WP 5 Project Deliverable D5.5

Software User Guide V2.0

Project Number	IST-2000-29266
Project Title	Virtual Real Time Fire Emergency Simulator
Deliverable Type	User Guide
Deliverable Class	Internal

Deliverable Number	D5.5
Title of Deliverable	Software User Guide V2.0
Nature of the Deliverable	User Guide
Contributing WPs	WP 5
Contractual Date of Delivery	December 31st, 2003
Actual Date of Delivery	January 8th, 2004
URL	www.virtualfires.org
Authors	Christian Redl (CD), Wilhelm Brandstätter (CD)
Contact Details	Institute for Structural Analysis / SiTu Research Univ. Prof. Dipl.-Ing. Dr. techn. Gernot Beer Lessingstrasse 25/II 8010 Graz / Austria Tel.: +43 316 8736180 Fax: +43 316 8736185 Email: gernot.beer@ifb.tu-graz.ac.at

Abstract	This report describes the physical and mathematical background as well as the turbulence modelling assumptions involved in the Virtualfires Software V2.0 named MPICE. The program structure of MPICE is presented in detail and all necessary input data to run the software in a parallel computing environment are specified. Finally a reference example of a fire simulation in a subway station is given.
Keywords	Tunnel Fires , Large-Eddy Simulation, Lattice Boltzmann Method

CONTENTS

1. DISCLAIMER

The Christian-Doppler-Laboratory for Applied Computational Thermofluiddynamics (CD) makes no warranty, expressed or implied, to users of the VIRTUALFIRES software named MPICE, and accepts no responsibility for its use. Users of MPICE assume sole responsibility for determining the appropriateness of its use in any particular application; for any conclusions drawn from the results of its use; and for any actions taken or not taken as a result of analyses performed using this tool.

Users are warned that MPICE is intended for use only by those competent in the fields of fluid dynamics, thermodynamics, combustion and heat transfer. It is intended only to supplement the informed judgement of the qualified user. The physical phenomena are too complex to be described in all detail using today's computer power. Therefore assumptions and simplifications are to be made and the user has to decide in which situations these are justified.

Contact information

Christian-Doppler-Laboratory for Applied Computational Thermofluiddynamics
at the University of Leoben
Franz-Josef-Strasse 18
A-8700 Leoben

Phone: +43/3842 402 – 8200
Fax: +43/3842 402 – 8202

Email: sekcdfd@unileoben.ac.at

2. INTRODUCTION

The rapid growth of computing power and the corresponding maturing of Computational Fluid Dynamics (CFD) has lead to the development of CFD-based "field" models applied to fire research problems. Virtually all this work is based on the conceptual framework provided by the Reynolds averaged form of the fundamental conservation equations and the effects of turbulence are taken into account by the so-called k-ε model [1]. The use of such a model was demonstrated in the first phase of the VIRTUALFIRES project (see D 5.1), where the commercial CFD solver FLUENT formed the basis of the analysis. However, CFD solvers like FLUENT have a fundamental limitation for fire applications – the averaging procedure at the root of the model equations. This causes a smoothed appearance of the results of even the most-highly resolved fire simulations. The smallest resolvable length scales are determined by the product of the local velocity and the averaging time, rather than the spatial resolution of the underlying computational grid.

Unfortunately, the evolution of large eddy structures characteristic of most fire plumes is lost with such an approach, as is the prediction of local transient events. The application of Large-Eddy Simulation (LES)-techniques is aimed at extracting greater temporal and special fidelity from simulations of fire. The basic idea behind the LES technique is that turbulent eddies that account for the fluctuation of the velocity field and mixing are large enough to be calculated with reasonable accuracy from the equations of fluid dynamics. Using LES-techniques in this way more realistic impressions of fires can be obtained.

In the first part of this report the underlying physical and mathematical framework, i.e. the fundamental conservation equations which describe the conservation of mass, momentum, energy and species, are described. Thereafter various turbulence models are discussed and the LES-technique is introduced. Then the Lattice Boltzmann (LB)-method, which is used for the numerical solution of the corresponding differential equations, is presented. Compared to conventional CFD methods as mentioned above, the LB-method is attractive for various reasons in context with fire simulations. As an explicit numerical scheme it does not require iterations. Secondly code parallelisation is extremely simply. Last but not least, as already discussed in the VIRTUALFIRES proposal, it shows potential for real time fire simulations.

The second part of the report is devoted to the documentation of the VIRTUALFIRES Software V2.0 named MPICE. This version is designed to run on parallel platforms. The domain decomposition is done automatically without any user interaction. According to the work program MPICE enables the user to simulate turbulent flame spread in tunnels containing arbitrarily shaped "objects" (cars, trucks and trains). It permits the computation and visualisation of the spatial and time-dependent smoke spread, temperature distribution, dynamic pressure variations, velocity field, fresh air/exhaust product stratification, etc. during the event of a hazard.

The structure of the MPICE is described in detail and all necessary files to run the program are specified. Finally a reference example for simulating a fire in a subway station is given.

3. GOVERNING EQUATIONS

The behaviour of continuum (both solid and fluid) is governed by the so-called transport equations based on the following basic laws of physics expressing balance (conservation) of:

- *mass,*
- *momentum (Newton's second law) and*
- *energy (First law of thermodynamics)*

The preferred way of writing equations expressing these laws is in differential form valid for an arbitrary point within the continuum:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j\right) = 0 \tag{1}$$

$$\frac{\partial \rho c_l}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j c_l\right) = \frac{\partial}{\partial x_j}\left[D_l \frac{\partial c_l}{\partial x_j}\right] + S_{c_l} \tag{2}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j u_i\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu \frac{\partial u_m}{\partial x_m}\delta_{ij}\right] + \rho g_i \tag{3}$$

$$\frac{\partial \rho h}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j h\right) = \frac{\partial}{\partial x_j}\left[\frac{\lambda}{C_P}\frac{\partial h}{\partial x_j}\right] + \frac{\partial p}{\partial t} \tag{4}$$

u_i is the velocity in direction x_i; ρ is density; p is pressure; g_i is the component of the gravitational acceleration vector in direction x_i; μ, D_l and λ are viscosity, diffusion coefficient of species l and heat conductivity respectively. S_{c_l} is the species source or sink.

In the case when the continuum is a mixture of various species:

- *which are mixed at the molecular level*
- *which share the same velocity, pressure and temperature fields, and*
- *in which the mass transfer between phases takes place by convection and diffusion*

the balance equation for the lth species concentration c_l or mass fraction defined as the ratio of the mass of the lth species m_l to the mass of the mixture m at a point

$$c_l = \frac{m_l}{m} \tag{5}$$

has to be solved. Since from the definition of mass fractions (Equ. 5) follows, that

$$\sum_{i=0}^{N} c_i = 1 \tag{6}$$

it is not necessary to solve transport equations (Equ. 2) for all the species. The species labelled by 0 is called *background* or *carrier fluid* and transport equations (Equ. 2) are solved for the mass fractions of the *additional species* (labelled from 1 to N).

Finally the total enthalpy is defined by:

$$h = C_p T + \frac{1}{2} u_i^2 \tag{7}$$

In this equation C_p represents the specific heat at constant pressure and T is the temperature.

Apart from the assumption, that the dissipation of kinetic energy into heat can be ignored for low Mach number flows, the above equations contain no approximations.

3.1 TURBULENCE MODELLING

Most engineering fluid flows are in a particular state of continuous instability called turbulence and can be said to be steady on an average basis only, since small scale, high frequency fluctuations of all the hydrodynamic variables in both space and time are always present. A flow exhibiting these macroscopic fluctuations is called turbulent flow.

Generally the degree of turbulence of a given flow is expressed by the Reynolds number (see Appendix A), e.g. for a pipe flow a Reynolds number higher than 2000 indicates a turbulent flow.

Assuming air as the ambient fluid and the tunnel diameter as a characteristic length one can suppose turbulent flows in case of tunnel fires.

The turbulent flow is well described by the Navier-Stokes equations. However their numerical solution requires a mesh with spacing smaller than the length scale of the smallest eddies and time steps smaller than the smallest time scale of turbulent fluctuations. Except for low Reynolds number flows in simple geometries this is not possible to achieve with today's computing power.

In a Large Eddy Simulation (LES) only the largest unsteady motions are resolved and all the smaller scales are modelled. This modelling can be done at a local basis for the Lattice BGK method using a Smagorinsky subgrid scale model. Details of LES can be found in the books by McComb [2], Pope [3], Wilcox [1] and Orlandi [4].

3.2 REYNOLDS AVERAGED NAVIER-STOKES

The RANS equations are obtained by using a statistical description of turbulent motion, formulated in terms of averaged quantities.
One such description uses the Reynolds averaging, whereby each dependent variable is expressed as the sum of its mean, or time-averaged value $\overline{\phi}$, and fluctuating component ϕ'.

$$\phi = \overline{\phi} + \phi', \tag{8}$$

where

$$\overline{\phi}(\mathbf{r},\mathbf{t}) = \frac{1}{\tau}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \phi\,(\mathbf{r},\mathbf{t}+\xi)\mathbf{d}\xi, \tag{9}$$

and the time interval τ is large enough with respect to the time scale λ_t of the turbulent fluctuations, but small with respect to the scale of other time dependent effects (see fig. 1). This practice is termed "ensemble-averaging".

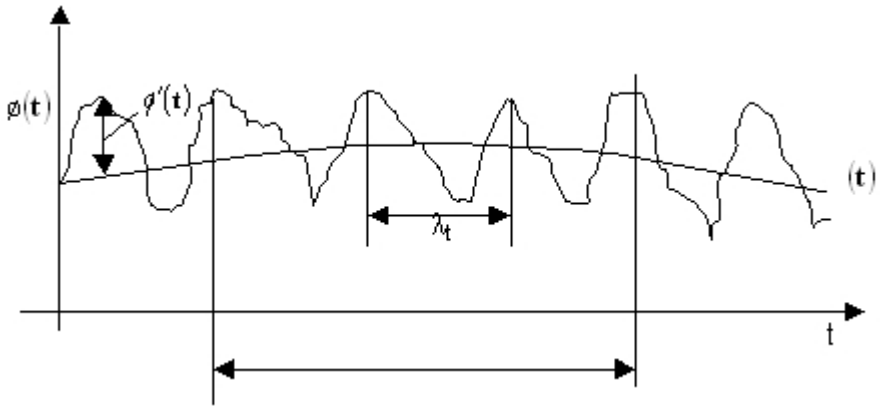


Figure 1: *Ensemble averaging procedure*

Applied to the governing equations of Section 3.1; the following equations for mass, energy and momentum balance in turbulent flows are obtained:

$$\frac{\partial\overline{\rho}}{\partial\mathbf{t}} + \frac{\partial}{\partial\mathbf{x_j}}\left(\overline{\rho}\,\overline{\mathbf{u}}_\mathbf{j}\right) = 0 \tag{10}$$

$$\frac{\partial\overline{\rho}\,\overline{\mathbf{c}}_\mathbf{l}}{\partial\mathbf{t}} + \frac{\partial}{\partial\mathbf{x_j}}\left(\rho\overline{\mathbf{u}}_\mathbf{j}\overline{\mathbf{c}}_\mathbf{l}\right) = \frac{\partial}{\partial\mathbf{x_j}}\left[\mathbf{D_l}\frac{\partial\overline{\mathbf{c}}_\mathbf{l}}{\partial\mathbf{x_j}} - \overline{\rho\mathbf{u}_\mathbf{j}'\mathbf{c}_\mathbf{l}'}\right] + \overline{\mathbf{S}_{\mathbf{c_l}}} \tag{11}$$

$$\frac{\partial\overline{\rho}\,\overline{\mathbf{u}}_\mathbf{i}}{\partial\mathbf{t}} + \frac{\partial}{\partial\mathbf{x_j}}\left(\overline{\rho}\,\overline{\mathbf{u}}_\mathbf{j}\overline{\mathbf{u}}_\mathbf{i}\right) = -\frac{\partial\overline{\mathbf{p}}}{\partial\mathbf{x_i}} + \frac{\partial}{\partial\mathbf{x_j}}\left[\mu\left(\frac{\partial\overline{\mathbf{u}}_\mathbf{i}}{\partial\mathbf{x_j}} + \frac{\partial\overline{\mathbf{u}}_\mathbf{j}}{\partial\mathbf{x_i}}\right) - \overline{\rho\mathbf{u}_\mathbf{j}'\mathbf{u}_\mathbf{i}'} - \frac{2}{3}\mu\frac{\partial\overline{\mathbf{u}}_\mathbf{m}}{\partial\mathbf{x_m}}\delta_{\mathbf{ij}}\right] + \rho\mathbf{g_i} \tag{12}$$

$$\frac{\partial\overline{\rho}\,\overline{\mathbf{h}}}{\partial\mathbf{t}} + \frac{\partial}{\partial\mathbf{x_j}}\left(\overline{\rho}\,\overline{\mathbf{u}}_\mathbf{j}\overline{\mathbf{h}}\right) = \frac{\partial}{\partial\mathbf{x_j}}\left[\frac{\lambda}{\mathbf{C_p}}\frac{\partial\overline{\mathbf{h}}}{\partial\mathbf{x_j}} - \overline{\rho\mathbf{u}_\mathbf{j}'\mathbf{h}'}\right] + \frac{\partial\overline{\mathbf{p}}}{\partial\mathbf{t}} \tag{13}$$

It is evident by comparing (Equ. 1) to (Equ. 4) with the above that the averaging procedure has produced equations where all instantaneous variables are replaced by their ensemble-averaged counterpart. Additionally new terms containing products of fluctuating quantities (so called

"correlations") have appeared. In what follows the averaging signs (overbars) are retained only for the correlations and all dependent variables are considered as averaged values.

The averaging procedure produced a set of new unknowns in the momentum and energy conservation equations, respectively:

Turbulent mass flux

$$\mathbf{q}_c^t = -\overline{\rho \mathbf{c}_l' \mathbf{u}_j'} \tag{14}$$

Turbulent momentum flux (Reynolds stress)

$$\mathbf{T}^t = -\overline{\rho \mathbf{u}_j' \mathbf{u}_i'} \tag{15}$$

Turbulent heat flux

$$\mathbf{q}_h^t = -\overline{\rho \mathbf{u}_j' \mathbf{h}'} \tag{16}$$

Since these quantities are unknown, the averaged equations are accompanied by the so-called *turbulence models*, which provide these unknowns by expressing the correlations of the fluctuations in terms of the mean quantities. To do so, one has to rely on experimental data and knowledge obtained from DNS. No single model can be expected to reproduce well the effects of turbulence on the mean flow in all practical applications.

The most popular turbulence models are *eddy-viscosity models*, which postulate an analogy between the turbulent and viscous diffusion (Boussinesq eddy-viscosity hypothesis) and model the effects of turbulence by introducing turbulent diffusivity and viscosity coefficients:

$$\begin{aligned}
\mathbf{q}_c^t &\approx \rho \mathbf{D}_{l,t} \frac{\partial \overline{\mathbf{c}}_l}{\partial \mathbf{x}_j}, \\
\mathbf{T}^t &\approx 2\mu_t \left(\frac{\partial \overline{\mathbf{u}}_i}{\partial \mathbf{x}_j} + \frac{\partial \overline{\mathbf{u}}_j}{\partial \mathbf{x}_i} \right) - \frac{2}{3} \left(\mu_t \frac{\partial \overline{\mathbf{u}}_m}{\partial \mathbf{x}_m} + \rho \mathbf{k} \right) \delta_{ij}, \\
\mathbf{q}_h^t &\approx \lambda_t \frac{\partial \mathbf{T}}{\partial \mathbf{x}_j},
\end{aligned} \tag{17}$$

where μ_t represents the turbulent viscosity and $\mathbf{k}$ stands for the turbulent kinetic energy. These coefficients are non-linear functions of the flow parameters and usually vary several orders of magnitude within the flow region.

3.3 FILTERED NAVIER-STOKES EQUATIONS

Contrary to Reynolds averaging in conventional CFD simulations the Navier-Stokes equations are required in filtered form to be used with LES.
The filter operation can be done either in spectral space (components greater than a given cut off frequency are suppressed) or in physical space (weighted averaging in a given volume).

The filtering can be described mathematically by

$$\bar{\mathbf{u}}_i(\mathbf{x},\mathbf{t}) = \int\int\int \mathbf{G}(\mathbf{x}-\xi,\Delta)\mathbf{u}_i(\xi,\mathbf{t})\mathbf{d}^3\xi \tag{18}$$

For the application with LBGK models the Navier-Stokes equations are filtered in physical space using a volume-average box filter.

$$\mathbf{G}(\mathbf{x}-\xi,\Delta) = \begin{cases} \frac{1}{\Delta^3}, & |\mathbf{x}_i - \xi_i| < \frac{\Delta \mathbf{x}_i}{2} \\ 0, & \textbf{otherwise} \end{cases} \tag{19}$$

In what follows the filtering operation is denoted by $\overline{\mathbf{u}_i} = \mathbf{G} * \mathbf{u}_i$.

Filtering the instantaneous balance equations leads to equations formally similar to the Reynolds averaged balance equations.

The instantaneous velocity can be decomposed into a resolvable filtered velocity and a sub grid scale velocity:

$$\mathbf{u}_i = \bar{\mathbf{u}}_i + \mathbf{u}_i' \tag{20}$$

Using the definitions above the incompressible continuity and Navier-Stokes equations can be written as:

$$\frac{\partial \bar{\mathbf{u}}_i}{\partial \mathbf{x}_i} = 0$$
$$\frac{\partial \bar{\mathbf{u}}_i}{\partial \mathbf{x}_i} + \frac{\partial}{\partial \mathbf{x}_j}(\overline{\mathbf{u}_i \mathbf{u}_j}) = -\frac{1}{\rho}\frac{\partial \bar{\mathbf{p}}}{\partial \mathbf{x}_i} + \nu \frac{\partial^2 \bar{\mathbf{u}}_i}{\partial \mathbf{x}_i \partial \mathbf{x}_j} \tag{21}$$

The filtered non-linear term can be written as

$$\overline{\mathbf{u}_i \mathbf{u}_j} = \mathbf{G} * (\mathbf{u}_i \mathbf{u}_j) = \mathbf{G} * (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_j + \mathbf{u}_i' \bar{\mathbf{u}}_j + \bar{\mathbf{u}}_i \mathbf{u}_j' + \mathbf{u}_i' \mathbf{u}_j') \tag{22}$$

The first term on the right hand side contains only explicit scales and the filter function and can be written as:

$$\mathbf{G} * (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_j) = \bar{\mathbf{u}}_i \bar{\mathbf{u}}_j + \mathbf{L}_{ij} \tag{23}$$

$\mathbf{L}_{ij}$ is usually referred to as Leonard stresses. There is no fundamental closure problem to calculate the above term, but if a finite-difference scheme of second order (as the LBGK model described in chapter 5) is used the Leonard stresses are in the same order as the truncation error. Therefore they are implicitly accounted for.
The remaining terms containing sub grid scales can be lumped together as

$$\mathbf{G}*(\mathbf{u}_i'\overline{\mathbf{u}}_j+\overline{\mathbf{u}}_i\mathbf{u}_j'+\mathbf{u}_i'\mathbf{u}_j')=\overline{\overline{\mathbf{u}}_i\mathbf{u}_j'}+\overline{\mathbf{u}_i'\overline{\mathbf{u}}_j}+\overline{\mathbf{u}_i'\mathbf{u}_j'}=\mathbf{C}_{ij}+\mathbf{R}_{ij} \tag{24}$$

where $\mathbf{C}_{ij}=\overline{\overline{\mathbf{u}}_i\mathbf{u}_j'}+\overline{\mathbf{u}_i'\overline{\mathbf{u}}_j}$ are the cross-term stresses and $\mathbf{R}_{ij}=\overline{\mathbf{u}_i'\mathbf{u}_j'}$ is the Reynolds stress tensor. As mentioned above the filtered equations are somewhere similar to the Reynolds averaged form but not identical.

Neglecting the Leonard stress the filtered Navier-Stokes equations can be rearranged

$$\frac{\partial\overline{\mathbf{u}}_i}{\partial\mathbf{x}_i}+\frac{\partial}{\partial\mathbf{x}_j}(\overline{\mathbf{u}_i\mathbf{u}_j})=-\frac{1}{\rho}\frac{\partial\overline{\mathbf{p}}}{\partial\mathbf{x}_i}+\nu\frac{\partial^2\overline{\mathbf{u}}_i}{\partial\mathbf{x}_i\partial\mathbf{x}_j}+\frac{\partial\tau_{ij}}{\partial\mathbf{x}_j} \tag{25}$$

where τ_{ij} contains the cross-term and Reynolds stress.

3.4 LARGE EDDY SIMULATION

The basic idea of a Large Eddy Simulation (LES) is only to resolve the largest unsteady turbulent motions. The role of the small eddies is limited to give the satisfactory role of dissipation that it is concentrated at small scales. The large energy carrying length scales of turbulence are very problem dependent, whereas the small scales are assumed to be more universal.

Referring to the filtered equations given above the modelling effort of a LES is taken on the sum of the cross-term and Reynolds stress. The subgrid scale model most often used is based on the Eddy viscosity concept and was derived by Smagorinsky in 1963.

Using the Boussinesq approximation τ_{ij} can be expressed as

$$\tau_{ij}=-2\nu_t\overline{\mathbf{S}}_{ij} \tag{26}$$

Assuming the eddy viscosity to be proportional to a characteristic length and velocity of the small scales, the expression for υ_t is given by

$$\upsilon_t=\mathbf{l}^2\sqrt{2\overline{\mathbf{S}}_{ij}\overline{\mathbf{S}}_{ij}} \tag{27}$$

The resolved strain rate tensor can be obtained locally within the LBGK method as a linear combination of the non-equilibrium distribution functions and reads in continuum form as

$$\overline{\mathbf{S}}_{ij}=\frac{1}{2}\left(\frac{\partial\overline{\mathbf{u}}_i}{\partial\mathbf{x}_j}+\frac{\partial\overline{\mathbf{u}}_j}{\partial\mathbf{x}_i}\right) \tag{28}$$

The length l is represents the scale of the small eddies and it must be related to the filter size. The expression best suited for a LBGK method working on uniform grids with spacing Δ is

$$\mathbf{l}=\mathbf{C}_s\Delta^2 \tag{29}$$

In the above equation C_S is the so-called Smagorinsky constant.

3.5 FILTER WIDTH AND SMAGORINSKY CONSTANT

Fig. 2 shows the energy spectrum for a turbulent flow in log-log scale.

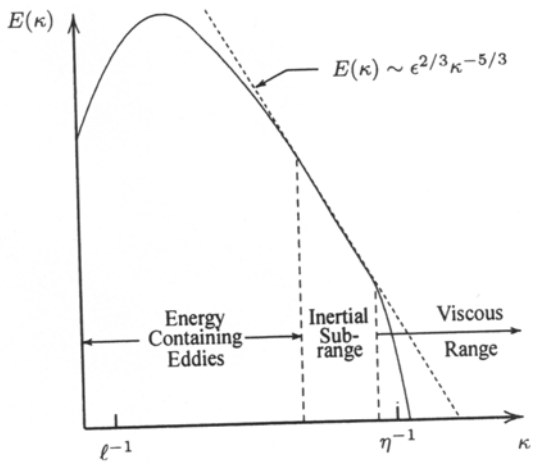


Figure 2: *Turbulent energy spectrum*

If the filter size were in the inertial sub range and sufficiently larger than the Kolmogorov viscous length scale the Smagorinsky constant would be universal and its theoretical value about 0.18 [1]. As this is rarely the case the Smagorinsky "coefficient" has to be calibrated for different kind of flows. Usually it lies in the range of 0.10 and 0.28. A lot of research is devoted to overcome the deficiency of the fixed static Smagorinsky constant. In these models the Smagorinsky constant is computed during the simulation from the flow variables.

If the grid were too coarse to resolve a substantial fraction of the turbulent kinetic energy (TKE) spectrum the method is classified as Very Large Eddy Simulation. Pope [3] claims a limiting value of 80 [%] of the resolved TKE as the criterion for a LES. As the fraction of the resolved energy is seldom estimated it is not always clear whether a simulation is a LES or a VLES.

As MPICE currently does not use wall functions to account for near wall effects and the grid is very coarse it is appropriate to classify it as a VLES.

4. REPRESENTATION OF FIRE

The fire is represented as a source of smoke and energy in a pre-defined region. This model, which does not simulate the combustion process itself, is known as Volumetric Heat Source (VHS) model. It is widely used for simulating fires in enclosures.

The heat release rate can be computed by a simple equation [5, 6, 7]

$$\mathbf{Q} = \dot{\mathbf{m}}_{\mathbf{fuel}} \mathbf{H}_{\mathbf{fuel}} \eta \tag{30}$$

where $\dot{\mathbf{m}}_{\mathbf{fuel}}$ is the fuel consumption, $\mathbf{H}_{\mathbf{fuel}}$ is the heating value of the fuel and η is a combustion efficiency parameter.

Radiation heat transfer can simply be accounted by using the radiative fraction approach, which changes (Equ. 30) to the following form

$$\mathbf{Q} = \dot{\mathbf{m}}_{\mathbf{fuel}} \mathbf{H}_{\mathbf{fuel}} \eta (1 - \chi_{\mathbf{R}}) \tag{31}$$

A fixed fraction $\chi_{\mathbf{R}}$ of the total heat released is assumed to be lost to the surroundings. Thermal radiation in the participating medium is ignored. As stated by [5] this fraction lies between 0.2 and 0.4 for non-premixed flames.

In MPICE it is possible to use pre-defined or user-defined smoke and energy emission charts like the one in Fig. 3 [8]. The energy and some emission charts can be described by defining the initial slope (increasing fire load), the final slope (decreasing fire load) and the overall duration of the fire.

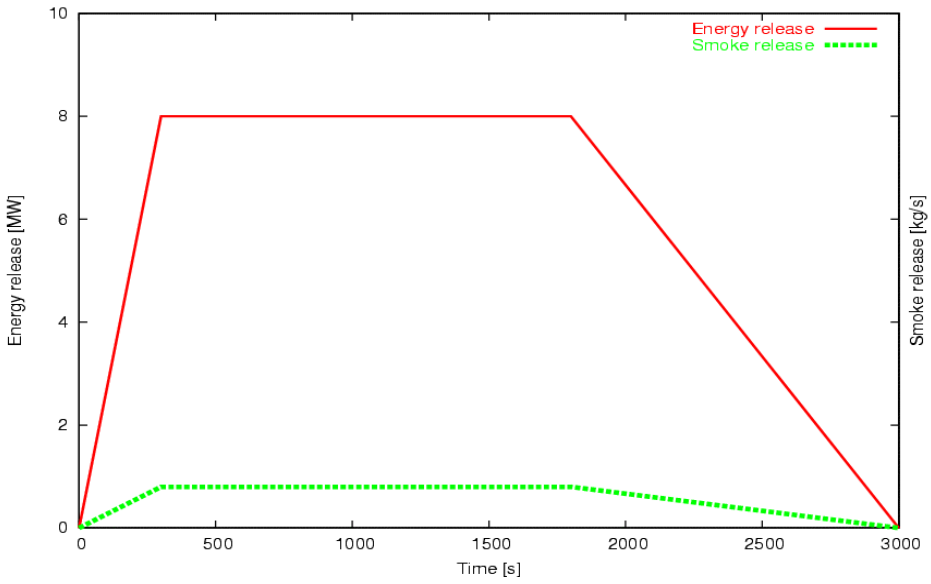


Figure 3: *Energy and smoke release rates*

The evolution of smoke is governed by the transport equation for a scalar species with source terms in pre-defined regions. The convective transport is calculated using the velocity of the carrier fluid.

If the energy equation is treated as an active scalar equation with the energy source region coinciding with the smoke source region the transport equations of energy and smoke become similar. Due to the difference in boundary conditions in general both equations would not be identical. Assuming a unity Lewis number and the walls to be adiabatic (which may be a first approximation in the early stage of fire development) the transport equation for energy and smoke are linearly dependent and only one of them has to be considered.

5. LATTICE BOLTZMANN METHOD

5.1 LATTICE BOLTZMANN EQUATION FOR FLUID FLOW

The Lattice Boltzmann Method (LBM) is a recent tool for simulating fluid flow problems based on kinetic theory. The Lattice Boltzmann Equation (LBE) is a finite difference approximation to the discrete Boltzmann equation using a Bhatnagar-Gross-Krook (BGK) model for the collision term.

For the derivation and theoretical background the interested reader is referred to the excellent reviews published by Luo [9] and Chen and Doolen [10]. A lot of information can also be found in the recent book by Succi [11].

In what follows the notation introduced by Qian [12] is used. The term dXqY means a model in X-dimensional space using Y discrete lattice vectors. Currently MPICE supports 2 dimensional model using 9 lattice vectors and 3 dimensional model using 15 or 19 lattice vectors.

Contrary to ICE Version 1.0 MPICE uses a multiple relaxation time (MRT) formulation of the lattice Boltzmann equation. The basic idea is to perform the translation step within the velocity space and the relaxation within the moment space. The mapping between the velocity and moment space is done by a simple transformation operation. The single relaxation time is replaced by a diagonal matrix containing one relaxation parameter for each hydrodynamic mode. An overview about the MRT model can be found in [14].

The evolution equation for a particle distribution function on a lattice consisting of discrete lattice vectors is given by

$$\left| f_\alpha(x_i + e_{\alpha,} t+1)\right\rangle = \left| f_\alpha(x_{i,} t)\right\rangle - \mathbf{M}^{-1}\mathbf{S}[\left| m_\alpha(x_{i,} t)\right\rangle - \left| m_\alpha^{(eq)}(x_{i,} t)\right\rangle] + \mathbf{F} \tag{32}$$

where f_α is the particle distribution function related to the lattice vector e_α, "(eq)" denotes the equilibrium particle distribution, S is the diagonal relaxation matrix and F is the body force term due to buoyancy.

The relaxation matrix is given by

$$\mathbf{S} = diag\ (s_{0,} s_{1,} \ldots s_{ndir\ -1}) \tag{33}$$

The moments are related to the density (ρ), momentum (j), momentum flux (q), energy (e), kinetic energy squared (ε), elements of the stress tensor (p, π) and there are some moments without obvious physical meaning (m) :

$$\left| m\right\rangle = (\rho, e, \varepsilon, j_x, q_x, j_y, q_y, j_z, q_{z,} 3p_{xx,} 3\pi_{xx}, p_{ww}, \pi_{ww}, p_{xy,} p_{yz,} p_{xz,} m_x, m_y, m_z)^T \tag{34}$$

Note that the equation above is in dimensionless form. The lattice structure for the D3Q19 model used in MPICE is depicted in Fig. 4 [13].

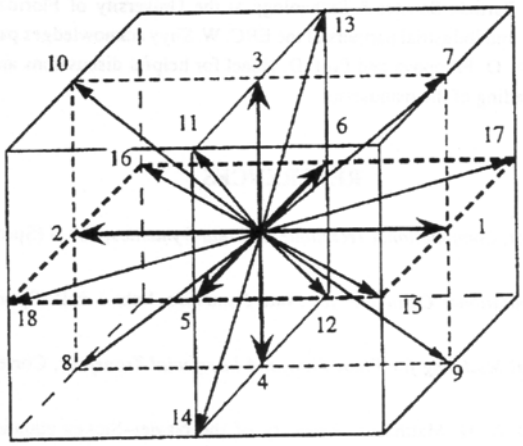


Figure 4: *D3Q19 model*

5.2 TRANSPORT EQUATIONS FOR SCALAR QUANTITIES IN THE LBGK FRAMEWORK

The transport equations for scalar quantities, e.g. smoke, are solved within the framework of the LBGK method by introducing an additional distribution function for each quantity.

The particle distribution functions for the scalar quantities evolve on the same lattice than the pressure distribution function for the fluid.

The LBE for the transport of species is given by

$$\mathbf{g}_i(\mathbf{x}+\mathbf{e}_i, t+1) = \mathbf{g}_i(\mathbf{x},t) + \frac{1}{\tau}\left[\mathbf{g}_i^{eq}(\mathbf{x},t) - \mathbf{g}_i(\mathbf{x},t)\right] + \mathbf{Q} \tag{35}$$

In the above equation Q is the volumetric heat source term. The equilibrium distribution function is given by

$$\mathbf{g}_\mathbf{i}^{\mathbf{eq}} = \mathbf{t}_\mathbf{i}\phi\left(1 + \frac{\mathbf{e}_{i\alpha}\mathbf{u}_\alpha}{\mathbf{c}_\mathbf{s}^2} + \frac{\mathbf{u}_\alpha\mathbf{u}_\beta}{2\mathbf{c}_\mathbf{s}^2}\left(\frac{\mathbf{e}_{i\alpha}\mathbf{e}_{i\alpha}}{\mathbf{c}_\mathbf{s}^2} - \delta_{\alpha\beta}\right)\right) \tag{36}$$

In principle it is possible to neglect terms quadratic in velocity and to use only a lattice consisting of the Cartesian lattice vectors, i.e. only 6 lattice vectors instead of 19 in the standard model. As relatively high flow velocities are expected to occur in most cases it is not advisable to use this reduced model as it introduces a compressibility error, which increases quadratically with increasing Mach number. In addition for high Peclet numbers the numerical error can be substantially.

6. INITIAL CONDITIONS

For initialising a simulation run values of velocity, pressure and concentration are required for each cell. Hence the equilibrium distributions calculated from the initial values are assigned to each cell prior to the start of the simulation.

In principle it is possible to include the first order deviations from the equilibrium state into the initialisation. The spatial and temporal derivatives of the hydrodynamic variables have to be known. It is assumed that for the temporal evolution of tunnel fires the non-equilibrium deviations are not important in the initialisation process.

If the .INI file is not present the simulation is initialised with zero velocity, unit pressure and zero concentration for each cell. A warning is issued to the user.

7. BOUNDARY CONDITIONS

REMARK

Inlet and outlet conditions are assigned to areas rather to cells, i.e. a certain prescribed hydrodynamic variable is assigned to all cells within the specified boundary area. If the boundary values are required to vary over a certain area this area has to be subdivided into an appropriate number of boundary areas.

For all boundary conditions the equilibrium distributions are calculated based on the prescribed or extrapolated hydrodynamic variables.

7.1 INLET BOUNDARY CONDITION

7.1.1 VELOCITY INLET

The velocity inlet condition is used to prescribe a certain velocity at an inlet opening. The pressure is extrapolated from the corresponding neighbour cells. For stability reasons this extrapolation is of first order.

7.1.2 VELOCITY INLET ACCORDING TO LADD

Ladd [16] proposed a velocity boundary condition based on the bounce back concept (see chapter 7.3). The velocity and the pressure at the inlet have to be specified by the user.

7.1.3 MASSFLUX INLET

The mass flux inlet condition is used to prescribe a certain mass flux at an inlet opening. The velocity is calculated using the fluid density according to the inlet temperature. If the calculated velocity is above a critical limit (Ma > 0.3) the velocity is set to this limit and a warning is issued to the user.

7.2 OUTLET BOUNDARY CONDITIONS

7.2.1 OUTFLOW

For the outflow condition both the pressure and the velocity are obtained by an appropriate extrapolating from the corresponding neighbour cells. For flows with higher Mach number this boundary condition is subject to severe mass conservation errors. Therefore the outflow boundary conditions should be used with caution.

7.2.2 PRESSURE OUTLET

The pressure outlet condition uses a pressure specified by the user at the outlet. Again the velocity is obtained by an extrapolation procedure from the neighbour cells. Generally the pressure outlet condition leads to a higher numerical stability than the outflow boundary condition.

7.3 FIXED WALL BOUNDARY CONDITIONS

At fixed walls the bounce back rule is applied, which means that the momentum of an incoming particle distribution is reversed at the boundary node and the particle distribution is translated back to the node it comes from in the next time step. This results in a zero velocity at the wall, which is assumed to be located at the half distance between the last active fluid node and the first wall node.

Fig. 5 depicts the “Bounce Back” rule. The red arrows show the state before and subsequent to the bounce back.

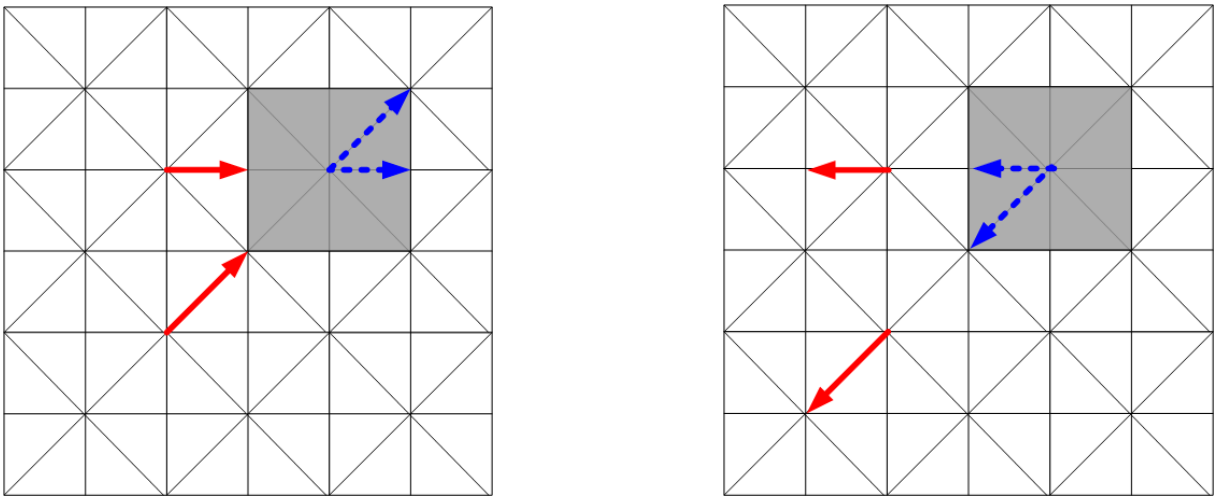


Figure 5: *Schematic representation of the bounce back rule*

For scalar quantities the fixed wall boundary conditions are accounted for by assigning the reference value specified in the .CAS file to the wall nodes.

7.4 SYMMETRY BOUNDARY

A symmetry boundary is an imaginary wall that defines a plane of flow symmetry. This means that neither flow nor scalar flux cross the boundary. However unlike a fixed wall the fluid behaviour near the symmetry boundary on one side is a mirror image of the fluid behaviour on the other side. The normal velocities are zero at symmetry boundaries.

Symmetry boundaries can be specified only for whole covering surfaces.

7.5 PERIODIC BOUNDARIES

At periodic or cyclic boundaries an exact copy of all variables of the corresponding boundary surface is assigned to the cells, i.e. the flow leaving the computational domain at one boundary re-enters the domain at the corresponding periodic boundary.

Periodic boundary conditions can be specified for the X-, Y- and Z-direction. Therefore both covering surfaces in the corresponding direction are set to periodic boundaries.

7.6 POTENTIAL PITFALLS REGARDING BOUNDARY CONDITIONS

As the fluid flow inside the computational domain is driven by the boundary conditions it is very important to specify physically meaningful boundary conditions. Inappropriate selection of boundary conditions may be the cause of failing of the simulation run.

Care has to be taken in selecting multiple outlet boundaries. It is not possible to combine an outflow condition with a constant pressure outlet as this problem is under specified.

For symmetry boundary conditions it may be important to note that symmetry of the computational domain does not imply that the flow possesses the same symmetry.

7.7 TIME DEPENDENT BOUNDARY CONDITIONS

The boundary conditions may vary during a simulation run. It is therefore possible for the user to specify these changes either in advance using a description file or interactively from within the VR system during run time. The syntax of the description file is described in a later section.

8. DOMAIN DECOMPOSITION

There exist a number of possibilities for splitting a computational domain onto different processors in a parallel computing environment. The most popular are geometry based methods (e.g. Cartesian coordinate bisection, coordinate bisection, recursive coordinate bisection) and graph based methods (e. g. recursive graph bisection, Greedy algorithm, recursive spectral bisection, Kernighan-Lin method).

For the strictly regular computational meshes used within the Virtual Fires project two domain decomposition methods promise the best performance due to a minimum amount of communication between the processors.

The simplest one is a one-dimensional decomposition (ODD) which works along the direction of longest extend of the computational domain. This method works for any number of processors, but will lead to bad surface/volume ratios for higher number of processors as can be inferred from fig. 6. The term surface/volume ratio characterises the number of computational mesh cells located on an inter-processor communication boundary to the number of computational mesh cells allocated to an individual processor.

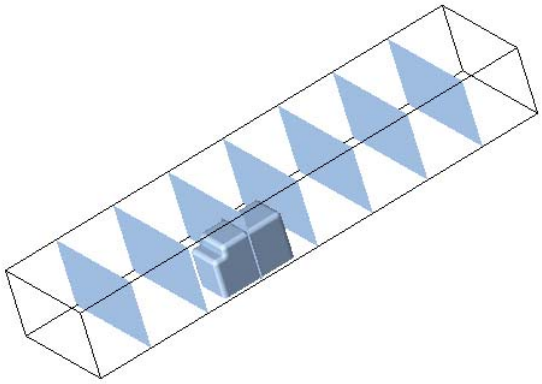


Figure 6: *1- dimensional domain decomposition*

Alternatively, for the recursive co-ordinate bisection method (RCB) the number of sub-domains must be a power of 2, but in most cases it delivers better topologies of the sub domains (fig. 7). The Send/Receive lists for the inter-processor communication are almost of equal size in contrast to the ODD method. The major disadvantage is that each processor has to communicate with a larger number of neighbours as in the ODD method (2 processor neighbours as a maximum).

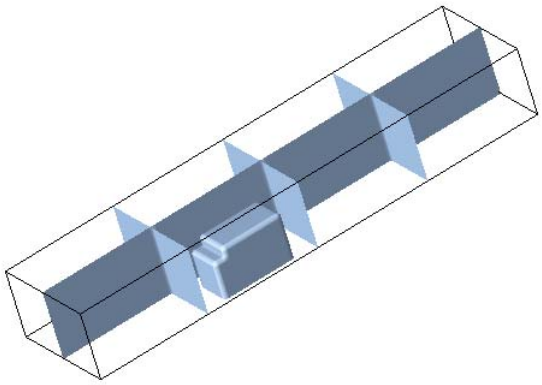


Figure 7: *Recursive co-ordinate bisection method*

Both decomposition methods have been implemented. It was observed at the performance tests below, that for the tested configurations the ODD method always shows higher performance. This is primary because of the larger number of processors involved in the communication process. As the ODD also offers more flexibility (arbitrary number of processors possible) it is the default domain decomposition used in MPICE from now on.

It should be pointed out that there is no user interaction necessary to perform the domain decomposition within MPICE. Once the number of processors is selected, the domain is subdivided in a suitable number of sub-domains using the above described one-dimensional domain decomposition methods.

9. COUPLING MPICE TO THE DATA MANAGEMENT CLIENT

The data management client (DMC) provides the necessary functionality to control MPICE interactively during run time. MPICE communicates with the DMC in a two-way-fashion. It sends data to and receives data from the DMC.

Currently the DMC provides the following basic functions:

`int InitConnection()`

Called at the beginning of a simulation run to connect to the data manager.

`void GetWD(char* WD, int &length)`

Communicates the present working directory to the simulation program

`void NewTSFinished(int TSID, float second)`

Called if a new data file for visualisation has been written. The `TSID` is the dimensionless time in the Lattice Boltzmann calculation, whereas `second` represents the corresponding real time.

`void NewRSFinished(int TSID, float second)`

Called if a new restart file has been written. The `TSID` is the dimensionless time in the Lattice Boltzmann calculation, whereas `second` represents the corresponding real time.

`int StopCalculation`

This function is called every time step. If it returns the value “1” the calculation is stopped.

`int GetBoundaries(float* pValues, int* pState, int* NumBnd)`

The data manager send the boundary states (“1”=on, “0”=off) and values (pressure) to the simulator. MPICE uses these boundary settings for the ongoing calculations.

`void getArray(float* array, int dim)`

This function communicates an one-dimensional array of real numbers to the data manager. The length of the array in `dim.` The data manager may process the array and send it back.

`void getArrayDouble(double* array, int dim)`

This function communicates an one-dimensional array of double precision numbers to the data manager. The length of the array in `dim.` The data manager may process the array and send it back.

`void ICEFinished(int Reason)`

This function is called either if the end of the simulation time is reached (return value = 0) or after the simulation has been stopped by the DMC (return value = 1).

10. PROGRAM STRUCTURE OF MPICE

MPICE is based on the MPI communication language [21]. In what follows an overview over the program structure is given by a pseudo code.

```
PROGRAM MPIce

!---Include the MPI library

INCLUDE “mpif.h”

!---Initialise MPI
  CALL MPI_INIT(IERR)
!---Communicate the number of processors
  CALL MPI_COMM_SIZE(MPI_COMM_WORLD, NP)
!---Every processor finds its ID
  CALL MPI_COMM_RANK(MPI_COMM_WORLD, MYID)

!---Processor 0 performs the data input
data_input : IF(MYID == 0) THEN

!---Read geometry data and perform the domain decomposition
           OPEN <ICE_Basename>.GEO
           READ geometry
           DO domain_decomposition
           DISTRIBUTE subdomain extensions to all processors

!---Set up the SEND and RECEIVE lists
           Loop over all boundary cells of each sub domain
           Determine SEND ID and corresponding RECEIVE ID
                for each boundary distribution
           Sort SEND and RECEIVE IDs and store it
                in the SEND and RECEIVE lists
           Communicate SEND and RECEIVE lists

!---Read case file, boundary definitions and initial data
           READ <ICE_Basename>.CAS, .INI or .CGNS
           DISTRIBUTE case and data information

ELSEIF(MYID /= 0)
           RECEIVE subdomain information, SEND and RECEIVE
                lists, case and data information

ENDIF data_input
```

```
!---All processors perform the calculation
solve_equations : DO time = 1, number_of_timesteps

!---Check if boundary conditions are to be updated
     IF(DMC_Update_BC) Update boundary conditions

     Impose boundary conditions at boundary points
     Update the solution at all grid points in the
          interior of the sub domains
     Copy DISTRIBUTION -> SEND list
     Communicate (MPI_SEND/MPI_RECEIVE) distribution
          functions via SEND/RECEIVE lists
          Copy RECEIVE list -> DISTRIBUTION

!---Check if results are required by the DMC
          dump_results : IF(DMC_Require_data) THEN
!---All data are gathered by Processor 0
                CALL MPI_GATHER(data)
!---Processor 0 send the data to
                IF(MYID == 0) CALL SENT_DATA_TO_DATABASE
          end dump_results : ENDIF(DMC_Require_data)

END DO solve equations

!---Finalize the parallel execution
CALL MPI_FINALIZE(IERR)

END PROGRAM MPIce
```

A graphical representation of the structure of MPICE and its communication with the DMC is displayed in fig. 8.

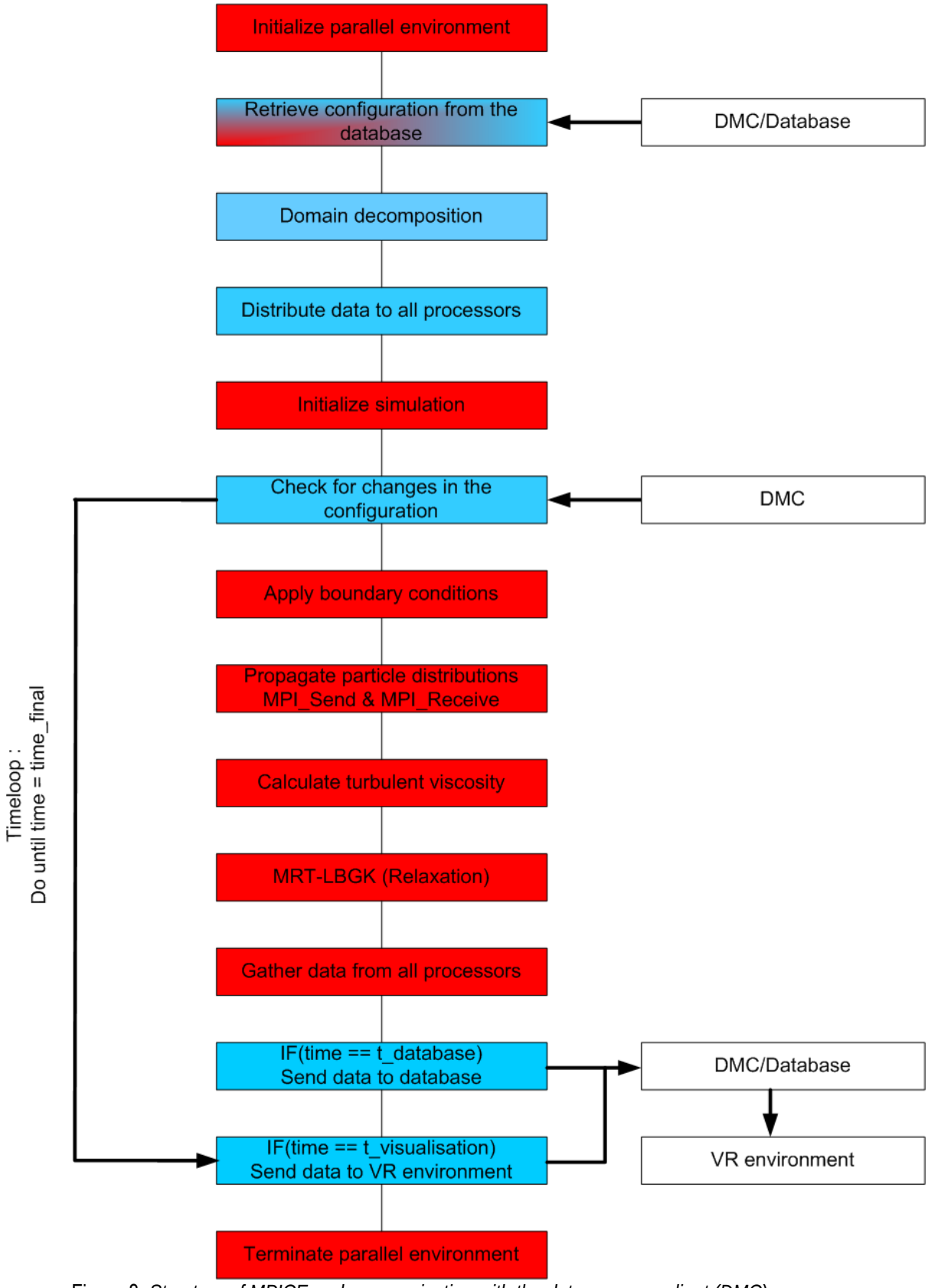


Figure 8: *Structure of MPICE and communication with the data manager client (DMC)*

11. SPECIFICATION OF THE SIMULATION BASE NAME AND FILE NAME CONVENTIONS

11.1 ICE_BASENAME

For a simulation run all file names are constructed using a common name called ICE_BASENAME. The simulation program searches all its necessary input files in the directory it has been started.

The first file read is ICE_INPUT. It simply contains the ICE_BASENAME consisting of exactly six letters. The ICE_BASENAME must be the first entry in ICE_INPUT.

11.2 FILE NAME CONVENTIONS

All file names are based on ICE_BASENAME. In what follows the file name conventions are highlighted:

<ICE_BASENAME>.CAS	Case file
<ICE_BASENAME>.GEO	Geometry description
<ICE_BASENAME>.INI	File containing the initial values
<ICE_BASENAME>.TBC	Descrition file for time dependent boundary conditions
<ICE_BASENAME>.LOG	File containing log information and error messages
<ICE_BASENAME>.RES	Restart file containing velocity, pressure and concentrations
<ICE_BASENAME><TIMEID>.cngs	Result file in CGNS data format. The <TIMEID> consists of 6 integer values, e.g “001200” for time step “1200”. The corresponding real time is available via the data manager.
<ICE_BASENAME>_Grid.cgns	This file contains the real world coordinates of the grid points.
<ICE_BASENAME>.<DAT>	Result file containing velocity and pressure in data format <DAT>
<ICE_BASENAME>_SPEC<ID>.<DAT>	Result file containing concentration of species <ID> in data format <DAT>

Fig. 7 shows the file structure used by MPICE.

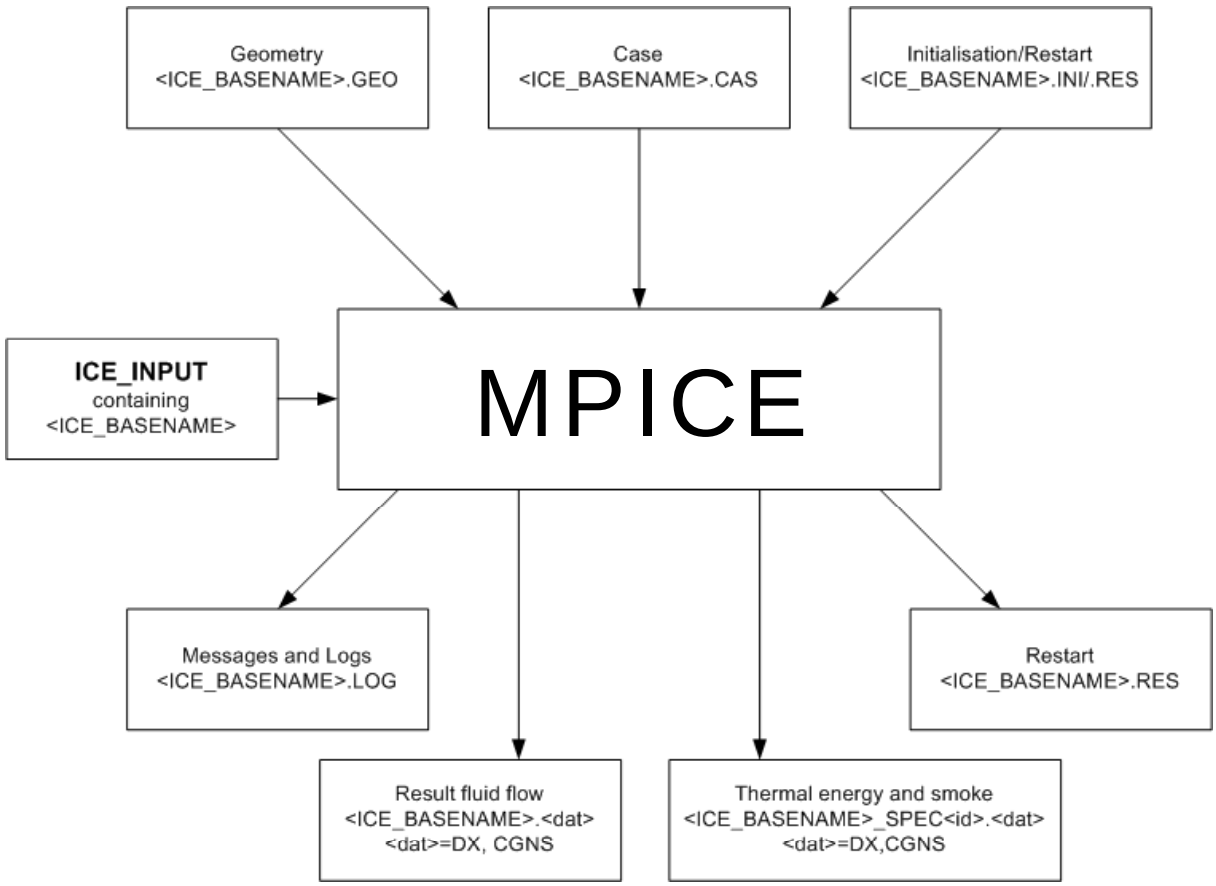


Figure 9: *MPICE file structure*

12. SPECIFICATION OF THE CASE FILE <ICE_BASENAME>.CAS

The file <ICE_BASENAME>.CAS contains all information regarding the fluid and species properties, models, turbulence parameter and time management.

Specifiers marked in blue are default setting and should not be changed by the user.
Specifiers marked in red are to be chosen by the user.

12.1 STRUCTURE

```
MODEL
        NDIM[INT]                   NDIR_FLUID[INT]          NDIR[SPECIES]
NCOMP
        SOLVE_FLUID_FLOW[BOOLEAN]
        NCOMP[INT]                  NSPECIES[INT]
LTURB
        USE_LES[BOOLEAN]
        SMAGORINSKY[FLOAT]
FLUI[id]
        DENSITY[FLOAT]              VISCOSITY[FLOAT]
SPEC[id]
        REF_QUANTITY[FLOAT]         DIFFUSIVITY[FLOAT]
GRAVITY
        USE_GRAVITY[BOOLEAN]
        GRAV_X[FLOAT]               GRAV_Y[FLOAT]       GRAV_Z[FLOAT]
TIMEM
        SimTime[FLOAT]              SimTimeMax[FLOAT]   CheckTime[FLOAT]
LWRESTART
        WRITE_SOLUTION[BOOLEAN]
        WRITE_INTERVAL[FLOAT]
LRFLOW
        READ_FLOW[BOOLEAN]
        READ_FLOW_TS[INT]
LRPSC
        READ_SPECIES[BOOLEAN]
LTDBC
        TIME_DEP_BNDC[BOOLEAN]
BND[id]
        TYPE[INT]
        VEL_X[FLOAT]                VEL_Y[FLOAT]            VEL_Z[FLOAT]
        PRESSURE[FLOAT]
        TEMPERATURE[FLOAT] ....     CONCENTRATION[nspecies][FLOAT]
#EOF#
```

Remarks: No blank lines allowed!

Keywords are in bold face. They may be left out in case the default values are used.

12.2 VARIABLE NAMES

MODEL

NDIM — Number of spatial dimensions.
Input value : 2 3
Default input : 3

NDIR_FLUID — Number of discrete lattice vectors
Input value : NDIM = 2 : 9
NDIM = 3 : 15, 19
Default : NDIR_FLUID = 19
Attention : The number of lattice vectors must correspond to the spatial dimensions.

NDIR_SPECIES — Number of discrete lattice vectors for the species transport
Input value : NDIM = 2 : 4, 9
NDIM = 3 : 6, 15, 19
Default : NDIM = 3 : 19
Attention : The number of lattice vectors must correspond to the spatial dimensions.

NCOMP

SOLVE_FLUID_FLOW — Specifies if the equation for the flow field is to be solve.
Input values : T F
Default : T

NCOMP — Number of (fluid) components
Input values : 1
Default : 1
Currently only a single fluid is considered.

NSPECIES — Number of species
Input values : 0 or positive integer
Default : 0
In principle the number of species is limited only by memory issues.

LTURB

USE_LES — Use a Smagorinsky type SGS – LES model
Input values : T F
Default : F

SMAGORINSKY — Smagorinsky constant
Input values : Positive float
Default : 0.18
Theoretical value is 0.18; usual values are between 0.10 and 0.30.
This value has to be change by an experienced user only!!!

FLUI[id]

DENSITY — Fluid density [kg/m³]
Input value : Positive float
Default : 1.0

VISCOSITY — Fluid viscosity [Pa s]
Input value : Positive float
Default : 0.02

SPEC[id]

REF_CONCENTRATION — Reference concentration
Input value : Positive float
Default : 1.0

DIFFUSIVITY — Species diffusion constant
Input value : Positive float
Default : 0.02

GRAVITY

USE_GRAVITY — Specifies if gravity is to be considered in the simulation
Input values : T F
Default : F

GRAV_X
GRAV_Y
GRAV_Z — Gravity vector
Input values : Float Float Float
Default : 0.0 0.0 0.0

TIMEM

SimTime — Simulation time [s]
Input value : Positive float
Default : 1.0

SimTimeMax — Maximum simulation time [s]
Input value : Positive float ≥ SimTime
Default : 1.0

CheckTime — Interval for convergence check [s]
Input value : Positive float
Default : 1.0

LWRESTART

WRITE_SOLUTION — Dump a restart file ICE_BASENAME<TSID>.cgns
Input value : Boolean
Default : F

WRITE_INTERVAL — Time interval for writing a restart file [s]
Input value : Positive float
Default : 10.0

LRFLOW

READ_FLOW	Read the flow field from ICE_BASENAME<READ_FLOW_TS>.cgns	
	Input value :	Boolean
	Default :	F
READ_FLOW_TS	Restart time step ID	
	Input value:	Positive Integer
	Default :	0

LRPSC

READ_SPECIES	Read the species field from ICE_BASENAME_SPEC[id].RES	
	Input value :	Boolean
	Default :	F

LTDBC

TIME_DEP_BNDC	Activate time dependent boundary conditions	
	Input value:	Boolean
	Default :	F

BND[id]

TYPE	Specifies the type of the boundary	
	Input value :	1 Velocity inlet
		2 Massflow inlet
		3 Inflow according to Ladd
		101 Outflow
		102 Pressure outlet
		401 Energy source region
		699 Potential fire region
		701 Fan
	Default :	501 Inactive

12.3 SPECIFICATION OF THE BOUNDARIES IN <ICE_BASENAME>.CAS

The file <ICE_BASENAME>.CAS contains a description of the initial state of the boundary condtions.

The structure is as follows for inlet boundaries (1,2, 3), outlet boundaries (101,102) and fans (701):

<BndID>
    <BndType>
    <Vel_X> <Vel_Y> <Vel_Z>
    <Pressure>
    <Temperature> <Concentration_2> ... <Concentration_N>

The structure for fire regions (401) and potential fires regions (601) is as follows:

<BndID>
<BndType>
<FireReleaseTime> <FireReleaseIncrease> <FireReleaseDecrease>
<IgnitionTemperature>
<FireLoad> <SmokeLoad>

The boundary type has to be specified according to the previous chapter.

All values have to be specified regardless the selected boundary. The first entry in the line of the species specification is always treated as temperature.

Valid ranges of boundary values:

Velocity : - 10.1 – 10.0 [m/s]

Pressure : - 5000.0 – 5000.0 [Pa]

A pressure of 0.0 means standard pressure. Note that in an incompressible flow the absolute pressure has no influence on the solution!

Temperature: 300 – 2500 [K]

The specified temperature must be in some meaningful relation to the specified reference temperature (e.g. Ref_Temp = 300 and BND_Temp = 700 is ok).

Concentration : 0 – 1 []

The concentration is dimensionless and as it does not influence the flow simulation it can be easily scaled in the post processing to the required value.

13. SPECIFICATION OF THE GEOMETRY FILE <ICE_BASENAME>.GEO

13.1 STRUCTURE

DIMENSIONS
    NI[INT] NJ[INT] NK[INT]
SYMMETRY
    SYMMETRY_BND[INT, DIMENSION(2*NDIMENSION)]
PERIODIC
    PERIODIC_BND[INT, DIMENSION(NDIMENSION)]
NBOUND
    NBOUNDARY[INT] NBOUNDARYCELLS[INT]
BNDIDX
    BOUNDARY_SID[INT, DIMENSION(NBOUNDARY)]
    BOUNDARY_EID[INT, DIMENSION(NBOUNDARY)]
BNDCOORD
    BOUNDARY_CELL[INT, DIMENSION(NBOUNDARYCELLS)]
    BOUNDARY_NBC[INT, DIMENSION(NBOUNDARYCELLS)]
WALLFLAGS
    CELL_ID(OBSTACLE(1)) [INT] WALLFLAG(OBSTACLE(1)) [INT]
    .
    .
    .
    CELL_ID(OBSTACLE(NOBS)) [INT] WALLFLAG(OBSTACLE(NOBS)) [INT]
NOBSTACLES
    NOBST[INT] NOBST_INNER[INT]
#EOF#

Remarks : No blank lines allowed!

Keywords are in bold face. Except of DIMENSION and #EOF# they may be left out if not required. The order of their appearance is insignificant.

13.2 VARIABLE NAMES

DIMENSION
    NI Number of cells in X-direction
    NJ Number of cells in Y-direction
    NK Number of cells in Z-direction

SYMMETRY

Symmetry boundaries may be specified for all enclosing surfaces of the computational domain.

SYMMETRY_BND(+X, -X, +Y, -Y, +Z, -Z)

An integer flag for each surface has to be specified. A number greater zero means that the corresponding surface is defined as symmetry boundary.

PERIODIC

Periodic boundary condition may be specified in each direction.

PERIODIC_BND(X-direction, Y-direction, Z-direction)

An integer flag for each direction has to be specified. A number greater zero means that the corresponding surfaces (WEST-EAST, NORTH-SOUTH, HIGH-LOW) are defined as symmetry boundaries.

BNDIDX

The start and end indices specifies which cells belong to a distinct boundary. Therefore it is important to keep a strict ordering of the cell co-ordinates.

BOUNDARY_SID	Boundary start index
BOUNDARY_EID	Boundary end index

BNDCOORD

As mentioned above the boundary cell co-ordinates MUST be in a specific order corresponding to the boundary start and end indices.

BOUNDARY_CELL	Cell index
BOUNDARY_NBC	Index of the neighbour cell

WALLFLAG

The cell indices are only given for occupied cells (wall cells)

CELL_ID	Cell index of the wall cell
WALLFLAG	Flag

NOBSTACLE

NOBST	Total number of obstacles
NOBST_INNER	Number of obstacles in the domain [(2,NI-1), (2,NJ-1), (2,NK-1)]

14. SPECIFICATION OF THE FILE <ICE_BASENAME>.INI

The file <ICE_BASENAME>.INI contains the initial values for all hydrodynamic variables.

14.1 FILE STRUCTURE

The structure is as follows:

PRES<id>
PRESSURE_<id>(1:NIJK)[FLOAT]
VELO<id>
VEL_X_<id>(1:NIJK)[FLOAT]
VEL_Y_<id>(1:NIJK)[FLOAT]
VEL_Z_<id>(1:NIJK)[FLOAT]
CONC<id>
CONCENTRATION_<id>(1:NIJK)[FLOAT]

Remark: No blank lines are allowed.

Keywords are in bold face and can appear in any order. Each component or species requires its own data set within the <ICE_BASENAME>.INI file.

14.2 VARIABLE NAMES

PRES<id>

The pressure has to be specified for each component in each cell as an array of reals.

VELO<id>

The fluid velocity for each component has to be specified in each cell. For each spatial direction one array of real is required.

CONC<id>

The concentration of each species has to be specified for each cell as an array of reals.

14.3 EXAMPLE <ICE_BASENAME>.INI FILE

Example 1 : Homogenous flow field

The following file is used to initialise a homogenous flow field of 1.0 [m/s] and uniform pressure. The concentration field of the first species is set equal “300.0” everywhere, whereas the concentration of species 2 is set equal “1.0” throughout the whole computational domain.

```
PRES1
        300000*0.0
VELO1
        300000*1.0
        300000*0.0
        300000*0.0
CONC1
        300000*300.0
CONC2
        300000*1.0
```

Example 2: Non-homogenous flow field

The following file initialises a flow field in a 2*2*2 domain with non-homogenous values for pressure, velocity and concentrations. Note that this example does not contain very meaningful values but it should only be considered as a demonstration.

```
PRES1
        1000.0 1200.0 1030.0 1100.0 1003.0 –1000.0 –2000.0 –1700.0
VELO1
        10.0 10.1 10.2 10.3 10.4 10.5 10.6  10.7
        0.1 0.1 0.3 0.2 0.2 0.0 0.3 0.3
        -0.03 -0.03 0.01 -0.04 -0.03 -0.02 -0.08 0.01
CONC1
        8*300.0
CONC2
        0.987 0.888 0.999 0.987 0.987 0.876 0.998 0.988
```

15. SPECIFICATION OF THE FILE <ICE_BASENAME>.TBC

The file <ICE_BASENAME>.TBC contains a description of the variation of the boundary conditions in time.

The structure is as follows:

<NoTDepBndCond>
<ActivationTime>
    <NoBnd>
        <BndID>
            <BndType>
            <Vel_x> <Vel_Y> <Vel_Z>
            <Pressure>
            <Temperature> <SmokeConcentration>
        <BndID>
            <BndType>
            <FireReleaseTime> <FireReleaseIncrease> <FireReleaseDecrease>
            <IgnitionTemperature>
            <FireLoad> <SmokeLoad>

<NoTDepBndCond>	Number of “branch points”
<ActivationTime>	Activation time of “branch point”
<NoBnd>	Number of boundaries to be changed at that specific branch point
<BndID>	ID of boundary to be changed
<Vel_X> ...	Boundary values

16. RESULT FILE FORMAT

Currently three output file formats are supported by MPICE.

16.1 OPEN DATA EXPLORER

Open DX is the open source version of IBM's Visualization Data Explorer [18].

Velocity, pressure and geometry information are contained in one data file. For each scalar quantity a single data file is written.

The file name convention is as follows:

<ICE_BASENAME>.DAT for the fluid variables

<ICE_BASENAME>_SPEC<ID>.DAT for species <ID>

For each data file a corresponding header file is created. The file name convention for the header files is as follows:

<ICE_BASENAME>.general for the fluid variable data file

<ICE_BASENAME>_SPEC<ID>.general for the data file of species <ID>

16.2 CGNS

CGNS (CFD General Notation System) [19] is an effort to standardise CFD data including grid, flow solution, boundary conditions, etc.

The computational domain is build of one single zone structured grid.

16.2.1 Grid file

At the beginning of the simulation a CGNS file containing grid information is written. The file name is as follows :

```
<ICE_BASENAME>_Grid.cgns
```

The grid information consists of four scalar fields:

`CoordinateX`	Cell coordinates in X-direction [RealDouble]
`CoordinateY`	Cell coordinates in Y-direction [RealDouble]
`CoordinateZ`	Cell coordinates in Z-direction [RealDouble]
`InterpolantsDonor`	Wallflag [RealDouble]

16.2.2 Result file

If a result file is required at time step <TimeStep> a CGNS file containing only the flow solution is written.

The file name is as follows:

```
<ICE_BASENAME><TimeStep>.cgns
```

The time step ID in the results file name is a six digit integer with leading zeros if required.

The flow solution consists of the following RealDouble scalar field with obvious meaning:

```
VelocityX
VelocityY
VelocityZ
Temperature
Pressure
```

16.3 USER DEFINED OUTPUT FILE FORMAT

It is possible to specify a user defined output file format within the subroutine ice_usr_out.f90, which has the following header specification :

```
SUBROUTINE ICE_USR_OUT(ICE_BASENAME, NIJK, NI, NJ, NK, VEL_X, VEL_Y, VEL_Z,
PRESSURE, CONCENTRATION, GRID, NCOMP, NSPEC)


!---Simulation base name
CHARACTER(LEN=*), INTENT(IN) :: ICE_BASENAME
!---Number of cells
INTEGER, INTENT(IN) :: NIJK
!---Number of cells in X-, Y- and Z-direction
INTEGER, INTENT(IN) :: NI, NJ, NK
!---Fluid velocity
REAL(HIGH), DIMENSION(:), INTENT(IN) :: VEL_X, VEL_Y, VEL_Z
!---Pressure
REAL(HIGH), DIMENSION(:), INTENT(IN) ::  PRESSURE
!---Species concentration
REAL(HIGH), DIMENSION(:), INTENT(IN) :: CONCENTRATION
!---Cell information flag
INTEGER, DIMENSION(:), INTENT(IN) :: GRID
!---Number of phases and species
INTEGER, INTENT(IN) :: NCOMP, NSPEC
```

To access the different species it is necessary to add a shift variable to the actual cell number. This shift parameter is calculated within ICE_USER_OUT by the following command:

```
!---GET CELL SHIFT
CELLSHIFT = NIJK*(ISPEC – 1)
```

where ISPEC is the actual species ID.

The actual concentration of a species in cell NCELL can be accessed by

```
CONCENTRATION(NCELL + CELLSHIFT)
```

17. PARALLEL SIMULATIONS

In what follows it is assumed that a suitable parallel environment is running on the target machines.

DEC Alpha ES 40

As the 4 processors are located on only one host it is not necessary to create a hostfile and the parallel calculation can be started by entering the following command:

```
dmpirun –np <NProc> mpice.exe
```

Where `<NProc>` is the number of processors used.

Linux cluster using LAM (CD)

Assume that 3 nodes each equipped with 2 processors are available. To include these nodes in a parallel simulation a hostfile (e.g. `lamhosts`) as to be created containing the following lines:

```
george cpu=2
werner cpu=2
otto cpu=2
```

where `george, werner` and `otto` are the alias names of the nodes.

Start your LAM session using the following commands (`-v` means verbose mode):

```
recon –v lamhosts
lamboot –v lamhosts
```

To start the parallel calculation it is then only necessary to execute the following command:

```
mpirun –np <NProc> mpice.exe
```

where again `<NProc>` corresponds to the number of processors (a maximum of 6 in the present example).

Linux cluster using MPICH (KTH - Lucidor)

To use MPICH on Lucidor you have to add the corresponding modules:

```
module add mpich easy
```

To run a parallel simulation on the interactive nodes it is only necessary to execute the following commands:

```
spattach –I –p<NNodes>
mpirun –np <NProc> -machinefile $SP_HOSTFILE ./mpice.exe
```

As every node on Lucidor contains 2 processors <NProc> can be twice <Nnodes> (e.g. for <Nnodes> = 6 a maximum of 12 Processors is possible).

18. DESCRIPTION OF A REFERENCE CASE

As a reference example a simplified model of a subway station provided by the Fire Department Dortmund is presented. The geometry is displayed in fig. 10.

It is assumed that only natural ventilation is existing within the station. As a consequence of this a pressure difference is applied between the two sides of the computational domain, which is represented by the doted pink box depicted in fig. 10. For modelling the open ceiling within the computational domain, also a constant pressure boundary condition is specified.

Within the simplified geometry a subway train consisting of two wagons is located. This train is sub-divided into three "virtual" parts, each of them representing a potential fire region, as indicated by the red circles numbered from 1 to 3. The fire region number 1 is ignited at the simulation start time. The other fire regions are activated as the solution progresses according to a temperature threshold level. If the computed surface temperature of a train section exceeds an assumed ignition temperature of 400 [K] this condition is met. Within activated fire regions additionally smoke is released.

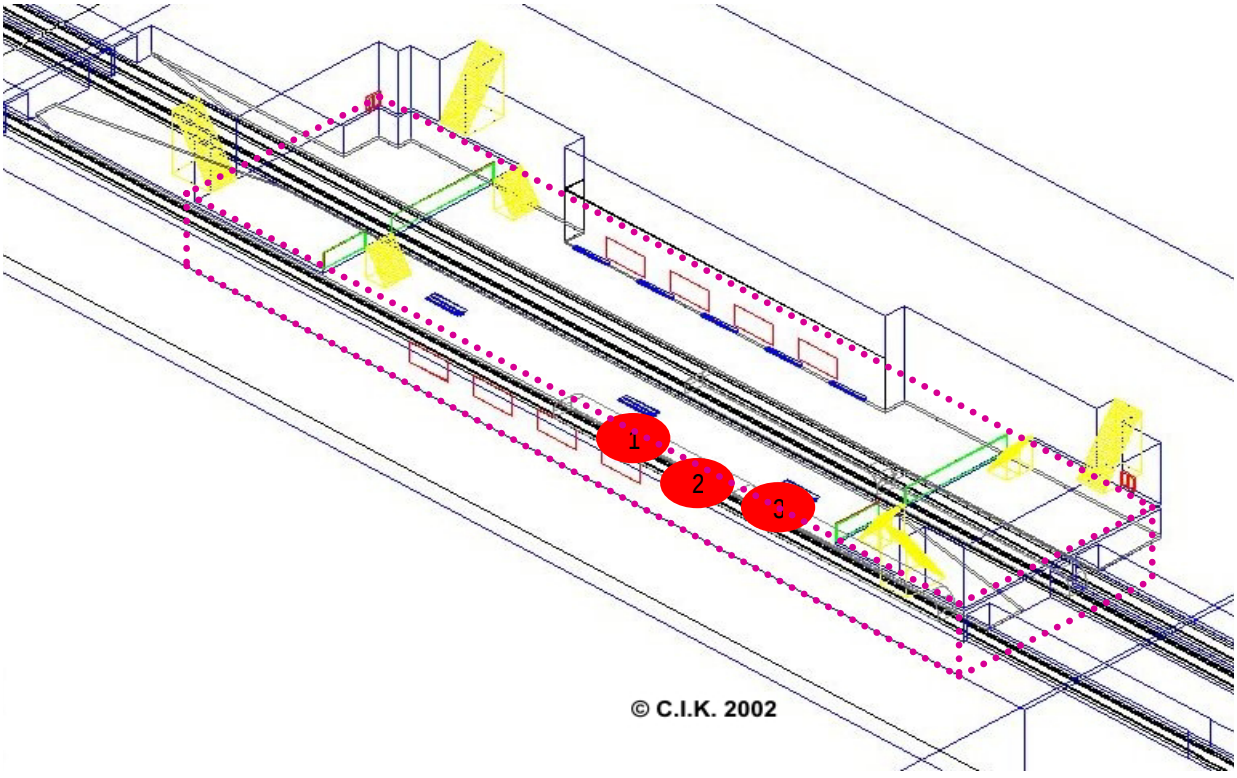
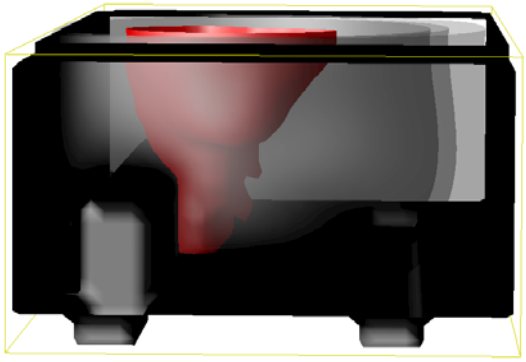


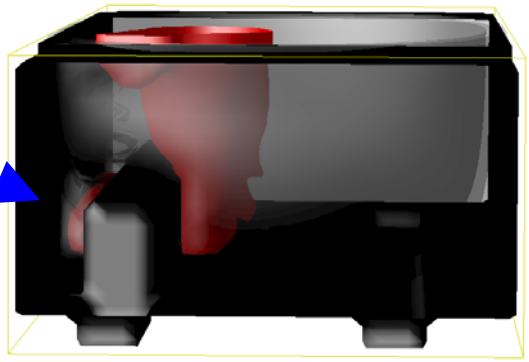
Figure 10: *Scenario "Subway Station Dortmund"*

In fig. 11 the temporal evolution of the hazard scenario is displayed in a perspective frontal view. Within this figure a temperature iso-surface of 450 [K] is shown in red and three different iso-surfaces of smoke concentration are displayed in grey colour. Furthermore in fig. 12 a perspective side view of the same results is depicted to help interpretation.



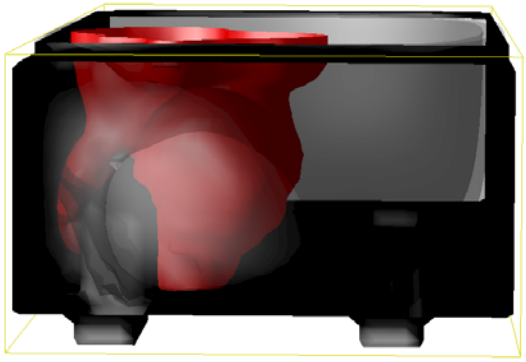
t = 100 [s]

Second train section starts burning.

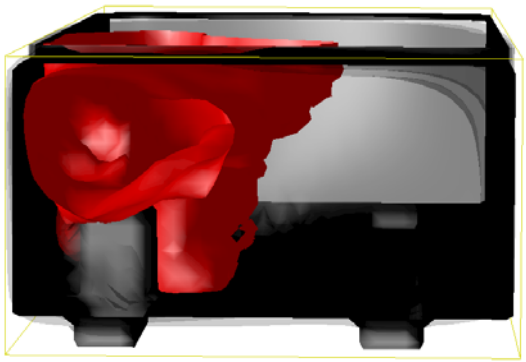


t = 300 [s]

Third train section is burning.



t = 500 [s]



t = 800 [s]

Figure 11: *Perspective frontal view of the temporal evolution of the hazard scenario for a temperature iso-surface of 450 [K] shown in red and three different smoke concentration levels displayed in grey.*

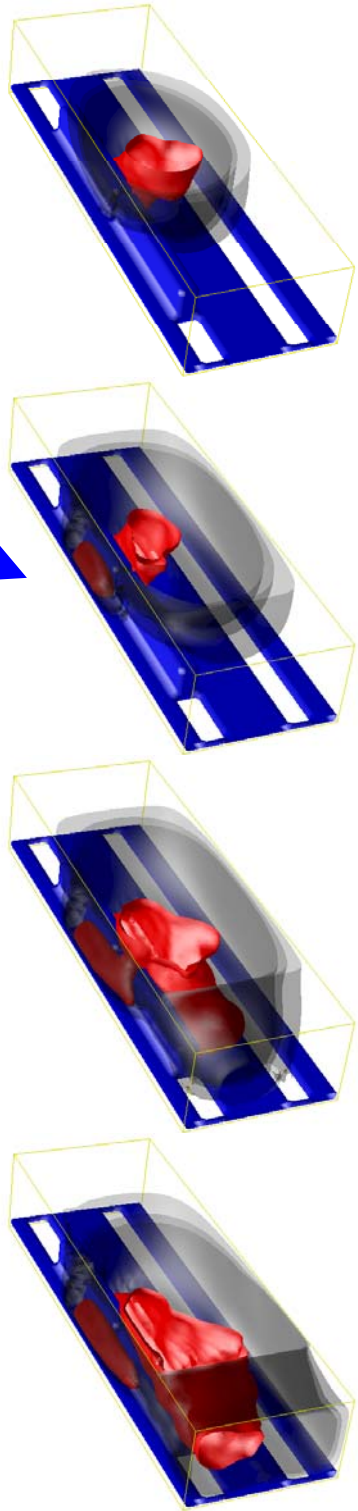


Figure 11: *Perspective side view of the temporal evolution of the hazard scenario for a temperature iso-surface of 450 [K] shown in red and three different smoke concentration levels displayed in grey.*

19. FILES USED FOR COMPUTING THE REFERENCE CASE

19.1 ICE_INPUT

The ICE_INPUT file contains only the ICE_BASENAME:

```
Subway
```

19.2 Subway.CAS

The "case"-file for the simulation of the subway tunnel fire is given below:

```
# 3 spatial dimensions
# d3q19 model for both the carrier and the species
      MODEL
      3 19 19
# Solve the fluid flow equation
# 1 fluid, 2 species (temperature and smoke concentration)
      NCOMP
      T
      1 2
# Turbulence parameter
# Use LES turbulence model with Cs = 0.18
      LTURB
      t
      0.18
# Parameter for Temperature
# Reference value and diffusivity
      SPEC1
      300.0 0.008
# Parameter for Smoke concentration
      SPEC2
      0.0   0.008
# Fluid properties
# Reference density and viscosity
      FLUI1
      1.0 2.5e-05
# Consider buoyancy
# Definition of the gravity vector
      GRAVITY
      T
      0.0000172 0.0 0.0000375
# Time management data
      TIMEM
      100.0 200.0 20.0
# Restart data
# Write restart file
# IF(Interval==0) only a restart file at
# the end of the simulation is written
      LWRESTART
      T
      5.0
# Read restart file from TSID(Flow)
      LRFLOW
      F
```

```
      50.0
# Read restart file (Species)
      F
#
# Boundary specifications
# Tunnel portals WEST and EAST
# Specifies a pressure of STP + 1000.0 [Pa]
# The inlet temperature is 300.0 [K]
      BND1
      102
      0.0 0.0 0.0
      1000.0
      300.0 0.0
      BND2
      102
      0.0 0.0 0.0
      1000.0
      300.0 0.0 0.0
#Roof
#Extraction specified using a pressure of STP – 1000.0 [Pa]
      BND3
      102
      0.0 0.0 0.0
      -1000.0
      300.0 0.0
# Initial fire region
# A fire with a heat release of 50 [MW] and
# a release time of 100 [s] is specified
      BND4
      401
      100.0 0.0 0.0
      400.0
      50.0 1.0
# Specification of potential fire regions emitting 15.0 [MW],
# which are acitivated if the surface temperature exceeds 400 [K]
# Potential fire region
      BND5
      699
      100.0 0.0 0.0
      400.0
      15.0 1.0
# Potential fire region
      BND6
      699
      100.0 0.0 0.0
      400.0
      15.0 1.0
# Source
      BND7
      699
      100.0 0.0 0.0
      400.0
      15.0 1.0
#EOF#
```

19.3 Subway.GEO

In what follows the geometry file Subway.GEO is shown. As it contains about 600 pages of text only a few exemplary lines of the geometry definitions are given.

```
DIMENSIONS
         110           37           22
SYMMETRY
           0            0            0            0            0            0
PERIODIC
           0            0            0
NBOUND
           7         4157
BNDIDX
           1          631         1261         3711         3795         3879         3978
         630         1260         3710         3794         3878         3977         4157
…
```

…Only the first three lines of the boundary coordinates are shown

```
…
BNDCOORD
       12212        12213        12214        12215        12216        12217
       12218        12219        12220        12221        12222        12223
       12224        12225        12226        12227        12228        12229
…
```

… The corresponding neighbour nodes to the boundary nodes above

```
…
       12249        12250        12251        12252        12253        12254
       12255        12256        12257        12258        12259        12260
       12261        12262        12263        12264        12265        12266
…
```

…Only the first three lines of the cell flags are shown

```
…
WALLFLAGS
           1 1
           2 1
           3 1
...
...
...
NOBSTACLES
       18870         8780
NACTIVECELLS
       70670        66820
#EOF#
```

19.4 Subway.INI

```
PRES1
300000*1.0
VELO1
300000*0.0
300000*0.0
300000*0.0
CONC1
```

```
300000*300.0
CONC2
300000*0.0
```

20. REFERENCES

[1] Wilcox, D. C.
Turbulence modelling for CFD
DCW Industries, 2000.

[2] McComb, W. D.
The Physics of Fluid Turbulence
Brady, J. M., Cullen, A. L., Jones, T. V., Van Bladel, J., Woods, L. C., Wroth, C. P. (eds.)
Oxford Engineering Science Series, Claredon Press (Oxford), 25, 1990.

[3] Pope, S. B.
Turbulent flows
Cambridge University Press, 2000.

[4] Orlandi, P.
Fluid Flow Phenomena
Fluid Mechanics and its Applications, Kluwer Academic Publishers, 2001.

[5] Karki, C., Patankar, S. V., Rosenbluth, E. M., Levy, S. S.
CFD Model for Jet Fan Ventilation Systems
Proc. 10th Int. Symposium on Aerodyn. and Ventilation of Vehicle Tunnels
Boston USA, November 1st – 3rd, 2000.

[6] Novozhilov, V.
Computational fluid dynamics modelling of compartment fires
Prog. Eng. Comb. Sci. 27 (2001), 611 – 666.

[7] Xue, H., Ho, J. C., Cheng, Y. M.
Comparison of Different Combustion Models in Enclosure Fire Simulation
Fire Safety Journal, 36 (2001), 37 – 54.

[8] Centre d'Etudes des Tunnels
Les Etudes Specifiques des Danger (ESD) pour les Tunnels de Reseau Routier
Ministére de l'Equipment, des Transports et du Logement, Direction des Routes, 2001.

[9] Luo, L.-S.
The Lattice Gas and Lattice Boltzmann Methods: Past, Present, and Future
Proceedings Int. Conf. Appl. Comp. Fluid Dyn., Beijing, China, October 17 – 20, 2000.

[10] Chen, S., Doolen, G. D.
Lattice Boltzmann Method for Fluid Flows
Annu. Rev. Fluid Mech. 30, 329 – 364 (1998).

[11] Succi, S.
The Lattice Boltzmann Equation for Fluid Dynamics and Beyond
Numerical Mathematics and Scientific Computation, Oxford University Press, 2001.

[12] Qian, Y. H., D'Humieres, D., Lallemand, P.
Lattice BGK models for the Navier-Stokes equations
Europhy. Lett, 17 (1992), 479 – 484.

[13] Mei, R., Shyy, W., Yu, D., Luo, L.-S.
Lattice Boltzmann Method for 3-D Flows with Curved Boundaries
J. Comp. Phys. 161 (2000), 680 – 699.

[14] D'Humieres, D., Ginzburg, I., Krafczyk, M., Lallemand, P., Luo, L.-S.
Multiple-Relaxation-Time Lattice Boltzmann Models in Three Dimensions
Phil. Trans. R. Soc. Lond. A 360 (2002), 437 – 451.

[15] Lallemand, P., Luo, L.-S.
Hybrid Finite-Difference Thermal Lattice Boltzmann Equation
submitted to Int. J. Mod. Phy. B (2002).

[16] Ladd, A. J. C.
Numerical Simulation of Particulate Suspensions via a Discretized Boltzmann Equation
J. Fluid Mech., (1994) 271 – 285.

[17] VirCinity, http://www.vircinity.com

[18] OpenDx, http://www.opendx.org

[19] CGNS, http://www.cgns.org

[20] Pucher, K.
Konzepte der Tunnel-Brandrauchentlüftung, Längs-, Quer- oder Vollquerlüftung?
Sicherheit im Tunnel – Austroschutz 99, Österreich (1999).

[21] http://www.mpi-forum.org