Computer Assisted Mechanics and Engineering Sciences, 8: 79–98, 2001.
Copyright © 2001 by Institute of Fundamental Technological Research, Polish Academy of Sciences

Object-oriented analysis of the numerical modelling of castings solidification

Norbert Sczygiol
Technical University of Częstochowa, Institute of Mechanics and Machine Design
ul. Dąbrowskiego 73, 42-200 Częstochowa, Poland

(Received September 29, 1999)

The paper concerns a method of implementation for the numerical modelling of the solidification process in which the finite element method was used. Modern techniques of software engineering were applied to reach the aim. The decomposition of the problem domain, for the needs of object-oriented analysis, was carried out. The relationships between parts of the analysed problem were discussed. At first, the object-oriented analysis was investigated in general for the wide range of problems solved by the finite element method, e.g. thermomechanic problems of castings, and then it was investigated in detail for the solidification process. The most important specialisation of classes for object implementation of the solidification model were also discussed. The enthalpy solidification formulations were used in the numerical modelling. The three models of solid phase growth, used for solidification modelling of two-component alloys, were described. The method for determining the dependence of enthalpy in relation to temperature and the formula for calculating the solid fraction were shown for each of the three models.

1. Introduction

The development of information systems, including systems for physical phenomena simulation, requires a growing number of factors to be taken into account. This causes an increase in the difficulties of implementing such systems.

Nowadays, the object-oriented methodology is the most commonly used technique to comprehend the complexity of the problem domain [7, 12, 16, 23]. It allows the gap between reality and the reasoning of data and processes to be greatly reduced, being an intuitive tool for conceptual modelling purposes. The basic objectivity mechanisms are:

- abstraction mechanism; this is convenient for building larger and larger software units and for operating them without penetrating their internal construction,
- composition and decomposition mechanism; this allows the details of the project to be enclosed into larger and larger units, and complex structures to be decomposed into their fragments, which allows them to be considered separately from each other and the total,
- reuse mechanism; this allows earlier created project components to be made use of.

The object-oriented methodology has its own notation, as with every other kind of methodology. A system of notation, which is in accordance with the *Unified Modelling Language (UML)* [21, 24], is the most often applied. It is a way of overcoming hitherto disorder, and it is considered to be the standard method to visualise object-oriented analysis results. The basic component of this notation are object diagrams. These include object classes (the attributes and methods are specified using classes), generalised relations, associated relations as well as the number of these relations. The object diagram arises at the stage of problem domain analysis and takes place independently of programming language, which ensures better problem domain mapping [5, 27].

2. Mathematical description of solidification

The description of phenomena occurring during the transition from the liquid to the solid state can be made on two levels: micro — which applies to single crystals, or macro — which applies to all solid phase growth. In the first case, we talk about crystallisation, in the second — about solidification. We can take into account some of the effects connected with the microscale in the solidification description, but without going into detail, e.g. the size of grains, the distance of secondary arms in dendrites, etc.

Solidification can proceed at a constant temperature or at a range of temperatures. If solidification proceeds at a constant temperature, the so-called "Stefan problem" or solidification problem with a zero-width phase-change interval should be addressed. A sharp separation between liquid and solid phases occurs in the Stefan problem. The two phases are in contact with each other forming a solidification surface (front). The solidification of metal alloys generally proceeds at certain temperature intervals, so-called "temperature solidification intervals". There is no sharp separation between liquid and solid phases in this case as both phases are separated from each other by a so-called "mushy zone" in which both liquid and solid phases appear at the same time. It is said that the solidification front is dispersed. The width of the mushy zone depends on the chemical constitution of the solidifying alloy and on the velocity of solidification (solid phase growth), and therefore on the conditions for carrying away heat.

Because of the complexity of the calculation, numerical methods are most often used to model the solidification process. Analytical ones are hardly ever used. The finite element method [6, 17, 18, 22] is the most commonly used numerical method, but the boundary element method [2, 3, 8, 9] and the finite difference method [14] can also be applied.

Solidification is stated by a quasi-linear heat conduction equation

$$\nabla \cdot (\lambda \nabla T) + \dot{q} = c\rho \frac{\partial T}{\partial t}, \tag{1}$$

containing the term of heat source $\dot{q}$, which describes the rate of latent heat evolution and is equal to

$$\dot{q} = \rho_s L \frac{\partial f_s}{\partial t}, \tag{2}$$

where λ is the thermal conductivity coefficient, c is the specific heat, ρ is the density (subscript s refers to the solid phase), L is the latent heat of solidification and f_s is the solid phase fraction. This equation, together with suitable initial and boundary conditions, forms the basis of the thermal description of solidification.

Taking into consideration the enthalpy, defined as follows [6]

$$H(T) = \int_{T_{\text{ref}}}^{T} c\rho \, \mathrm{d}T + \rho_s L(1 - f_s(T)), \tag{3}$$

where T_{ref} is the reference temperature, one can pass to the so-called enthalpy description of the solidification process. The prevailing ones are [6, 17, 22]:

1. the basic enthalpy formulation

$$\nabla \cdot (\lambda \nabla T) = \frac{\partial H}{\partial t}, \tag{4}$$

which is obtained by differentiating the enthalpy, given by Eq. (3), with respect to time

$$\frac{\partial H}{\partial t} = c\rho \frac{\partial T}{\partial t} - \rho_s L \frac{\partial f_s}{\partial t}, \tag{5}$$

2. the apparent heat capacity formulation

$$\nabla \cdot (\lambda \nabla T) = c^*(T) \frac{\partial T}{\partial t}, \tag{6}$$

which is obtained by differentiating Eq. (3) with respect to temperature

$$\frac{\mathrm{d}H}{\mathrm{d}T} = c\rho - \rho_s L \frac{\mathrm{d}f_s}{\mathrm{d}T} = c^*(T). \tag{7}$$

One can distinguish four types of boundary condition in the modelling of solidification — these are the same as those found in thermal conductivity modelling:

1. in the first type of boundary condition (Dirichlet condition), the temperature (T_{giv}) is given on the boundary Γ of Ω domain

$$\Gamma: \qquad T = T_{\mathrm{giv}}, \tag{8}$$

2. in the second type boundary condition (Neumann condition), the heat flux (q_{giv}) is given on the boundary Γ of Ω domain

$$\Gamma: \qquad q = q_{\mathrm{giv}}, \tag{9}$$

3. in the third type boundary condition (Newton condition), heat exchange with environment takes place on the boundary Γ of Ω domain

$$\Gamma: \qquad q = \alpha(T - T_{\mathrm{env}}), \tag{10}$$

where α is the heat exchange coefficient, T is temperature in the body on the boundary Γ, and T_{env} is environmental temperature,

4. in the fourth type boundary condition (continuity condition), on the boundary Γ, which separates Ω_1 and Ω_2 domains, where the heat flow takes place, two cases are possible:
   - ideal contact

$$\Gamma: \begin{cases} -\lambda_1 \dfrac{\partial T^{(1)}}{\partial n} = -\lambda_2 \dfrac{\partial T^{(2)}}{\partial n}, \\ T^{(1)} = T^{(2)}, \end{cases} \tag{11}$$

   where

$$\frac{\partial T^{(1)}}{\partial n} = T^{(1)}_{,i} n_i, \qquad \frac{\partial T^{(2)}}{\partial n} = T^{(2)}_{,i} n_i, \tag{12}$$

   and $\mathbf{n}$ is the normal vector to Γ boundary, while i indicates spatial directions,
   - shortage of ideal contact (contact through additional layer)

$$\Gamma: \begin{cases} -\lambda_1 \dfrac{\partial T^{(1)}}{\partial n} = -\lambda_2 \dfrac{\partial T^{(2)}}{\partial n} = \dfrac{\beta}{\delta}(T^{(1)} - T^{(2)}), \\ T^{(1)} \neq T^{(2)}, \end{cases} \tag{13}$$

   where β is the thermal conductivity coefficient of the material in the separating layer, and δ is the thickness of this layer.

The heat radiation, which performs a crucial role when the mould cavity is filled with liquid metal, is introduced to the mathematical description through the third type boundary condition. The α coefficient in Eq. (10) is in this case composed of two parts: convectional and radiant.

3. ENTHALPY FORMULATIONS OF SOLIDIFICATION BY THE FINITE ELEMENT METHOD

The equations describing particular types of solidification formulations are transformed into the finite element method equations. The weight residue method is used for this purpose. This method is based on finding an approximate solution to the differential equation which describes the phenomenon of solidification in the region Ω. This is done by multiplying this equation by a certain weight function and then by integrating over the region Ω. The Bubnov–Galerkin method is one of the most popular methods used for further investigation. This method uses the same weight and shape functions which describe the behaviour of the sought quantity in the finite element. The application of the Green–Gauss theorem enables the transformation of the discussed equation into an ordinary differential equation (containing the time derivative). The procedure described above is called semi-discretisation because it is applied only to the discretisation over the space. The equations are usually represented in a matrix notation.

The semi-discretisation of Eq. (4), which determines the basic enthalpy formulation of solidification, gives [6, 17]

$$\mathbf{M}\dot{\mathbf{H}} + \mathbf{K}(T)\mathbf{T} = \mathbf{b}(T), \tag{14}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the conductivity matrix, $\mathbf{H}$ is the enthalpy vector, $\mathbf{T}$ is the temperature vector and $\mathbf{b}$ is the right-hand side vector, whose values are calculated on the boundary conditions basis. The values of the matrix and vector coefficients, for single finite elements, are calculated according to the following equations

$$\mathbf{M}^e = \int_{\Omega^e} \mathbf{N}^T\mathbf{N}\,\mathrm{d}\Omega, \tag{15}$$

$$\mathbf{K}^e(T) = \int_{\Omega^e} \lambda(T)\nabla^T\mathbf{N}\cdot\nabla\mathbf{N}\,\mathrm{d}\Omega, \tag{16}$$

$$\mathbf{b}^e(T) = \int_{\Gamma^e} \mathbf{N}\lambda(T)\nabla\mathbf{T}\cdot\mathbf{n}\,\mathrm{d}\Gamma, \tag{17}$$

where $\mathbf{N}$ is the shape functions vector.

In the case of the apparent heat capacity formulation, Eq. (6), semi-discretisation results in the following equation [6, 17]

$$\mathbf{M}(T)\dot{\mathbf{T}} + \mathbf{K}(T) = \mathbf{b}(T), \tag{18}$$

in which

$$\mathbf{M}^e(T) = \int_{\Omega^e} c^*\mathbf{N}^T\mathbf{N}\,\mathrm{d}\Omega, \tag{19}$$

while matrix $\mathbf{K}$ and vector $\mathbf{b}$ are calculated according to Eqs. (16) and (17).

The ordinary differential equation (containing the time derivative) received as a result of semi-discretisation, is integrated over time using one of the linear multi-step schemes [25]. Because properties of the casting material depend on temperature, it is the best to apply an appropriate time integration scheme which eliminates the necessity of finding the actual values of the material properties for the calculated temperatures. The two-step Dupont II scheme can be applied for this purpose [6, 17]. However, the application of a two-step scheme requires the use of a one-step scheme as a first calculation. For this purpose the modified Euler-backward (EB) scheme [18, 25] can be applied.

The application of the modified Euler-backward scheme to the basic enthalpy formulation gives [18]

$$\mathbf{M}\mathbf{H}^{n+1} + \Delta t\,\mathbf{K}^n\mathbf{T}^{n+1} = \mathbf{M}\mathbf{H}^n + \Delta t\,\mathbf{b}^{n+1}, \tag{20}$$

while its application to the apparent heat capacity formulation brings in turn

$$(\mathbf{M}^n + \Delta t\,\mathbf{K}^n)\mathbf{T}^{n+1} = \mathbf{M}^n\mathbf{T}^n + \Delta t\,\mathbf{b}^{n+1}. \tag{21}$$

Value of the coefficient $a = 1/4$ was assumed for use in the Dupont II scheme. This assures the greatest accuracy and solution stability [6]. Moreover, it was established that matrix and vector coefficients are calculated for extrapolated temperature as follows

$$T = \frac{3}{2}T^{n+1} - \frac{1}{2}T^n, \tag{22}$$

and was denoted by superscript $(^0)$ in the following equations. As stated above, this allows long-time iterative calculations to be avoided in each time step. The application of Dupont II scheme to the basic enthalpy formulation gives [18]

$$\mathbf{MH}^{n+2} + \frac{3}{4}\Delta t\,\mathbf{K}^0\mathbf{T}^{n+2} = \mathbf{MH}^{n+1} - \frac{1}{4}\Delta t\,\mathbf{K}^0\mathbf{T}^n + \frac{3}{4}\Delta t\mathbf{b}^{n+2} + \frac{1}{4}\Delta t\mathbf{b}^n, \tag{23}$$

while for the apparent heat capacity formulation one can obtain

$$\left(\mathbf{M}^0 + \frac{3}{4}\Delta t\,\mathbf{K}^0\right)\mathbf{T}^{n+2} = \mathbf{M}^0\mathbf{T}^{n+1} - \frac{1}{4}\Delta t\,\mathbf{K}^0\mathbf{T}^n + \frac{3}{4}\Delta t\,\mathbf{b}^{n+2} + \frac{1}{4}\Delta t\,\mathbf{b}^n. \tag{24}$$

The function describing the relationship between enthalpy and temperature plays a basic role in the enthalpy solidification formulations from the pouring temperature to the temperature at which the casting is removed from the mould. It is possible to obtain analytical equations which describe the dependence of solid phase fraction on temperature for two-component metal alloys. These functions will be discussed for the succeeding models of solid phase growth in two-component alloys. Three models of solid phase growth for two-component alloys: *equilibrium, non-equilibrium* and *indirect* were taken into account in both formulations.

4. The solid phase growth models for two-component metal alloys

The behaviour of metal alloys in terms of the temperature and chemical constitution is presented with the help of phase diagrams. A fragment of a phase diagram for a two-component alloy with eutectic transformation is displayed in Fig. 1. The chemical constitution of the alloy is stated on the x-axis and temperature on the y-axis. The melt temperature of the basic component equals T_M, and eutectic temperature is T_E. The beginning of alloy solidification is marked as T_L. The end of solidification is marked as T_S. This temperature is not shown in Fig. 1 because it depends on the solidification model. The temperatures of the beginning and the end of solidification are called the liquidus and solidus temperatures, respectively. The pressure can also have an influence on the shape of the phase diagram, but this kind of influence is omitted in all forms of casting, except for pressure die casting.

Three methods for numerical modelling the solid phase growth can be distinguished when considering two-component alloys with eutectic transformation. In the case of *non-equilibrium solidification model*, given by Scheil equation, the eutectic temperature is always reached by the solidifying alloy (line 2). This means that a certain last portion of the metal solidifies at a constant temperature. For the *equilibrium solidification model* the temperature at which solidification ends depends on the chemical constitution (line 1). The model is described by the lever rule [10]. The temperature at which solidification ends equals the eutectic temperature only for alloys in which the solute concentration is bigger than its maximal solubility in solid phase. The solidification run depends on the diffusion path length of solute and so on the grain size in the solidifying microstructure, in the indirect solidification model (line 3), the limits of which are determined by lines illustrating equilibrium and non-equilibrium solidification models. For this reason it is called the *between solidification model*. The average grains size in the casting microstructure depends on cooling velocity [19].

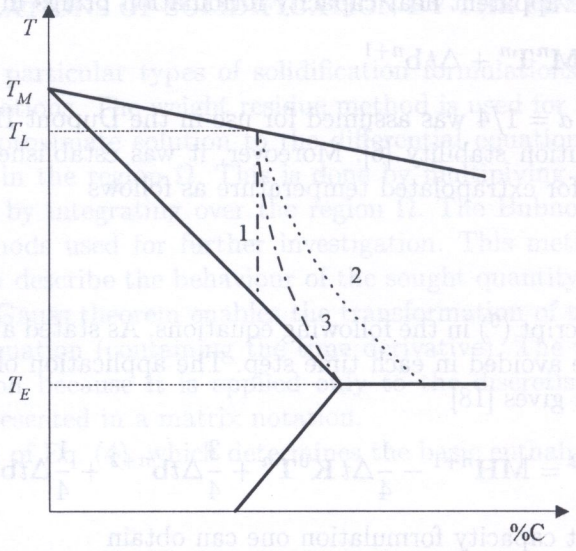

Fig. 1. The solid phase growth models in the two-component alloys. 1 – equilibrium solidification model (*lever*), 2 – non-equilibrium solidification model (*Scheil*), 3 – indirect solidification model (*between*)

4.1. The solid phase growth according to the equilibrium solidification model (*lever*)

The solid fraction equation can be obtained from the solution of a diffusion equation for the solute, after introducing some simplifications [10, 11, 15]

$$f_s = \frac{1}{1-k}\frac{T_L - T}{T_M - T}, \tag{25}$$

where k is the solute partition coefficient, defined as

$$k = \frac{C_s^*}{C_l^*}, \tag{26}$$

where C_s^* is the solute concentration on the solidification surface in the solid phase, and C_l^* is the solute concentration on the solidification surface in the liquid phase. The solute partition coefficient is strictly connected with liquidus (T_L) and solidus (T_S) temperatures in the equilibrium solidification model. Two cases are possible there, depending on the initial solute concentration:

- solidification ends without reaching the eutectic temperature T_E; whole solidification process takes place in some temperature interval,
- solidification ends after reaching the eutectic temperature; the solidification process takes place at a constant temperature from the moment eutectic temperature is reached.

4.1.1. The apparent heat capacity formulation

Substituting the final solidification temperature into Eq. (25) shows if the solidification will take place in the temperature interval, or if the constant solidification temperature will be reached

$$f_s = \begin{cases} = 1, & \text{solidification in temperature interval,} \\ < 1, & \text{the final solidification stage takes place at a constant tempetature.} \end{cases} \tag{27}$$

The solid phase fraction is calculated according to Eq. (25), if the solidification proceeds without reaching the eutectic temperature. The diagram of enthalpy, as a function of temperature for this

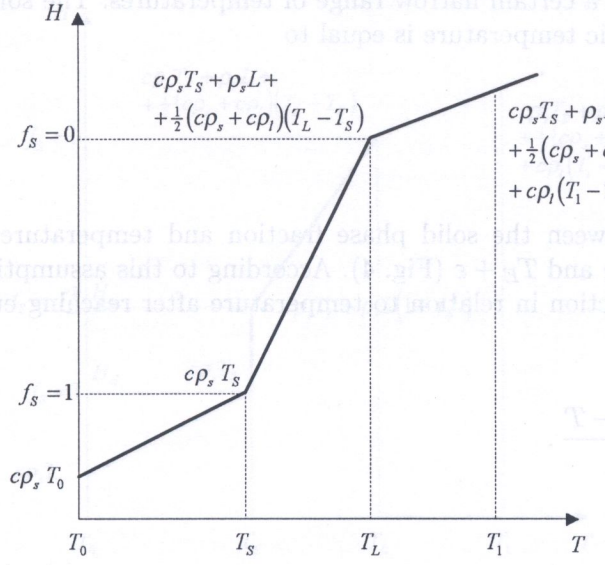

Fig. 2. The relationship of two-component alloy enthalpy to temperature (the range of solute full solubility)

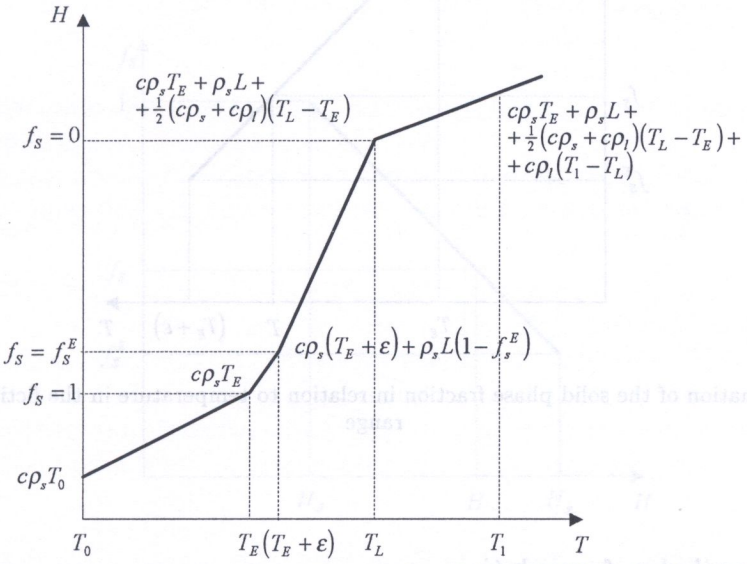

Fig. 3. The relationship of two-component alloy enthalpy to temperature (alloy with assumed fictitious solidification range)

case, is shown in Fig. 2[1]. The beginning of the solidification ($f_s = 0$) and the end of solidification ($f_s = 1$) are additionally shown on the y-axis.

If the solidification proceeds reaching the constant solidification temperature (eutectic temperature), the relationship of enthalpy to temperature is modified by assuming that a certain fictitious range of temperatures exists. Its width is ε (Fig. 3). Obtaining identical temperatures in every node of one finite element is avoided in this way, which means indeterminacy during calculating effective heat capacity c^* is avoided (compare Eq. (7)). In fact the solidification at a constant temperature

[1]To construct the enthalpy to temperature diagrams it was assumed that the heat capacity in the solidification temperature interval is an average of the solid and liquid phase heat capacities. Between the beginning and end of solidification temperatures the enthalpy was interpolated linearly.

is modelled by solidification in a certain narrow range of temperatures. The solid phase fraction at the moment of reaching eutectic temperature is equal to

$$f_s^E = \frac{1}{1-k}\frac{T_L - T_E}{T_M - T_E}. \tag{28}$$

The linear dependence between the solid phase fraction and temperature is assumed in the temperature range between T_E and $T_E + \varepsilon$ (Fig. 4). According to this assumption the equation for calculating the solid phase fraction in relation to temperature after reaching eutectic temperature can be educed. Its form is

$$f_s = 1 - (1 - f_s^E)\frac{(T_E + \varepsilon) - T}{\varepsilon}. \tag{29}$$

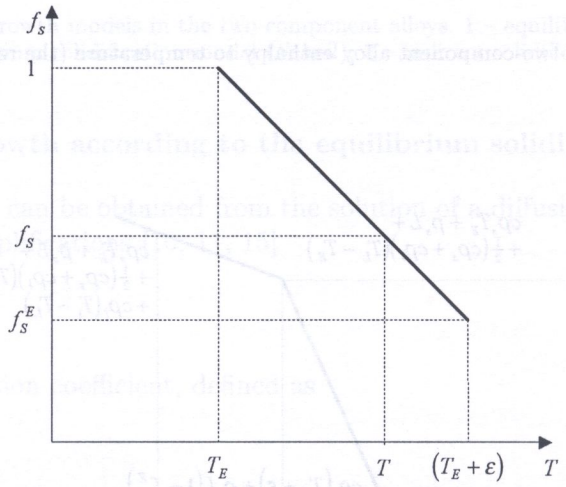

Fig. 4. Approximation of the solid phase fraction in relation to temperature in the fictitious solidification range

4.1.2. The basic enthalpy formulation

The relationship of enthalpy to temperature is shown in Fig. 2. This takes place for solidification in the temperature interval, and was tested by Eq. (27). If solidification finishes at a constant temperature, then the relationship of enthalpy to temperature is subject to modification as shown in Fig. 5. The solid phase fraction is calculated according to Eq. (25) until eutectic temperature is reached. The linear relationship between the solid phase fraction and enthalpy (Fig. 6) is established for constant solidification temperature. According to this assumption, the equation used for calculating the solid phase fraction can be educed, but this time in relation to enthalpy, after reaching the eutectic temperature. The equation is as follows

$$f_s = 1 - (1 - f_s^E)\frac{H - H_d}{H_g - H_d}. \tag{30}$$

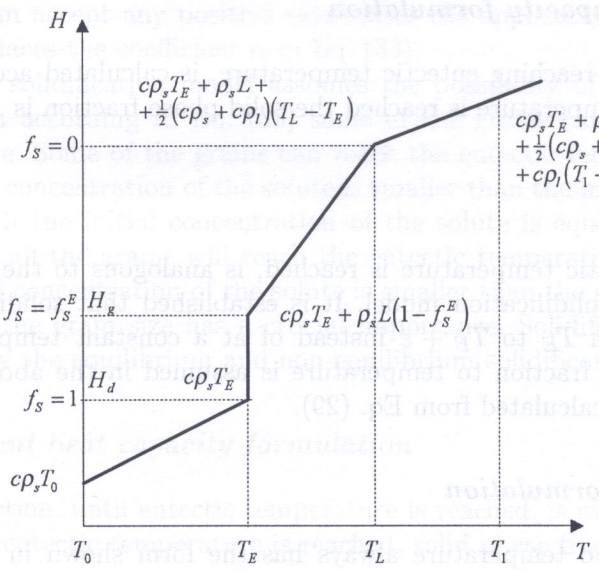

Fig. 5. The relationship of two-component alloy enthalpy to temperature (alloy with eutectic transformation)

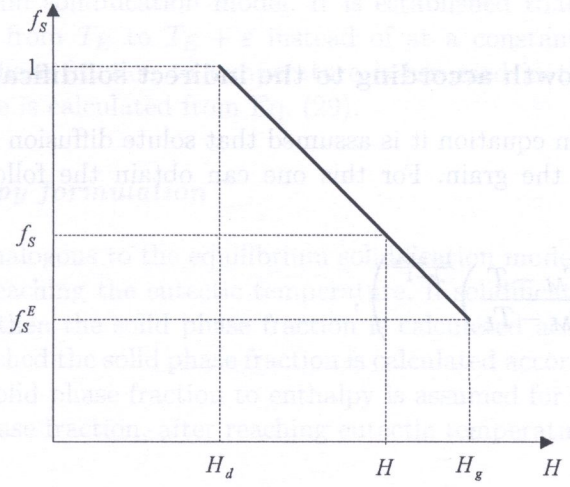

Fig. 6. Approximation of solid phase fraction as the function of enthalpy at the eutectic temperature

4.2. The solid phase growth according to the non-equilibrium solidification model (*Scheil*)

Again, one can obtain the following solid phase fraction equation from the solution of a diffusion equation for solute, but after introducing different simplifications [10, 15]

$$f_s = 1 - \left(\frac{T_M - T}{T_M - T_L}\right)^{\frac{1}{k-1}}. \tag{31}$$

The equation describing solute distribution, from which the above equation is educed, is called the Scheil equation. This equation is characteristic because it always gives the eutectic composition. This means that part of the solid phase always solidifies at a constant (eutectic) temperature.

4.2.1. The apparent heat capacity formulation

The solid phase fraction, until reaching eutectic temperature, is calculated according to Eq. (31). At the moment the eutectic temperature is reached the solid phase fraction is

$$f_s^E = 1 - \left(\frac{T_M - T_E}{T_M - T_L}\right)^{\frac{1}{k-1}}. \tag{32}$$

The procedure, after the eutectic temperature is reached, is analogous to the solid phase growth according to the equilibrium solidification model. It is established that solidification takes place in the temperature range from T_E to $T_E + \varepsilon$ instead of at a constant temperature. The linear relationship of the solid phase fraction to temperature is assumed in the above range. The solid phase fraction in this range is calculated from Eq. (29).

4.2.2. The basic enthalpy formulation

The relationship of enthalpy to temperature always has the form shown in Fig. 5 because the solidification always finishes at a constant temperature. Until the moment the eutectic temperature is reached, the solid phase fraction is calculated according to Eq. (31). At the moment the eutectic temperature is reached, the solid phase fraction is given by Eq. (32). The linear relationship is again established for the constant solidification temperature. The solid phase fraction, after the eutectic temperature is reached, is given by Eq. (30).

4.3. The solid phase growth according to the indirect solidification model (*between*)

For the solution of diffusion equation it is assumed that solute diffusion in the solid phase depends on the size and shape of the grain. For this one can obtain the following solid phase fraction equation [17, 19]

$$f_s = \frac{1}{1 - nk\alpha}\left(1 - \left(\frac{T_M - T}{T_M - T_L}\right)^{\frac{1-nk\alpha}{k-1}}\right), \tag{33}$$

where

$$\alpha = \frac{D_s}{r_g^2} t_f, \tag{34}$$

and D_s is the solute diffusion coefficient in the solid phase, t_f is the so-called local solidification time, r_g is characteristic grain size and n is a coefficient engaging the grain shape

$$n = \begin{cases} 2, & \text{for plane grain,} \\ 4, & \text{for cylindrical grain,} \\ 6, & \text{for spherical grain.} \end{cases} \tag{35}$$

The $D_s t_f$ product can be treated as a material parameter. The application of Eq. (33) gives physically unreal results for a wide range of coefficient α values above a certain value depending on grain shape. This means that the solid phase fraction is equal to 1 for the temperature higher than solidus temperature. One can avoid this inconvenience by introducing appropriate correction for the α value. In this paper correction was introduced only for the plane grains, this means for $n = 2$. It equals [4]

$$\Omega(\alpha) = \alpha\left(1 - \exp\left(-\frac{1}{\alpha}\right)\right) - \frac{1}{2}\exp\left(-\frac{1}{2\alpha}\right). \tag{36}$$

The coefficient α can accept any positive value after the application of the above correction. The correction $\Omega(\alpha)$ replaces the coefficient α in Eq. (33).

If the numerical solidification model assumes the possibility of different grain sizes occurring in the casting, then according to Eq. (33) some of the grains can solidify without reaching the eutectic temperature. Some of the grains can reach the eutectic temperature during solidification, assuming the initial concentration of the solute is smaller than the maximum solubility of the solute in the solid phase. If the initial concentration of the solute is equal to or bigger than the above concentration, then all the grains will reach the eutectic temperature during solidification. In the first case (the initial concentration of the solute is smaller than the maximal solubility of the solute in the solid phase) the grain size has a crucial importance. Solidification can take place between limits determined by the equilibrium and non-equilibrium solidification models.

4.3.1. The apparent heat capacity formulation

The solid phase fraction, until eutectic temperature is reached, is calculated according to Eq. (33). At the moment the eutectic temperature is reached, solid phase fraction is equal to

$$f_s^E = \frac{1}{1-nk\alpha}\left(1-\left(\frac{T_M-T_E}{T_M-T_L}\right)^{\frac{1-nk\alpha}{k-1}}\right). \tag{37}$$

The procedure, after the eutectic temperature is reached, is analogous to the solid phase growth according to the equilibrium solidification model. It is established that solidification takes place in the temperature range from T_E to $T_E+\varepsilon$ instead of at a constant temperature. The linear relationship of the solid phase fraction to temperature is assumed in the above range. The solid phase fraction in this range is calculated from Eq. (29).

4.3.2. The basic enthalpy formulation

Subsequent procedure is analogous to the equilibrium solidification model because the solidification can end with or without reaching the eutectic temperature. If solidification ends without reaching the eutectic temperature, then the solid phase fraction is calculated according to Eq. (33). When eutectic temperature is reached the solid phase fraction is calculated according to Eq. (37). Again the linear relationship of the solid phase fraction to enthalpy is assumed for a constant temperature of solidification. The solid phase fraction, after reaching eutectic temperature, is calculated according to Eq. (30).

5. Object-oriented analysis of the finite element method

After analysing the system for physical phenomena modelling using the Finite Element Method, a certain abstract part was isolated and grouped in the *FEMCore* software package. The basic classes of FEM, apart from the problem tackling type, are collected in this package. The classes, describing real modelling problems, are created using an inheritance mechanism (Fig. 7).

5.1. *FEMCore* package

The basic FEM components (Fig. 8): a node — *Node* class, a finite element — *Element* class, boundary conditions — *BoundaryCondition* class, material properties for a group of elements — *Region* class and class solving system of equations — *SystemOfEquationsSolver* are included in *FEMCore* package composition. They are abstract classes, this means that the number of node coordinates, the number of nodes in an element and interpolation functions, type of boundary

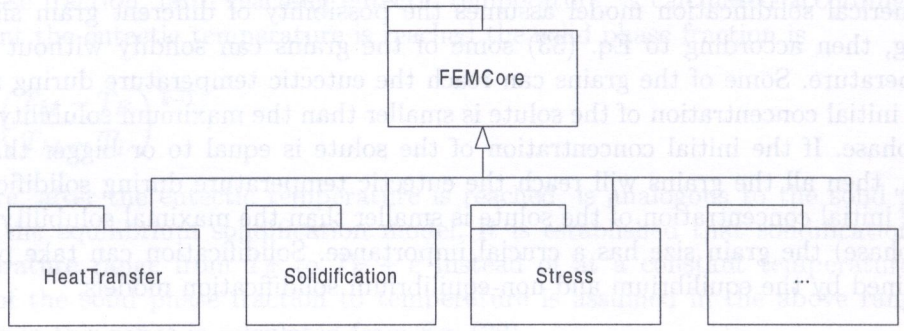

Fig. 7. The division of system into packages

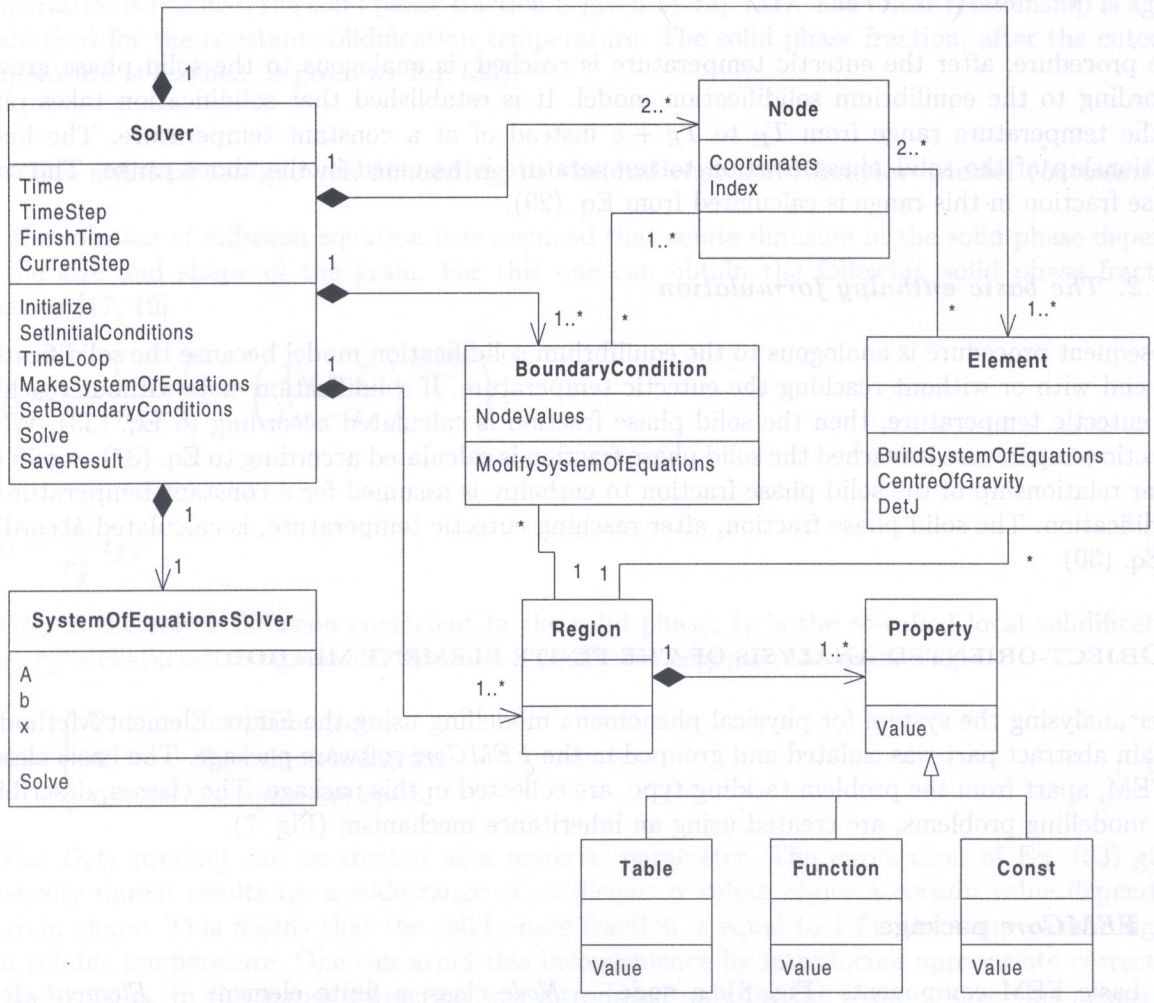

Fig. 8. *FEMCore* package content

conditions, material properties and methods of solving system of equations are not given in them. All this information is detailed at the real problem modelling stage.

The *Solver* class, through aggregation relation, is the consolidation class. It stores all object classes of nodes, elements, boundary conditions and material properties occurring in the task. *Node*, *Element*, *Region* and *BoundaryCondition* classes describe in turn: a single node, a single finite element, properties for a single domain and introduced boundary conditions for a single element or a node. The objects collections of those classes are created as the task is solved. Their number is readjusted to the task size, this means to the number of nodes, elements, boundaries and domains. The attributes and methods associated with the structure and solution of the task are contained in the classes. The initial-boundary task is assumed and hence, the solver class components are connected with a time loop.

The *Element* class was designed in such a way that it creates a system of equations for only one finite element. The construction of a global system of equations for the whole task relies on calling *BuildSystemOfEquations* method for all the objects in the finite element collection. The identical treatment of different geometry elements, different interpolation functions and elements from domains of different material properties is possible in this way. This identical treatment of the collection of elements has also been used for the *BoundaryCondition* class, where boundary conditions are introduced by using the same *ModifySystemOfEquations* method, independently of the type of boundary condition.

The collection of objects describing the material properties goes into the *Region* class. Those objects have inherited attributes and methods from the *Property* class and can be described by a constant value or by a one parameter function (e.g. temperature). The situation in which the function describing material property values is unknown or very complicated was also foreseen. It can be represented in a discrete form and the values between discrete points are interpolated linearly. The access to particular properties is achieved by using appropriate methods inherited from *Region*.

The problem of solving a system of equations is left open in *FEMCore* package. The *SystemOfEquationtsSolver* class is the common interface for direct and iterative methods. This makes it possible to choose a suitable solving method and a suitable preconditioner for the system of equations for implementing task. Moreover, it is possible to choose a method to store the system of equations coefficients matrix: as a full matrix (square), a band matrix (rectangle) or a sparse matrix (only non-zero coefficients and their matrix subscripts are stored) [1, 13].

5.2. The operation of *FEMCore* package

The state/activity diagram showing the methods of classes included in the package should be used to illustrate the operation of the package. Many figures need to be inserted with such diagrams, which could make reviewing difficult. Despite this, it was decided to present a short description of an operation which illustrates package functionality.

First the *Solver* class object is created in the computer memory and the *Initialize* method is started. The aim of this method is to read in the model parameters, e.g. time-step, steps number etc. Afterwards, the method reads in information about regions which produce the Region class objects, aggregating objects of particular material properties. The next operation stage of the *Initialize* method is to read in the finite elements mesh. The nodes whose objects are initialised by coordinate values (*Coordinates*) and by the node number (*Index*) are read in first. The node number determines the position of the corresponding equation (or equations) in the global system of equations. Next, *Element* class objects are created in the computer memory and connected with the region objects and node objects on which they are described. Then the boundary conditions are read in creating *BoundaryCondition* class objects and joining them with suitable region and node objects. The initial values are read in for non-stationary tasks. These are introduced into the vector solution in the earlier formed *SystemOfEquationsSolver* class object.

After all the information is read in, the *Solver* class object starts to solve the problem by calling *TimeLoop* method. The first step is to set to zero matrix A and vector b of the *SystemOfEquationsSolver* object. Then, the *MakeSystemOfEquations* method is called. Its function is to start the *BuildSystemOfEquations* method in all the *Element* class objects. This method, using material information obtained from the attached *Region* class objects and *Node* class objects, builds the global system of equations. As a next step, the boundary conditions of the task are introduced into the *TimeLoop* method. This is done by the *SetBoundaryConditions* method which starts the *ModifySystemOfEquations* method in all the *BoundaryCondition* class objects. The coefficients' values in the global system of equations are suitably changed. These changes are based on the material properties information and on the place of their introduction (node number). The system of equations formed in this way is solved by the *SystemOfEquationsSolver* class object. The obtained results are recorded by the *SaveResult* method. These steps, from setting to zero of the system of equations coefficients to the recording of results, are repeated until the *Time* attribute reaches the *FinishTime* value. Then the *Solver* object, during the ending of the task, releases the memory assigned to all the objects of the task.

6. Object-oriented analysis of solidification models

The specialisation of *FEMCore* package classes is the next stage for the considered problem. The specialisation of *FEMCore* package abstract classes for the modelling of solidification are shown in the following diagrams. The obtained class hierarchies are closed in the *Solidification* package (Fig. 7).

Modelling in the problem domain was restricted to two solidification formulations: the apparent heat capacity formulation and the basic enthalpy formulation. The abbreviated notations of those formulations: *AHC* — for apparent heat capacity formulation and *BEF* — for basic enthalpy formulation, were used during analysis. Two regions, which were the subject of modelling, are distinguished: *Cast* — representing the solidifying casting and *Mould* — representing the casting mould (possibly containing casting cores). It was also assumed that the distribution of cooling velocity, solidification kinetics and the temperature field will be results of the calculation.

6.1. *Solver* class specialisation

The *SolidificationSolver* class was worked out from the *Solver* class (Fig. 9). It inherits all the *Solver* class properties expanding to the requirements of the solidification simulation system. The system should model the solidification process according to one or other of the two formulations together with three models of solid phase growth. It gives six different procedures to describe the solidification process. This requires the *Model* attribute in *SolidificationSolver* class to be loaded. The *Model* attribute identifies the classes of objects created. The objects of classes describe finite elements, boundary conditions, material properties and method for introducing initial conditions.

The inheriting *TimeLoop* method is completed by calling the service of velocity cooling calculation which was realised by the *CalculateVelocity* method. This method calls the *CoolingVelocityInNodes* method (Fig. 10) in all the *Element* class objects (specialised for solidification modelling needs). The *CoolingVelocityInNodes* method determines the solidification velocity value in the finite element nodes.

The unknowns in the system of equations for AHC are temperatures, while for BEF — enthalpies. The temperature field is a common initial condition for both formulations. The *SetInitialConditions* method recalculates the temperature field to enthalpy field for BEF.

The action of the *SaveResult* method was developed by calling: *SaveVelocity* — to save cooling field velocity, *SaveTemperature* — to save temperature field (for BEF recalculated from enthalpy), and *SaveSolidFraction* — to save solidification kinetics.

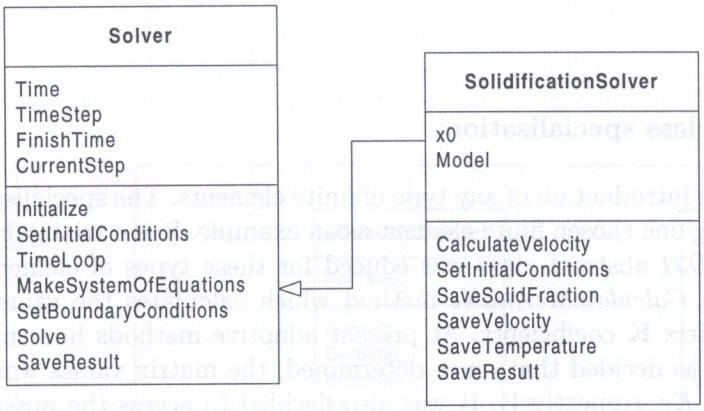

Fig. 9. *Solver* class specialisation

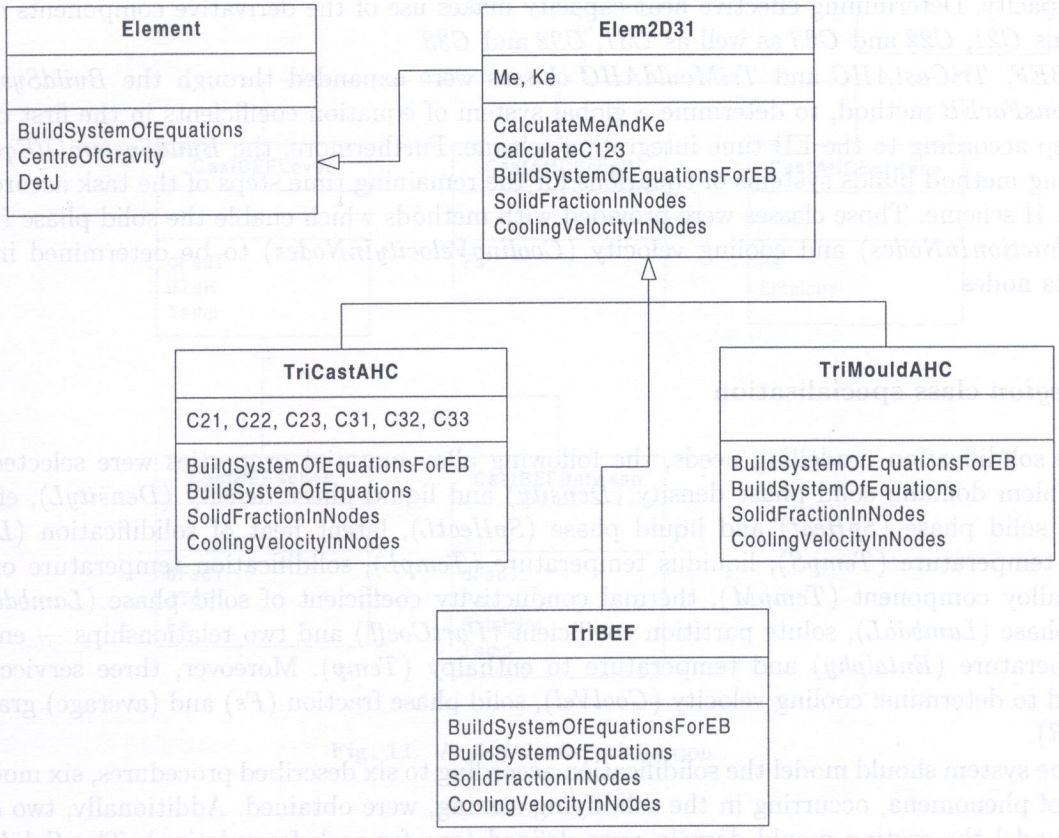

Fig. 10. *Element* class specialisation

The three level time integration, according to Dupont II scheme, needs an extra vector ***x0*** to be introduced to store "earlier" calculation results. Substituting the resulting values of solution vector $\boldsymbol{x}$ from the *SystemOfEquationSolver* class into ***x0*** vector was realised by a suitable widening of the *TimeLoop* method.

6.2. The *Element* class specialisation

The system allows the introduction of any type of finite elements. The specialisation of *Element* class will be discussed using one chosen finite element as an example. It is a triangular element with linear interpolation. *Elem2D31* abstract class was educed for these types of elements from the *Element* class. It contains the *CalculateMeAndKe* method which calculates the values of mass matrix $\mathbf{M}$ and conductivity matrix $\mathbf{K}$ coefficients. At present adaptive methods have not been implemented in the system, so it was decided that, once determined, the matrix values would be stored as class attributes in *Me* and *Ke*, respectively. It was also decided to access the possibility of determining the constant components of the nodal function derivatives with respect to suitable directions. This is realised by the *CalculateC123* method.

The class hierarchy for triangular finite element modelling was educed from the *Elem2D31* class for both formulations. *TriBEF* class models the triangular element for the basic enthalpy formulation independently of this element's attachment to the casting domain or the casting mould. All details, connected with this were moved to the classes modelling material property domains. The finite elements were discriminated between casting (*TriCastAHC*) and mould (*TriMouldAHC*) modelling for apparent heat capacity formulation because of the specificity of determining effective heat capacity. Determining effective heat capacity makes use of the derivative components of node functions *C21*, *C22* and *C23* as well as *C31*, *C32* and *C33*.

TriBEF, *TriCastAHC* and *TriMouldAHC* classes were expanded through the *BuildSystemOfEquationsForEB* method, to determine a global system of equation coefficients in the first calculation step according to the EB time integration scheme. Furthermore, the *BuildSystemOfEquations* inheriting method builds systems of equations for the remaining time steps of the task according to Dupont II scheme. Those classes were provided with methods which enable the solid phase fraction (*SolidFractionInNodes*) and cooling velocity (*CoolingVelocityInNodes*) to be determined in finite elements nodes.

6.3. *Region* class specialisation

For the solidification modelling needs, the following alloy material properties were selected from the problem domain: solid phase density (*Density*) and liquid phase density (*DensityL*), effective heat of solid phase (*SpHeat*) and liquid phase (*SpHeatL*), latent heat of solidification (*LHeat*), solidus temperature (*TempS*), liquidus temperature (*TempL*), solidification temperature of basic (pure) alloy component (*TempM*), thermal conductivity coefficient of solid phase (*Lambda*) and liquid phase (*LambdaL*), solute partition coefficient (*PartCoeff*) and two relationships — enthalpy to temperature (*Entalphy*) and temperature to enthalpy (*Temp*). Moreover, three services were specified to determine: cooling velocity (*CoolVel*), solid phase fraction (*Fs*) and (average) grain size (*GrainR*).

As the system should model the solidification according to six described procedures, six modelling classes of phenomena, occurring in the solidifying casting, were obtained. Additionally, two classes which model the casting mould domain were defined (one for each formulation). The *SolidRegion* abstract class was worked out from the *Region* class to make for easier management of the class hierarchy describing material properties (Fig. 11). The methods, stored there, make a common interface for all classes modelling particular domains. The inheritance from this class was taken in two paths: for solidifying casting, and for casting mould and casting cores.

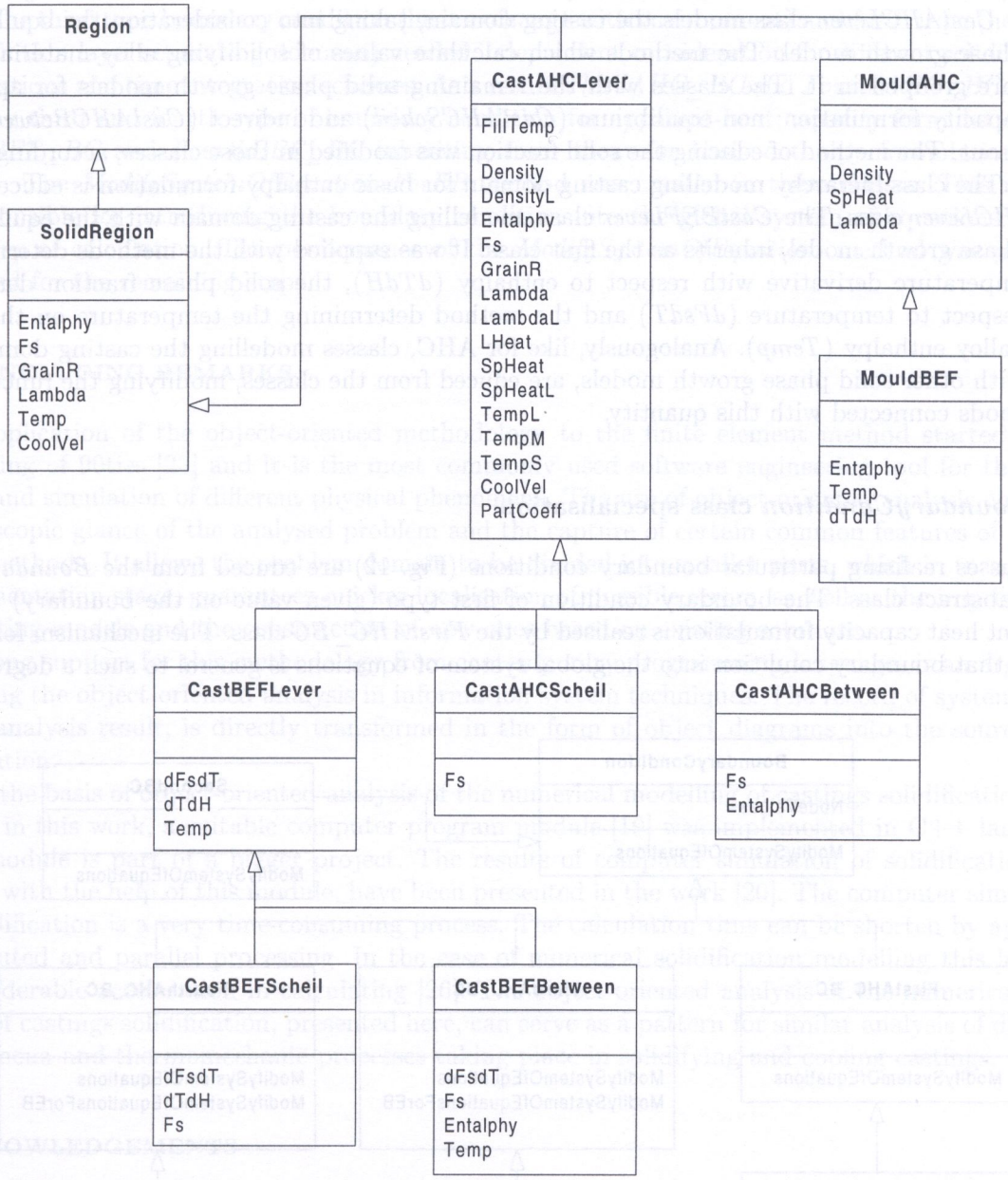

Fig. 11. *Region* class specialisation

The *MouldAHC* class contains the casting mould material properties description for the apparent heat capacity formulation. The methods, stored there, enable necessary material properties to be determined for this type of domain. The *MouldBEF* class was worked out from this class. It contains the material property description for the domains of this type, but only for the basic enthalpy formulation. It was expanded using the conversion possibility of temperature to enthalpy and vice versa. The method for determining the value of the temperature derivative with respect to enthalpy (*dTdH*) was also added.

The *CastAHCLever* class models the casting domain, taking into consideration the equilibrium solid phase growth model. The methods which calculate values of solidifying alloy material properties are grouped in it. The classes with the remaining solid phase growth models for apparent heat capacity formulation: non-equilibrium (*CastAHCScheil*) and indirect (*CastAHCBetween*) are worked out. The method of educing the solid fraction was modified in those classes, according to the model. The class hierarchy modelling casting domain for basic enthalpy formulation is educed from *CastAHCLever* class. The *CastBEFLever* class, modelling the casting domain with the equilibrium solid phase growth model, inherits as the first class. It was supplied with the methods determining: the temperature derivative with respect to enthalpy (*dTdH*), the solid phase fraction derivative with respect to temperature (*dFsdT*) and the method determining the temperature on the basis of the alloy enthalpy (*Temp*). Analogously, like for AHC, classes modelling the casting domain for BEF with other solid phase growth models, are educed from the classes, modifying the functioning of methods connected with this quantity.

6.4. *BoundaryCondition* class specialisation

The classes realising particular boundary conditions (Fig. 12) are educed from the *BoundaryCondition* abstract class. The boundary condition of first type (given value on the boundary) for the apparent heat capacity formulation is realised by the *FirstAHC_BC* class. The mechanism for introducing that boundary condition into the global system of equations is general to such a degree that

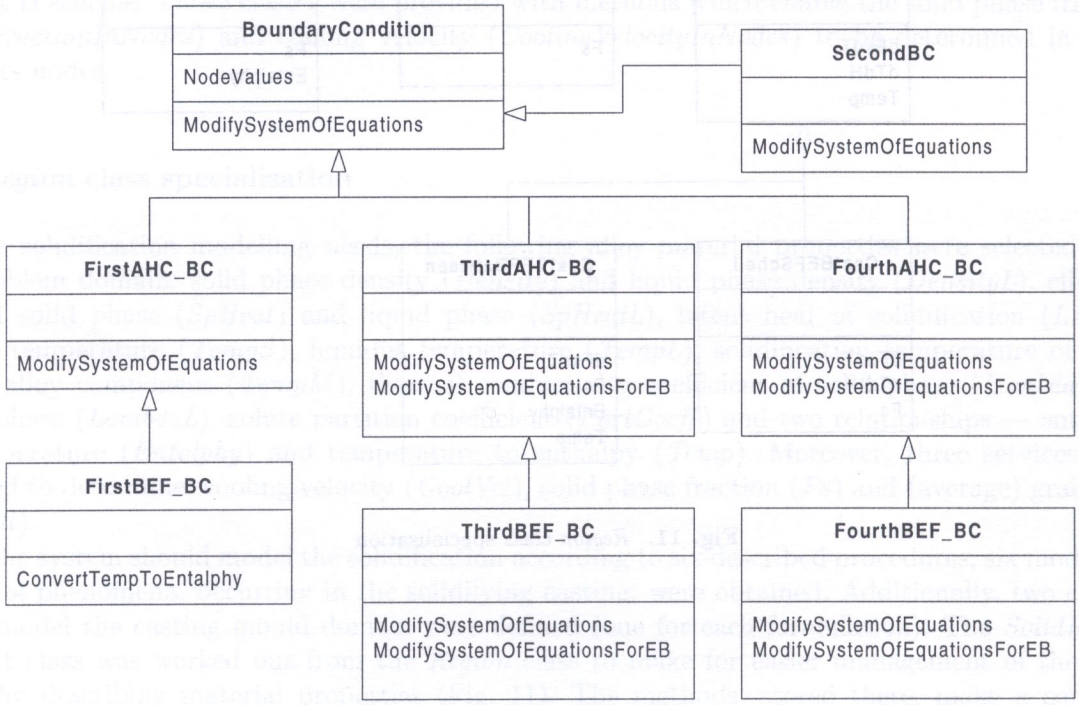

Fig. 12. *BoundaryCondition* class specialisation

there was no necessity to modify the *ModifySystemOfEquations* method for the *FirstBEF_BC* child class. The additional *ConvertTempToEntalphy* method was introduced for the basic enthalpy formulation, for which the basic quality is enthalpy. The *ConvertTempToEntalphy* method recalculates the boundary node temperature for enthalpy.

The mechanism for introducing the second type boundary condition (heat flux given on the boundary), as for the first type condition, does not depend on the solidification formulation or on the time integration scheme. For this reason it is modelled by one, common *SecondBC* class. On the other hand, the third type boundary condition (heat exchange with environment) and forth type (heat flow through thin layer separating domains) needed independent consideration for different solidification formulations and time integration schemes. As a result *ThirdAHC_BC* and *FourthAHC_BC* classes, realising third and fourth type of boundary condition for apparent heat capacity formulation, and *ThirdBEF_BC* and *FourthBEF_BC* inheriting from them for basic enthalpy formulation, were formed. The *ModifySystemOfEquationsForEB* method was created in these classes. This method is responsible for introducing the boundary condition into the global system of equations for the first step of calculation. The performance of the *ModifySystemOfEquations* method was properly modified for the remaining steps.

7. Concluding remarks

The application of the object-oriented methodology to the finite element method started at the beginning of 90ties [23] and it is the most commonly used software engineering tool for the modelling and simulation of different physical phenomena. The use of object-oriented analysis enables a macroscopic glance of the analysed problem and the capture of certain common features of the applied methods. It allows the problem domain to be divided into smaller parts, which in turn, at the implementation stage, guarantees quicker localisation of possible errors, as well as the improvement of existing models and the construction of new ones based on existing solutions.

Strong support for this methodology from contemporary programming languages is an argument for using the object-oriented analysis in information system techniques. The record of system ideas, as an analysis result, is directly transformed in the form of object diagrams into the source code application.

On the basis of object-oriented analysis of the numerical modelling of castings solidification, presented in this work, a suitable computer program module [19] was implemented in C++ language. This module is part of a bigger project. The results of computer simulation of solidification, obtained with the help of this module, have been presented in the work [20]. The computer simulation of solidification is a very time-consuming process. The calculation time can be shorten by applying distributed and parallel processing. In the case of numerical solidification modelling this leads to a considerable acceleration in calculating [26]. The object-oriented analysis of the numerical modelling of castings solidification, presented here, can serve as a pattern for similar analysis of different phenomena and thermomechanic processes taking place in solidifying and cooling castings.

Acknowledgements

The current work is sponsored by the State Committee for Scientific Research of the Republic of Poland for 1997–1999 under Grant No. 8 T11F 016 12.

References

[1] R. Barrett, et al. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* `http://www.netlib.org/templates/Templates.html` .

[2] A. Bokota. Application of boundary elements for solving the solidification problems. *Bulletin of the Polish Academy of Sciences, Technical Sciences*, **37**: 497–509, 1989.

[3] A. Bokota. Erstarrung der gleichaxialen Körner, berechnet mit der Randelementmethode. *ZAMM* **73**: T762–T765, 1993.
[4] T.W. Clyne, W. Kurz. Solute redistribution during solidification with rapid solid state diffusion. *Metall. Trans.*, **12A**: 965–971, 1981.
[5] P. Coad, E. Yourdon. *Object-Oriented Analysis* (in Polish). Oficyna Wydawnicza Read Me, Warszawa, 1994.
[6] A.J. Dalhuijsen, A. Segal. Comparison of finite element techniques for solidification problems. *Int. J. Numer Methods Eng.*, **23**: 1807–1829, 1986.
[7] R.R. Gajewski, T. Kowalczyk. A prototype object-oriented finite element method program: Class hierarchy and graphic user interface. *Computer Assisted Mechanics and Engineering Sciences*, **3**: 65–74, 1996.
[8] E. Majchrzak, B. Mochnacki. The BEM application for numerical solution of non-steady and nonlinear thermal diffusion problems. *Computer Assisted Mechanics and Engineering Sciences*, **3**: 327–346, 1996.
[9] B. Mochnacki, E. Majchrzak. Application of the BEM in thermal theory of foundry. *Engineering with Boundary Elements*, **16**: 99–121, 1995.
[10] R. Parkitny, N. Sczygiol. Volume solidification equations of two-components metallic alloys (in Polish). *Solidification of Metals and Alloys*, **12**: 29–44, 1987.
[11] R. Parkitny, N. Sczygiol. Erstarrungsthermomechanik eines Gußstückes unter Berücksichtigung seines Kornaufbaus. *ZAMM*, **69**: T514–T515, 1989.
[12] A. Radomski, O. Gajl. Object oriented software methodology: a case study of systems for dam monitoring, state diagnosis and visualisation. *Computer Assisted Mechanics and Engineering Sciences*, **2**: 271–287, 1995.
[13] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing, New York, 1995.
[14] P.R. Sahm, W. Richter, F. Hediger. Das rechnerische Simulieren und Modellieren von Erstarrungsvorgängen bei Formguß. *Giesserei-Forschung*, **35**: 35–42, 1983.
[15] N. Sczygiol. Das rechnerische Modellieren der Beimengungsverteilung in erstarrenden Gußstücken aus binären Legierungen. *ZAMM*, **74**: T619–T622, 1994.
[16] N. Sczygiol, G. Szwarc. Application of object-oriented finite element programming for castings solidification problems. *Solidification of Metals and Alloys*, **23**: 55–60, 1995.
[17] N. Sczygiol. Solidification equations by means of finite elements method. *Solidification of Metals and Alloys*, **30**: 221–232, 1997. (in Polish)
[18] N. Sczygiol, G. Szwarc, T. Olas, A. Nagórka. Object-oriented analysis of solidification modelled by finite elements. *Solidification of Metals and Alloys*, **30**: 233–242, 1997. (in Polish)
[19] N. Sczygiol. Numerical modelling of castings solidification concerning the forming grained microstructure (in Polish). *Archives of Mechanical Technology and Automatization*, **18**: 275–285, 1998.
[20] N. Sczygiol, G. Szwarc. Application of enthalpy formulations for numerical simulation of castings solidification. *Computer Assisted Mechanics and Engineering Sciences*, **8**: 99–120, 2001.
[21] K. Subieta. *Objectivity in design and data bases* (in Polish). Akademicka Oficyna Wydawnicza PLJ, Warszawa 1998.
[22] K.K. Tamma, R.R. Namburu. Recent advances, Trends and new perspectives via enthalpy-based finite element formulations for applications to solidification problems. *Int. J. Numer. Methods Eng.*, **30**: 803–820, 1990.
[23] D.A. Taylor. *Object-oriented Technology.* Helion 1994. (in Polish)
[24] UML Guide, `http://www.rational.com/uml` .
[25] W.L. Wood. *Practical Time-stepping Schemes.* Clarendon Press, Oxford, 1990.
[26] R. Wyrzykowski, N. Sczygiol, T. Olas, J. Kanevski. Parallel finite element modeling of solidification processes. *Lecture Notes in Computer Science*, No. 1557, 183–195, 1999.
[27] T. Zimmermann, Y. Dubois-Pelerin, P. Bomme. Object-oriented finite element programming: I. Governing principles. *Comp. Meth. App. Mech. Eng.*, **98**: 291–303, 1992.