are defined as the principal variables of the flow one then seeks to qualify relative to those mean values. This leads to a decomposition of the motion (the so-called Reynolds decomposition) into mean and fluctuating fields, the latter being the measure of the instantaneous local difference between each actual quantity and its mean (Figure). These fluctuations represent the turbulence and cover a major part of the Kolmogorov spectrum.(1)

This operation considerably lowers the number of degrees of liberty of the problem, making it amenable to computational treatment. It does also involve many difficulties: first, it should be noted that, precisely due to the non-linearity of the equations of motion, any average process leads to new, unknown terms that must be estimated. By closing the door on complete, deterministic description of the phenomenon, we open one to modeling, i.e. to the representation of the effects of turbulence on mean variables.

Many advances have been made since the early models (Prandtl, 1925). Modeling schemas have moved unabated towards greater complexity, grounded on the generally verified fact that any new extension allows the previously gained properties to be preserved. It should also be noted that, even if many new developments are emphasizing anew the need to treat flows by respecting their *non-stationary* character, the most popular modeling techniques were developed in the context of *stationary* flows, for which, consequently, only a representation of the flow's temporal mean can be achieved: in the final mathematical model, the effects of turbulence thus stem wholly from the modeling process.

It is equally remarkable that, despite extensive work, no modeling has yet been capable of accounting for all of the processes influencing turbulence or influenced by it (transition, non-stationarity, stratification, compression, etc.). Which, for the time being, would seem to preclude statistical modeling from entertaining any ambitions of universality.

Despite these limitations, most of the common statistical modeling techniques are now available in commercial codes and industrial tools. One cannot claim that they enable predictive computations in every situation. They are of varying accuracy, yielding useful results for the engineer in controlled, favorable situations (prediction of drag to an accuracy of 5–10%, sometimes better, for some profiles), but sometimes inaccurate in situations that subsequently turn out to lie outside the model's domain of validity. Any controlled use of modeling is based, therefore, on a qualification specific to the type of flow to be processed. Alternative modeling techniques, meeting the requirement for greater accuracy across broader ranges of space and time scales, and therefore based on a "mean" operator of a different nature, are currently being developed and represent new ways forward.

The landscape of turbulence modeling today is highly complex, and the unification of viewpoints and of the various modeling concepts remains a challenge. The tempting goal of modeling with universal validity thus remains out of order. Actual implementation proceeds, in most cases, from compromises, guided as a rule by the engineer's know-how.

(1) One may mention the spectral distribution of turbulent kinetic energy known as the "Kolmogorov spectrum," which illustrates very simply the hierarchy of scales, from large, energy-carrying scales to ever smaller, less energetic scales.
(2) This range results from the non-linearities of the equations of motion, giving rise to a broad range of spatial and temporal scales. This range is an increasing function of the Reynolds number, Re, which is a measure of the inertial force to viscous force ratio.
(3) The hypothesis that complete resolution of the Navier–Stokes equations allows simulation of turbulence is generally accepted to be true, at any rate for the range of shock-free flows.
(4) This is a problem governed by initial and boundary conditions.

Frédéric Ducros
Nuclear Energy Division
CEA Grenoble Center

Analytical experiments and integral experiments

To achieve better understanding and more in-depth knowledge, experiments focusing on investigation of a single phenomenon are carried out first, often at a smaller scale. These so-called **analytical**, *elementary* or *detailed* **experiments** enable individual evaluation of each phenomenon, or at any rate investigation of *separate effects*, by seeking to restrict the influence of other phenomena. The findings are then integrated as data used by the physics models in **computation codes** (**software** programs).

In the nuclear domain, the neutron-balance equation for a fission reactor (**Boltzmann equation**) provides an example of linearity; thus an experiment carried out on a low-power critical reactor such as Éole is representative of configurations found in power-generating reactors for key parameters of their design, such as power distribution or absorbing agent efficiency. However, the physics of **thermonuclear fusion** is non-linear: it is therefore impossible to extrapolate, given that thresholds have to be reached.

Experiments taking into account all of the elementary processes and thus – which is essential – their interactions are called **overall** or "**system**" **experiments**. Their purpose is to reproduce, possibly scaled down but with all of the elements of the system being investigated, the concatenation of the essential physical (or, as the case may be, chemical and biological) processes characterizing their operation, whether in normal conditions or "envelope" conditions, or even outside of those limits (accident situations, for instance, with the Bethsy loop in thermalhydraulics and the Cabri reactor in fuel thermalmechanics). These experiments highlight *system effects* and enable data acquisition, together with the definition of criteria, and are necessary to verify that the computation codes integrating all of this knowledge provide relevant modeling of reality.

Advances in software engineering

In the early days of scientific computation (in the period 1950–70), the physicist would be a "jack of all trades": physical and mathematical **modeling**, **numerical analysis**, programming, use of the tool once developed, analysis of the results (see Box A, **What is a numerical simulation?**).

Then, the complexity of the problems addressed, the need to control result accuracy and the stability of **numerical methods**, the quest for minimizing computation times (up to *real time* nowadays) led to "numericists" providing support to physicists for the design of **computation codes**. This evolution naturally raised a few temporary difficulties: the activity became a cross-disciplinary effort, and physicists lost some of their autonomy, giving place to synergy between two cutting-edge areas of expertise.

Then, the problems grew even more in scope and complexity: nowadays, nuclear reactor cores are calculated with the description of every one of the 40,000 **fuel** rods of varying composition and, simultaneously, of the **turbulent** water–steam flow surrounding them! (Box F, ***Modeling and simulation of turbulent flows***.) Issues of ergonomics, flexibility of use, sequencing, coupling and adaptation to computers with rapidly changing architectures also had to be addressed (see Box B, ***Computational resources for high-performance numerical simulation***). In turn, the integration of this third domain of expertise, software engineering, enabling more accurate or more relevant calculations, brought some organizational difficulties of its own.

The three areas of expertise (physics, numerical analysis and software engineering) are of course closely interwoven, and the know-how of each one directly impacts the others. It has become crucial to train engineers and researchers to master at least two of these domains, while having better than rudimentary knowledge of the third.

Within the framework of joint development by CEA and EdF of new computation and modeling platforms for nuclear power generation (**neutronics**, **thermalhydraulics**, fuel), the software architecture is just as important as physical models, numerical methods, or qualification. The issue has been examined for each individual project and globally between projects over a two-year period, ultimately leading to a five-level common architecture that will take on board user requirements for ergonomics, while enabling the physicists and numericists to express their know-how in optimum fashion.

These five software layers govern transitions between "real-world objects" (a fuel assembly, a steam pipe, etc.) and computational objects (**meshes**, arrays, solvers, fields, etc.).

The distance covered is vast, since the initial calculations performed in the 1950s on a slide rule and mainly based on rules of three, and today's calculation codes that can solve non-linear systems of **partial differential equations** on **supercomputers**. The physicist's and the engineer's know-how nevertheless remains intact: such is the experimental validation, whether intuitive or deductive, of the mathematical and physical modeling devised and integrated into the calculation codes. The true physicist retains the ability to predict and verify the orders of magnitude of the results yielded by these codes… using rules of three!

Thierry N'kaoua
Nuclear Power Division
CEA Saclay Center