AKADEMIA GÓRNICZO-HUTNICZA IM. ST. STASZICA W KRAKOWIE
WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I ELEKTRONIKI

---

mgr inż. Sławomir Bieniasz

TECHNIKI SYMULACJI AGENTOWEJ W ZASTOSOWANIU DO BADANIA PROCESÓW CIEPLNYCH

Rozprawa doktorska

Promotor:
dr hab. inż. Krzysztof Cetnarowicz

Kraków 2006

Podziękowania

Chciałbym serdecznie podziękować Panu dr hab. inż. Krzysztofowi Cetnarowiczowi za pomoc i opiekę w trakcie pisania pracy.

Jestem także bardzo wdzięczny Pani prof. dr hab. inż. Stanisławie Kluska-Nawareckiej oraz Panu dr Henrykowi Połcikowi za życzliwość i współpracę, dzięki której możliwe było opracowanie koncepcji i eksperymentów zawartych w pracy.

Dziękuję również Panu prof. dr hab. inż. Edwardowi Nawareckiemu za wiele cennych, merytorycznych uwag, które pozwoliły na nadanie prowadzonym rozważaniom większej ścisłości.

Spis treści

1. Wstęp

1.1. Obszar problemowy i cel pracy

Przedmiotem rozważań niniejszej pracy są problemy modelowania złożonych systemów dynamicznych, w których kilka procesów przebiega w tym samym czasie i w tym samym środowisku.

Istota zaproponowanej metody modelowania polega na odwzorowaniu procesów elementarnych oraz ich interakcji prowadzącym do uzyskania sumarycznych efektów wynikających z współbieżnej realizacji tych procesów.

Proponowana koncepcja oparta została na zastosowaniu podejścia agentowego, w którym poszczególne agenty realizują elementarne działania składowe, wynikiem których są złożone procesy zachodzące w wirtualnym świecie, reprezentowanym przez model.

Celem pracy jest wykazanie, że takie podejście umożliwia przybliżone lecz adekwatne odwzorowanie realnych zjawisk, a w konsekwencji przewidywanie ich przebiegu oraz wprowadzenie działań korygujących (sterowania).

Dziedziną analizowanych zjawisk są procesy cieplne, ze szczególnym uwzględnieniem zjawisk występujących w trakcie stygnięcia i krystalizacji odlewów metalowych. Jednakże, zaprezentowana metoda jest propozycją modelowania i badania szerszej klasy złożonych systemów dynamicznych.

1.2. Struktura pracy

W rozdziale 2 przedstawiona jest charakterystyka rozważanych procesów cieplnych i ich klasyczne modele komputerowe oparte o znane metody numeryczne. W tym kontekście na końcu tego rozdziału zaprezentowana jest teza niniejszej pracy.

Rozdział 3 zawiera wybrane zagadnienia związane z systemami agentowymi. Opisane są własności środowiska agentowego oraz wybrane modele teoretyczne agentów.

Kolejne rozdziały (4 – 7) zawierają prezentację proponowanych zastosowań podejścia agentowego do symulacji procesów cieplnych.

Pierwszym podejściem do symulacji złożonych zjawisk fizycznych (rozdział 4) jest połączenie symulacji w komercyjnym systemie ABAQUS z symulacją dynamicznego elementu chłodzącego w stworzonym dla tego celu systemie agentowym ABAG. W systemie ABAG przemieszczające się agenty modelują ruch i wymianę ciepła, natomiast środowisko,

w którym one działają spełnia rolę interfejsu do systemu elementów skończonych ABAQUS, który symuluje formę i odlew. Za pomocą procedury użytkownika DISP, ustalającej warunki brzegowe, system ABAQUS wymienia informacje z środowiskiem systemu ABAG.

Drugie podejście (rozdział 5) obejmuje samodzielny agentowy system MAFES-1 symulujący proces stygnięcia i krystalizacji, z dodatkowym procesem przemieszczania się medium chłodzącego, pełniącego rolę regulatora dla sterowania tym procesem. Zadaniami spełnianymi przez agentów są w tym podejściu wydzielone role w modelu symulacji i sterowania (agent obliczeniowy, agent regulator, agent sterujący).

Kolejne, trzecie podejście (rozdział 6) obejmujące system MAFES-2 wprowadza podział ról spełnianych przez agenty na dwie kategorie: agenty zjawisk fizycznych i agenty zadań. Pierwsza kategoria pozwala na wydzielenie osobnych typów agentów dla każdego zjawiska, każdy typ reprezentuje poszczególne procesy / zjawiska (dokładniej samą ich dynamikę i zmiany wprowadzane w środowisku). Druga kategoria (agenty zadań) pozwala na agentową realizację zadań typowych dla danej symulacji (typu wizualizacja czy sterowanie).

W systemie MAFES-2 wyróżniono środowisko, przechowujące stan modelu, będące obiektem działań agentów. Zastosowano uniwersalny wzorzec agenta, którego działanie sprowadza się do obserwacji i modyfikacji środowiska (oddziaływania na środowisko).

Ostanie podejście w postaci systemu MAFES-3 (rozdział 7) polega na próbie połączenia zalet systemów MAFES-1 i MAFES-2. Tutaj starano się zachować minimalną ilość typów agentów (tak jak to miało miejsce w przypadku systemu MAFES-1) wykorzystując dekompozycję zjawisk (podobnie jak w podejściu MAFES-2), dzięki zastosowaniu programowania aspektowego. Zjawiska są tu definiowane w osobnych jednostkach kompilacji (aspektach).

Dla każdego z proponowanych podejść zrealizowano serię eksperymentów symulacyjnych, które umożliwiły ocenę uzyskanych rezultatów.

Końcowy fragment pracy (rozdział 8) zawiera podsumowanie prezentowanego materiału oraz dalsze zamierzenia badawcze.

2. Sformułowanie problemu i istniejące rozwiązania

2.1. Problem wymiany ciepła w środowisku niejednorodnym

Ponieważ proponowane w pracy rozwiązania ukierunkowane są głównie na zastosowanie praktyczne (posiadają charakter narzędziowy), postanowiono skoncentrować się na odniesieniach do konkretnych obszarów zastosowania, za które wybrano zagadnienia związane ze zjawiskami cieplnymi zachodzącymi w procesie odlewniczym – a ściślej biorąc procesy stygnięcia i krzepnięcia odlewów.

Rozważana problematyka pojawiła się w wyniku poszukiwań efektywnych metod symulacji i sterowania procesami cieplnymi, w których poza odlewem i formą mogą również uczestniczyć dynamiczne elementy chłodzące, w postaci umieszczonych w formie przewodów z medium chłodzącym. Elementy takie służą realizacji sterowania – w postaci dodatkowego (poza formą) oddziaływania na rozkład pola temperatur w stygnącym odlewie, którego postać ma istotny wpływ na powstającą strukturę krystaliczną odlewu, a co za tym idzie na jego własności mechaniczne.

Proponowane w pracy koncepcje zastosowania technik agentowych dotyczą symulacji zachodzących współbieżnie zjawisk fizycznych:

- wymiany ciepła,
- krystalizacji metalu,
- wymuszonego ruchu medium chłodzącego.

Połączenie nakładających się na siebie zjawisk wymiany ciepła i ruchu ma miejsce w zestawach odlewniczych złożonych z odlewu, formy oraz układów chłodzących.

Połączenie zjawiska krystalizacji i wymiany ciepła zachodzi w trakcie przejścia fazowego podczas stygnięcia odlewu.

Aktualnie istnieją metody numeryczne do modelowania zjawiska wymiany ciepła w połączeniu ze zjawiskiem krystalizacji. Zjawisko ruchu rozpatrywane oddzielnie jest dużo łatwiejsze do modelowania, można je opisywać analitycznie. Natomiast stworzenie modelu problemu nakładania się na siebie zjawisk wymiany ciepła i ruchu prowadzi do skomplikowanych i wyrafinowanych metod numerycznych, które są dopiero w fazie opracowywania.

Modelowanie wymiany ciepła

Matematycznym modelem wymiany ciepła jest równanie różniczkowe cząstkowe typu parabolicznego (1). Jest to model uproszczony, zakładający niezmienność przewodności, ciepła właściwego i gęstości wraz ze zmianami temperatury [45, 58].

$$\frac{\partial T}{\partial t} = \frac{k}{c\,g}\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) \tag{1}$$

gdzie:

T – temperatura, jako funkcja zmiennych przestrzennych x, y, z i czasu t,

k – przewodność cieplna,

c –ciepło właściwe,

g – gęstość.

Dla zjawiska wymiany ciepła podstawowymi pojęciami są ciepło i temperatura. Ponadto w powyższym modelu używa się parametrów, opisujących własności termiczne materiału, są to: przewodność cieplna i ciepło właściwe. Wielkością, która wpływa również na wymianę ciepła jest gęstość. Ciepło właściwe jest odpowiednikiem bezwładności zmian temperatury. Natomiast przewodność odgrywa ważną rolę w przemieszczaniu się ciepła w przestrzeni i wymianie ciepła między różnymi ośrodkami przy ich wzajemnym kontakcie termicznym.

Równanie różniczkowe wymiany ciepła (1) pokazuje, że przyrost temperatury w wybranym punkcie w czasie zależy od pochodnych cząstkowych przestrzennych temperatury (otoczenia danego punktu) oraz od parametrów materiału.

Najczęściej równanie wymiany ciepła (1) rozwiązywane jest przy wykorzystaniu metody różnic skończonych lub elementów skończonych, dla których istnieją gotowe programy symulacyjne (m. in. ABAQUS, MAGMA).

W celu wyznaczania przepływu ciepła między dwoma różnymi ośrodkami (np. między odlewem a formą) potrzebny jest odpowiedni model warunków brzegowych. Zgodnie z [26] przyjęto następujący opis matematyczny warunków brzegowych:

$$k_1 \frac{\partial T_1}{\partial x} = k_2 \frac{\partial T_2}{\partial x} \qquad k_1 \frac{\partial T_1}{\partial y} = k_2 \frac{\partial T_2}{\partial y} \qquad k_1 \frac{\partial T_1}{\partial z} = k_2 \frac{\partial T_2}{\partial z} \tag{2}$$

gdzie:

k_1 – przewodność cieplna ośrodka pierwszego,

k_2 – przewodność cieplna ośrodka drugiego,

T_1 – temperatura ośrodka pierwszego,

T_2 – temperatura ośrodka drugiego.

Zjawisko krystalizacji

Przyjętym w pracy modelem krystalizacji jest model równowagi eutektycznej [26] opisany równaniami (3) – (7):

$$\frac{\partial T}{\partial t} = \frac{k}{c\,g}\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) + \frac{L_H}{c}\frac{\partial F_S}{\partial t} \tag{3}$$

$$\Delta T(t) = T_E - T(t) \tag{4}$$

$$N(t) = \Psi \Delta T(t)^2 \tag{5}$$

$$\frac{dR}{dt} = \mu \Delta T(t)^2 \tag{6}$$

$$F_S(t) = nN(t)\frac{4}{3}\pi R^3(t) \tag{7}$$

gdzie:

F_S – udział fazy stałej,

N – ilość kryształów w jednostce objętości,

R – średni promień kryształów,

T_E – temperatura eutektyki,

Wielkości F_S, N, R są funkcjami współrzędnych x, y, z oraz czasu t. Współczynniki Ψ, μ, n są to stałe dobierane doświadczalnie.

Przedstawiony model krystalizacji rozszerza przedstawione w postaci (1) prawo wymiany ciepła. Zjawisko krystalizacji związane jest ze zmianą stanu skupienia z ciekłego w stały, nazywaną również przemianą fazową. Procesowi temu towarzyszy wydzielanie ciepła, reprezentowane w prawej stronie równania (3) drugim składnikiem sumy.

Przyrost temperatury wynikający z przemiany fazowej jest bezpośrednio zależny od zmiany współczynnika F_S. Wartość tego współczynnika, określona równaniem (7), wynika z wartości N i R. Z kolei obie te wielkości zależne są od różnicy bieżącej wartości temperatury (T) oraz temperatury eutektyki (T_E).

Obliczanie wszystkich współczynników rozpoczyna się od momentu uzyskania przez temperaturę wartości T_E. Przyrost liczby kryształów (N) następuje do momentu zwanego *przechłodzeniem*, czyli do momentu gdy szybkość malenia temperatury (pochodna) osiągnie wartość 0 a następnie zacznie rosnąć (gdy ilość ciepła wydzielanego w skutek krystalizacji zaczyna przewyższać ilość ciepła oddawanego otoczeniu). Po tym momencie występuje tylko przyrost promienia kryształów (R). Proces krystalizacji kończy się z osiągnięciem przez F_S wartości 1, co oznacza, że 100% materiału przeszło w stan stały (uległo krystalizacji).

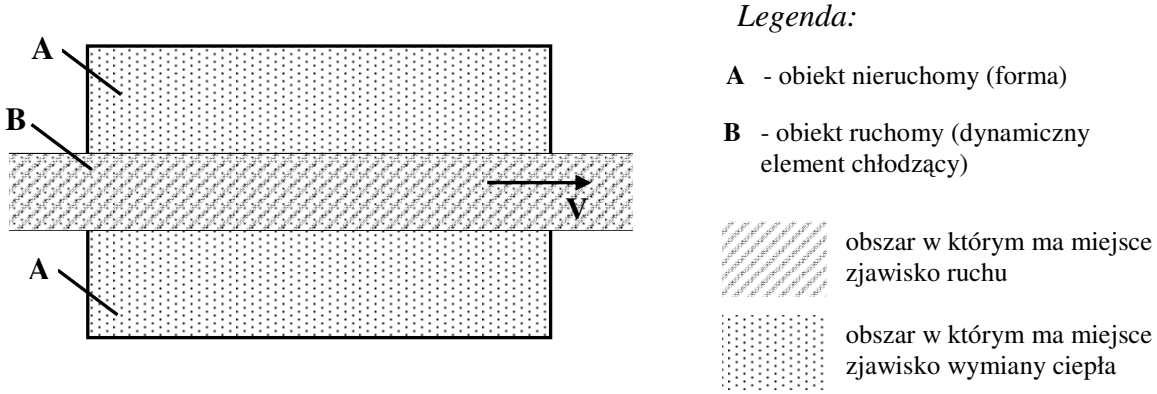
Zjawisko ruchu

W badanym procesie uwzględnia się dynamiczne elementy chłodzące zainstalowane w formie. Oddziaływanie na proces stygnięcia realizowane jest poprzez odpowiednie zmiany szybkości przepływu i temperatury przepływającego medium chłodzącego. Występują tutaj

równocześnie dwa zjawiska: ruch płynu i wymiana ciepła (między nim a otoczeniem i w nim samym).

Zjawisko ruchu traktowane jest tutaj w sposób uproszczony, co oznacza między innymi pomijanie efektów zawirowań. Założono, że realizacja ruchu odbywa się w ośrodku o takich samych parametrach termofizycznych, zatem przemieszczanie się medium chłodzącego sprowadzone jest do przenoszenia tylko pola temperatur (zgodnie z dynamiką ruchu).

Przykład modelu, w którym występuje nakładanie się zjawiska wymiany ciepła i ruchu przedstawiono na rys. 2.1. Zjawisko wymiany ciepła występuje tu w całym rozważanym modelu (w obiektach A i B). Natomiast zjawisko ruchu dotyczy tylko obszaru obiektu B, a zatem w obiekcie tym ma miejsce nakładanie się dwóch zjawisk.



Rys. 2.1. Przykład ilustrujący nakładanie się zjawisk wymiany ciepła i ruchu

2.2. Metody numeryczne w modelowaniu procesów cieplnych

Do najważniejszych metod numerycznych używanych w symulacji procesów cieplnych należą Metoda Różnic Skończonych [56] (FDM – ang. *Finite Difference Method*) oraz Metoda Elementów Skończonych [76] (FEM – ang. *Finite Element Method*). Każda z tych metod dysponuje szeroko rozwiniętym aparatem matematyczno – numerycznym, jednakże ich teoria nie obejmuje przypadków, w których proces wymiany ciepła jest w pewnych obszarach zaburzany przez zewnętrznie wymuszony ruch (tak jak to ma miejsce w dynamicznych elementach chłodzących rozważanych w pracy).

W dążeniu do uwzględnienia złożonych zjawisk powstało wiele specjalistycznych metod numerycznych, których przykładem jest metoda MEM [5] (ang. *Mortar Element Method*). Metody tej klasy ogólnie biorąc polegają na wprowadzaniu specjalnych typów elementów kontaktowych oddzielających środowisko ruchome od statycznego (siatek podstawowych elementów tych obszarów). W przypadku zmiany konfiguracji wzajemnej obiektów ruchomych i statycznych konieczna jest rekonstrukcja siatek elementów.

Realizacja komputerowa metod numerycznych operuje z reguły na macierzach oraz procedurach służących do ich przetwarzania. Taki model komputerowy jest odległy koncepcyjnie od fizykalnej interpretacji analizowanego problemu. W okresie powstawania większości metod numerycznych koncepcje modelowania komputerowego były na początkowym etapie rozwoju. Można więc zauważyć iż model komputerowy wynikający z metod numerycznych odzwierciedla w pewnym stopniu koncepcje języków programowania tego początkowego etapu ich rozwoju.

W dalszej fazie rozwoju metod numerycznych pojawiły się nowe koncepcje modelowania komputerowego oparte m. in. o programowanie obiektowe [16, 23] lub zastosowanie systemów wieloagentowych. Niektóre możliwości jakie stwarza to ostatnie podejście są właśnie przedmiotem rozważań prowadzonych w niniejszej pracy.

2.3. Teza pracy

Jako punkt wyjścia do sformułowania tezy pracy przyjęto stwierdzenie, że modelowany proces fizyczny można potraktować jako złożenie zjawiska ruchu oraz zjawiska rozpływu ciepła.

Przy takim podejściu, w przekonaniu autora można postawić tezę, że zastosowanie koncepcji agenta i technologii agentowych pozwala na adekwatną i efektywną reprezentację złożonych zjawisk obejmujących wymianę ciepła w połączeniu z ruchem fizycznych elementów ośrodka.

Stworzony w ten sposób model, pozwala na uniknięcie stosowania metod numerycznych o dużej złożoności, zachowując równocześnie naturalną a więc łatwiejszą do interpretacji relację z fizyczną postacią symulowanego procesu.

3. Systemy wieloagentowe

3.1. Charakterystyka ogólna

Badania nad systemami wieloagentowymi (MAS, ang. *Multiagent Systems*) zaliczane często do dziedziny określanej jako rozproszona sztuczna inteligencja (DAI, ang. *Distributed Artificial Intelligence*), zostały zainicjowane w latach osiemdziesiątych [20, 70]. Aktualnie, teoria systemów wieloagentowych jest nie tylko przedmiotem badań naukowych, ale również stała się domeną nauczania akademickiego oraz znalazła zastosowanie w budowie przemysłowych i komercyjnych aplikacji [70].

System wieloagentowy to system, w którym grupa współdziałających agentów dąży do osiągnięcia pewnego zbioru celów lub wykonuje pewien zbiór zadań [70]. Agent to autonomiczna jednostka (taka jak np. program komputerowy lub robot), która postrzega swoje środowisko i może na nie oddziaływać.

Eksponowana często cecha agenta - autonomiczność oznacza, że ma on w pewnym zakresie kontrolę nad swoim zachowaniem oraz może działać bez interwencji człowieka lub innego programu (obiektu).

Agent realizuje cele lub wykonuje zadania w taki sposób, aby spełnić założenia projektowe przyjęte przy jego konstruowaniu.

W pracy [43] przedstawione są następujące cechy systemu wieloagentowego:
- każdy agent dysponuje niepełną informacją i ma ograniczone możliwości działania,
- sterowanie w systemie jest zdecentralizowane,
- dane są rozproszone,
- obliczenia przebiegają w sposób asynchroniczny.

Systemy MAS mogą się różnić sposobem modelowania i realizacji agenta, interakcji między agentami i środowiska, w którym agenty działają.

Interakcja agentów może się odbywać w sposób pośredni poprzez środowisko, w którym są one osadzone (np. przez obserwację stanu środowiska lub przeprowadzenie akcji, która modyfikuje stan środowiska) lub w sposób bezpośredni poprzez komunikację przy pomocy wspólnego języka (poprzez przekazywanie informacji).

Zastosowania systemów wieloagentowych

Zarówno aktualne jak i perspektywiczne zastosowania systemów agendowych mogą być bardzo różnorodne. Jako najbardziej charakterystyczne [43], można wymienić:

- Handel elektroniczny i rynki elektroniczne, gdzie kupujące i sprzedające agenty nabywają i sprzedają towary w imieniu ich użytkowników.
- Monitorowanie i zarządzanie sieciami telekomunikacyjnymi, gdzie agenty odpowiadają np. za przekazanie rozmowy, przełączanie czy transmisję.
- Modelowanie i optymalizacja systemów transportowych (w obszarze gospodarstwa domowego, miejskim, narodowym, światowym), gdzie agenty reprezentuja np. pojazdy transportujące lub przewożone towary czy klientów.
- Przetwarzanie informacji w środowiskach informacyjnych typu Internet, gdzie wiele agentów odpowiada np. za filtrację informacji czy jej gromadzenie.
- Organizacja ruchu miejskiego czy powietrznego, gdzie agenty odpowiadają za odpowiednie interpretowanie danych pojawiających się w różnych punktach pomiarowych.
- Zautomatyzowanie planów spotkań, gdzie agenty działając w imieniu ich użytkowników ustalają szczegóły spotkań takie jak miejsce, czas, porządek spotkania.
- Optymalizacja przemysłowych procesów produkcyjnych, gdzie agenty reprezentują agregaty produkcyjne lub przedsiębiorstwa.
- Analiza procesów biznesowych wewnątrz lub pomiędzy przedsiębiorstwami, gdzie agenty reprezentują ludzi lub różne wydziały zaangażowane w te procesy na różnych etapach i różnych poziomach.
- Elektroniczna rozrywka i interakcyjne, z wirtualną rzeczywistością, gry komputerowe, gdzie np. animowane agenty wyposażone w różne charaktery grają przeciwko sobie lub przeciwko ludziom.
- Projektowanie i reinżynieria wzorców przepływu informacji i sterowania w organizacjach naturalnych, technicznych i hybrydowych wielkiej skali, gdzie agenty reprezentują jednostki odpowiedzialne za te wzorce.
- Badanie społecznych aspektów inteligencji i symulacja złożonych społecznych zjawisk takich jak ewolucja ról, norm i struktur organizacyjnych, gdzie agenty spełniają rolę członków naturalnej społeczności będącej przedmiotem rozważań.

Wspólnymi cechami wymienionych zastosowań są:

a) *wbudowane rozproszenie* – są one wewnętrznie rozproszone w takim znaczeniu, że dane przetwarzane:
   - występują w różnych miejscach geograficznych („rozproszenie przestrzenne”),
   - występują w różnym czasie („temporalne rozproszenie”),
   - mogą być zgrupowane w klastry, dostęp do których i użycie wymaga znajomości różnych ontologii i języków („semantyczne rozproszenie”),
   - mogą być również zgrupowane w klastry, dostęp do których i użycie wymaga zdolności oddziaływania, percepcyjnych, klasyfikujących („funkcjonalne rozproszenie”).

b) *wbudowana złożoność* – są one wewnętrznie złożone w tym znaczeniu, że są zbyt duże by być rozwiązane przez pojedynczy, zcentralizowany system, z powodu ograniczeń na danym poziomie technologii sprzętu i oprogramowania komputerowego. Zastosowanie zcentralizowanego systemu, który spełnia wymagania rozważanej złożoności zwykle jest bardzo trudne, wymaga dużo czasu i kosztów. Poza tym, takie rozwiązania często prowadzą do rezultatów, które są nietrwałe i stają się nieużyteczne gdy tylko wymagania im stawiane zmieniają się nieznacznie (niestabilność).

Realizacja wewnętrznie rozproszonych i złożonych problemów w zcentralizowany sposób jest nie tylko niezgodne z intuicją, lecz często wręcz niemożliwa. Alternatywą jest rozbicie procesu rozwiązania takich problemów pomiędzy wiele jednostek zdolnych do inteligentnego współdziałania, czyli zastosowanie podejścia agentowego.

Własności systemów wieloagentowych

Dwa ważne powody skłaniają do zainteresowania się i badań nad systemami wieloagentowymi [70]:

- *Potrzeby technologiczne i aplikacyjne* – Systemy wieloagentowe oferują obiecującą i nowoczesną metodę pozwalającą zrozumieć, zarządzać i używać rozproszone, wielkiej skali, dynamiczne, otwarte, heterogeniczne systemy informacyjne i obliczeniowe. Sieć Internet jest widocznym przykładem takich systemów. Innym przykładem są systemy wielo-bazodanowe czy obsługujące sprzęt gospodarstwa domowego. Komputery i aplikacje komputerowe odgrywają coraz większe znaczenie i wpływają na codzienne życie, stają się coraz bardziej doskonałe i mocniej wzajemnie powiązane przez lokalne i globalne sieci, powiązane z ludźmi przez interfejsy użytkownika. Te systemy są zbyt złożone by można je było w pełni scharakteryzować i dokładnie opisać. Sterowanie w nich staje się coraz bardziej zdecentralizowane, ich komponenty działają jak „indywidua", które można określać takimi atrybutami jak autonomiczność, racjonalność, inteligencja a nie tylko jak „część". DAI dąży nie tylko do szukania metod budowania skomplikowanych, interaktywnych systemów, ale również do łączenia istniejących, odziedziczonych systemów, takich które działają spójnie jako samodzielna całość. Jak żadna inna dyscyplina, DAI dąży do tego by dostarczyć rozwiązań dla wewnętrznie rozproszonych i wewnętrznie złożonych aplikacji, takich, którym trudno sprostać przy zcentralizowanym podejściu.
- *Naturalne spojrzenie na systemy inteligentne* – Systemy wieloagentowe oferują naturalny sposób na spojrzenie i charakterystykę systemów inteligentnych. Inteligencja i współdziałanie są głęboko powiązane i systemy wieloagentowe właśnie to realizują. Naturalne inteligentne systemy, jak ludzie, nie funkcjonują w izolacji. Są nie tylko częścią środowiska, lecz mogą na nie aktywnie oddziaływać. Ludzie współdziałają na różne

sposoby i na różnych poziomach i większość ludzi osiąga wyniki dzięki współdziałaniu. DAI może umożliwić wgląd i zrozumienie słabo zbadanych interakcji pomiędzy naturalnymi inteligentnymi istotami, szczególnie w przypadku ich organizacji w różne grupy, stowarzyszenia, społeczności mające na celu osiągnięcie postępu.

Dodatkowo systemy wieloagentowe, jako systemy rozproszone mogą zaoferować szereg ważnych możliwości [15], takich jak:

- *Wydajność* – agenty mogą działać asynchronicznie i równolegle.
- *Niezawodność* – awaria jednego lub kilku agentów nie oznacza awarii całego systemu, ponieważ inne agenty mogą przejąć ich zadania.
- *Skalowalność i elastyczność* – system może zostać zaadaptowany do problemu o większym rozmiarze dzięki dodaniu nowych agentów i niekoniecznie oddziałuje to na działanie pozostałych agentów (problem skalowalności w systemach agentowych nie jest w pełni rozwiązany, jest przedmiotem badań).
- *Redukcja kosztów* – angażują mniejsze koszty niż podejście zcentralizowane, gdyż można rozwiązać problem wykorzystując proste, dużo tańsze podsystemy.
- *Elastyczność w zakresie projektowania i wykorzystania* – pojedyncze agenty mogą być projektowane przez oddzielnych specjalistów (albo od podstaw albo na bazie istniejących rozwiązań), cały system może być testowany i utrzymywany łatwiej, istnieje możliwość rekonfiguracji i ponownego użycia agentów w różnych sytuacjach.

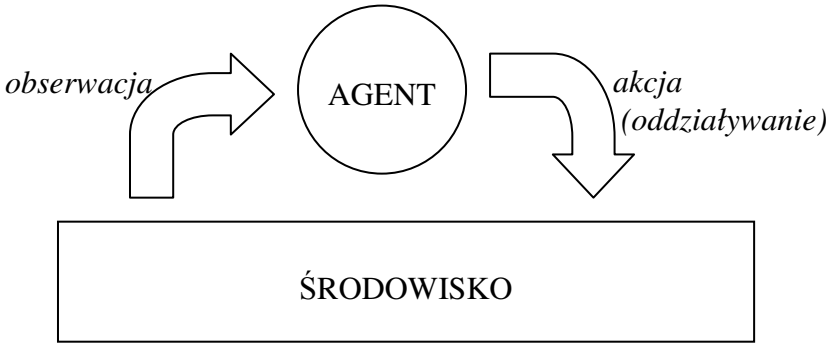
Dostępne technologie informatyczne (rozwój sieci, programowania obiektowego, równoległego, rozproszonego, itd.) dostarczają możliwości realizowania i stosowania systemów wieloagentowych, o różnej funkcjonalności i obszarach zastosowania.

3.2. Koncepcja agenta

Trudno znaleźć w literaturze powszechnie zaakceptowaną, uniwersalną definicję agenta. Najczęściej wymienianym atrybutem związanym z agentowością jest autonomia. Natomiast waga przykładana do innych cech agendowości, zależy od rozważanego obszaru zastosowań. Niekiedy zdolność agenta do uczenia się jest najważniejsza, a w innych przypadkach zdolność ta nie tylko nie jest istotna, lecz nawet niepożądana [72].

W pracy [73] autorzy definiują agenta jako system komputerowy, który jest umiejscowiony w określonym środowisku i jest zdolny do autonomicznych działań, służących realizacji jego celów projektowych.

Autonomia jest również pojęciem trudnym do zdefiniowania. Często określa się ją jako zdolność agentów do działania bez interwencji człowieka czy innych systemów, wówczas agenty mają kontrolę nad swoim wewnętrznym stanem i swoim zachowaniem.



Rys. 3.1. Agent i jego środowisko – model poglądowy

Rys. 3.1 przedstawia abstrakcyjny model agenta, w którym agent obserwuje stan środowiska i generuje akcje wpływające na to środowisko. Występowanie w działaniu agenta funkcji obserwacji i oddziaływania na środowisko, jest powszechne dla wielu modeli agenta, do których należy koncepcja M-Agenta przedstawiona w pracy [20].

3.3. Środowisko w systemach agentowych

Środowisko dla danego agenta może być dostępne w różny sposób. W pracy [20] zaproponowany został następujący podział środowiska na charakterystyczne obszary (rys. 3.2):

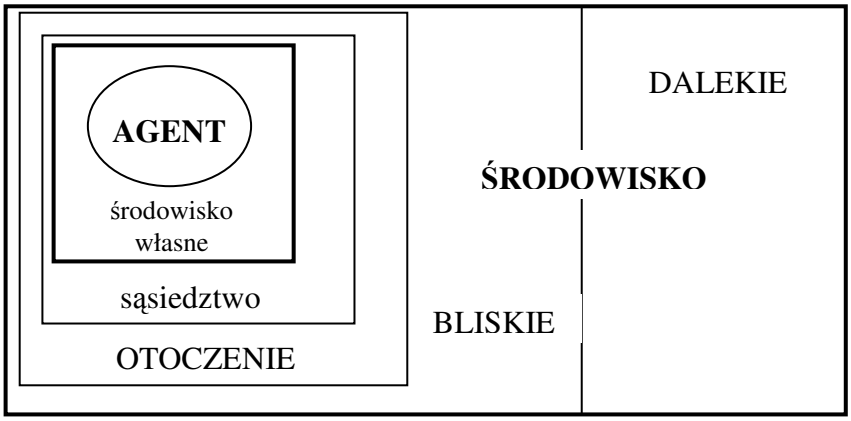
Środowisko dalekie (lub środowisko) – zbiór danych, które są w zasięgu postrzegania danego agenta bezpośrednio lub pośrednio (agent może być zmuszony do przedsięwzięcia pewnych czynności nie należących do procedur obserwacji w celu dostrzeżenia części lub elementów wspomnianego środowiska np. agent musi się przemieścić, aby dostrzec pewne jego elementy).

Środowisko bliskie – jest to środowisko, które jest w zasięgu obserwacji danego agenta przy użyciu wyłącznie procedur postrzegania.

Otoczenie – część środowiska (bliskiego), która jest w zasięgu możliwości sprawczych agenta, tzn. na którą dany agent może oddziaływać, przetwarzać ją i zmieniać.

Sąsiedztwo – część otoczenia, na którą dany agent może oddziaływać z innymi, większymi uprawnieniami (a tym samym możliwościami) niż inne agenty.

Środowisko własne – jest to ta część sąsiedztwa, która może być zmieniana (modyfikowana) wyłącznie przez danego agenta – zwanego właścicielem. Inne agenty mogą dokonywać zmian tej części środowiska wyłącznie za pośrednictwem i za zgodą właściciela. Agent wraz ze swoim środowiskiem własnym tworzą podstawową jednostkę budowy systemów zdecentralizowanych.



Rys. 3.2. Podział środowiska wokół agenta, z punktu widzenia jego dostępności

W pracy [65] zaproponowano następującą klasyfikację własności środowiska:

- *Dostępne – niedostępne*.
  Środowisko dostępne to takie, w którym agent może otrzymać kompletną, dokładną, aktualną informację o stanie środowiska. Większość bardziej złożonych środowisk (np. świat fizyczny, Internet) jest niedostępna. Im bardziej dostępne środowisko, tym prościej można zaprojektować agenta, który działa w nim.
- *Deterministyczne – niedeterministyczne*.
  Deterministyczne środowisko to takie, w którym każda akcja prowadzi to pojedynczego gwarantowanego efektu – po wykonaniu akcji nie występuje niepewność odnośnie wynikowego stanu. Środowisko niedeterministyczne stwarza większe problemy projektantowi agenta, gdyż efekty akcji nie są dokładnie znane.
- *Epizodyczne – nieepizodyczne*.
  W środowisku epizodycznym rezultat działania agenta jest zależny od pewnej liczby dyskretnych epizodów, nie ma związku z rezultatami działania agenta w dłuższym horyzoncie (przykładem środowiska epizodycznego może być system sortujący pocztę). Epizodyczne środowisko jest prostsze z perspektywy projektanta, ponieważ agent może decydować, którą akcję wybrać bazując tylko na bieżącym epizodzie – nie potrzebuje analizować zależności pomiędzy tym a przyszłymi epizodami.

- *Statyczne – dynamiczne*.
  Statyczne środowisko to takie, gdy zakłada się iż jego stan może zostać zmieniony tylko w wyniku działania agenta. Dynamiczne środowisko zawiera inne procesy działające w nim i w związku z tym zmienia swój stan niezależnie od działania agenta.
- *Dyskretne – ciągłe*.
  Środowisko jest dyskretne jeżeli występuje w nim stała, skończona liczba akcji i obserwacji (przykładem środowiska dyskretnego jest gra w szachy, natomiast prowadzenie pojazdu jest przykładem środowiska ciągłego).

Gdy środowisko jest dostatecznie złożone, fakt że jest ono deterministyczne nie ma dużego znaczenia. Najbardziej złożonym przypadkiem środowiska jest środowisko niedostępne, niedeterministyczne, nieepizodyczne, dynamiczne i ciągłe.

3.4. Abstrakcyjne architektury agentów

Zgodnie z pracą [72] można dokonać formalnego opisu abstrakcyjnego pojęcia agenta. Po pierwsze załóżmy, że stany środowiska agenta można scharakteryzować jako zbiór $S = \{s_1, s_2, \ldots\}$. Zakłada się, że w każdym momencie środowisko znajduje się w jednym z tych stanów. Z kolei zdolność oddziaływania agenta niech będzie reprezentowana przez zbiór akcji $A = \{ a_1, a_2, \ldots\}$. Przy tych oznaczeniach, agenta (standardowego agenta) można reprezentować poprzez funkcję:

$action : S^* \rightarrow A$

która przyporządkowuje sekwencji stanów środowiska akcję. Intuicyjne wytłumaczenie tego zapisu jest takie, iż agent podejmuje decyzję, którą akcję wykonać na podstawie historii – jego dotychczasowych doświadczeń. Doświadczenia te są reprezentowane jako sekwencja stanów środowiska – tych, z którymi agent się zetknął.

Zachowanie (niedeterministyczne) środowiska może być modelowane jako funkcja:

$env : S \times A \rightarrow \varphi(S)$

Funkcja ta przyjmuje jako argument bieżący stan środowiska $s \in S$ i akcję $a \in A$ (wykonaną przez agenta) i przyporządkowuje im zbiór stanów środowiska $env(s, a)$ – tych, które mogą wyniknąć z wykonania akcji a w stanie s. Jeśli zbiory te dla wszystkich wartości funkcji env są zbiorami jednoelementowymi, wówczas środowisko jest deterministyczne i jego zachowanie może być dokładnie przewidziane.

Słabo reaktywne agenty

Pewne typy agentów podejmują decyzję o wyborze akcji bez odniesienia do historii stanów. Decyzja ta jest oparta na bieżącym stanie środowiska. Taki typ agentów określany jest jako agenty *słabo reaktywne* [72], gdyż ich działanie polega na prostym reagowaniu na stan środowiska. Formalnie zachowanie słabo reaktywnych agentów może być reprezentowane przez funkcję:

$action : S \rightarrow A$

Percepcja

Dokonując głębszej analizy zachowania agenta można wyróżnić wewnątrz agenta wynik operacji obserwacji *see* w postaci percepcji (obrazu) stanu środowiska. Zakładając, że *P* jest niepustym zbiorem percepcji, można określić operację obserwacji jako odwzorowanie:

$see : S \rightarrow P$

Wówczas operacja akcji agenta będzie określona jako odwzorowanie:

$action : P^* \rightarrow A$

Te proste definicje pozwalają zauważyć pewne interesujące własności agentów i percepcji. Przypuśćmy, że dla dwóch stanów środowiska $s_1 \in S$ i $s_2 \in S$, takich, że $s_1 \neq s_2$, ich percepcje $see(s_1) = see(s_2)$. Zatem dla dwóch różnych stanów środowiska ich percepcje są identyczne. Wówczas agent uzyskuje tę samą percepcyjną informację przy dwóch różnych stanach. Można powiedzieć, że w takim przypadku stany środowiska s_1 i s_2 są dla agenta nieodróżnialne.

Agenty ze stanem

Wykorzystując wprowadzone określenia, możliwe jest rozszerzenie pojęcia agenta na agenta przechowującego stan.
Zakładając, że *I* jest zbiorem wewnętrznych stanów agenta i funkcja percepcji ma postać:

$see : S \rightarrow P$

dla agenta ze stanem wybór akcji jest określony odwzorowaniem:

$action : I \rightarrow A$

prowadzącym od określonego stanu agenta do akcji. Zatem o wyborze przez agenta akcji decyduje tutaj jego bieżący stan wewnętrzny.

Wewnętrzny stan agenta może ulec zmianie w wyniku uwzględnienia aktualnej percepcji. W tym celu określono funkcję *next*, która odwzorowuje wewnętrzny stan i percepcję w nowy stan wewnętrzny agenta:

$next : I \times P \rightarrow I$

W pracy [72] pokazano, że każdy agent z wewnętrznym stanem może być sprowadzony do agenta bezstanowego.

Opisane w sposób abstrakcyjny architektury agentów, znajdują bezpośrednie lub pośrednie odbicie w konkretnych architekturach, opisywanych w literaturze realizacji [72].
W szczególności, warto wśród nich wskazać:

- *agenty bazujące na logice* [33] – w których podejmowanie decyzji odbywa się przez logiczną dedukcję,
- *agenty reaktywne* [25] – w których podejmowanie decyzji ma formę bezpośredniego odwzorowania sytuacji w akcję,
- *agenty BDI* (ang. *belief-desire-intention*) [17] – w których podejmowanie decyzji opiera się na manipulacji w ramach struktur danych reprezentujących przekonania, pragnienia, zamiary,
- *architektury warstwowe* [59] – w których podejmowanie decyzji jest realizowane poprzez wykorzystanie warstw, odpowiadających rozumowaniu o środowisku na różnych poziomach abstrakcji,
- *M-agenty* [20] – gdzie kolejne decyzje określane są w oparciu o porównanie efektu (stanu) zamierzonego, z rzeczywiście uzyskanym (zestawienie modelu z rzeczywistością).

4. FEM w połączeniu z podejściem agentowym

4.1. Wprowadzenie

Jak już wspomniano, w celu ukonkretnienia proponowanych koncepcji modeli symulacyjnych, prowadzone rozważania odniesiono do procesu stygnięcia i krystalizacji odlewów w formach, co nie ogranicza możliwości zastosowania danego podejścia do innych złożonych procesów termofizycznych.

Przeprowadzenie symulacji dla ustalonych warunków początkowych konfiguracji odlew – forma, jest możliwe do realizacji w większości komercyjnych systemów symulacyjnych. Zadaniem trudniejszym do wykonania jest wprowadzanie do komputerowego modelu dynamicznych elementów (np. przewodów z medium chłodzącym o zmiennym przepływie – opisanych w rozdziale 2.1). W przypadku wprowadzenia sterowania przebiegiem symulacji, wymagane jest odwzorowanie zmian zachowania odpowiednich elementów dynamicznych, wynikających z działania algorytmów, których zadaniem jest zapewnienie uzyskanie odpowiedniego przebiegu wybranych parametrów, w różnych obszarach odlewu.

Wymagania tego typu mają duże znaczenie praktyczne, w szczególności stawiane są w celu zapewnienia pożądanej jakości odlewów. Struktura stopu, która jest kształtowana w trakcie procesu krzepnięcia zależy nie tylko od chemicznego składu danego stopu, ale również od gradientu temperatury i szybkości krzepnięcia. Przy wysokich wymaganiach w stosunku do jakości odlewów nie wystarcza zatem tylko obserwowanie parametrów w metalu i formie dla zadanej konfiguracji odlew – forma, lecz konieczne jest stworzenie odpowiednich oddziaływań, które zapewnią właściwy przebieg dynamicznych zmian w krzepnącym metalu, poprzez kształtowanie pola temperatur.

Prezentowana w tym rozdziale koncepcja powstała jako wynik przeprowadzenia szeregu eksperymentów symulacyjnych przy wykorzystaniu systemu ABAQUS [34, 50, 51], podczas których pojawiły się trudności w odwzorowaniu działań sterujących, w postaci dynamicznych elementów chłodzących.

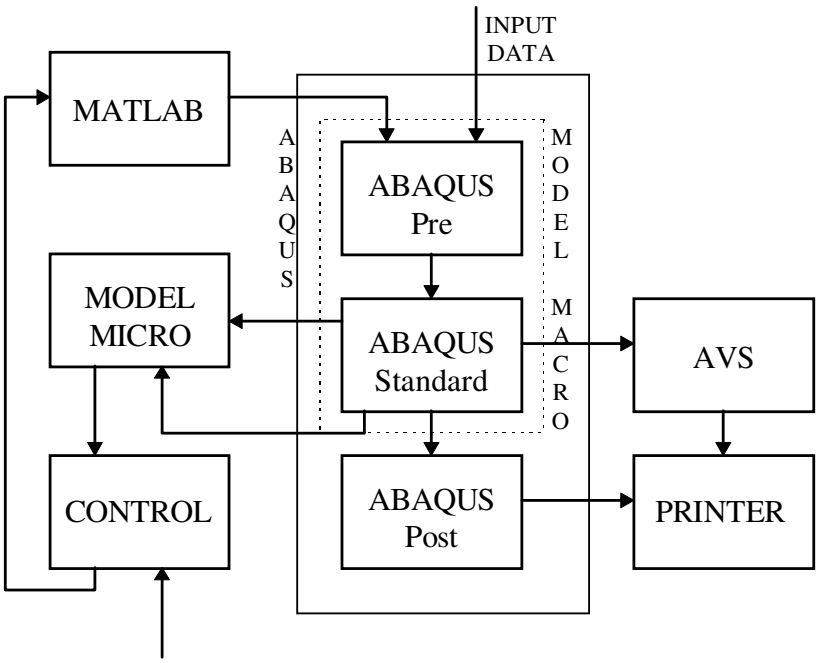
Zaproponowano tu model symulacyjny pozwalający na badanie efektów sterowania procesem chłodzenia odlewu. Używa on systemu ABAQUS [1], wykorzystującego metodę elementów skończonych oraz systemu agentowego ABAG [7], zapewniającego możliwość oddziaływania na model podstawowy. Połączenie tych dwóch systemów daje możliwość sterowania warunkami brzegowymi, w sposób analogiczny do użycia medium chłodzącego w rzeczywistych procesach odlewniczych.

Przeprowadzone przy takim podejściu eksperymenty symulacyjne są pomocne nie tylko w projektowaniu optymalnego kształtu formy, ale również w znalezieniu odpowiednich warunków chłodzenia (sterowania), zapewniających pożądany przebieg pól temperatur w odlewie. W dalszej części rozdziału zaprezentowano zasady współpracy systemów ABAQUS i ABAG. Przedstawiono również wybrane rezultaty eksperymentów symulacyjnych.

4.2. Realizacja symulacji oparta na systemie ABAQUS

Struktura modelu symulacyjnego opartego na systemie ABAQUS, w formie blokowego diagramu została przedstawiona na rys. 4.1.

ABAQUS jest systemem uniwersalnym, z bogatą biblioteką różnych rodzajów elementów skończonych. Znajduje zastosowanie w różnych dziedzinach, między innymi przy rozwiązywaniu zagadnień mechanicznych oraz termofizycznych.



Rys. 4.1. Struktura modelu symulacyjnego zrealizowana w oparciu o system ABAQUS

Rys. 4.1 pokazuje trzy główne moduły tego systemu: ABAQUS/Pre, ABAQUS/Standard i ABAQUS/Post.

Pierwszy z nich (ABAQUS/Pre) jest przeznaczony do tworzenia modelu wejściowego dla symulacji. Pozwala on między innymi na:

- budowanie obiektów geometrycznych,
- definiowanie materiałów,

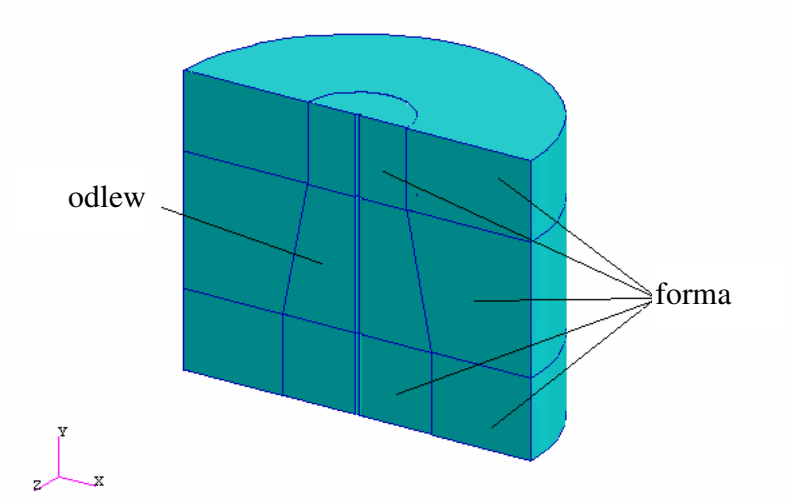
- ustalanie warunków początkowych,
- generowanie siatki elementów skończonych wybranego typu,
- określanie warunków przebiegu symulacji, takich jak np. ilość iteracji, krok czasowy, czy dopuszczalną zmianę parametrów modelu symulacji w jednym kroku obliczeniowym.

ABAQUS/Standard realizuje obliczenia dla przygotowanego modelu. W wyniku jego działania powstaje szereg plików wynikowych, na podstawie których można przeprowadzić wizualizację. Do tego celu służy program ABAQUS/Post. W nowszych wersjach systemu ABAQUS, preprocesor ABAQUS/Pre i postprocesor ABAQUS/Post zostały połączone w jeden program o nazwie ABAQUS/CAE (Complex Abaqus Environment).

W zadaniach symulacyjnym opartych na systemie ABAQUS można korzystać z modułów zewnętrznych (AVS – specjalistyczna wizualizacja, MATLAB – realizacja sterowania), co zaznaczono na rys. 4.1.

Poniżej przedstawiono przykładowy model symulacyjny procesu stygnięcia odlewu, zrealizowany w systemie ABAQUS. Schemat modelowanej formy i odlewu (przekrój) pokazano na rys. 4.2.

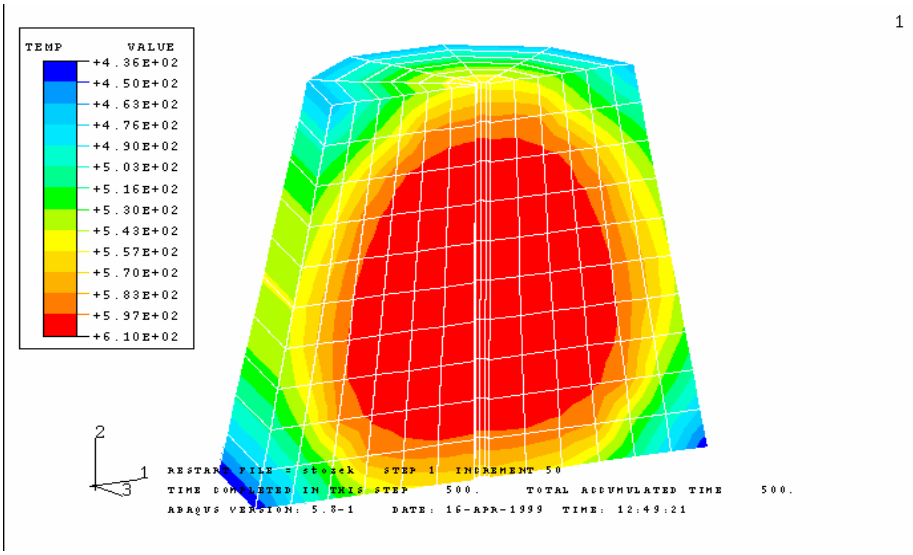
Wynik eksperymentu symulacyjnego w postaci rozkładu temperatur został przedstawiony na rys. 4.3. Dynamika zmian temperatury była zróżnicowana w zależności od rozpatrywanego obszaru odlewu. Prędkość zmian temperatury (stygnięcia) była większa blisko formy – w górnej i dolnej części odlewu.



Rys. 4.2. Schemat formy i odlewu (przekrój) modelu symulacyjnego realizowanego w systemie ABAQUS

Zaobserwowane, znaczne różnice w szybkości schładzania powodują, że przebieg krystalizacji jest również zróżnicowany w obszarze odlewu. Oznacza to, iż mamy do

czynienia z różną prędkością powstawania zarodków kryształów i różnymi wartościami średnich promieni kryształów, w zależności od miejsca odlewu. Przy tych warunkach, techniczne parametry odlewu w poszczególnych jego podobszarach, mogą być wyraźnie zróżnicowane.



Rys. 4.3. Rozkład temperatur w odlewie po 500 sekundach symulacji procesu stygnięcia

Jak już wspomniano, najczęściej dąży się do uzyskania takiej samej – jednolitej struktury krystalicznej w różnych obszarach odlewu, co zależy od intensywności schładzania poszczególnych jego części.

Można wskazać kilka metod umożliwiających oddziaływanie na szybkość schładzania odlewu:

- dobór warunków początkowych (temperatura początkowa odlewu lub formy),
- dobór odpowiedniego kształtu formy oraz nadlewu,
- dodanie do formy wkładek z materiałów o innych parametrach termicznych niż parametry formy.

Jak wykazują doświadczenia praktyczne, znacznie lepszą metodą dokonywania zmian warunków chłodzenia, jest zastosowanie dynamicznych elementów chłodzących (rurka z płynem chłodzącym, umiejscowiona w analizowanym przypadku w osi symetrii odlewu). Przy takim rozwiązaniu powinna być stworzona możliwość doboru:

- temperatury wpływającego medium,
- prędkości przepływu medium w trakcie stygnięcia odlewu.

Rozwiązanie w postaci dynamicznego elementu chłodzącego, można traktować jako dodatkową formę, która zapewnia absorpcję ciepła w środku odlewu (dla przykładowego modelu). Absorpcja ta może być sterowana w czasie symulacji, na przykład w taki sposób,

aby zapewnić możliwie zbliżony przebieg temperatur w czasie, w dwóch wybranych punktach odlewu. Algorytm sterujący powinien monitorować temperaturę w tych punktach w trakcie symulacji i wykorzystując predykcję, wyznaczać odpowiednie wartości temperatury wpływającego medium i prędkość jego przepływu.

Realizacja tego typu oddziaływania w systemie ABAQUS napotyka na trudność polegającą na tym, że w bibliotece tego systemu nie występuje element skończony uwzględniający jednocześnie wymianę ciepła i zjawisko ruchu oraz związane z tym przemieszczanie własności materiału (temperatury).

W systemie ABAQUS istnieje natomiast możliwość rozszerzania symulacji o dodatkowe procedury wprowadzane przez użytkownika, które są wywoływane przy każdym kroku obliczeniowym w czasie symulacji. Procedura, która służy ustalaniu wartości brzegowych nosi nazwę DISP.

W rozważanym eksperymentalnym modelu, w osi symetrii odlewu zachowano puste miejsce na rurkę chłodzącą. Wartości temperatury w węzłach brzegowych elementów sąsiadujących z rurką, w pierwszym przybliżeniu zostały ustawiane w trakcie symulacji na jednakową wartość, dzięki procedurze DISP. Podejście takie może być jednak uznane za przybliżone i uzasadnione tylko w określonych warunkach. Można je mianowicie dopuścić, biorąc pod uwagę zgodność z obiektem rzeczywistym, tylko wtedy gdy prędkość przepływu medium jest tak duża iż w pojedynczym kroku obliczeniowym dokonuje się przemieszczenie medium chłodzącego o całą długość rurki (wymiana medium w całej rurce).

W przypadku mniejszych wartości prędkości, konieczne jest uwzględnianie w procedurze DISP efektu nagrzewania się medium w trakcie przepływu. Dodatkowo, procedura odwzorowująca proces nagrzewania medium komplikuje się w wyniku realizacji sterowania (np. przez zmianę przepływu).

Zaproponowane w tym rozdziale rozwiązanie wykorzystuje procedurę DISP tylko w celu komunikacji z drugim systemem ABAG, który symuluje oddzielnie zjawisko dynamicznego przepływu medium chłodzącego. Dzięki temu, odwzorowane jest zjawisko złożone z wymiany ciepła pomiędzy odlewem a płynem chłodzącym, wymiany ciepła w obrębie medium chłodzącego oraz jego przemieszczenia. Dzięki wzajemnej komunikacji, współpracujące systemy ABAQUS i ABAG umożliwiają realizację zadania, które było trudne do wykonania w samym systemie ABAQUS.

4.3. System ABAG

Idea systemu ABAG polega na korygowaniu działania systemu ABAQUS w czasie symulacji. Korekcja ta ma uwzględniać wpływ schładzania wlewka za pomocą medium chłodzącego, którego dwa parametry:

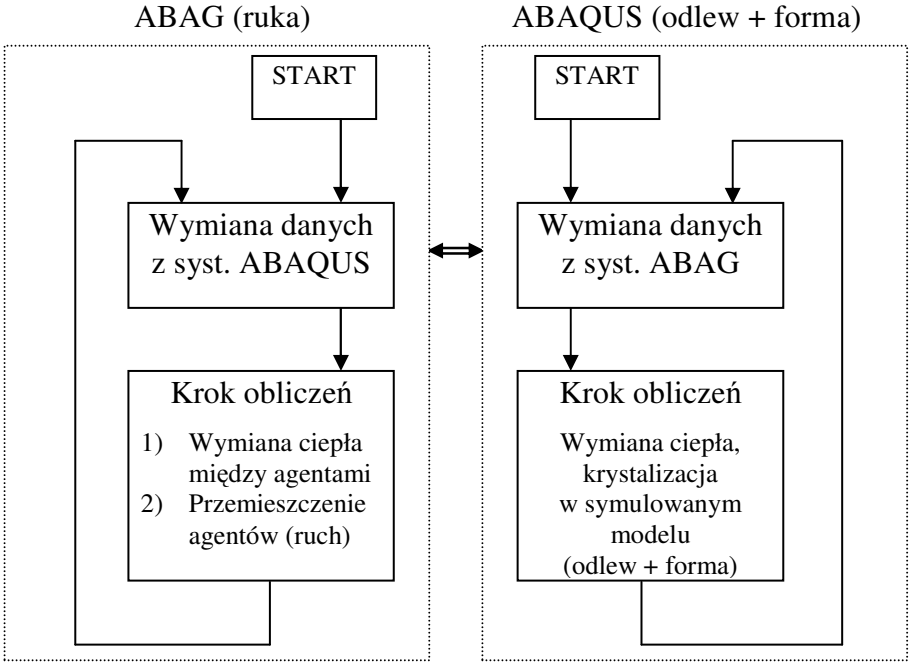
- prędkość przepływu,
- temperatura wpływającego medium,

są regulowane (ich wartości początkowe są zmieniane według zadanego algorytmu w czasie symulacji).

Proponowany model symulacyjny, wykorzystujący systemy ABAQUS i ABAG, zapewnia ich wzajemną współpracę w trakcie symulacji. Pierwszy z nich (ABAQUS) symuluje odlew z formą, w ich osi symetrii jest puste miejsce - kanał reprezentujący rurkę chłodzącą. Warunki brzegowe węzłów sąsiadujących z rurką są ustalane w kolejnych krokach symulacji systemu ABAQUS, przez wspomnianą procedurę DISP.

Drugi system (ABAG) symuluje samą rurkę. Komunikacja między procedurą DISP systemu ABAQUS a modułem komunikacyjnym systemu ABAG, odbywa się przez mechanizm soketów systemu operacyjnego. Dzięki temu, istnieje możliwość uruchamiania obydwu systemów na tym samym komputerze, lub na osobnych komputerach będących we wspólnej sieci.

Na rys. 4.4 przedstawiono schemat współpracy omawianych systemów. Przed każdym krokiem symulacji, systemy wymieniają informację (dane o parametrach miejsc sąsiadujących) między sobą.



Rys. 4.4. Współpraca systemów ABAG i ABAQUS

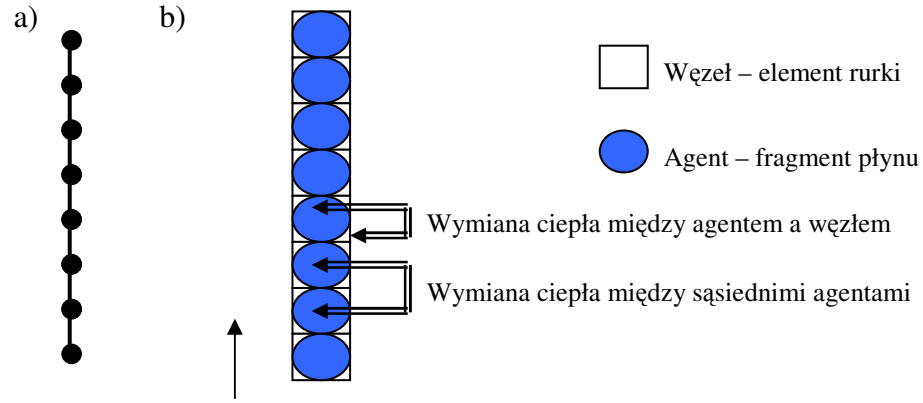
W przedstawionym na schemacie trybie współpracy systemów ABAQUS i ABAG występuje przesyłanie danych w dwóch kierunkach:

1. *Od systemu ABAQUS do systemu ABAG*. Z modelu symulacyjnego w systemie ABAQUS do systemu ABAG przekazywane są wartości temperatur w elementach brzegowych, otaczających rurkę.

2. *Od systemu ABAG do systemu ABAQUS*. Wartości temperatur w węzłach rurki chłodzącej są informacją dla systemu ABAQUS przesyłaną z systemu ABAG, które są używane przez procedurę użytkownika DISP, ustalającą warunki brzegowe w modelu symulacyjnym.

System ABAG buduje środowisko w postaci prostej liniowej siatki, w węzłach której umiejscowione są przemieszczające się agenty, reprezentujące fragmenty (porcje) płynu chłodzącego (rys. 4.5).

Wymiana ciepła następuje pomiędzy sąsiadującymi agentami oraz pomiędzy agentami a węzłami, w których się znajdują. Temperatury w węzłach reprezentują temperatury elementów odlewu sąsiadujących z rurką. Wartości ich aktualizowane są dzięki wymianie danych z modelem symulacyjnym w systemie ABAQUS.



Rys. 4.5. Siatka węzłów z agentami systemu ABAG: a – liniowa siatka węzłów reprezentujących rurkę, b – schemat agentów związanych z węzłami rurki

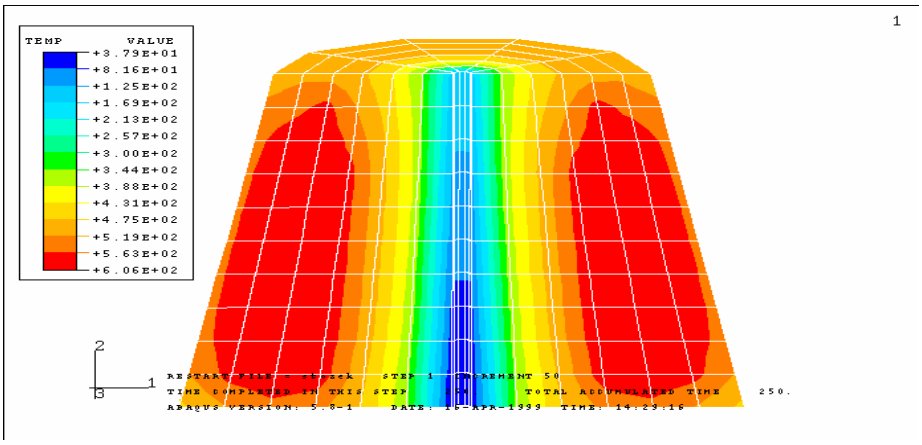
Globalnymi zmiennymi systemu ABAG są:

- V(t) – prędkość przemieszczania się agentów (przepływu płynu chłodzącego),
- T(t) – temperatura wpływającego medium (stan agentów wejściowych).

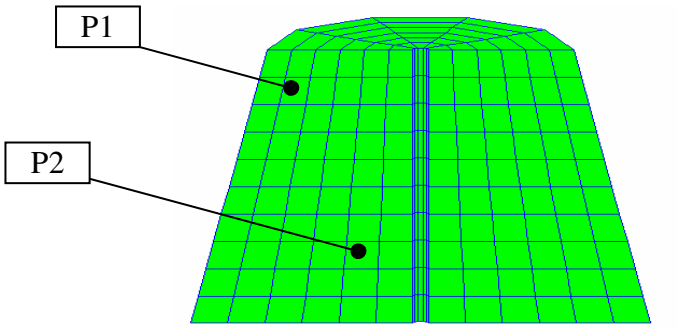
Przeprowadzono eksperyment symulacyjny, w którym w całym przebiegu symulacji, temperatura T(t) oraz prędkość V(t) miały stałą wartość.

Wpływ rurki chłodzącej jest widoczny w wynikach symulacji podanych na rys. 4.6. Wpływ ten uwidacznia się nie tylko w obszarach otaczających rurkę, ale również w bardziej odległych częściach odlewu.

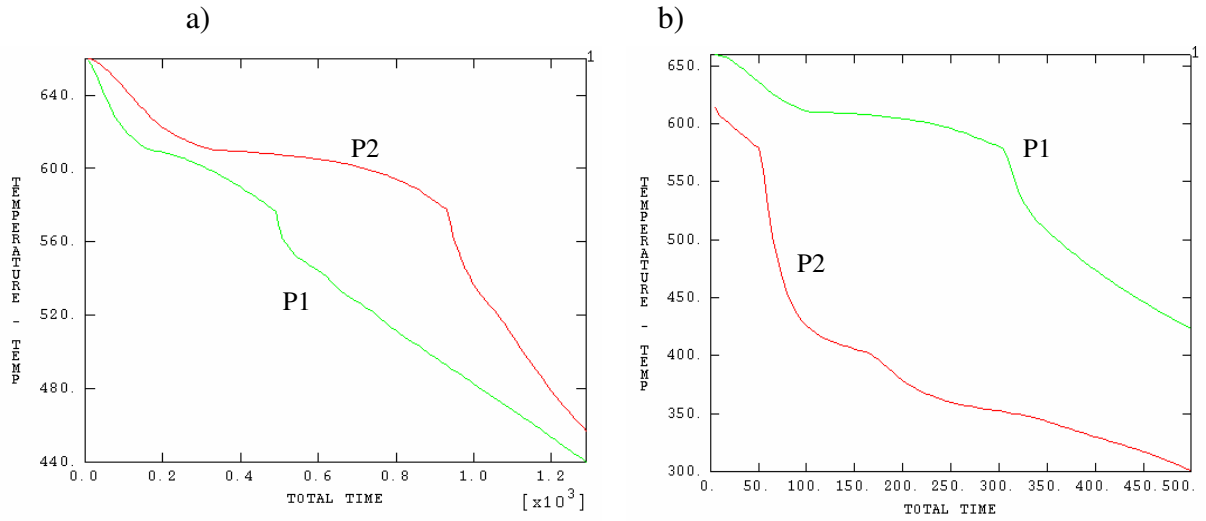
W celu ilościowego przedstawienia wpływu rurki chłodzącej, wybrano dwa elementy siatki odlewu, pokazane na rys. 4.7. Przebiegi temperatur w tych punktach, dla symulacji z rurką i bez rurki chłodzącej, zostały pokazane na rys. 4.8.



Rys. 4.6. Rozkład temperatury w odlewie
po 250 sekundach symulacji z rurką chłodzącą (ABAG)



Rys. 4.7. Dwa elementy (P1, P2) odlewu wybrane dla dokonywania porównania



Rys. 4.8. Porównanie zmian temperatur w dwóch elementach siatki odlewu:
a) symulacja bez rurki, b) symulacja z rurką

Analizując otrzymane rezultaty można dokonać następującej obserwacji:

a) dla symulacji bez rurki, temperatura w elemencie P1 w każdym kroku symulacji jest mniejsza niż w elemencie P2 (rys. 4.8.a),

b) po wprowadzeniu rurki chłodzącej w każdym kroku symulacji temperatura w elemencie P2 jest mniejsza niż w elemencie P1 (rys. 4.8.b) – czyli zachodzi sytuacja odwrotna w stosunku do punktu a).

Z powyższych eksperymentów symulacyjnych wynika, że powinien istnieć algorytm sterujący, który wykorzystując oddziaływanie na proces schładzania, poprzez zdolności regulacyjne rurki (T(t), V(t)), doprowadziłby do wyrównania (a przynajmniej zbliżenia) przebiegu temperatur w tych dwóch elementach (np. zmniejszając prędkość przepływu, lub wyłączając całkowicie przepływ – sterowanie dwupołożeniowe).

4.4. Podsumowanie

Przedstawiono podejście, w którym model symulacyjny oparty jest na współpracy dwóch systemów: ABAQUS i ABAG. System agentowy ABAG wspomaga działanie systemu ABAQUS likwidując ograniczenia tego systemu. Metoda taka może być przydatna w sytuacji, gdy dysponujemy gotowym, sprawdzonym systemem (np. ABAQUS), lecz nie realizuje on (w danej wersji) specyficznych wymagań. Wówczas system agentowy pełni rolę uzupełniającą dla tego systemu podstawowego.

W systemie ABAG przemieszczające się agenty modelują ruch i wymianę ciepła, natomiast środowisko, w którym działają te agenty, spełnia rolę interfejsu do systemu elementów skończonych ABAQUS.

Możliwe jest rozszerzenie tej metody, gdzie zadaniem systemu agentowego byłaby integracja wielu gotowych systemów symulacyjnych, z których każdy wyróżnia się pewną specyfiką w modelowaniu wybranego fragmentu (aspektu) złożonego procesu fizycznego.

Przedstawione rozwiązanie, rozszerzające znacznie możliwości podstawowego systemu (w przedstawionym przypadku systemu ABAQUS) nie jest pozbawione pewnych wad, do których należą:

- konieczność korzystania z więcej niż jednej platformy symulacyjnej,
- konieczność zapewnienia współdziałania systemów,
- wydłużenie czasu obliczeń symulacyjnych, związane z potrzebą komunikacji między systemami,
- limitowane przez konstrukcję systemu podstawowego (np. ABAQUS) możliwości rozszerzania jego funkcji.

5. System agentowy bez modelowanego środowiska

5.1. Wymagania ogólne

Modele analityczne oparte na teoretycznych rozważaniach opisujące proces krzepnięcia, a szczególnie krystalizacji metali i ich stopów, nie są w stanie odwzorować wspomnianych procesów w przypadku bardziej zaawansowanych metod odlewania.

Dotyczy to m.in. odlewania skomplikowanych kształtów z miedzi lub jej stopów, przy wysokich wymaganiach jakości odlewów (takich jak dysze wielkopiecowe lub elementy dla elektroenergetyki). Konieczne jest przy tym zapewnienie właściwej mikro i makrostruktury, posiadających istotny wpływ na możliwości występowania wad (np. rzadzizny, pęcherze).

W takim przypadku można modelować wspomniane procesy i przewidywać cechy otrzymanych rezultatów wyłącznie na drodze symulacji komputerowej.

Pomimo, że istnieje już znaczna ilość narzędzi informatycznych, większość dostępnych systemów do symulacji procesów schładzania i krzepnięcia odlewów nie uwzględnia wszystkich zjawisk związanych z tym zagadnieniem (ABAQUS, MAGMA, NASTRAN).

W związku z tym, zaproponowano odmienne podejście do tworzenia tej klasy systemów, zrealizowane w wersji badawczej w postaci systemu MAFES-1.

Tworząc nowy system do symulacji procesów odlewniczych można wskazać następujące cechy, jakie powinien taki system posiadać:

a) wykorzystanie metody obliczeniowej, która minimalizuje i w jak najmniejszym stopniu uzależnia błąd obliczeń od: rozmiaru siatki, kształtu jej elementów, warunków początkowych i zmian warunków brzegowych w trakcie symulacji,
b) możliwość wprowadzania do modelu symulacji dodatkowych dynamicznych elementów reprezentujących działania sterujące – najczęściej w postaci dodatkowego chłodzenia,
c) umożliwienie sterowania procesem schładzania poprzez zmiany parametrów fizycznych elementów wspomnianych w punkcie b), które dotyczą takich parametrów jak: prędkość przepływu, temperatura wpływającego medium,
d) modelowanie w procedurze obliczeniowej procesu krystalizacji w sposób uwzględniający wszystkie najważniejsze właściwości tego zjawiska, w szczególności efekt przechłodzenia w trakcie krystalizacji. Ważne jest by system posiadał parametryzowaną bibliotekę podstawowych modeli krystalizacji, która będzie otwarta na wprowadzanie nowych modeli (np. dla nowych materiałów),

e) powinna być zachowana szeroko rozumiana otwartość systemu na możliwość odwzorowania w nim problemów pojawiających się w praktyce symulacyjnej (co ma miejsce gdy chcemy coraz dokładniej odwzorowywać rzeczywistość).

Należy przypomnieć, że zasadniczy tok prowadzonych w pracy rozważań (zgodny z jej tezą) dotyczy stworzenia koncepcji i pilotowej realizacji modelu symulacyjnego spełniającego powyższe wymagania.

Z praktycznego punktu widzenia, można przewidywać kilka wariantów wykorzystania tego modelu.

Klasycznym zadaniem jest poszukiwanie najkorzystniejszych rozwiązań projektowych dotyczących kształtu i wymiarów formy, konfiguracji nadlewu, warunków początkowych i brzegowych. Ten typ zastosowań realizowany jest aktualnie przez istniejące (wspomniane powyżej) systemy symulacyjne.

Bardziej zaawansowane zastosowania dotyczą badania oddziaływań o charakterze dynamicznym. Do takich właśnie zadań adresowane jest proponowane rozwiązanie (system MAFES-1). W tej grupie zastosowań wyodrębnić można:

- badania symulacyjne różnych sposobów oddziaływań dynamicznych, kształtujących przebieg procesu stygnięcia i krystalizacji, mające na celu wybór odpowiednich rozwiązań projektowych (symulacja a priori),
- eksperymenty symulacyjne dotyczące możliwości sterowania w czasie rzeczywistym, to jest bieżącego oddziaływania na przebieg realnego procesu.

W dalszej części tego rozdziału przedstawiono założenia i podstawy formalne, na których oparto koncepcję uproszczonego modelu matematycznego procesu krzepnięcia oraz jego realizacji w postaci modelu symulacyjnego z wykorzystaniem technologii agentowej. Do proponowanego rozwiązania dostosowano odpowiednie procedury matematyczne, które zweryfikowane zostały w oparciu o serię eksperymentów symulacyjnych.

5.2. Cel i założenia systemu symulacyjnego MAFES-1

Celem stworzenia nowego systemu była potrzeba uwzględniania coraz większej liczby zjawisk zachodzących w procesach odlewniczych i potrzeba sterowania nimi (np. za pomocą dodatkowych dynamicznych elementów chłodzących).

Szereg eksperymentów przeprowadzonych w oparciu o system ABAQUS [34, 50, 51] pozwolił ustalić granice jego możliwości.

Trzy przedstawione poniżej ograniczenia dotyczące tego systemu, stały się przyczyną prac nad nowym rozwiązaniem:

a) mała możliwość ingerencji w działanie wewnętrznych mechanizmów, związana z jego komercyjnym charakterem,
b) uproszczony, jednolity model procesu krystalizacji, opisywany trzema parametrami: temperaturą solidus, temperaturą liquidus oraz ciepłem krystalizacji, co powoduje, że niektóre zjawiska (istotne dla procesu odlewniczego) zachodzące w trakcie tego procesu nie mogą być uwzględniane,
c) trudna i ograniczona możliwość modelowania elementów dynamicznych, opisana w rozdziale 4 (oparta na procedurze użytkownika DISP).

Dążąc do wyeliminowania tych niedogodności, opracowano badawczą wersję nowego systemu symulacyjnego o nazwie MAFES-1 (ang. Multi Agent Finite Element System).

Koncepcja konstruowanego systemu MAFES-1 różni się w sposób istotny od innych systemów symulacyjnych (ABAQUS, MAGMA), poprzez zastosowanie technologii agentowej, która stanowi jeden z kierunków rozwojowych systemów informatycznych.

Nowe możliwości jakie stwarza proponowane rozwiązanie dotyczą w szczególności większej precyzji modelowania procesu krystalizacji oraz możliwości oddziaływania na kształtowanie struktury krystalicznej, poprzez dostosowanie (dynamiczne zmiany) warunków chłodzenia.

Podstawową, charakterystyczną cechą proponowanego rozwiązania (która uzasadnia także nazwę systemu: Multi Agent Finite Element System) jest wprowadzenie trzech typów agentów:

- Pierwszy typ – *Agent Obliczeniowy* (**AO**). Każdy z agentów tego typu reprezentuje jeden element (Finite Element), z których składa się model. Agent Obliczeniowy przechowuje stan tego elementu oraz realizuje w nim zjawiska zachodzące (w rozważanym przypadku wymianę ciepła i krystalizację).
- Dugi typ agentów – *Agent Regulator* (**AR**), służy do reprezentacji dynamicznych elementów modelu, najczęściej układów chłodzących (np. rurek z medium chłodzącym). Agent ten nadzoruje grupę agentów obliczeniowych, tych które odpowiadają elementom modelu reprezentującym dany obiekt dynamiczny oraz realizuje decyzje sterujące. Posiada właściwości dynamiczne reprezentujące ruch – przemieszczanie się medium chłodzącego. Ten typ agentów można rozważać jako agentów drugiego, wyższego poziomu, zmieniających własności swojej grupy agentów obliczeniowych, kształtując je zgodnie z dynamiką ruchu.
- Trzeci typ agentów – *Agent Sterujący* (**AS**) – to kolejny, wyższy poziom agentów, które wpływają na zachowanie agentów regulatorów. Agenty typu AS podejmują decyzje (realizowane przez AR) sterujące dążące do osiągnięcia wyznaczonych im celów sterowania. Aby to wykonać muszą stale obserwować stan agentów obliczeniowych, tych

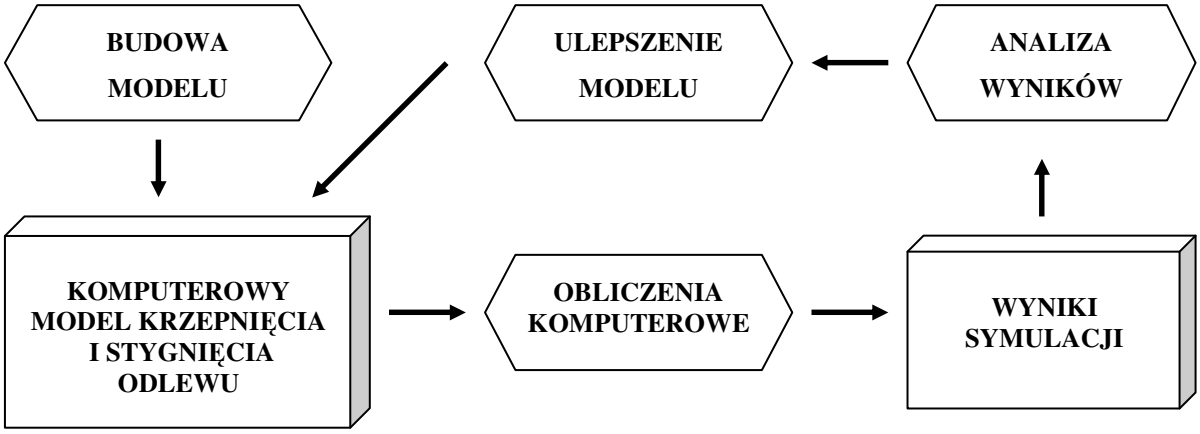
elementów modelu, których własności podlegają ocenie przy wyznaczaniu decyzji sterujących.

Przedstawiany system umożliwia realizację następujących funkcji (które w większości innych systemów są trudne do wprowadzenia):

a) Modelowanie przepływu płynów (media chłodzące jako istotne dodatki do klasycznego układu odlew-forma).
b) Symulacja procesów krystalizacji uwzględniająca możliwie dokładne, znane modele przemian faz *liquidus – solidus*.
c) Sterowanie dynamicznym elementem chłodzącym podczas symulacji zgodnie z wyznaczonym celem sterowania.
d) Sterowanie warunkami początkowymi procesu odlewniczego przeprowadzane w serii eksperymentów symulacyjnych. Wyniki otrzymane z dotychczasowych eksperymentów tej serii służą do zmiany konfiguracji warunków początkowych modelu symulacyjnego, dla kolejnego eksperymentu symulacyjnego. Zmiana konfiguracji warunków początkowych ma na celu uzyskanie pożądanych wyników symulacyjnych.
e) Możliwość modelowania nowych zjawisk i zadań pojawiających się w związku z rozwojem technologii odlewniczych, które są trudne do realizacji w większości systemów symulacyjnych ze względu na ich zamkniętość, związaną z ich komercyjnym charakterem i dążeniem do modelowania szerokiej gamy zjawisk fizycznych.
f) Modelowanie czujników temperatury (np. termopary) dających zniekształcony obraz rzeczywistych zmian temperatur w badanych obszarach. Przy weryfikacji wyników symulacji, wyniki z eksperymentów fizycznych (odczytane np. z termopary) porównywane są z wynikami symulacji uzyskanymi na wyjściu symulowanej termopary. Zgodność tych wyników pozwala na uznanie wyników symulacji będących sygnałem wejściowym dla termopary, za wyniki rzeczywiste. W przypadku braku modelowanych czujników temperatury, w celu porównania wyników eksperymentalnych i symulacyjnych konieczne jest opracowanie metody pozwalającej otrzymać wyniki rzeczywiste na podstawie wyników odczytanych z czujnika. Dopiero po otrzymaniu tych wyników, można je porównywać z wynikami symulacji. Ta ostatnia funkcja związana jest z konkretną realizacją eksperymentu fizycznego i nie będzie szerzej rozważana w dalszym ciągu pracy.

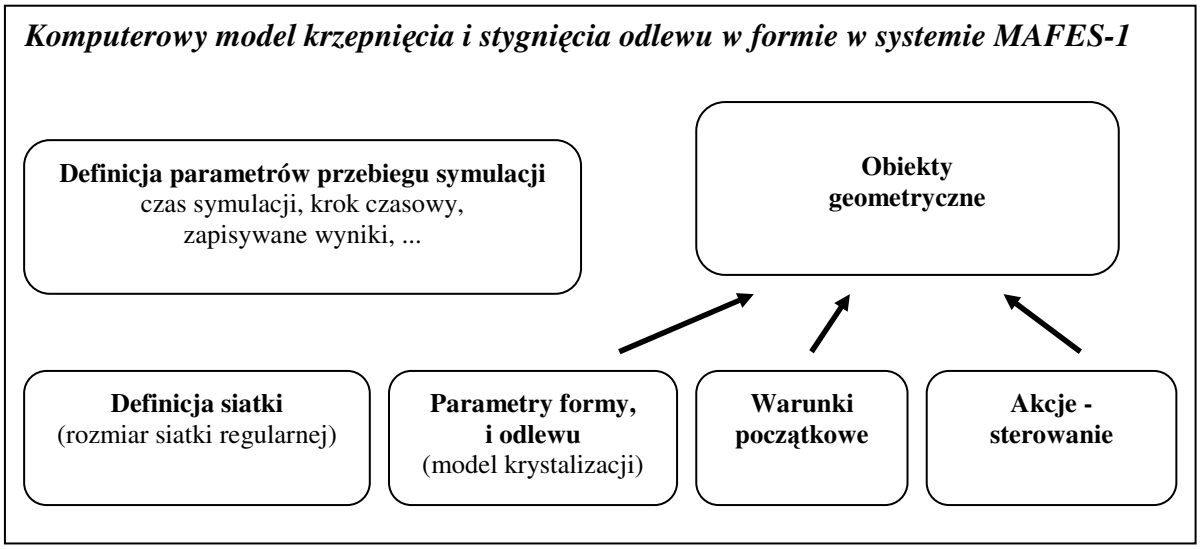
Ogólny schemat koncepcyjny, ilustrujący metodykę tworzenia, udoskonalania i wykorzystania modelu komputerowego przedstawiony został na rys. 5.1. Schemat ten przyjęto za podstawę realizacji systemu MAFES-1.

Wyniki obliczeń symulacyjnych muszą podlegać weryfikacji - porównaniu z wynikami uzyskiwanymi z eksperymentów fizycznych. Wszelkie niezgodności są podstawą dla szukania coraz to lepszego modelu komputerowego procesów krzepnięcia i stygnięcia.



Rys. 5.1. Organizacja analizy komputerowej procesu krzepnięcia i stygnięcia

Schemat struktury modelu wejściowego procesu krzepnięcia i stygnięcia odlewu w formie, realizowanego za pomocą systemu MAFES-1 przedstawiono na rys. 5.2.



Rys. 5.2. Model wejściowy procesu krzepnięcia i stygnięcia odlewu w systemie MAFES-1

Można wyróżnić następujące komponenty, wykorzystane przy realizacji powyższego modelu:

- narzędzia umożliwiające tworzenie obiektów geometrycznych (takich jak prostopadłościany, stożki, itp.),
- metody umożliwiające ustalenie własności materiałów tworzonych obiektów (takich jak gęstość, ciepło właściwe, itp.),
- metody umożliwiające ustalenie warunków początkowych dla tworzonych obiektów (np. temperatura początkowa),

- metody umożliwiające ustalenie parametrów siatki, za pomocą której modelowana jest struktura tworzonych obiektów,
- metody umożliwiające ustalenie podstawowych parametrów procesu symulacji (takich jak czas symulacji, krok czasowy, wybór parametrów do rejestracji wyników i określenie sposobu rejestracji, itp.).

5.3. Agent Obliczeniowy

Zadaniem agenta obliczeniowego jest reprezentacja elementu skończonego (Finite Element) przestrzeni fizycznej modelowanego obszaru. Reprezentacja ta dotyczy zarówno stanu jak i dynamiki zmian tego stanu, związanej z zachodzącymi w tym elemencie zjawiskami (np. wymiana ciepła i krystalizacja).

Opis działania agenta obliczeniowego, wymaga odniesienia do modeli matematycznych procesu wymiany ciepła i krystalizacji, zachodzących w stygnącym odlewie.

Matematyczny model procesu wymiany ciepła i jego realizacja przez agenta obliczeniowego

Poniżej przedstawiono uproszczony, matematyczny model procesu stygnięcia, i jego komputerową realizację przez jedną z funkcji agenta obliczeniowego.

Proces wymiany ciepła w środowisku określa równanie różniczkowe cząstkowe typu parabolicznego (dla niezmiennych od temperatury własności termofizycznych):

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}-\frac{1}{a^2}\frac{\partial T}{\partial t}=0 \tag{5.1}$$

$$a^2=\frac{k}{cg} \tag{5.2}$$

gdzie:

- T = T(x, y, z, t) – temperatura jako funkcja współrzędnych przestrzennych x, y, z i czasu t,
- a – współczynnik grupujący własności termofizyczne,
- k – przewodność cieplna,
- c – ciepło właściwe,
- g – gęstość materiału.

Równanie (5.2) może być przekształcone do postaci różnicowej:

$$\frac{T(x_{i+1})-2T+T(x_{i-1})}{h^2}+\frac{T(y_{j+1})-2T+T(y_{j-1})}{h^2}+\frac{T(z_{k+1})-2T+T(z_{k-1})}{h^2}-\frac{1}{a^2}\frac{T(t_{m+1})-T}{\Delta t}=0 \tag{5.3}$$

gdzie x_i, y_j, z_k, i t_m mogą być przedstawione w postaci dyskretnej następująco:

$x_i = i * \Delta x$ (dla i = 1, ... , N_x) – kolejne dyskretne wartości współrzędnej x,
$y_j = j * \Delta y$ (dla j = 1, ... , N_y) – kolejne dyskretne wartości współrzędnej y,
$z_k = k * \Delta z$ (dla k = 1, ... , N_z) – kolejne dyskretne wartości współrzędnej z,
$t_m = m * \Delta t$ (dla m = 0, 1, ..., N_t) – kolejne dyskretne wartości czasu t,
$\Delta x = \Delta y = \Delta z = h$

Liczby N_x , N_y , N_z oznaczają ilości węzłów siatki różnicowej w kierunkach X, Y, Z. Przyjęto jednakową odległość między węzłami tej siatki równą h. Wartość N_t reprezentuje ilość kroków badanego przedziału czasowego, natomiast Δt to wartość tego kroku.

Dla przejrzystości zapisu przyjęto uproszczenie, że brak oznaczenia któregoś z argumentów funkcji T(x, y, z, t) oznacza, że są one równe x_i , y_j , z_k lub t_m . Zatem samodzielnie występujący symbol T oznacza $T(x_i , y_j , z_k , t_m)$.

Równanie (5.3) można upraszczać w celu zminimalizowania kosztu obliczeń łącząc te same składniki. Można również tak uprościć to równanie by pokazać pewne zależności między sąsiednimi węzłami:

$$T(t_{m+1}) = T + \Delta T \tag{5.4}$$

gdzie:

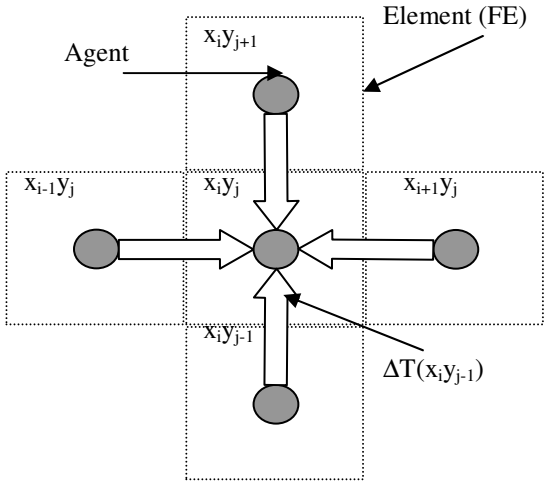
$$\Delta T = \Delta T(x_{i+1}) + \Delta T(x_{i-1}) + \Delta T(y_{j+1}) + \Delta T(y_{j-1}) + \Delta T(z_{k+1}) + \Delta T(z_{k-1}) \tag{5.5}$$

$$\Delta T(x_{i+1}) = b * [\, T(x_{i+1}) - T \,] , \ldots \tag{5.6}$$

$$b = \frac{a^2 \Delta t}{h^2} \tag{5.7}$$

Przyrost temperatury T w punkcie (x_i, y_j, z_k) – równanie (5.4), jest sumą sześciu składników – równanie (5.5). Każdy z nich – równanie (5.6), zależy od różnicy pomiędzy temperaturą w badanym punkcie a temperaturą w jednym z sześciu sąsiednich punktów oraz współczynnika proporcjonalności b – określonego przez parametry termofizyczne, rozmiar siatki i wartość kroku czasowego – równanie (5.7).

Przyjmijmy, że w każdym z tych punktów umiejscowiony jest agent obliczeniowy. Ma on pewne atrybuty (np. temperatura) i funkcje, które operują na tych atrybutach. Proces wymiany ciepła pomiędzy sąsiednimi agentami obliczeniowymi, w sposób schematyczny pokazano na rys. 5.3 (ze względu na czytelność rysunku ograniczono się do dwu wymiarów).



Rys. 5.3. Schemat wymiany ciepła pomiędzy sąsiednimi agentami obliczeniowymi (przekrój dwuwymiarowy)

Dla siatki regularnej o dwóch wymiarach każdy kwadrat (długość boku równa h), będący centralnym otoczeniem badanych punktów, ma czterech sąsiadów (w trzech wymiarach ich liczba wynosi sześć). Położeniem agenta będzie środek takiego kwadratu w 2D (w 3D sześcianu). Agent reprezentuje nie tylko ten punkt lecz cały kwadrat (sześcian) czyli pewien skończony element (Finite Element - FE). Dany agent obliczeniowy wymienia ze swoimi sąsiadami ciepło, proporcjonalne do różnicy temperatur między nimi i zależne od parametrów fizycznych środowiska w którym się znajdują.

Wychodząc od równania różniczkowego (5.1), opisującego prawo fizyczne wymiany ciepła dla pewnego obszaru, dochodzimy do oddziaływań pomiędzy lokalnymi obszarami, elementami reprezentowanymi przez agentów obliczeniowych (równania (5.4) – (5.7)). Pomiędzy sąsiednimi agentami tego typu występuje interakcja w postaci wymiany ciepła.

Matematyczny model procesu krzepnięcia i jego realizacja przez agenta obliczeniowego

Ważnym elementem symulacji procesów cieplnych jest modelowanie procesu wydzielania ciepła krystalizacji w trakcie powstawania fazy stałej w stygnącym odlewie. Równanie przewodzenia ciepła, zawierające składnik uwzględniający powstawanie źródeł ciepła związanych z tworzącymi się i rosnącymi kryształami, przedstawiono poniżej (równanie (5.8)). Rozważany jest model krystalizacji – równowagi eutektycznej, opisany równaniami (5.9) – (5.12).

$$\frac{\partial T}{\partial t} = \frac{k}{c\,g}\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right)+\frac{L_H}{c}\frac{\partial F_S}{\partial t} \tag{5.8}$$

$$\Delta T(t) = T_E - T(t) \tag{5.9}$$

$$N(t) = \Psi \Delta T(t)^2 \tag{5.10}$$

$$\frac{dR}{dt} = \mu \Delta T(t)^2 \tag{5.11}$$

$$F_S(t) = nN(t)\frac{4}{3}\pi R^3(t) \tag{5.12}$$

gdzie:

L_H – ciepło krystalizacji,

F_S – udział fazy stałej (wartość rzeczywista od 0 do 1),

T_E – temperatura eutektyki,

N – ilość kryształów w jednostce objętości,

R – średni promień kryształów

Wielkości F_S, N, R są funkcjami współrzędnych x, y, z oraz czasu t. Współczynniki Ψ, μ, n to stałe dobierane doświadczalnie.

Obliczanie wartości N następuje do momentu zwanego *przechłodzeniem*, czyli do pojawienia się efektu wzrostu temperatury, wynikającego z przewagi ilości wydzielanego ciepła wskutek krystalizacji nad ciepłem oddawanym otoczeniu danego punktu.

Z równań (5.8) – (5.12) wynika, że agent obliczeniowy wyznaczając nową temperaturę $T(t_{m+1})$ powinien uwzględniać nie tylko środowisko otaczające w postaci sąsiadujących z nim agentów – ich temperatury, lecz również wartość ΔF_S – opisującej przyrost (lub ubytek) fazy stałej dla przedziału czasowego $\Delta t = t_{m+1} - t_m$. Zachowanie to dotyczy obszarów, w których może zachodzić krystalizacja, czyli wartość ciepła krystalizacji L_H jest większa od zera.

Oznaczając zmianę temperatury wynikłą ze zjawiska wymiany ciepła (równanie (5.5)) przez ΔT_1, natomiast zmianę temperatury związaną z krystalizacją przez ΔT_2, zmianę atrybutu temperatura agenta obliczeniowego (ΔT_{AO}) w pojedynczym kroku symulacji, można wyrazić przez sumę tych dwóch przyrostów:

$$\Delta T_{AO} = \Delta T_1 + \Delta T_2 \tag{5.13}$$

gdzie zgodnie z równaniem (5.8) drugi przyrost wynosi:

$$\Delta T_2 = a_2 \Delta F_S \tag{5.14}$$

W powyższym równaniu ΔF_S oznacza przyrost atrybutu agenta obliczeniowego, będącego współczynnikiem udziału fazy stałej, w pojedynczym kroku symulacji. Wartość ta jest zależna od zmian parametrów N i R (równanie (5.12)).

Atrybuty agenta obliczeniowego

Podstawowy agent systemu MAFES-1 – agent obliczeniowy, reprezentuje między innymi stan odpowiadającego mu elementu przestrzeni badanego obszaru.

Ze względu na rozpatrywaną w pracy dziedzinę procesów cieplnych, stan ten określony jest przez zestaw atrybutów - parametrów termofizycznych:

a) położenie w środowisku,
b) parametry termofizyczne materiału (przewodność cieplna, ciepło właściwe, gęstość),
c) dodatkowe parametry termofizyczne związane z realizowanym modelem procesu krystalizacji (temperatura eutektyki, ciepło krystalizacji)
d) parametry obliczane podczas symulacji: temperatura T, ilość kryształów N, średni promień kryształów R, współczynnik udziału fazy stałej F_S, flaga CGN oznaczająca czy nastąpiło przechłodzenie w modelu krystalizacji – równowagi eutektycznej,
e) lista sąsiadujących z nim agentów obliczeniowych.

Funkcje agenta obliczeniowego

Podstawowymi zadaniami agenta obliczeniowego są:

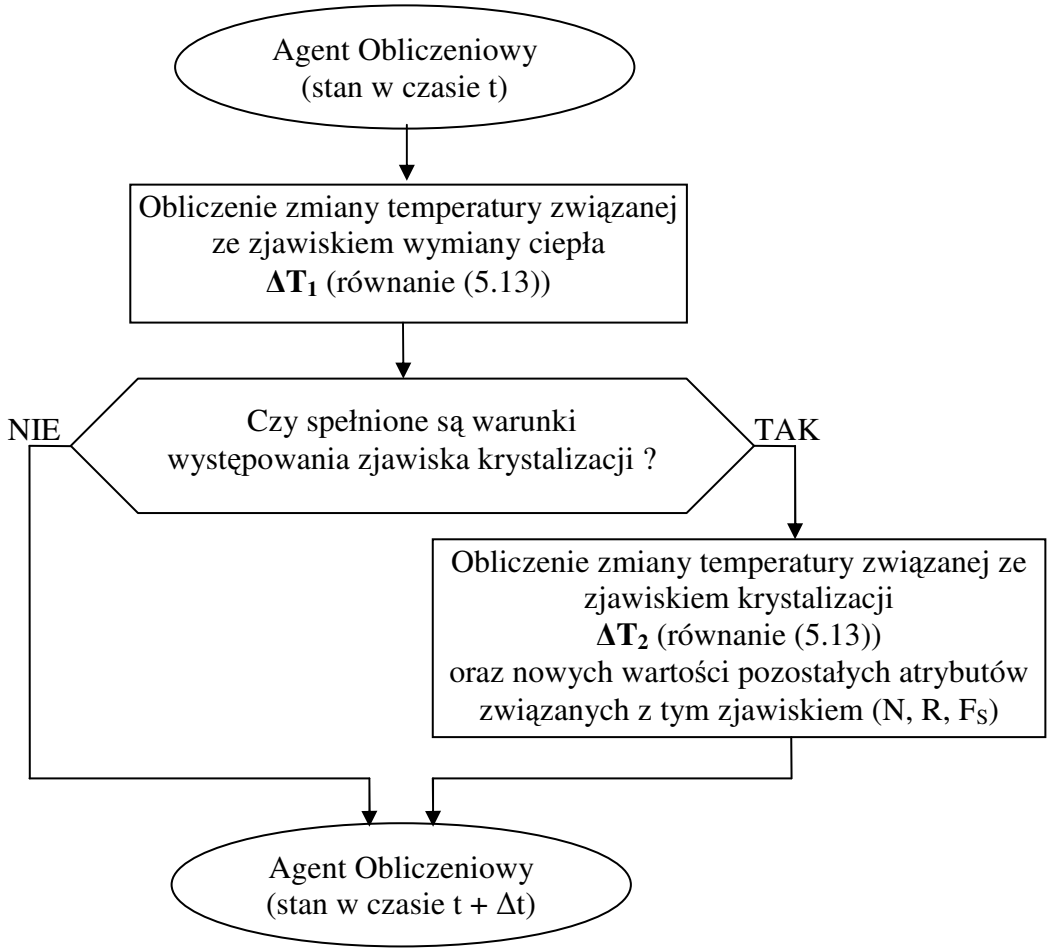
a) **Inicjalizacja** – ustala następujące dane agenta:
   - współrzędne położenia agenta zgodnie z wyznaczonym dla niego miejscem (środek reprezentowanego elementu – elementarnego sześcianu),
   - wartości atrybutów termofizycznych agenta, które są określone przez parametry obiektu geometrycznego wewnątrz którego został on umieszczony,
   - wartości obliczanych atrybutów agenta są ustalane w oparciu o wartości początkowe.
b) **Wyznaczenie nowego stanu** – ma na celu obliczenie wszystkich dynamicznych atrybutów agenta obliczeniowego w nowym kroku dyskretnego czasu obliczeń.

Algorytmy realizujące zjawiska występujące w elemencie reprezentowanym przez agenta obliczeniowego wykorzystują stan danego agenta i stany agentów sąsiednich z poprzedniej iteracji. Procedurę tę przedstawiono schematycznie na rys. 5.4.

Obliczenie zmiany temperatury wynikłej ze zjawiska wymiany ciepła wymaga komunikacji z sąsiednimi agentami, w celu uzyskania ich bieżącej wartości temperatury.

Warunkiem występowania zjawiska krystalizacji jest większa od zera wartość ciepła krystalizacji L_H , obniżenie się wartości temperatury poniżej temperatury eutektyki T_E oraz mniejsza od jedności wartość współczynnika F_S.

Nowy stan agenta obliczeniowego dla elementów, w których nie występuje zjawisko krystalizacji, związany jest ze zmianą temperatury $\mathbf{\Delta T_1}$ – wynikającą ze zjawiska wymiany ciepła. Natomiast w przypadku występowania w danym elemencie zjawiska krystalizacji zmiana stanu agenta obliczeniowego dotyczy atrybutu temperatura (zmiana o wartość $\mathbf{\Delta T_1 + \Delta T_2}$) oraz atrybutów związanych ze zjawiskiem krystalizacji (N, R, F_S).



Rys. 5.4. Algorytm wyznaczenia nowego stanu agenta obliczeniowego

5.4. Agent Regulator

Omawiany tutaj rodzaj agentów przeznaczony jest dla reprezentacji w modelu symulacyjnym obiektów związanych z dynamicznymi elementami chłodzącymi (np. rurek z medium chłodzącym). Należy podkreślić, że słowo „regulator" nie jest interpretowane tutaj zgodnie z teorią sterowania. W prezentowanym modelu agent regulator spełnia rolę elementu wykonawczego.

Agent regulator nadzoruje grupę agentów obliczeniowych znajdujących się w obszarze geometrycznym, w którym przemieszcza się medium chłodzące. Parametry tego agenta ustalane są przez agenta sterującego.

Atrybuty agenta regulatora

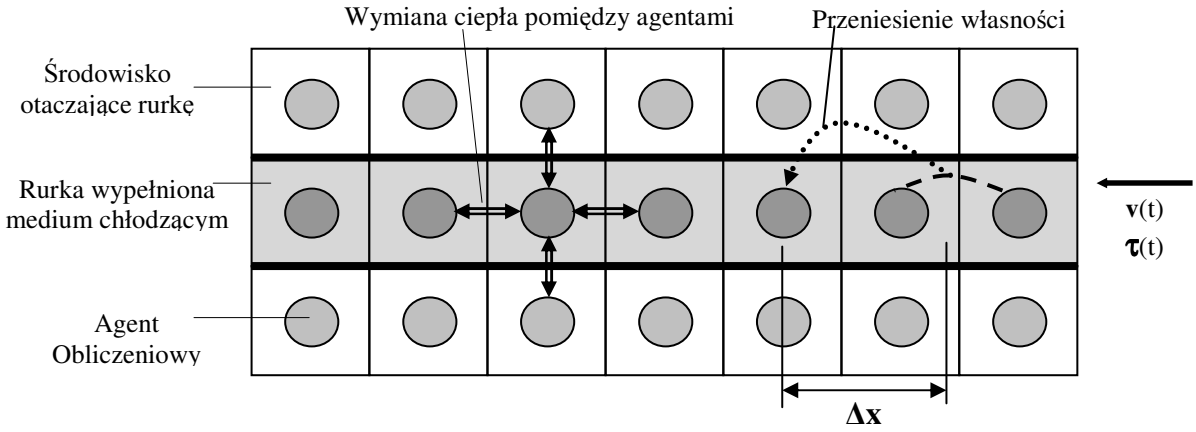
W rozważanym przypadku atrybutami agenta regulatora są:

a) prędkość przepływu medium chłodzącego – **v**(t),

b) temperatura wpływającego płynu chłodzącego – $\tau(t)$.

Funkcje agenta regulatora

a) odczyt decyzji sterujących przekazywanych od agenta sterującego, w celu ustalenia zmian wartości atrybutów danego agenta regulatora,
b) przemieszczenie własności wyznaczanych przez agentów obliczeniowych obejmowanych działaniem tego agenta, zgodnie z parametrami układu chłodzącego (np. $\mathbf{v}(t)$, $\tau(t)$).



Rys. 5.5. Agent regulator reprezentujący zbiór odpowiednich agentów obliczeniowych oraz parametry medium chłodzącego (jako działania sterujące) – uproszczenie dwuwymiarowe

Rys. 5.5 przedstawia przykładową, trójwarstwową siatkę agentów obliczeniowych, z których druga warstwa podlega działaniu agenta regulatora. Dla wybranego agenta obliczeniowego zaznaczono podwójną linią proces wymiany ciepła z sąsiednimi agentami obliczeniowymi.

Dla innego agenta obliczeniowego (rys. 5.5) zaznaczono linią przerywaną przeniesienie własności dokonywane przez agenta regulatora, dla przemieszczenia $\Delta\mathbf{x}$, związanego z ruchem medium chłodzącego o prędkości $\mathbf{v}(t)$. Przemieszczenie $\Delta\mathbf{x}$ w tym przypadku nie jest wielokrotnością odległości między węzłami siatki. Powoduje to, że przenoszone własności muszą być odpowiednio uśrednione na podstawie atrybutów dwóch agentów obliczeniowych, pomiędzy którymi znajduje się punkt źródłowy przesunięcia.

Agenty regulatory są wywoływane w procesie obliczeniowym systemu MAFES-1 po obliczeniu nowych stanów wszystkich agentów obliczeniowych i następującej po nim analizie bieżącej sytuacji przez agenta sterującego.

W bloku inicjalizacji procesu obliczeniowego, tworzenie agentów regulatorów następuje po skonstruowaniu wszystkich agentów obliczeniowych, poprzedza tworzenie agentów sterujących.

Stan agenta regulatora

Poniżej zapisano symbolicznie przejście między dwoma kolejnymi stanami agenta regulatora:

$$\mathbf{SAR}_{t+\Delta t} = \boldsymbol{m}\ (\ \boldsymbol{s}\ (\ \mathbf{SAR}_t\ )\ ) \tag{5.15}$$

gdzie:

$\mathbf{SAR}_t$, $\mathbf{SAR}_{t+\Delta t}$ – odpowiednio, aktualny i następny stan agenta regulatora;

$\boldsymbol{m}$ – operator reprezentujący zmianę atrybutów agentów obliczeniowych dokonywaną przez agenta regulatora.

W wyrażeniu tym $\boldsymbol{s}$ symbolizuje zmianę stanu agentów obliczeniowych, nadzorowanych przez agenta regulatora, w kroku obliczeniowym t całego systemu (zmiana realizowana przez agenty obliczeniowe). Natomiast operator $\boldsymbol{m}$ określa zmianę atrybutów tych agentów obliczeniowych (związaną z ruchem medium chłodzącego).
Stan agenta regulatora można również opisać w postaci wyrażenia:

$$\mathbf{SAR}_t = \{\ \{\mathbf{SAO}_{i,\,t} \text{ dla } i = 1 \ldots N_R\ \}\ ,\ \mathbf{v}\ ,\ \boldsymbol{\tau}\ \} \tag{5.16}$$

Tutaj, stan $\mathbf{SAR}_t$ zostaje określony przez stany wszystkich agentów obliczeniowych $\mathbf{SAO}_{i,\,t}$ podległych temu regulatorowi (w liczbie N_R), oraz parametry układu chłodzącego (np. $\mathbf{v}$, $\boldsymbol{\tau}$).

5.5. Agent Sterujący

Agenty sterujące mają za zadanie sterowanie agentami AR. Decyzja o ustaleniu parametrów agentów regulatorów powstaje w wyniku analizy dotychczasowego stanu modelu (lub jego części).

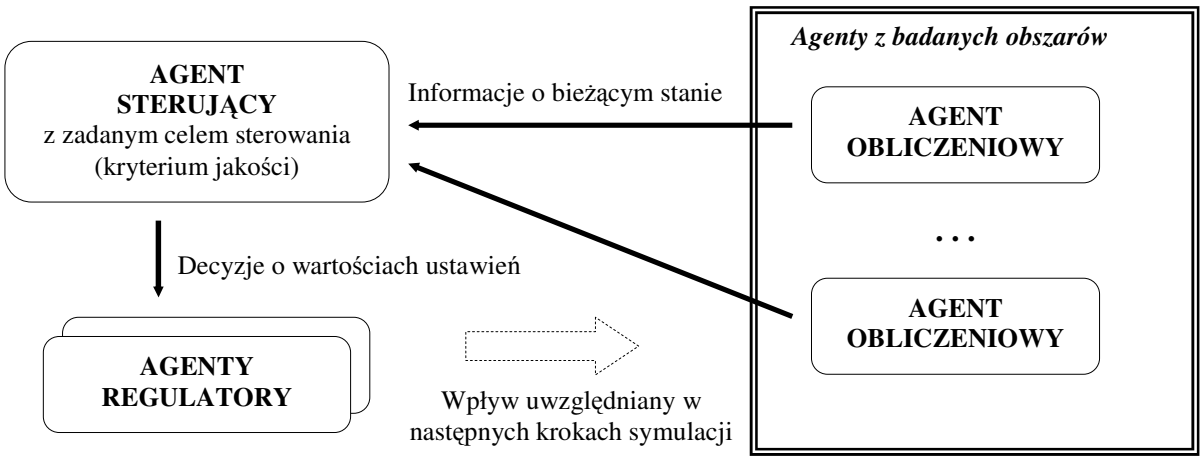
Na rys. 5.6 przedstawiono sposób współdziałania agenta sterującego z dwoma pozostałymi typami agentów. Agent sterujący dokonuje obserwacji stanu wybranych agentów obliczeniowych. Na ich podstawie oraz historii ich zmian, agent sterujący wypracowuje decyzje sterujące, przekazywane agentom regulatorom.

Przyjęto, że w rozważanym przypadku decyzje sterujące dotyczyć mogą następujących wielkości określających działanie agentów regulatorów:

- otwarcie przepływu medium chłodzącego w danym regulatorze,
- zamknięcie przepływu medium,
- zmiana prędkości ($\mathbf{v}_i$) przemieszczania się medium chłodzącego dla danego regulatora ($\mathbf{AR}_i$),
- zmiana temperatury wpływającego medium ($\boldsymbol{\tau}_i$).

Dwa ostatnie sposoby oddziaływania odpowiadają sterowaniu o charakterze parametrycznym, zaś dwa pierwsze sprowadzają się do sterowania dwupołożeniowego.

Wpływ dynamicznego zachowania agenta regulatora uwidacznia się przez zmianę stanu nadzorowanych przez niego agentów obliczeniowych. Powoduje to zmianę stanu modelu w tej jego części, którą reprezentuje dany agent regulator. Nowy stan oddziałuje na sąsiadujące obszary w kolejnych krokach procesu symulacji.



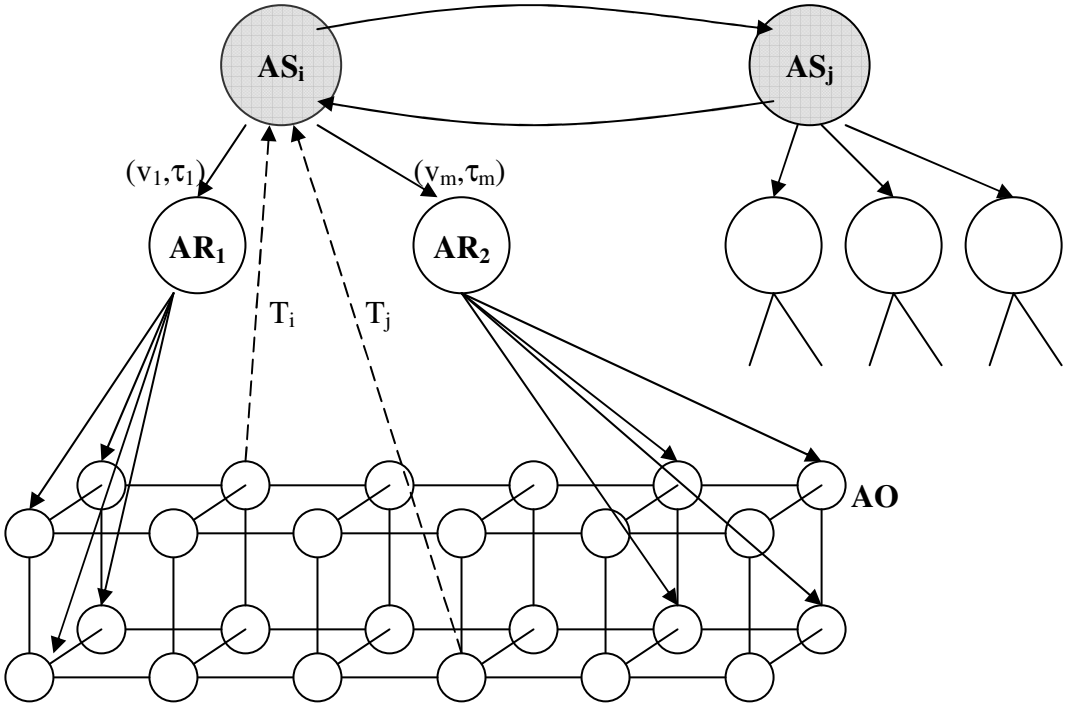
Rys. 5.6. Schemat działania agenta sterującego

5.6. Działanie systemu MAFES-1

W agentowej strukturze organizacyjnej systemu MAFES-1 (rys. 5.7), pomiędzy jego elementami występują powiązania o charakterze czysto informacyjnym oraz związki przyczynowo-skutkowe wynikające z przekazywania decyzji (realizacji działań).

Można wyróżnić następujące rodzaje przepływu informacji między agentami:

- agent obliczeniowy (**AO**) pobiera informację o wartościach atrybutów od sąsiadujących z nim agentów obliczeniowych,
- agent sterujący (**AS**) pobiera informację o wartościach atrybutów od powiązanych z nim (monitorowanych) agentów obliczeniowych,
- agent sterujący przekazuje decyzyje do powiązanych z nim (nadzorowanych) agentów regulatorów (**AR**),
- agent regulator realizuje działanie sterujące poprzez zmiany atrybutów nadzorowanych agentów obliczeniowych

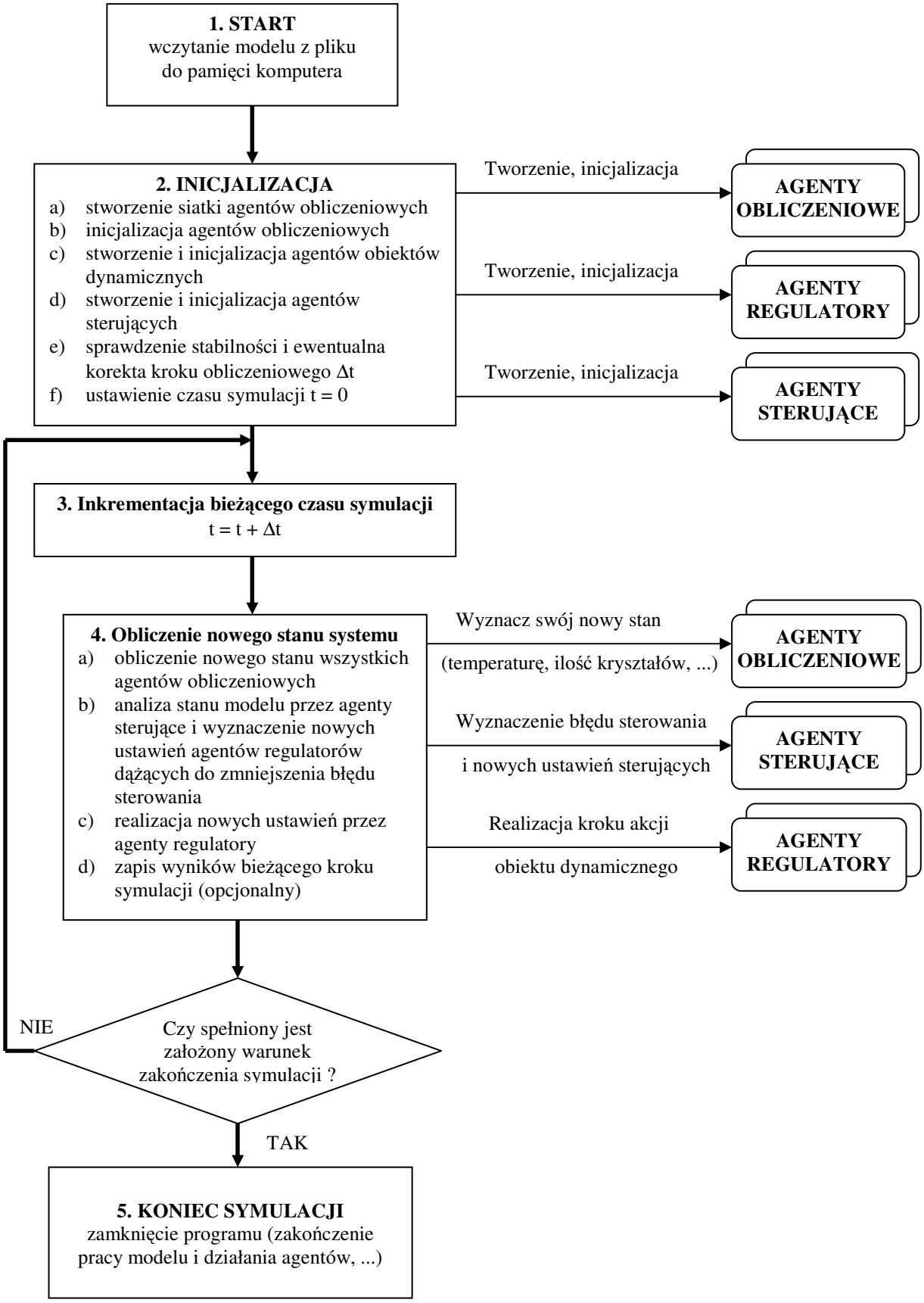


Rys. 5.7. Struktura organizacyjna agentowego systemu MAFES-1

Algorytm procesu symulacji w systemie MAFES-1 (rys. 5.8) obejmuje następujące operacje:
- inicjalizacja wszystkich typów agentów występujących w modelu,
- realizacja następujących po sobie kroków symulacji, w każdym kroku:
  a) każdy agent obliczeniowy wyznacza swój nowy stan,
  b) agenty sterujące obserwują stan agentów obliczeniowych badanych obszarów, podejmują decyduje dotyczące parametrów podległych im agentów regulatorów,
  c) agenty regulatory realizują krok przemieszczenia medium chłodzącego, przenosząc własności odpowiednich agentów obliczeniowych,
  d) zapisywane są wyniki bieżącego stanu modelu,
- zakończenie symulacji.

Sprawdzenie stabilności obliczeń (rys. 5.8, blok inicjalizacji) związane jest z koniecznością zapewnienia odpowiedniej zależności pomiędzy krokiem czasowym symulacji a pozostałymi jej parametrami. Maksymalna wartość stabilnego kroku jest odwrotnie proporcjonalna do wartości odległości między sąsiednimi węzłami siatki agentów obliczeniowych. Ograniczenie to zapobiega wzrostowi wartości temperatury danego agenta obliczeniowego powyżej wartości temperatur w sąsiadujących z nim agentach (równania (5.5) – (5.7), rozdział 5.3).

Najczęściej warunkiem zakończenia symulacji jest osiągnięcie przez bieżący czas symulacji wartości maksymalnej, zadawanej w konfiguracji modelu wejściowego.



Rys. 5.8. Struktura procesu symulacji w systemie MAFES-1

5.7. Środowisko agentowe w systemie MAFES-1

Środowiskiem wspólnym dla wszystkich agentów obliczeniowych jest przestrzeń, w której się one znajdują, pozwalająca określać relację sąsiedztwa między agentami obliczeniowymi.

Natomiast dla pojedynczego agenta obliczeniowego, otoczeniem (częścią środowiska) jest grupa sąsiadujących z nim agentów tego samego typu, których własności obserwuje wyznaczając (obliczając) swój stan.

Relacje agenta regulatora z otoczeniem składają się z dwóch części:
- oddziaływanie na nadzorowanych przez niego agentów obliczeniowych, których atrybuty zmienia zgodnie z dynamiką ruchu (np. przepływem medium chłodzącego),
- odbieranie komunikatów od zarządzających nim agentów sterujących.

Agent sterujący dokonuje następujących interakcji z zewnętrznym środowiskiem:
- obserwacja zmian stanu agentów obliczeniowych z obszarów, których parametry podlegają ocenie w kontekście sterowania,
- przekazywanie decyzji sterujących agentom regulatorom, które zmieniają ich zachowanie (np. prędkość przepływu medium, czy temperatura wpływającej cieczy),
- negocjacje z innymi agentami sterującymi odnośnie podejmowania decyzji sterujących dla regulatorów, które są wspólnie nadzorowane przez więcej niż jednego agenta sterującego.

5.8. Symulacja uproszczonego zadania sterowania

Sterowanie w procesach stygnięcia i krzepnięcia można interpretować jako zbiór działań mających na celu doprowadzenie do bardziej równomiernego rozkładu temperatur w stygnącym odlewie, a w konsekwencji uzyskanie wymaganych parametrów mechanicznych odlewów, które uzależnione są od kształtowanej w trakcie przejścia fazowego struktury kryształów. Trudność w określeniu takiego sterowania polega na tym, że w czasie krzepnięcia temperatura odlewu nie jest funkcją zależną jedynie od intensywności chłodzenia. W początkowym okresie, następującym po momencie przechłodzenia pojawia się etap wzrostu temperatury związanego z wydzielaniem ciepła krystalizacji. W konsekwencji, przebieg temperatury ma charakter nieliniowy i niemonotoniczny, co prowadzi do tego, że wyznaczenie sterowań w oparciu o metody analityczne jest trudne lub nawet niemożliwe.

W tej sytuacji, proponuje się aby badania dotyczące sterowania rozważanym procesem przeprowadzić w oparciu o model symulacyjny – w danym przypadku system MAFES-1.

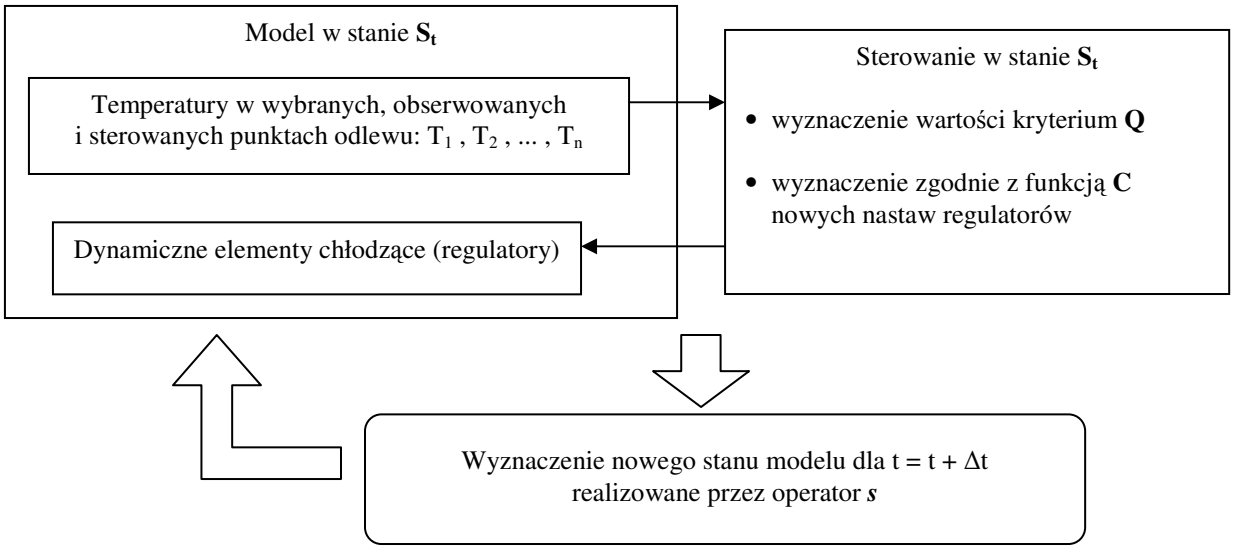
Należy zaznaczyć, że konstrukcja efektywnych algorytmów decyzyjnych sterujących procesem stygnięcia i krystalizacji, stanowi poważny problem badawczy, wykraczający poza zakres niniejszej pracy.

Równocześnie jednak, proponując określone rozwiązanie modelu symulacyjnego, autor pragnie uzyskać potwierdzenie, że model ten rzeczywiście nadaje się do prowadzenia tego typu badań.

W efekcie, w przedstawionych poniżej rozważaniach omówiono funkcjonowanie modelu symulacyjnego (MAFES-1) w połączeniu z działaniami sterującymi, jednakże wprowadzone działania dotyczą znacznie uproszczonej wersji zadania sterowania. W danym przypadku zadanie to dotyczy minimalizacji różnicy temperatur dla dwu wybranych punktów modelu.

Tak postawione zadanie, tylko w ograniczonym zakresie odpowiada wymogom technologicznym, pozwala jednak na weryfikację działania samego modelu, jak też potwierdza możliwość uzyskania pożądanych zmian w rozkładzie pola temperatur stygnącego odlewu.

Na rys. 5.9 przedstawiono schemat organizacji procesu symulacji, uwzględniającego działanie agenta sterującego (symulacja przykładowego zadania sterowania).



Rys. 5.9. Schemat blokowy układu sterowania

Stan modelu symulacyjnego **S** w dwóch kolejnych chwilach czasowych można wyrazić w postaci:

$$\mathbf{S_{t+\Delta t}} = \boldsymbol{s}\ (\ \mathbf{S_t}\ ,\ \mathbf{C}(\ \mathbf{S_t}\ )\ ) \tag{5.17}$$

gdzie ***s*** jest operatorem reprezentującym zmianę stanu dokonywaną w pojedynczym kroku symulacji, natomiast **C** funkcją decyzyjną wyznaczającą wartości parametrów sterujących na podstawie bieżącego stanu modelu (wartości wielkości sterowanych).

Tak więc, na nowy stan modelu ma wpływ jego stan aktualny oraz wartości parametrów sterujących ustalone przez funkcję **C**.

Punktem wyjścia do sformułowania przykładowego zadania jest przyjęcie wskaźnika jakości, który w danym przypadku określono w postaci:

$$\mathbf{Q}(t) = \mathbf{Q}(T_1, T_2, ..., T_N) = \max (| T_i - T_j |) = h_{uw}(t), \text{ gdzie } i, j = 1..N, i \neq j \qquad (5.18)$$

gdzie: T_i , T_j – temperatury w wybranych (monitorowanych) punktach modelu symulacyjnego,

h_{uw} – różnica temperatur pomiędzy punktami o indeksach u, w, wyznaczonych w wyniku operacji (5.18).

Tak więc, obiektem sterowania jest model procesem stygnięcia i krystalizacji, którego wielkością wyjściową jest różnica temperatur $\mathbf{h_{uw}}(t)$, zaś decyzje sterujące dotyczą parametrów układu chłodzącego, czyli prędkości przepływu $\mathbf{v}(t)$ oraz temperatury $\boldsymbol{\tau}(t)$.

Celem sterowania jest uzyskanie możliwie równomiernego rozkładu temperatur w okresie krystalizacji, co odpowiada minimalizacji wskaźnika jakości **Q**.

Biorąc pod uwagę dyskretyzację czasu, wynikającą z cyklu działania modelu symulacyjnego, można napisać:

$$Q^* = \min_{v(t_i),\tau(t_i)} \sum_{i=l}^{s} h_{uw}(v(t_i), \tau(t_i)) \qquad (5.19)$$

gdzie: t_i – momenty czasu odpowiadające kolejnym etapom symulacji,

t_l – moment osiągnięcia temperatury liquidus (początek krystalizacji),

t_s – moment osiągnięcia temperatury solidus (zakończenie krystalizacji).

Z czysto formalnego punktu widzenia, rozwiązaniem zadania optymalizacji (5.19) jest strategia:

$$\{ (v(t_l), \tau(t_1)) , (v(t_{l+1}), \tau(t_{1+1})) , ..., (v(t_s), \tau(t_s)) \}^* \qquad (5.20)$$

zapewniająca uzyskanie minimalnej wartości wskaźnika Q^*.

Trzeba jednakże zwrócić uwagę, że wartości $h_{uw}(v(t_i), \tau(t_i)))$ w kolejnych krokach symulacji uzyskiwane są w wyniku operacji (5.18), zaś temperatury T_i, T_j będące jej argumentami są rezultatem działania modelu symulacyjnego. Co więcej, wartość maksymalnej różnicy $| T_i - T_j |$ uzyskana być może w kolejnych krokach dla różnych punktów modelu (dla różnych wartości indeksów u, w).

W konsekwencji prowadzić to może do sterowania rozproszonego realizowanego przez różnych agentów AS i związanych z nimi regulatorów AR.

W tak złożonej sytuacji, można mówić jedynie o poszukiwaniu strategii quasi optymalnej przy zastosowaniu metod inteligencji obliczeniowej. Szczególnie przydatnymi wydają się tu metody ewolucyjne lub sieci neuronowe.

Jak już wspomniano, problem sterowania wykracza poza ramy niniejszej pracy i zamiarem autora jest jedynie pokazanie, że proponowany model (MAFES-1) może służyć do tego typu badań.

Dlatego też, w opisanych dalej eksperymentach ograniczono się do bardzo prostego przypadku, gdy obserwowane są tylko 2 (a co najwyżej 3) punkty modelu, co w konsekwencji prowadzi do wykorzystania jednego agenta AS, współpracującego z nim agenta AR oraz grupy podległych mu agentów obliczeniowych (AO).

Decyzje sterujące agenta AS (które na pewno nie są optymalnymi) określane SA na podstawie oceny aktualnej w danym kroku symulacji wartości różnicy $h(t_i)$, a więc posiadają charakter sterowania nadążnego.

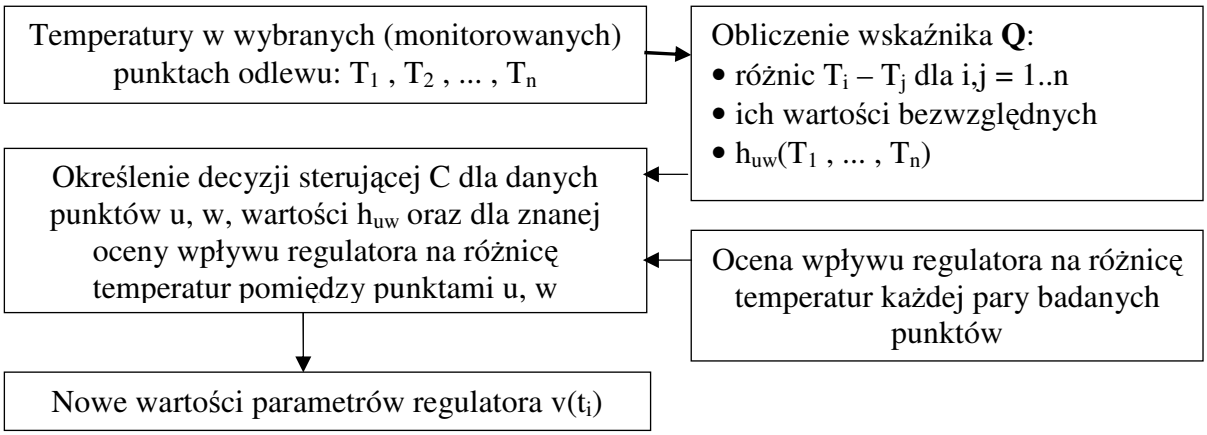
Po wstępnej serii eksperymentów symulacyjnych, dla ułatwienia interpretacji uzyskanych rezultatów, ograniczono się do zastosowania decyzji sterujących dotyczących wyłącznie prędkości przepływu medium chłodzącego (z zachowaniem stałej wartości temperatury $\tau(t_i)$).

Równocześnie zdecydowano się na przyjęcie dwupołożeniowego wariantu sterowania, a zatem wyrażenie:

$$v(t_{i+1}) = C\ (\ h_{uw}(t_i)\ ) \tag{5.21}$$

określa decyzję sterującą o włączeniu przepływu medium chłodzącego ($v(t_{i+1}) = V$), lub jego wyłączeniu ($v(t_{i+1}) = 0$).

Przyjęty w sposób arbitralny algorytm decyzyjny (5.21) przedstawiony zostanie przy opisie eksperymentów. Schemat ideowy stosowanego układu sterowania przedstawiono na rys. 5.7.



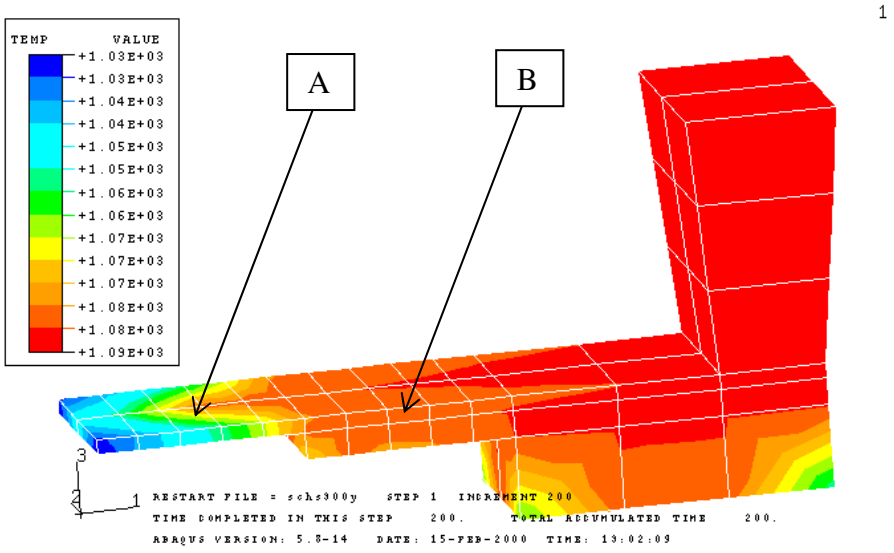
Rys. 5.10. Schemat blokowy układu sterowania

5.9. Eksperymenty symulacyjne w systemie ABAQUS

Celem tego rozdziału jest zaprezentowanie wybranych wyników symulacji w systemie ABAQUS dla porównania ich z wynikami symulacji w systemie MAFES-1 przedstawionymi w kolejnym podrozdziale.

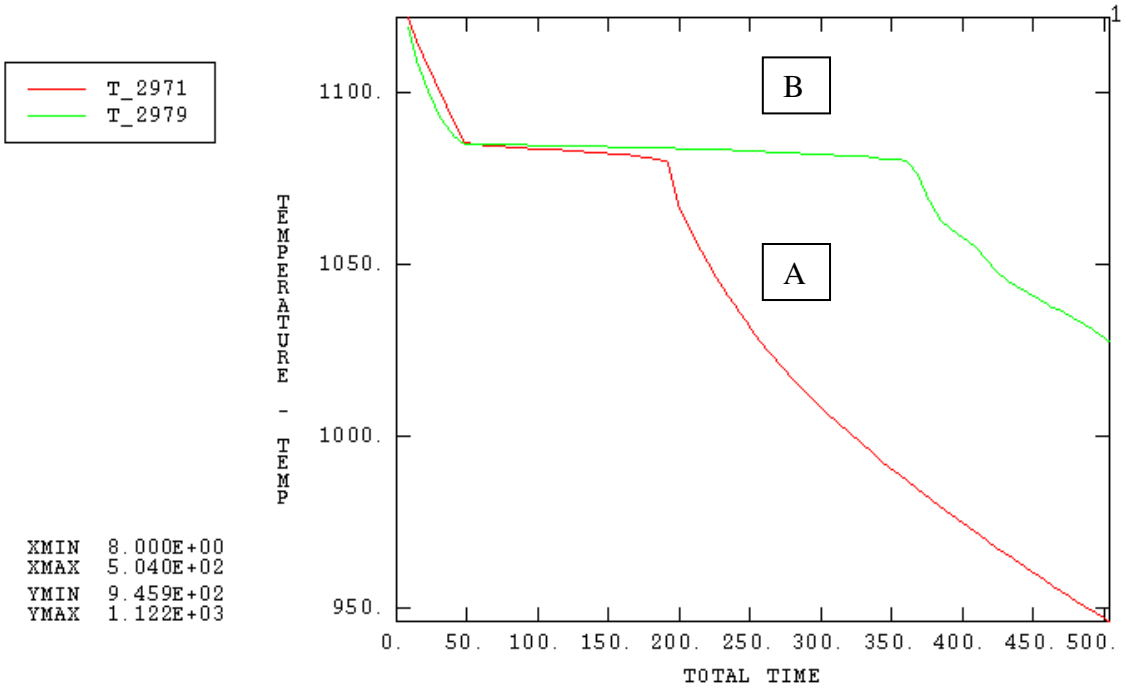
W oparciu o przeprowadzone badania systemu ABAQUS (prace [34, 50, 51]) można wskazać następujące niedoskonałości tego systemu (wymienione już w rozdziale 5.2):

a) proces krystalizacji definiowany jest przy budowaniu modelu symulacyjnego przez trzy parametry: ciepło krystalizacji, temperatury liquidus i solidus, efekt tego uproszczenia to brak na charakterystykach czasowych temperatury (rys. 5.12) chwilowego przyrostu wartości temperatury wynikającego ze zjawiska przechłodzenia (symulacja dla odlewu schodkowego – rys. 5.11),
b) wprowadzenie dynamicznych elementów chłodzących do modelu symulacyjnego jest trudna i nie uwzględnia ich rzeczywistego charakteru,



Rys. 5.11. Odlew schodkowy z miedzi, rozkład temperatury w czasie 124 sek. (ABAQUS)

Przedstawione na rys. 5.11, 5.12 wyniki eksperymentu symulacyjnego w systemie ABAQUS dotyczą odlewu schodkowego (rys. 5.11) ze stopu miedzi w formie piaskowej. Chociaż na rys. 5.12 widoczny jest efekt przejścia fazowego (krystalizacji) w postaci przejściowego zmniejszenia szybkości obniżania się temperatury to nie można zauważyć zjawiska przechłodzenia, w którym następuje chwilowy wzrost temperatury.



Rys. 5.12. Charakterystyki czasowe temperatury w małym (element A) i średnim (element B) schodku (ABAQUS)

5.10. Eksperymenty symulacyjne w systemie MAFES-1

W rozdziale tym przedstawiono wyniki dla dwóch modeli symulacyjnych zrealizowanych w systemie MAFES-1. W pierwszym eksperymencie zrealizowano sterowanie na podstawie temperatur w dwóch punktach odlewu. W drugim eksperymencie sterowanie dotyczy temperatur w trzech punktach odlewu.

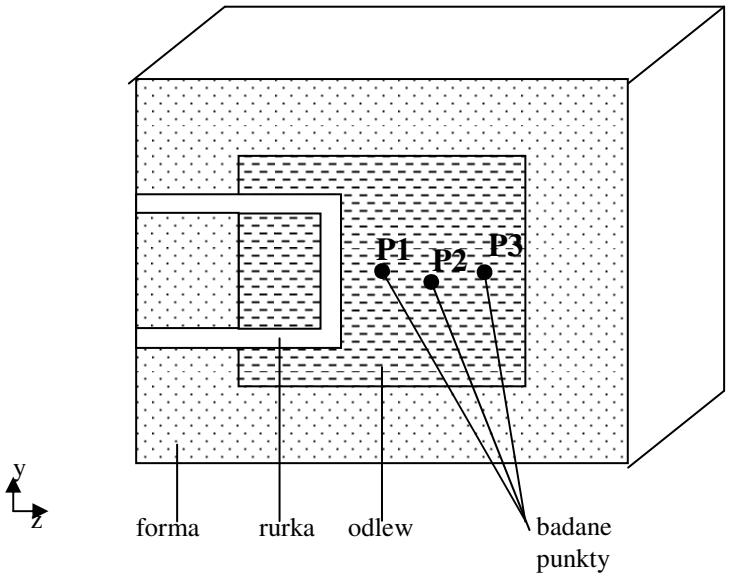
Komputerowy model oraz obliczenia realizowane są w trzech wymiarach, natomiast obserwacje wyników symulacji można prowadzić w dowolnym przekroju równoległym do płaszczyzn X–Y, Y–Z lub Z–X.

W obydwu modelach rurka z medium chłodzącym przechodzi przez odlew. Umieszczenie takie ma na celu osiągnięcie większego stopnia oddziaływania przez medium chłodzące na pole temperatur w symulowanych procesach stygnięcia odlewu. W rozwiązaniach stosowanych w praktyce najczęściej układy chłodzące usytuowane są w obszarze formy.

Eksperyment nr 1 w systemie MAFES-1 - odlew sześcienny, rurka chłodząca

Przedstawiony na rys. 5.13 przekrój symulowanego obiektu pokazuje trzy elementy wchodzące w jego skład:

- forma w kształcie sześcianu (forma metalowa),
- umieszczony centralnie odlew, posiadający również kształt sześcianu (ze stopu miedzi),
- medium chłodzące w postaci rurki, której wlot i wylot znajduje się w lewej ścianie bocznej formy.



Rys. 5.13. Przekrój symulowanego modelu

Punkty P1, P2, P3 zostały tak wybrane by można było zaobserwować zróżnicowanie wpływu chłodzenia formy na różne obszary odlewu. Układ chłodzący w postaci rurki z przepływającym medium, został umiejscowiony w taki sposób, by umożliwić zmniejszanie różnic pomiędzy temperaturami w punktach P1, P2, P3, w trakcie przebiegu symulacji.

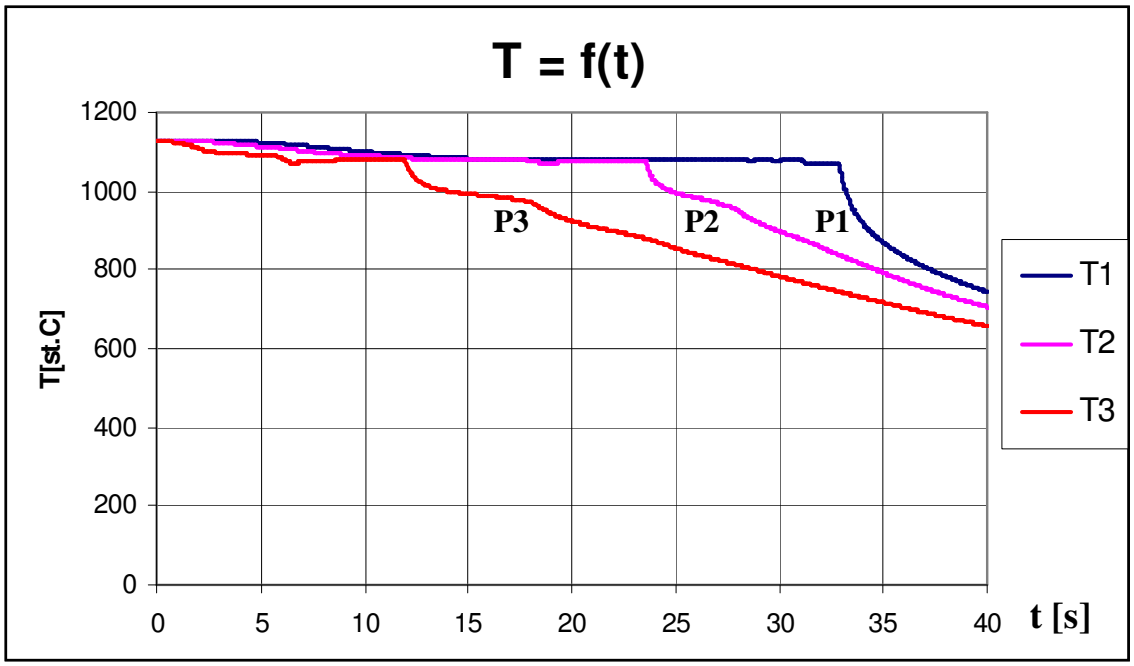
Przeprowadzono trzy eksperymenty symulacyjne:

- z wyłączonym przepływem medium chłodzącego (rys. 5.14),
- ze stałą prędkością przepływu medium (rys. 5.15.a),
- z przepływem medium sterowanym przez agenta AS (rys. 5.15.b).

Wykres przedstawiający zmiany temperatury w czasie, w trzech badanych punktach, dla symulacji bez przepływającego medium chłodzącego, został pokazany na rys. 5.14.

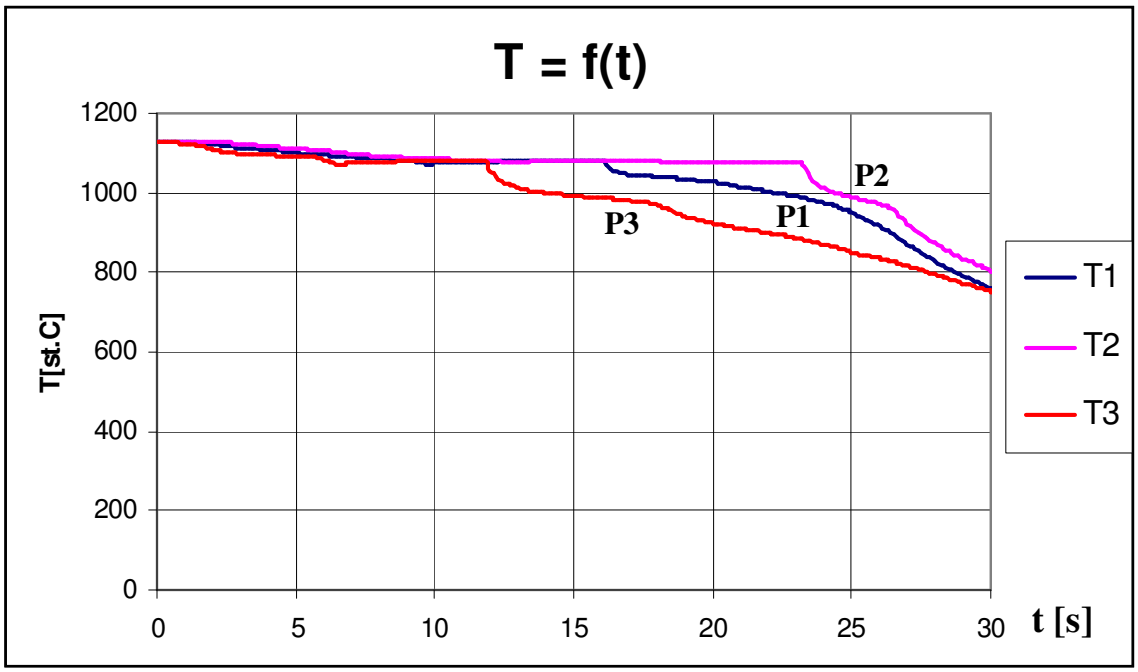
Na rysunku tym można dostrzec efekt przechłodzenia dla punktu P3, dla czasu t w przedziale 5 do 10 sekund. Przemiana fazowa dla punktu P3 trwała najkrócej, zakończyła

się dla czasu t równego około 12 sekund. Dla punktu P2 czas zakończenia przemiany fazowej to 24 sekundy, natomiast dla punktu P1 był on najdłuższy, wynosił około 33 sekundy.



Rys. 5.14. Charakterystyki czasowe temperatury w symulacji bez rurki

Na rys. 5.15 pokazano zmiany temperatury w czasie, w trzech badanych punktach, dla symulacji ze stałą prędkością przepływu medium chłodzącego.

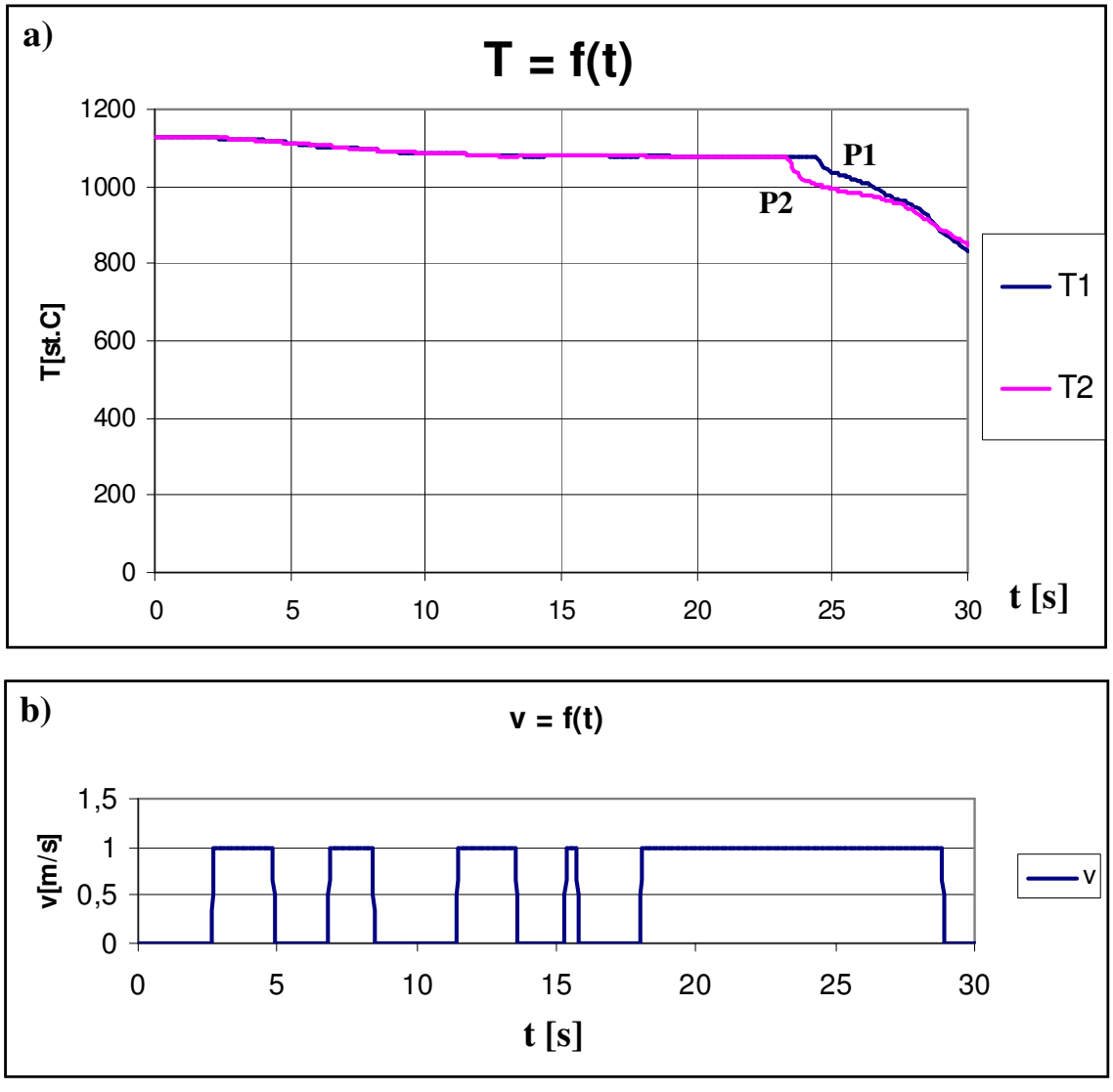


Rys. 5.15. Charakterystyki czasowe temperatury w symulacji z rurką, przy stałej prędkości przepływu medium chłodzącego, w trakcie całej symulacji: **v**(t) = 1 m/s

Jak wynika z rys. 5.15, wprowadzenie rurki zmieniło kolejność w szybkości stygnięcia punktów P1 i P2 (w porównaniu do rys. 5.14). Jednak w dalszym ciągu chłodzenie punktu P3 przez formę jest najszybsze.

W kolejnej symulacji zastosowano prosty algorytm sterowania („włącz-wyłącz") przepływem medium chłodzącego, mający na celu zmniejszenie różnicy pomiędzy temperaturami w punktach P1 i P2.

Wynik tej symulacji dla punktów P1 i P2 pokazano na rys. 5.16.a. Przebieg sterowania przepływem medium chłodzącego pokazano na rys. 5.16.b.



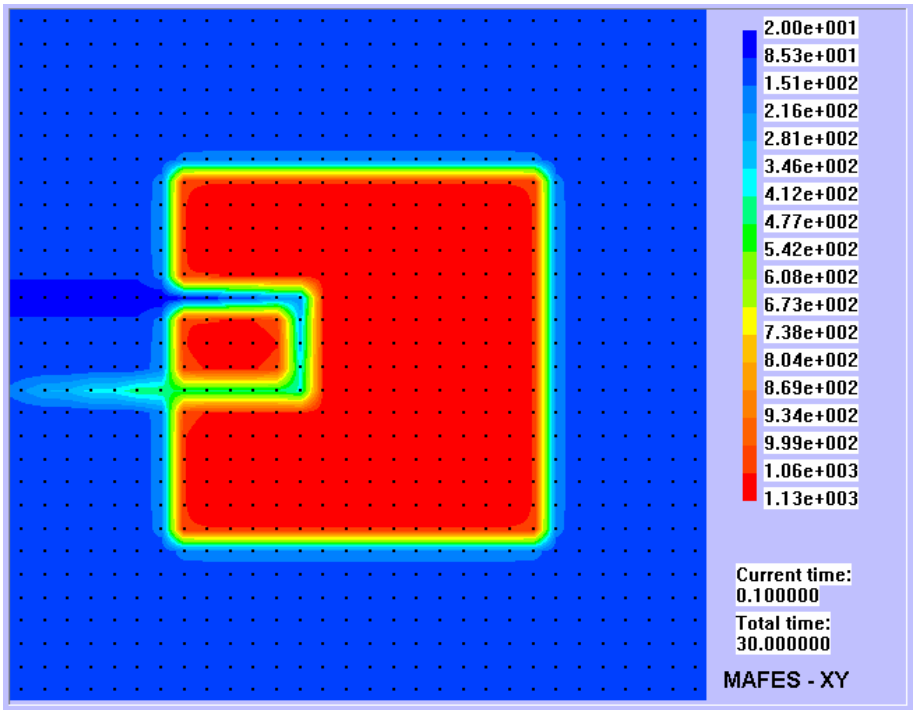
Rys. 5.16. Charakterystyki czasowe temperatury w symulacji z rurką i sterowaniem - a), sterowanie prędkością **v**(t) przepływu medium - b)

Porównując charakterystyki czasowe temperatur w punktach P1 i P2 przedstawione na rys. 5.14, 5.15 i 5.16.a, widoczne jest, iż wprowadzenie sterowania doprowadziło do znacznego zmniejszenia różnic temperatur między tymi punktami.

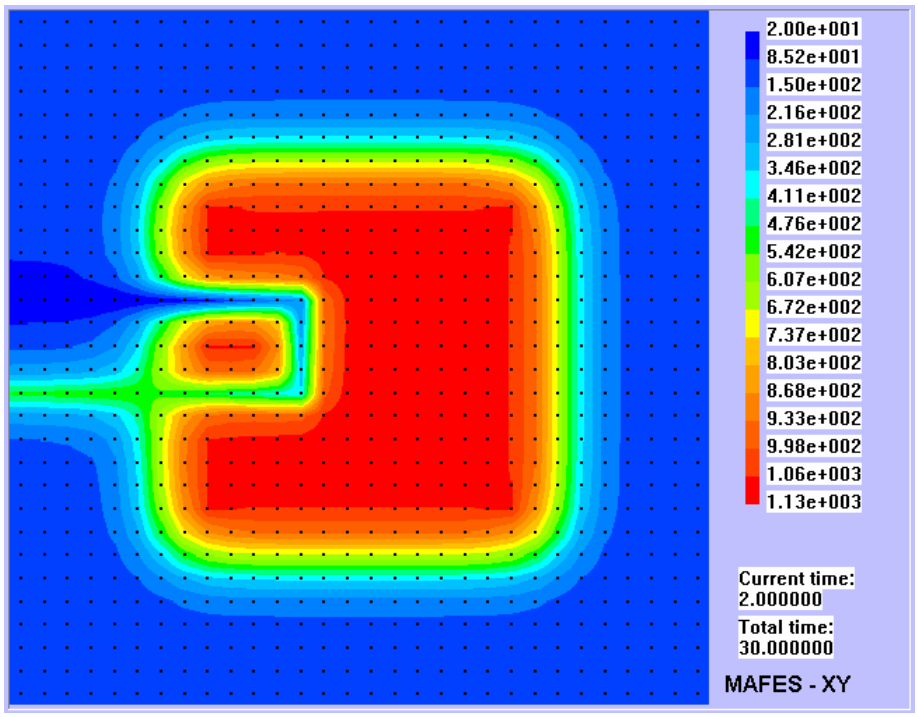
Realizacja symulacji za pomocą systemu MAFES-1 umożliwia obserwację nie tylko zmian temperatury w funkcji czasu, ale również rozkładu przestrzennego temperatury w wybranej chwili czasowej.

Rys. 5.17, 5.18 i 5.19 przedstawiają rozkład przestrzenny temperatury w modelowanym obiekcie (odlewie i formie) dla trzech kolejnych chwil symulacji (0.1 s, 2 s oraz 8 s). Skala temperatur jest tak dobrana, że kolor czerwony oznacza najwyższą wartość temperatury, natomiast kolor niebieski jej wartość najmniejszą. Liczbowe wartości temperatur, odpowiadające poszczególnym kolorom zaznaczono w prawej części rysunków.

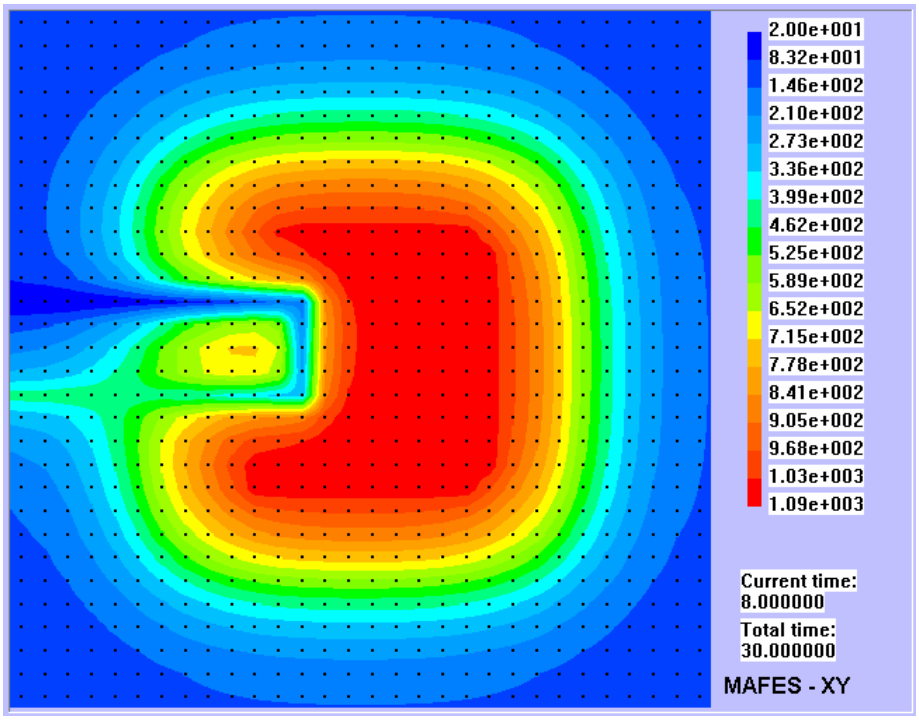
Na rysunkach tych widoczne jest odprowadzenie energii cieplnej przez medium chłodzące, z obszaru odlewu na zewnątrz.



Rys. 5.17. Rozkład temperatury w przekroju XY modelu dla czasu t = 0.1 s

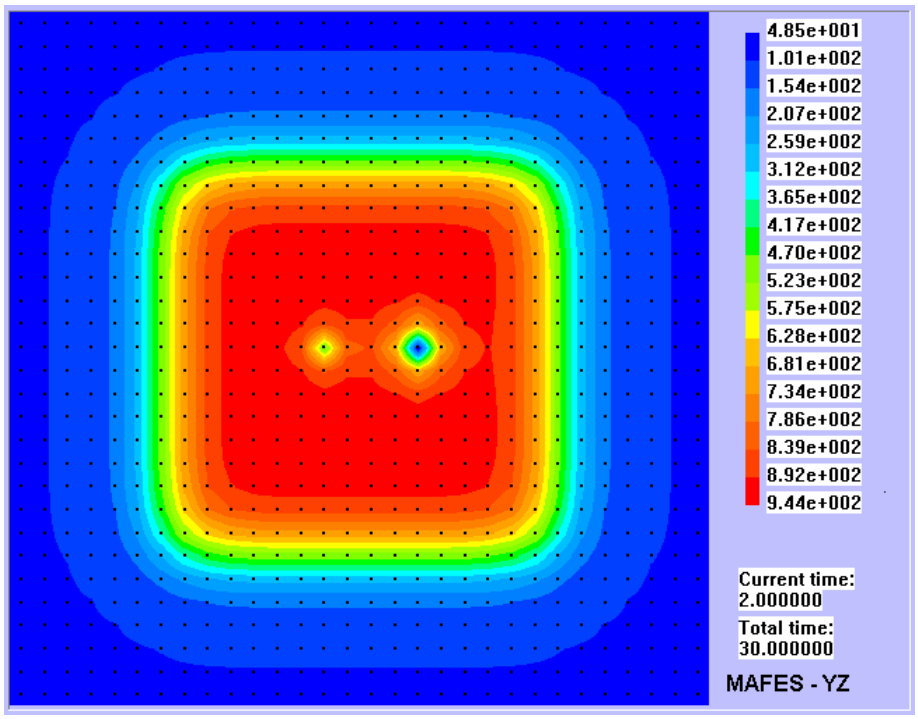


Rys. 5.18. Rozkład temperatury w przekroju XY modelu dla czasu t = 2 s



Rys. 5.19. Rozkład temperatury w przekroju XY modelu dla czasu t = 8 s

Na rys. 5.20 przedstawiono rozkład temperatur w formie i odlewie w przekroju YZ. Pokazuje to, że symulacja w systemie MAFES-1 realizowana jest we wszystkich trzech wymiarach. Widoczny jest również przekrój rurki z medium wpływającym (po prawej) i wypływającym (po lewej).

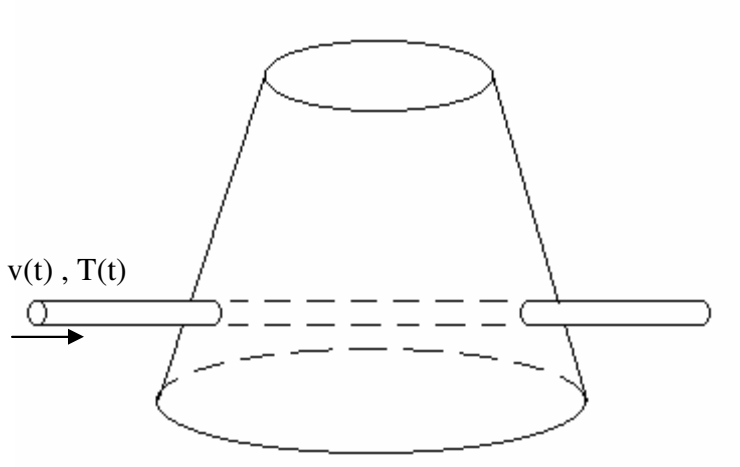


Rys. 5.20. Rozkład temperatury w przekroju YZ modelu dla czasu t = 2 s

Eksperyment nr 2 w systemie MAFES-1 - odlew stożkowy, rurka chłodząca

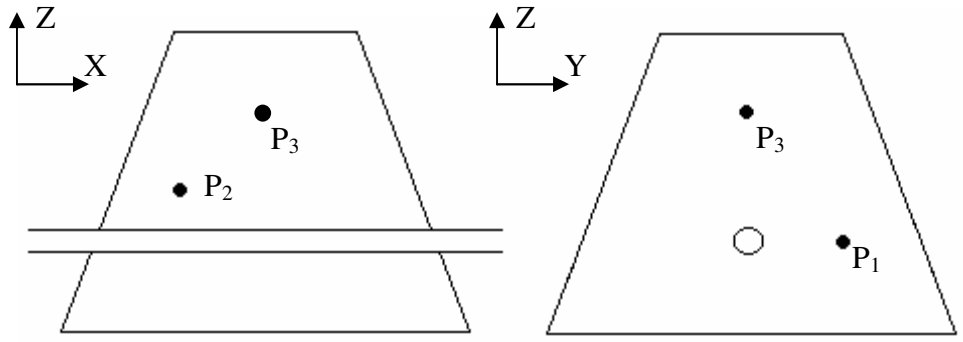
W poprzednim eksperymencie realizowano sterowanie na podstawie różnicy temperatur w dwóch punktach odlewu sześciennego. Celem drugiego eksperymentu jest realizacja sterowania na podstawie temperatur w trzech punktach odlewu.

W systemie MAFES-1 zamodelowano następujący układ odlewniczy: odlew w kształcie ściętego stożka, umiejscowiony centralnie w otaczającej go sześciennej formie. Przez formę i dolną część stożka poprowadzono rurkę chłodzącą. Odlew z rurką pokazano na rys. 5.21.



Rys. 5.21. Schemat odlewu (stożka) i rurki chłodzącej

W symulowanym odlewie wybrano trzy punkty monitorowane: P1, P2, P3, których umiejscowienie przedstawiono na rys. 5.22.

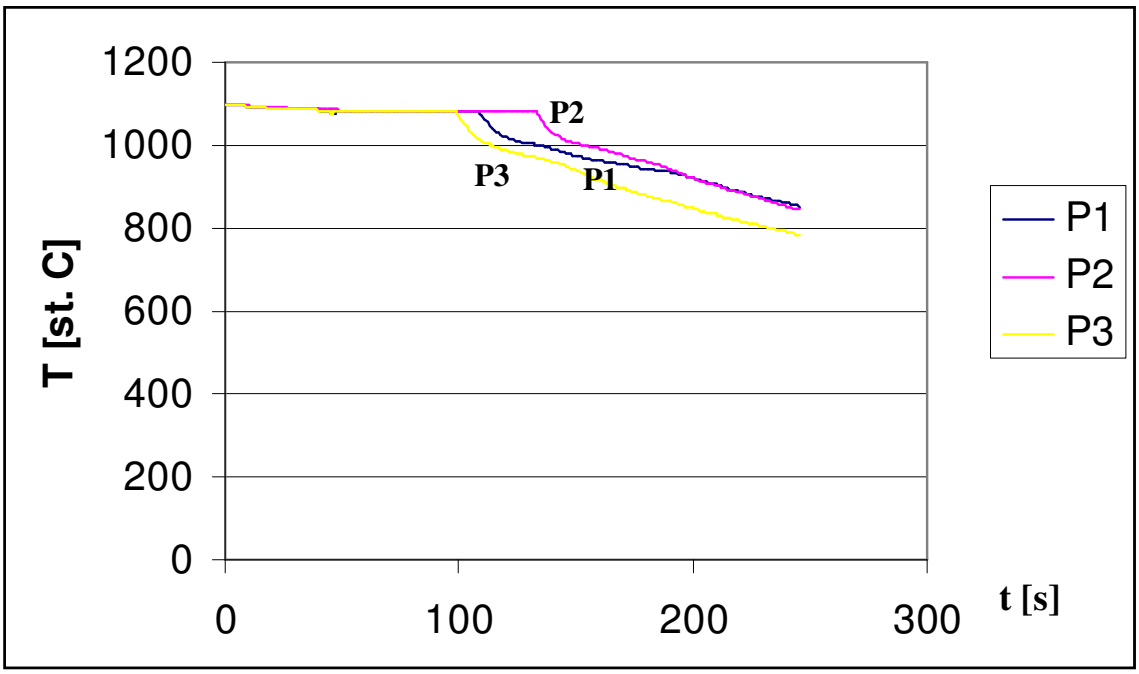


Rys. 5.22. Przekroje odlewu z zaznaczonymi badanymi punktami

Położenie rurki wynika z potrzeby chłodzenia dolnej części odlewu, która z powodu jego kształtu, jest chłodzona przez formę wolniej niż część górna. Oddziaływanie rurki chłodzącej zmierzać powinno do zmniejszenia różnic temperatury między punktami P1 a P3 oraz P2 a P3.

W przedstawionym eksperymencie założono, że chcemy uzyskać sytuację gdy temperatury w punktach P1 oraz P2 będą niższe niż w punkcie P3, ale jednocześnie aby różnice temperatur pomiędzy tymi trzema punktami były możliwie jak najmniejsze.

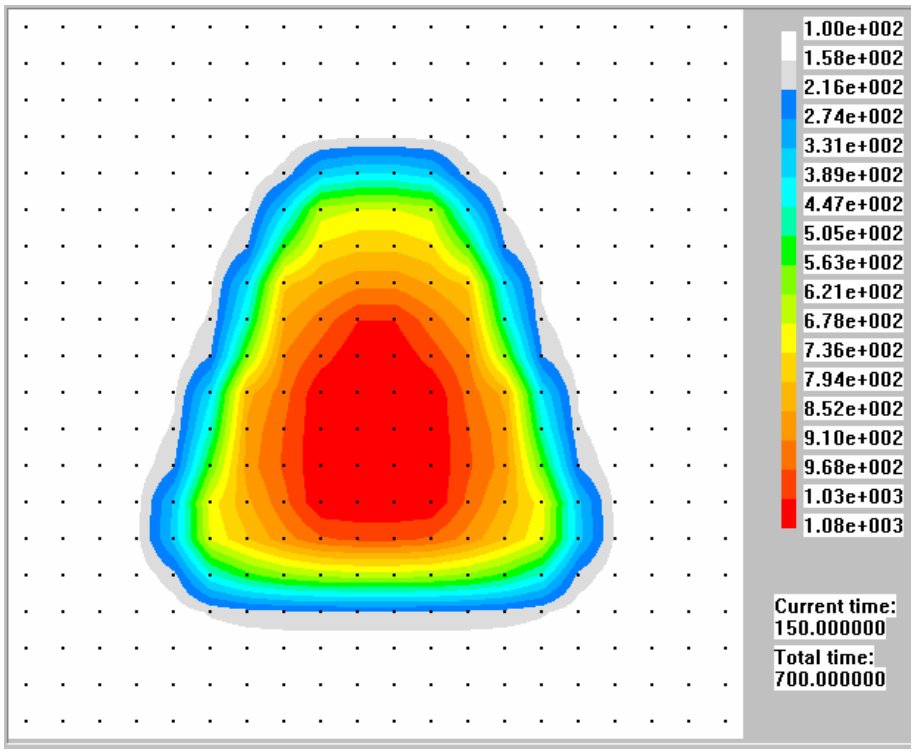
Na początku przeprowadzono symulację dla układu odlew forma bez dodatkowego chłodzenia. Wykres przebiegów temperatur w badanych punktach P1, P2, P3 pokazano na rys. 5.23.



Rys. 5.23. Wykres temperatur w trzech badanych punktach dla symulacji bez rurki chłodzącej

Jak można było przypuszczać, kształt geometryczny odlewu spowodował, że najszybciej schłodzony został punkt P3, położony w górnej, węższej, a więc bardziej chłodzonej przez formę części stożka. Natomiast najwolniej chłodzony był umieszczony w dolnej części odlewu punkt P2.

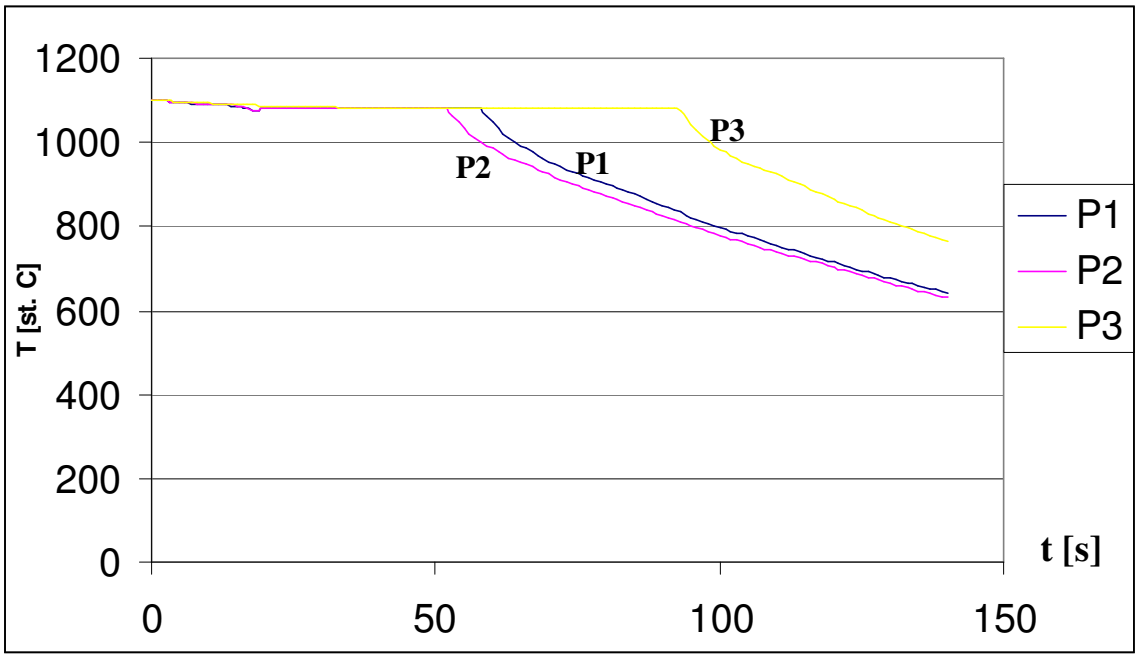
Przedstawione na rys. 5.24 pole temperatur, również potwierdza fakt szybszego stygnięcia górnej części stożka.



Rys. 5.24. Pole temperatur w przekroju X-Z dla czasu t = 150 s, symulacja bez zastosowania rurki chłodzącej

Po wprowadzeniu medium chłodzącego, płynącego przez rurkę przeprowadzoną w dolnej części stożka (odlewu), można spodziewać się efektu obniżenia temperatur w punktach P1, P2, natomiast wpływ rurki na temperaturę w punkcie P3 powinien być nieznaczny (rys. 5.22).

Wyniki przeprowadzonej symulacji przy wyłączonym sterowaniu, ze stałą prędkością przepływu medium chłodzącego (przez cały czas eksperymentu), w postaci charakterystyk czasowych w punktach P1, P2 i P3, pokazano na rys. 5.25. Można zaobserwować, że temperatury w punktach P1 i P2 są niższe aniżeli w punkcie P3 (rys. 5.23, 5.25), natomiast powstają duże różnice temperatur pomiędzy punktami P1 a P3 i P2 a P3.



Rys. 5.25. Wykres temperatur w trzech badanych punktach,
dla symulacji z rurką bez sterowania

Następnie wykonano symulację z chłodzeniem odlewu, ze sterowaniem prędkością przepływu medium chłodzącego, dążąc do zmniejszenia różnicy pomiędzy temperaturami w punktach P1 i P2, P1 i P3 oraz P2 i P3.

Do oceny wpływu przepływu medium chłodzącego na różnicę temperatur pomiędzy punktami P1 i P2, wykorzystano wyniki z rys. 5.25. W oparciu o przeprowadzony eksperyment z rurką chłodzącą ze stałym przepływem medium można stwierdzić, że przepływ medium chłodzącego obniża bardziej temperaturę w punkcie P2 aniżeli w punkcie P1. Jeżeli chodzi o pozostałe dwie pary różnic, to wpływ rurki chłodzącej obniża temperaturę w P1 i P2 bardziej niż w P3 (rys. 5.25).

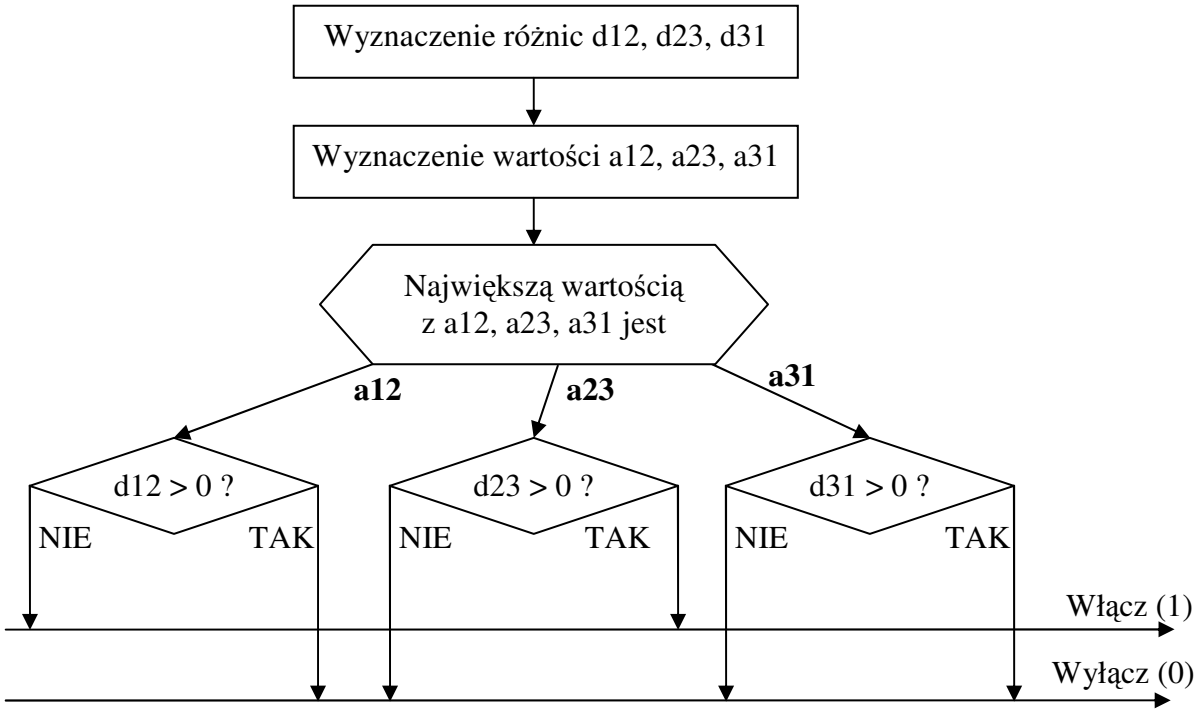
Dla celów przygotowania proponowanego dla tego modelu algorytmu sterowania określono różnice temperatur:

$$d12 = T_{P1} - T_{P2}\ ,\ \ d23 = T_{P2} - T_{P3}\ ,\ \ d31 = T_{P3} - T_{P1}\ ,$$

oraz wartości bezwzględnych tych różnic:

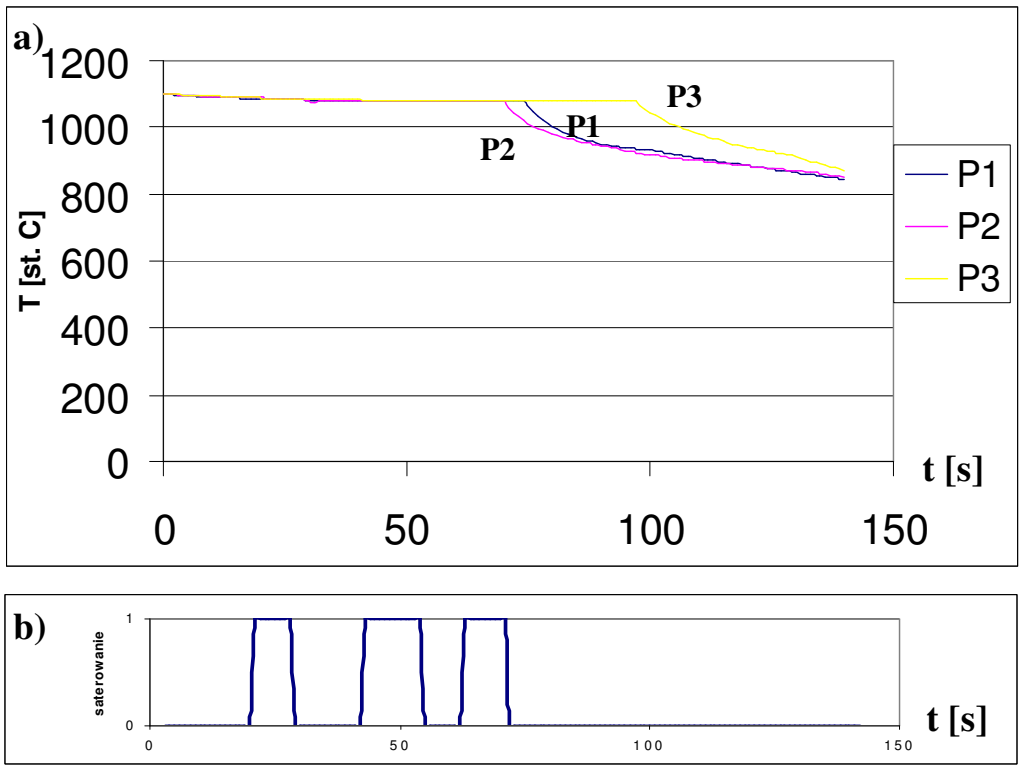
$$a12 = |T_{P1} - T_{P2}|\ ,\ \ a23 = |T_{P2} - T_{P3}|\ ,\ \ a31 = |T_{P3} - T_{P1}|$$

Oparty o tak określone współczynniki algorytm decyzyjny agenta sterującego pokazano na rys. 5.26 (w przypadku, gdy największą wartość reprezentują równocześnie dwie lub wszystkie trzy różnice a12, a23, a31, największa różnica wybierana jest w sposób losowy).

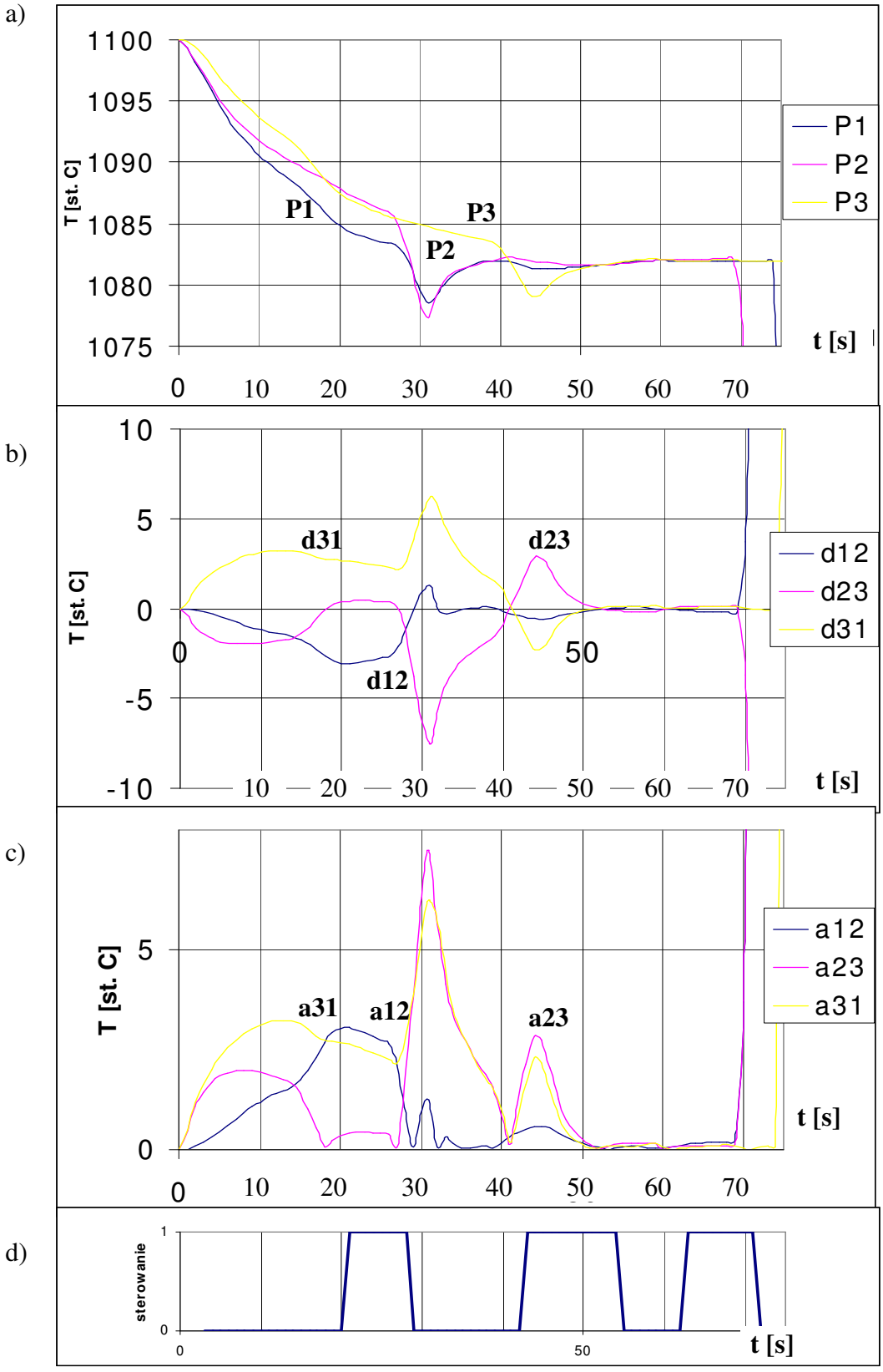


Rys. 5.26. Algorytm decyzyjny agenta sterującego AS (eksperyment nr 2)

Wyniki symulacji uzyskane po wprowadzeniu sterowania pokazano na rys. 5.27, 5.28.



Rys. 5.27. a) wykres temperatur w trzech badanych punktach, przy wprowadzeniu sterowania, b) sterowanie dwupołożeniowe prędkością przepływu medium chłodzącego



Rys. 5.28. a) temperatury w punktach P1, P2, P3, b) różnice tych temperatur, c) wartości bezwzględne tych różnic, d) przebieg sterowania

Na rys. 5.27.a przedstawiono zmiany temperatury w czasie w punktach P1, P2 i P3, natomiast na rys. 5.27.b pokazano przebieg wielkości sterującej.

Wprowadzenie sterowania zmniejszyło różnice temperatur pomiędzy wszystkimi trzema punktami P1, P2 i P3, co było celem eksperymentu.

Dla lepszego zilustrowania działania sterowania, fragment charakterystyki z rys. 5.27.a przedstawiono na rys. 5.28.a w powiększonej skali. Widoczny jest tu wyraźnie efekt przechłodzenia dla wszystkich trzech punktów (dla P1 i P2 przy t równym około 32 sekundy, dla P3 przy t równym 45 sekund). Tego efektu nie można było zaobserwować w symulacjach opartych na systemie ABAQUS (rys. 5.12).

Na rys. 5.28.b pokazano różnice temperatur d12, d23, d31, natomiast na rys. 5.28.c przedstawiony jest wykres wartości bezwzględnych tych różnic (a12, a23, a31). Na rys. 5.28.d przedstawiono przebieg sterowania.

Na rys. 5.28.a-c widać, że zmiany temperatury miały charakter oscylacyjny, co wynika z zastosowania bardzo prostego algorytmu sterowania.

Interesujące wydają się rezultaty podane na rys. 5.29, 5.30, przedstawiające wpływ sterowania na parametry związane bezpośrednio z procesem krystalizacji – udział fazy stałej (F_S), promień kryształów (R) oraz liczbę kryształów (N).

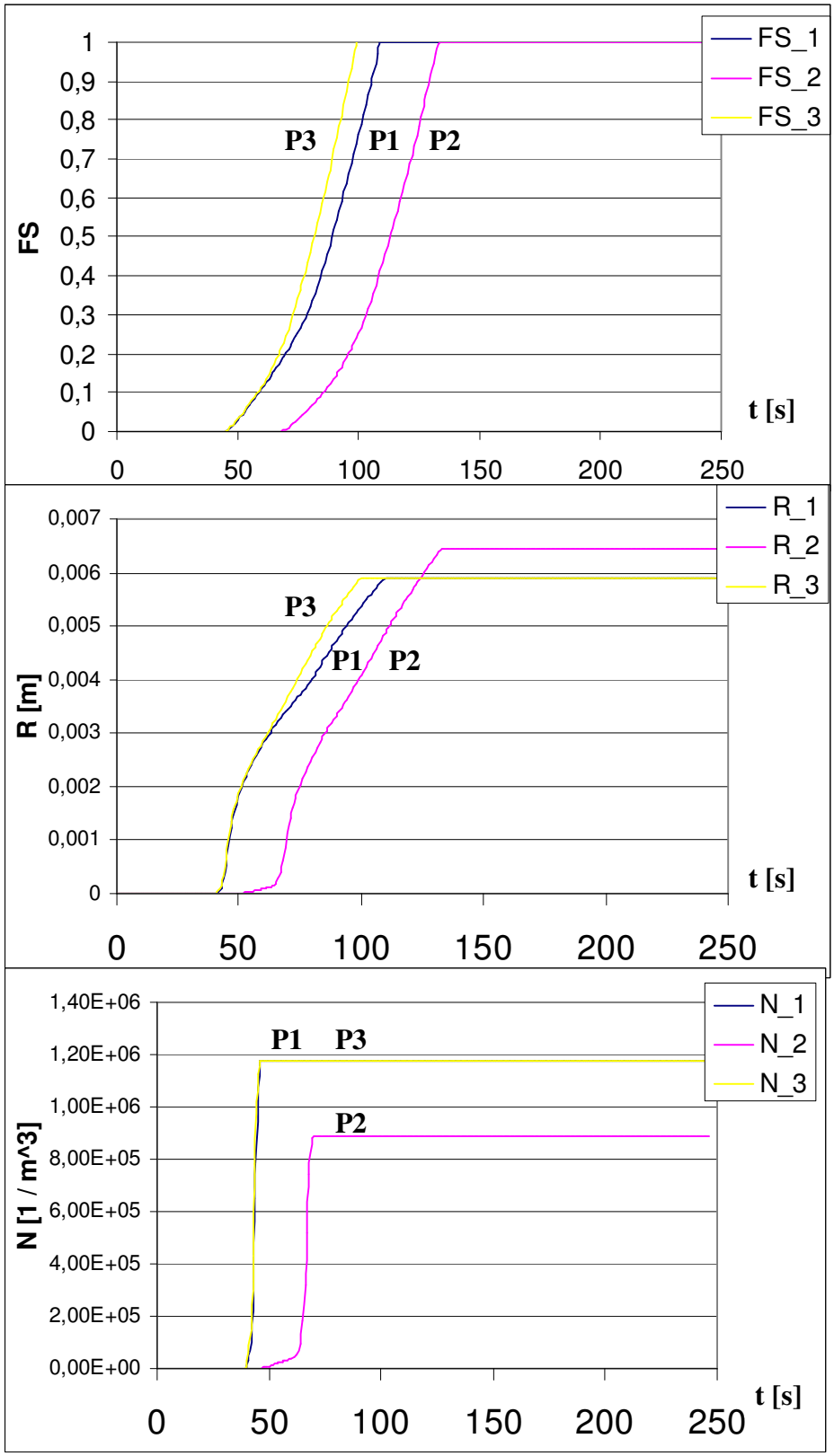
Można zaobserwować, że wprowadzenie sterowania zmienia wyraźnie wzajemne usytuowanie przebiegów powyższych parametrów w czasie.

Kolejna grupa rysunków ilustruje rozkłady temperatur modelowanego odlewu w różnych przekrojach. Rys. 5.31, 5.32 pokazują wpływ rurki chłodzącej w przekroju poziomym i pionowym, natomiast rys. 5.33, 5.34, ilustrują rozkłady temperatury w przekrojach odlewu w różnych momentach czasu dla symulacji z rurką chłodzącą i sterowaniem.

a)

b)

c)

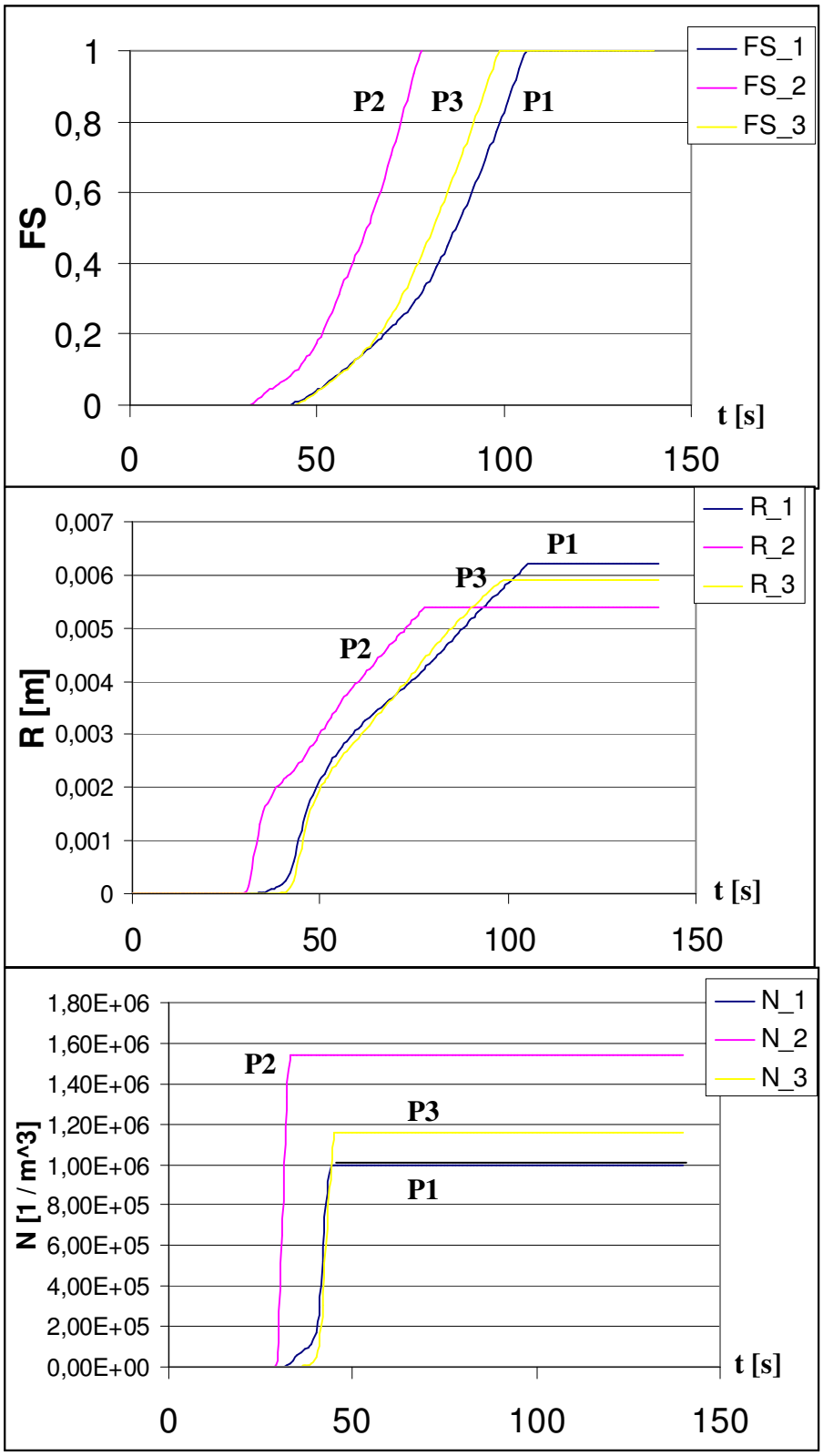


Rys. 5.29. Wyniki dla symulacji bez rurki chłodzącej: a) udział fazy stałej, b) promień kryształów, c) liczba kryształów

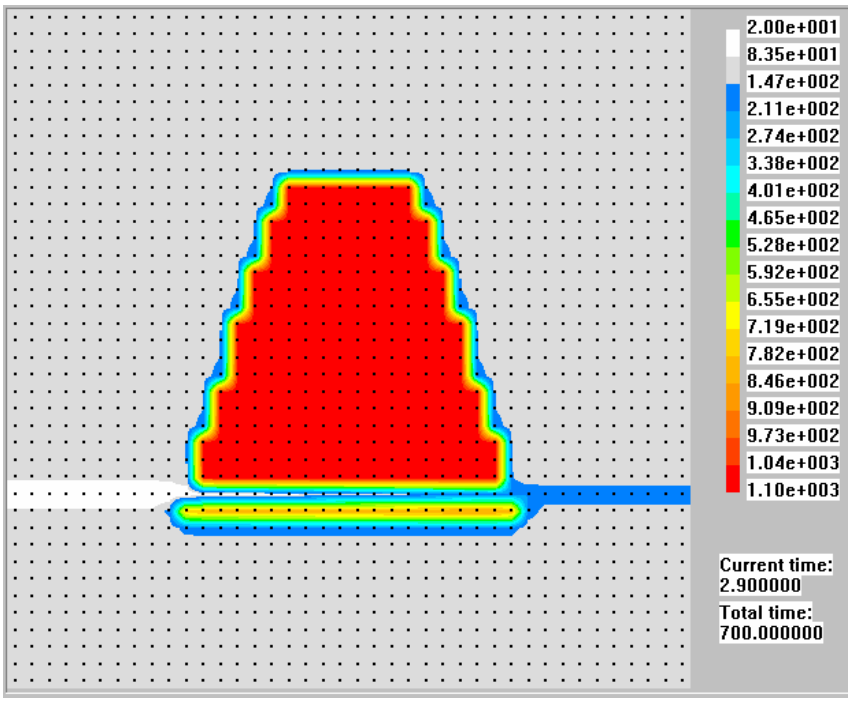
a)

b)

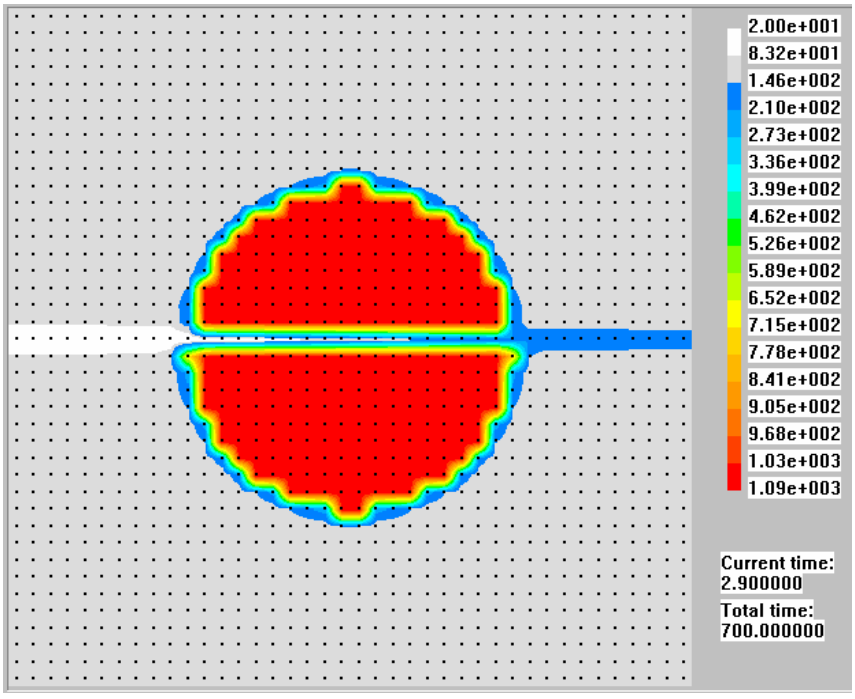
c)



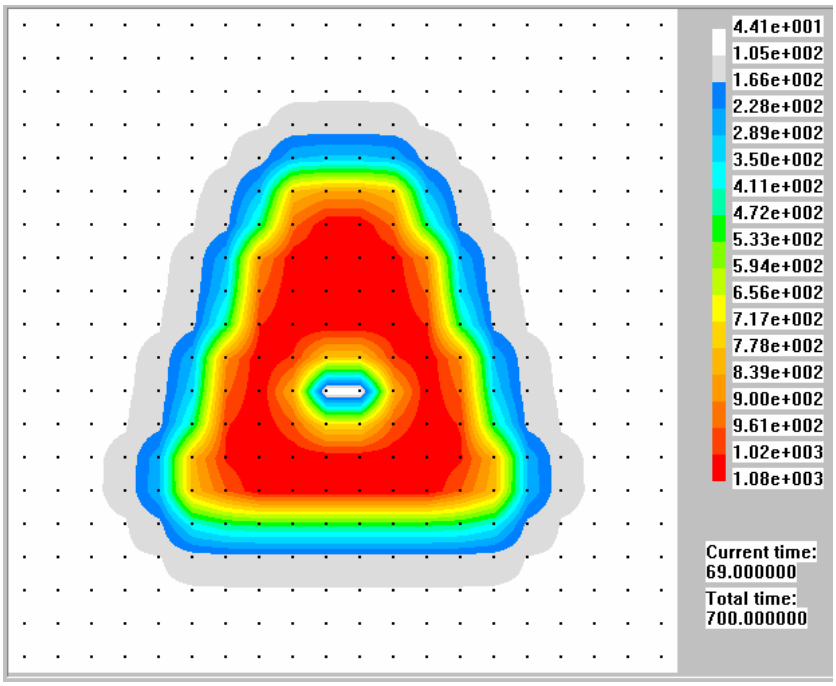
Rys. 5.30. Wyniki dla symulacji z rurką chłodzącą i sterowaniem: a) udział fazy stałej, b) promień kryształów, c) liczba kryształów



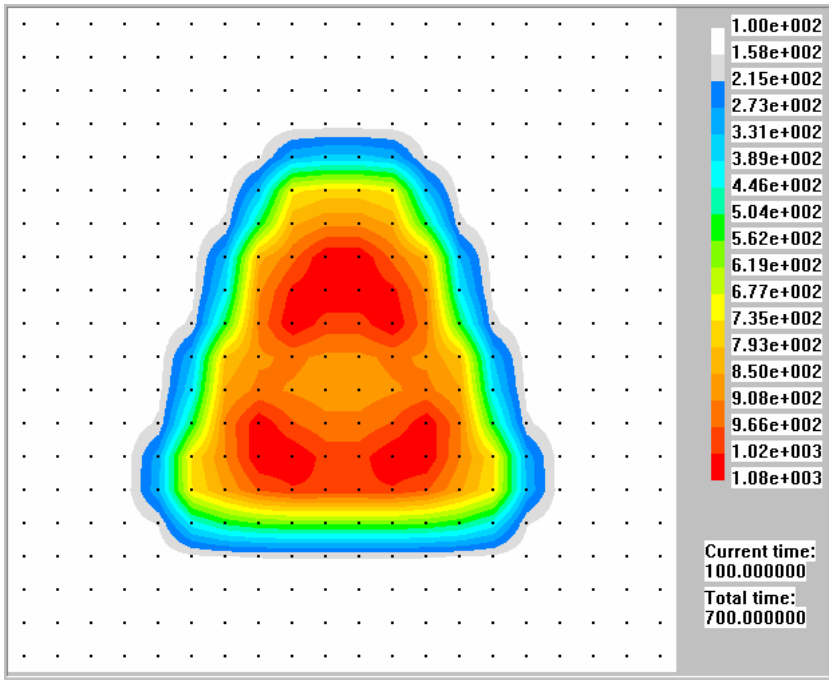
Rys. 5.31. Pole temperatur w przekroju pionowym
(wpływ przepływu rurki chłodzącej)



Rys. 5.32. Pole temperatur w przekroju poziomym
(wpływ przepływu rurki chłodzącej)



Rys. 5.33. Przekrój Y-Z, rurka chłodząca – sterowanie



Rys. 5.34. Przekrój Y-Z, rurka chłodząca - sterowanie

5.11. Podsumowanie

Podsumowując wyniki zaprezentowane w niniejszym rozdziale należy zauważyć, że pomimo krótkiego czasu powstawania systemu MAFES-1 oraz stosunkowo małych nakładów w porównaniu do komercyjnych systemów symulacyjnych (takich jak np. ABAQUS), udało się w nim rozwiązać problemy uciążliwe w realizacji w innych systemach komputerowych.

Dotyczy to przede wszystkim dokładniejszego uwzględnia w symulacji komputerowej procesu krzepnięcia oraz wprowadzenia dynamicznych elementów chłodzących pozwalających na uzyskiwanie odlewów o wysokiej jakości.

Dalsza rozbudowa systemu MAFES-1 dotyczyć może rozszerzenia modeli stygnięcia i krzepnięcia odlewów z miedzi i aluminium, udoskonalania algorytmu realizowanego przez agenta obliczeniowego, a także wizualizacji 3D.

Dotychczasowy regularny kształt siatki wymaga przy gęstej siatce (setki milionów agentów obliczeniowych) bardzo dużej mocy obliczeniowej (rozmiar pamięci, szybkość procesorów). Interesującym i pożądanym dążeniem jest więc opracowanie konstrukcji siatki nieregularnej, adaptującej się do warunków istniejących w symulowanym modelu komputerowym oraz związanych z taką konstrukcją siatki algorytmów obliczeniowych.

Oceniając całościowo rozwiązania i rezultaty opisane w tym rozdziale, sformułować można następujące stwierdzenia:

1. Zrealizowany system symulacyjny MAFES-1 stanowi uniwersalne narzędzie umożliwiające badanie procesu stygnięcia i krzepnięcia odlewów, z uwzględnieniem sterowania tym procesem poprzez zmianę warunków chłodzenia.
2. W porównaniu z oprogramowaniem komercyjnym (np. typu ABAQUS) system MAFES-1 jest bardziej elastyczny w odniesieniu do możliwości włączania dodatkowych elementów oprogramowania, szczególnie dotyczących monitoringu i sterowania.
3. Traktując uzyskane rezultaty eksperymentów symulacyjnych jako weryfikację poprawności działania systemu oraz wstępne potwierdzenie zasadności proponowanych rozwiązań w zakresie sterowania procesami cieplnymi w odlewie, można stwierdzić, że uzyskany wynik jest zadowalający.
4. Bardziej szczegółowa analiza otrzymanych wyników pokazuje, że efekty sterowania w znacznym stopniu zależą zarówno od rozmieszczenia i liczby monitorowanych punktów odlewu, jak też od konfiguracji odlewu i sposobu doprowadzenia medium chłodzącego.
5. Istotna wartość przeprowadzonych przy użyciu systemu MAFES-1 eksperymentów symulacyjnych polega na wykazaniu, że z zastosowaniem współczesnych środków informatycznych, możliwe jest sterowanie przebiegiem procesu stygnięcia i krystalizacji. Stwierdzenie to posiada duże znaczenie, szczególnie, że w dostępnej literaturze brak jest jakichkolwiek doniesień dotyczących sterowania przestrzennego (w 3D) procesami

opisanymi nieliniowymi równaniami cząstkowymi. Tak więc, choć otrzymane rezultaty nie pozwalają jeszcze na sformułowanie wniosków o charakterze ilościowym, wskazują jednak na celowość kontynuacji podjętych badań i potwierdzają skuteczność koncepcji opartej na zastosowaniu systemu symulacyjno-sterującego zrealizowanego w technologii agentowej.

6. W ramach kontynuacji danego kierunku badań należy zwrócić uwagę na dokładniejsze przebadanie struktury informacyjnej systemu, a szczególnie bardziej wnikliwe opracowanie algorytmu sterowania (m. in. poprzez wprowadzenie uczenia i predykcji).

6. System agentowy z własnym środowiskiem

6.1. Założenia koncepcyjne systemu MAFES-2

Współczesny poziom rozwoju informatyki dostarcza wiele metod komputerowego modelowania i symulacji szerokiego zakresu procesów zachodzących w rzeczywistym świecie. Zaliczyć do nich można ciągle rozwijające się i specjalizujące metody numeryczne, których istotnymi przykładami mającymi już swoją długoletnią historię są metoda różnic skończonych czy metoda elementów skończonych.

Metody te zazwyczaj służą do modelowania tylko pojedynczych zjawisk dla których istnieją modele matematyczno – fizyczne opisane odpowiednimi typami równań różniczkowych cząstkowych. Dlatego zaczęto udoskonalać te metody i rozwijać ich modele matematyczne, dla coraz większego obszaru zastosowań. Wskutek tego powstały liczne specjalistyczne odmiany wspomnianych metod, służące modelowaniu bardziej złożonych procesów. W rozwiązaniach tych kontynuowano rozwój obranego aparatu matematycznego będącego ich podstawą i uzasadnieniem, co doprowadziło stosowany aparat do dużej złożoności.

Równolegle do rozwoju metod numerycznych nastąpił rozwój metodologii programistycznych. Powstała koncepcja programowania obiektowego i szereg języków coraz lepiej ją realizujących. Okazało się, że pojęcie obiektu nie nadaje się zbytnio do odwzorowywania zjawisk fizycznych a tylko bytów, w których można wyróżnić pewien stan i operującą na nim funkcjonalność.

Kolejną koncepcją modelowania komputerowego stanowi podejście agentowe. W systemach tej klasy można definiować agentów, ich grupy, relacje wzajemne i role agentów oraz różne modele środowisk, w których one działają.

Wydaje się, że podejście agentowe może być szczególnie przydatne w dziedzinie symulacji zjawisk fizycznych, między innymi zjawisk cieplnych rozważanych w pracy.

Przeprowadzano badania symulacyjne dotyczące procesów cieplnych, bazujące na systemie ABAQUS, wykorzystującym metodę elementów skończonych, opisane w pracach [34, 50, 51]. Wynikiem tego rozpoznane zostały jego różne ograniczenia, które stały się przyczyną szeregu podejść, prowadzących do agentowego modelu procesów cieplnych.

Pierwsza z prezentowanych w pracy koncepcji (rozdział 4) dotyczyła realizacji systemu ABAG współpracującego z systemem ABAQUS. Podejście to polegało na łączeniu symulacji w systemie ABAQUS z symulacją dynamicznego elementu chłodzącego w systemie agentowym ABAG. W systemie ABAG przemieszczające się agenty modelowały ruch i wymianę ciepła, natomiast środowisko, w którym działały te agenty spełniało rolę

interfejsu do systemu elementów skończonych ABAQUS. Za pomocą procedury użytkownika DISP, ustalającej warunki brzegowe następowała wymiana informacji z środowiskiem systemu ABAG.

Wadą tego rozwiązania jest konieczność korzystania z więcej niż jednej platformy symulacyjnej oraz realizacji współdziałania systemów. Dodatkowo, komunikacja między systemami wydłuża czas obliczeń procesu symulacji.

Drugie podejście (system MAFES-1, rozdział 5) stanowiło samodzielny agentowy system symulujący proces stygnięcia i krystalizacji z dodatkowym zjawiskiem przemieszczającego się medium chłodzącego. Zadaniami spełnianymi przez agentów w tym podejściu były wydzielone role w modelu symulacji i sterowania (agent obliczeniowy, agent regulator, agent sterujący).

W rozwiązaniu tym występowała duża złożoność agenta obliczeniowego, wynikająca z realizacji przez niego wszystkich zjawisk występujących w reprezentowanym przez niego elemencie przestrzeni fizycznej. Poza tym, wprowadzanie nowych typów zjawisk wymagałoby dalszej rozbudowy agenta obliczeniowego, prowadzącej do jeszcze większej jego złożoności.

W systemie MAFES-1 miała również miejsce duża ilość interakcji wzajemnych między agentami obliczeniowymi, wynikająca z przechowywania przez agenty stanu. Dodatkowo w podejściu tym, większość zadań symulacyjnych typu wizualizacja czy rejestracja wyników realizowana była w sposób nieagentowy (strukturalny, obiektowy).

Kolejne, trzecie podejście (system MAFES-2), opisane w niniejszym rozdziale, dotyczy agentowego modelu tej samej klasy procesów cieplnych. Model ten jest bliski pojęciowo rozważanym typom zjawisk. Tutaj o agencie można powiedzieć, że nie służy wyłącznie realizacji wydzielonych ról (sterowanie, obliczenia, regulacja) – tak jak to miało miejsce w przypadku systemu MAFES-1. Agent w tym przypadku odwzorowuje zjawisko fizyczne, jest jego odpowiednikiem w modelu symulacyjnym. Wyróżnia się zjawiska proste, elementarne (typu: wymiana ciepła, krystalizacja, ruch) oraz złożone będące nałożeniem się na siebie zjawisk prostych (np. współdziałanie zjawiska krystalizacji i wymiany ciepła lub nałożenie zjawiska wymiany ciepła i ruchu).

Zakładając, że *proces* jest opisany przez zestaw atrybutów (parametrów) można zdefiniować *zdarzenie* jako skokową zmianę wybranego parametru. Wówczas można powiedzieć, że *agentowy model zjawiska* reprezentuje zmiany parametrów w określonym przedziale czasu (sekwencję zdarzeń).

Zjawisko złożone jest tutaj reprezentowane przez hierarchię struktur agentów, która odpowiada jego dekompozycji na procesy składowe. Interakcje tych agentów, poprzez modyfikacje wspólnego środowiska, odpowiadają współistnieniu i wzajemnemu oddziaływaniu procesów składowych.

Prezentowany tutaj model symulacyjny można określić jako wirtualny świat działających w wspólnym środowisku agentów, który odwzorowuje zjawiska fizyczne (cieplne, ruch) występujące w realnym świecie. Model ten ma służyć do wspomagania projektowania procesu odlewniczego, a w szczególności do określenia warunków stygnięcia odlewu, zapewniających uzyskanie odpowiedniej struktury krystalicznej metalu.

System realizujący prezentowaną metodę nazwano MAFES-2 (ang. Multi Agent Finite Environment System) podobnie jak system opisywany w rozdziale 5 – MAFES-1. Różnica polega na interpretacji czwartej litery jego nazwy. Tutaj oznacza ono środowisko, dla podkreślenia wyróżnionego w modelu symulacyjnym środowiska, przechowującego stan modelowanej rzeczywistości (w rozdziale 5 litera E związana była ze słowem Element, gdyż agent reprezentował w tamtym podejściu fragment – element przestrzeni fizycznej modelu).

Podstawowymi modelami występującymi w systemie MAFES-2 są:
a) *model wejściowy* – reprezentujący całą konfigurację symulowanego modelu (obiekty, zjawiska, zadania i parametry symulacji),
b) *model agentowy* – model symulacyjny stanowiący agentową reprezentację modelu wejściowego.

Dane wejściowe dla algorytmu działania systemu MAFES-2 (algorytm 6.1) stanowi tekstowy plik *imf*, opisujący konfigurację symulacji (linia 1). Plik ten zapisany jest w stworzonym specjalnie do tego celu języku MMD (ang. *Mafes Model Data*) przedstawionym szerzej w dodatku B.1. Opisuje on między innymi geometrie symulowanych obiektów, ich parametry materiałowe, czas symulacji, zapisywane wyniki, krok przestrzenny siatki węzłów.

```
1.  input: imf : MMD-file
2.  output: rsf : MRD-files, visualization
3.  inputModel := read ( imf )
4.  agentModel := AgentModel.new ( inputModel )
5.  agentModel.prepare
6.  repeat
7.          agentModel.step
8.  until agentModel.time >= inputModel.endTime
```

Algorytm 6.1. Nadrzędny algorytm symulacji w systemie MAFES-2

Rezultatami algorytmu (lina 2) są pliki MRD (ang. *Mafes Results Data*) i wyniki obserwowane w wizualizacji, która może być opcjonalnie dokonana w trakcie symulacji (zadania te realizowane są przez agentów wizualizacji i zapisywania wyników).

Algorytm działania systemu MAFES-2 rozpoczyna się od wczytania pliku *imf*, opisującego konfigurację symulacji (linia 3). Wynikiem operacji **read** jest obiekt *inputModel*, na podstawie którego tworzony jest agentowy model symulacyjny *agentModel* (linia 4). Operacja **prepare** modelu agentowego (linia 5) wymagana jest w celu przeprowadzenia wizualizacji stanu początkowego symulacji i jego utrwalenia w plikach MRD.

Pętla główna algorytmu to iteracja kroków modelu agentowego, (wyznaczanie nowego stanu, odświeżanie ewentualnej wizualizacji i zapisywanie aktualnego stanu symulacji), do momentu osiągnięcia czasu końcowego (linie 6-8).

Warto zwrócić uwagę, że operacje wizualizacji związane są z interakcją z użytkownikiem symulacji poprzez interfejs graficzny. Interakcja ta umożliwia rozpoczęcie symulacji, wstrzymanie i wznowienie symulacji. Interakcja z użytkownikiem poprzez interfejs graficzny w zakresie wizualizacji umożliwia tworzenie tzw. okien wizualizacji (przekroje, charakterystyki czasowe) dowolnego parametru obliczanego w symulacji.

6.2. Model agentowy

Model agentowy składa się ze środowiska oraz działających w tym środowisku grup agentów dwóch kategorii: zjawisk fizycznych i zadań symulacji.

Środowisko przechowuje stan modelowanej rzeczywistości, pewien jej obszar w którym występują badane obiekty i w którym zachodzą analizowane zjawiska.

Pierwsza kategoria agentów to agenty zjawisk fizycznych. Stanowią one szczególną cechę systemu MAFES-2. Wprowadzenie tej kategorii agentów pozwala na przybliżenie procesu analizy problemu i jego implementacji w systemie symulacyjnym. Agenty tej kategorii reprezentują zjawiska proste, natomiast ich współdziałanie odpowiada nakładaniu się zjawisk fizycznych na siebie.

Drugą kategorię stanowią agenty zadań, które wymagane są do realizacji przeprowadzanej symulacji. Agenty tej kategorii zostały już wprowadzone w systemie MAFES-1 (agent regulator, sterujący). W systemie MAFES-2 dodatkowo zrealizowano w sposób agentowy pozostałe zadania symulacji (wizualizacja, rejestracja wyników). Agenty tej drugiej kategorii współdzielą środowisko wraz z agentami zjawisk.

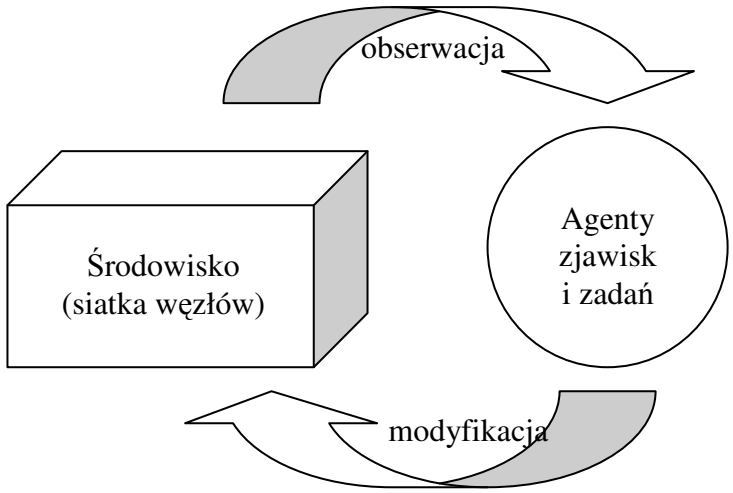
Zarówno agenty zjawisk fizycznych jak i agenty zadań przejawiają wspólny agentowy wzorzec działania w postaci dwóch operacji: obserwacji i modyfikacji środowiska.

Symulowane zjawiska fizyczne (ich dynamika) odwzorowywane są przez agentów zjawisk i ich grupy, które działając we wspólnym środowisku modyfikują jego stan (zmieniają własności środowiska zgodnie z dynamiką tych zjawisk).

Rozpatrywanymi zjawiskami (opisanymi szerzej w rozdziale 2.1) są:
- zjawisko wymiany ciepła – realizowane przez agenty typu **AgentT**,
- zjawisko krystalizacji – realizowane przez agenty typu **AgentCr**,
- zjawisko ruchu – realizowane przez agenty typu **AgentR**.

Agenty służą również do realizacji zadań takich jak:
- zapisywanie (rejestracja) wyników symulacji – realizowane przez agenty typu **AgentS**,
- wizualizacja wyników symulacji – realizowane przez agenty typu **AgentV**,
- predykcja przyszłego stanu symulacji – realizowane przez agenty typu **AgentP**,
- sterowanie – realizowane agenty typu **AgentC**.

Ogólna koncepcja struktury modelu symulacyjnego została przedstawiona na rys. 6.1.



Rys. 6.1. Ogólna struktura modelu symulacyjnego w systemie MAFES-2

Tworzenie modelu agentowego w systemie MAFES-2

Sposób w jaki tworzony jest model agentowy przedstawiono w algorytmie 6.2. Operacja **checkDt** (linia 4) sprawdza czy wartość kroku czasowego *dt* spełnia warunek stabilności obliczeń. Następnie tworzone jest środowisko (linia 5) na podstawie modelu wejściowego. Kolejnym krokiem jest stworzenie agentów zjawisk w operacji **createPhenomenaAgents** (linia 6) oraz agentów zadań w operacji **createTaskAgents** (linia 7).

Operację tworzenia struktury grup agentów zjawisk dla badanej dziedziny procesów cieplnych przedstawiono w algorytmie 6.3. Agent *aAll* reprezentuje korzeń tej struktury. Agent *aTG* grupuje agentów wymiany ciepła (typu **AgentT**). Agent *aCrG* grupuje agentów

krystalizacji (typu **AgentCr**). Zmienna *aRG* grupuje agentów ruchu (typu **AgentR**). Grupy te tworzone są w liniach 4-7.

```
1.    function AgentModel.new ( im : InputModel ) : AgentModel
2.    begin
3.        this.dt := im.dt
4.        checkDt ( im )
5.        this.env := Environment.new ( im , this.dt )
6.        createPhenomenaAgents ( im )
7.        createTaskAgents ( im )
8.        return this
8.    end function
```

Algorytm 6.2. Algorytm tworzenia nowego modelu agentowego

```
1.    function AgentModel.createPhenomenaAgents ( im : InputModel )
2.    var aAll , aTG , aCrG, aRG : AgentGroup
3.    begin
4.        aAll := AgentGroup.new
5.        aTG := AgentGroup.new
6.        aCrG := AgentGroup.new
7.        aRG := AgentGroup.new
8.        for each node in env do
9.            aTG.add( AgentT.new ( node ) )
10.           if node.material.latentHeat > 0 then
11.               aCrG.add ( AgentCr.new ( node ) )
12.           end if
13.       end for
14.       for each figureMotion in im do
15.           aRG.add ( AgentR.new ( figureMotion ) )
16.       end for
17.       aAll.add ( aTG )
18.       aAll.add( aCrG )
19.       aAll.addSync( aRG )`
20.       agent := aAll
21.   end function
```

Algorytm 6.3. Algorytm tworzenia agentowej hierarchii zjawisk dla procesów cieplnych (operacja **createPhenomenaAgents**)

Pierwsza pętla (linie 8-13) przebiega po wszystkich węzłach środowiska *env*, tworząc w każdym z nich agenta wymiany ciepła (linia 9). Dodatkowo, jeśli własności środowiska danego węzła oznaczają występowanie w nim efektu przemiany fazowej (*latentHeat* > 0), tworzony jest w nim agent krystalizacji (linia 11).

Druga pętla (linie 14-16) dla każdego obiektu ruchomego zdefiniowanego w modelu wejściowym tworzy odpowiadającego mu agenta ruchu (typu **AgentR**).

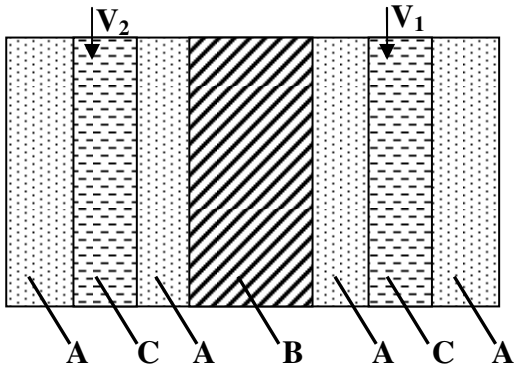
Wszystkie trzy grupy agentów dodawane są do grupy – korzenia, łączącej agentów wszystkich zjawisk (linie 17-19).

Zależności między agentami grupującymi *aAll*, *aTG*, *aCrG*, *aRG* dla przykładowego modelu przedstawione zostały na rys. 6.3.

Model fizyczny i jego model agentowy

Poniżej (rys. 6.2) przedstawiono przykładowy model fizyczny (odlew, forma, dynamiczne media chłodzące). Występują w nim wszystkie trzy zjawiska fizyczne rozpatrywane w pracy (wymiana ciepła, krystalizacja, ruch).

Zjawisko wymiany ciepła obejmuje wszystkie trzy obszary A, B i C. Zjawisko krystalizacji zachodzące w stygnącym odlewie nakłada się na zjawisko wymiany ciepła w obszarze B. Natomiast zjawisko ruchu (dynamicznych mediów chłodzących) współdziała ze zjawiskiem wymiany ciepła w obszarach C.



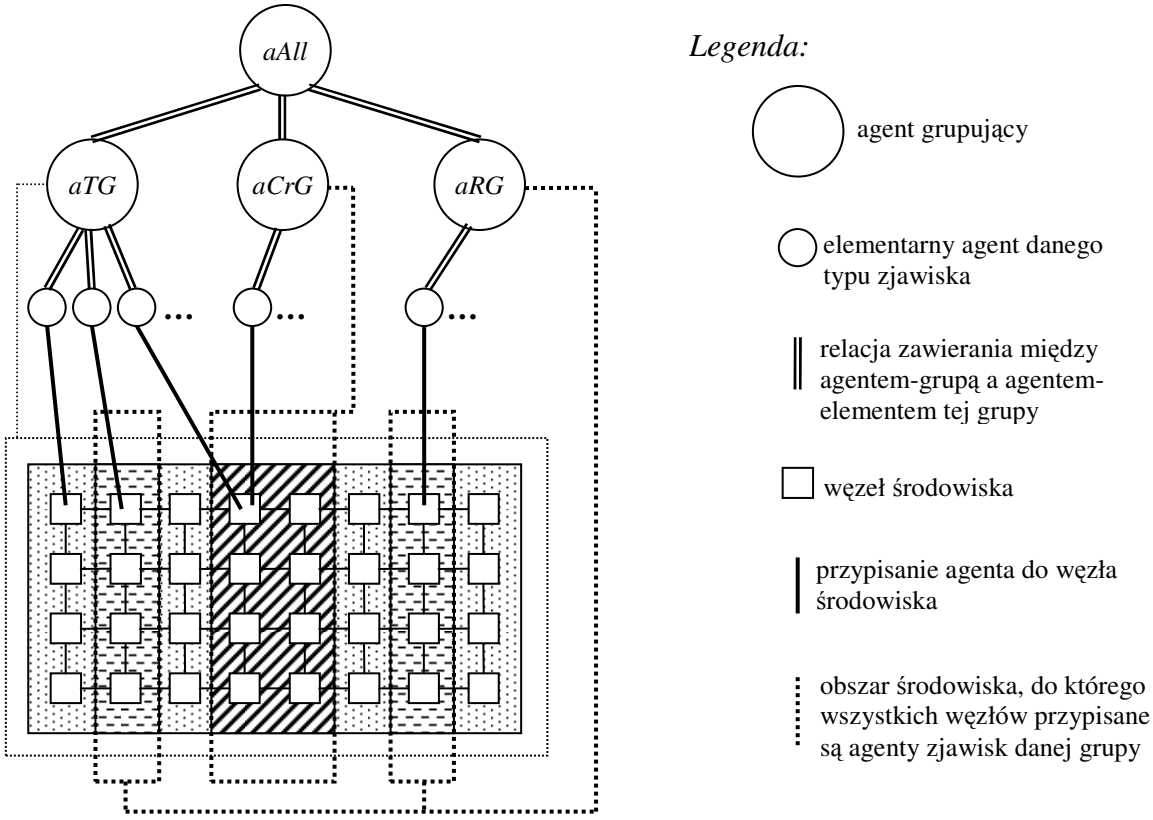
Legenda:

A - forma
B - odlew
C - dynamiczne medium chłodzące
V - prędkość przepływu medium

Rys. 6.2. Przykładowy model fizyczny, w którym występuje współdziałanie trzech typów zjawisk (wymiana ciepła, krystalizacja, ruch)

Strukturę zależności agentów dla przykładowego modelu fizycznego pokazano na rys. 6.3. Do węzłów środowiska przypisywane są agenty zjawisk występujące w danym obszarze środowiska. Współistnienie zjawisk oznacza przypisanie do węzła kilku typów agentów zjawisk.

Występujące etykiety nazw agentów grupujących (*aAll*, *aTG*, *aCrG*, *aRG*) są zgodne z nazwami występującymi w algorytmie 6.3. Gupa *aTG* łączy wszystkie agenty wymiany ciepła, reprezentując zjawisko wymiany ciepła w całym modelu. Grupa *aCrG* łączy agenty krystalizacji, reprezentując zjawisko krystalizacji w całym modelu. Grupa *aRG* łączy agenty ruchu, reprezentując zjawisko ruchu w całym modelu. Natomiast grupa *aAll* łączy te trzy grupy (*aTG*, *aCrG*, *aRG*), reprezentując wszystkie zjawiska występujące w modelu.



Rys. 6.3. Przykładowy model fizyczny, w którym występuje współdziałanie trzech typów zjawisk (wymiana ciepła, krystalizacja, ruch)

Agenty należące do grupy *aTG* przypisane są do wszystkich węzłów środowiska. Agenty grupy *aCrG* przypisane są do dwóch środkowych kolumn węzłów, objętych odpowiednią linią przerywaną doprowadzoną do grupy *aCrG*. Agenty ruchu (grupa *aRG*) przypisane są do drugiej i przedostatniej kolumny węzłów.

Zatem do węzłów drugiej kolumny przypisywane są zarówno agenty wymiany ciepła jak i agenty ruchu. Natomiast do kolumny czwartej i piątej przypisywane są agenty wymiany ciepła i krystalizacji. Do węzłów pozostałych kolumn przypisywane są tylko agenty wymiany ciepła.

6.3. Środowisko

Środowisko w systemie MAFES-2 zbudowane jest z siatki regularnie rozmieszczonych węzłów dla modelowanej przestrzeni. Każdy węzeł przechowuje stan odpowiadającego mu punktu modelowanej przestrzeni, jego własności. Do węzła może być przypisany jeden lub więcej agentów (np. agent zjawiska wymiany ciepła – **AgentT** oraz

agent zjawiska krystalizacji – **AgentCr**) w zależności od ilości zjawisk zachodzących w danym punkcie przestrzeni.

Przedstawiane środowisko (skończone – ang. *finite*), składa się z pewnej ograniczonej liczby węzłów siatki. Liczba ta jest tym większa im mniejsza jest odległość między sąsiadującymi węzłami w siatce. Odległość ta określa stopień dokładności odwzorowania modelu, gdyż im gęstsza siatka tym więcej będzie stworzonych agentów pewnego zjawiska w danym podobszarze (jeden agent dla jednego węzła tego podobszaru).

Atrybuty węzłów środowiska

Główną częścią operacji tworzenia środowiska na podstawie modelu wejściowego jest zbudowanie regularnej siatki węzłów i ustanowienie relacji sąsiedztwa między nimi. Każdy węzeł połączony jest z sąsiadującymi z nim węzłami. Połączenie zrealizowane jest za pomocą referencji. Każdemu węzłowi środowiska przypisywane są wartości parametrów odpowiadającego mu punktu w modelu wejściowym.

W przypadku rozważanych w pracy zjawisk cieplnych, węzeł środowiska zawiera następujące atrybuty służące do przechowywania stanu środowiska:

a) temperatura **T** (zjawisko wymiany ciepła),
b) współczynnik udziału fazy stałej $\mathbf{F_S}$ (zjawisko krystalizacji),
c) ilość kryształów w jednostce objętości **N** (zjawisko krystalizacji),
d) średni promień kryształów **R** (zjawisko krystalizacji),
e) zmienna oznaczająca czy nastąpił już punk przechłodzenia **CGN** (zjawisko krystalizacji),
f) odniesienie (referencja) do materiału, który jest zbiorem parametrów takich jak: przewodność cieplna, ciepło właściwe, gęstość, temperatura eutektyki, ciepło krystalizacji, potrzebnych do realizacji działań agentów zjawisk (wymiana ciepła, krystalizacja).

Rola środowiska w modelu agentowym systemu MAFES-2

Środowisko w prezentowanym systemie agentowym przechowuje stan modelu symulacyjnego, co minimalizuje ilość informacji koniecznej do pamiętania przez agenty zjawisk.

Dzięki przedstawionej koncepcji środowiska możliwe jest modelowanie współbieżnego działania zjawisk fizycznych, przy synchronicznej (iteracyjnej) realizacji agentowych oddziaływań na środowisko. Gdy jedno zjawisko (reprezentujący je agent) nie jest aktywne, to drugie zjawisko współdziała z nim poprzez uwzględnianie zmian w środowisku, które zostały wprowadzone przez tamto, pierwsze zjawisko, kiedy było ono aktywne.

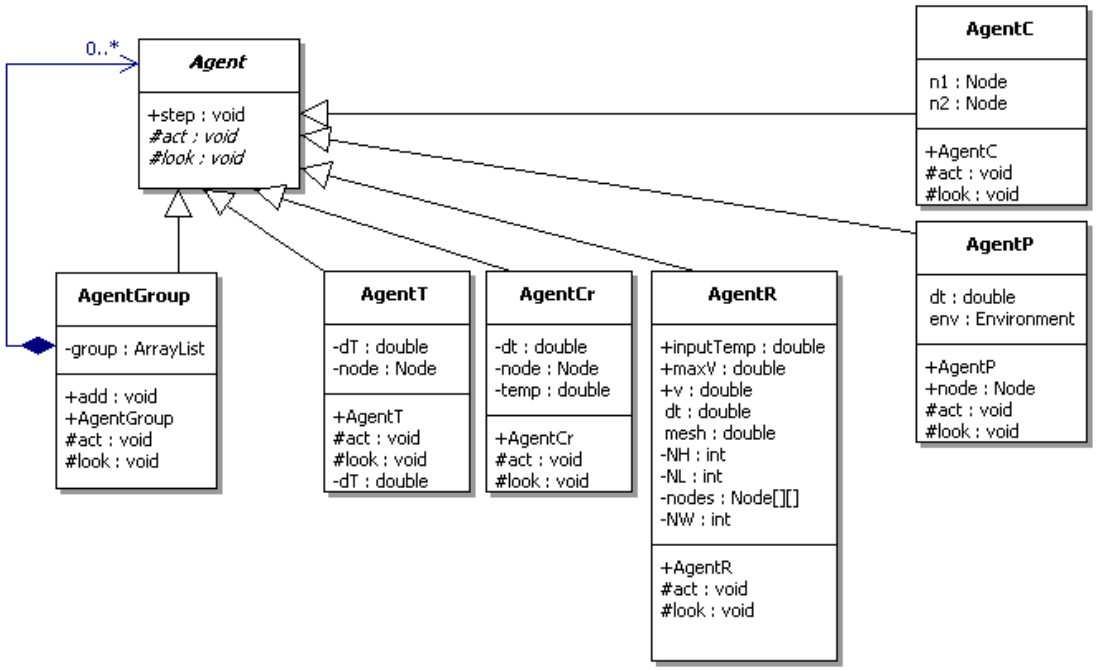
6.4. Agenty

Agent jest aktywnym elementem systemu MAFES-2. Działa w określonym miejscu środowiska (pojedynczy węzeł lub grupa węzłów) lub może nie być związany z węzłami środowiska, tylko z innymi agentami przez relację komunikacji.
Agent może reprezentować:

a) zjawisko (mogące się składać ze zjawisk elementarnych),
b) fragment zjawiska – dekompozycja przestrzenna,
c) zjawisko składowe zjawiska złożonego,
d) realizację określonego zadania (wizualizacja, sterowanie, zapisywanie wyników).

Odwzorowanie fragmentu zjawiska przez agenta zjawiska określonego typu (podpunkt b) oznacza podzielenie zjawiska o jednakowej naturze na grupę agentów, w związku z obejmowaniem przez to zjawisko pewnego podobszaru przestrzeni środowiska, składającego się z wielu węzłów (grupa agentów – w każdym węźle jeden agent tego samego typu – np. grupa agentów wymiany ciepła).

Reprezentacja zjawiska składowego (podpunkt c) wynika z modelowania zjawisk złożonych, składających się z podzjawisk, które to współistnieją w tym samym obszarze (np. zjawisko wymiany ciepła i zjawisko krystalizacji). Kilku agentów różnych typów zjawisk może być przypisanych do tego samego węzła środowiska. Oznacza to współdziałanie tych zjawisk w tym samym miejscu.



Rys. 6.4. Reprezentacja wybranych typów agentów systemu MAFES-2 za pomocą klas (diagram w notacji UML)

Na rys. 6.4 przedstawiono diagram UML klas agentów występujących w systemie MAFES-2 (bez agentów zadań: wizualizacji i zapisywania wyników).

W dalszej części rozdziału opisano dwa podstawowe typy agentów. Pierwszy z nich, *agent uogólniony* (klasa **Agent**) jest abstrakcją wszystkich typów agentów występujących w systemie MAFES-2. Drugim ważnym typem agenta, który jest wykorzystywany zarówno przy zastosowaniu agentów do reprezentacji zjawisk, jak i przy użyciu agentów do realizacji zadań jest *agent grupujący* (klasa **AgentGroup**).

Agent uogólniony (klasa Agent)

Agent ten zaimplementowany jest jako klasa w sensie teorii języków obiektowych. Jest on klasą bazową (wzorcem) dla wszystkich typów agentów występujących w systemie MAFES-2 (wszystkie pozostałe klasy agentów dziedziczą z tej abstrakcyjnej klasy).

Cechą tego modelu agenta jest występowanie w jego działaniu dwóch funkcji:
- *look* (obserwacja środowiska),
- *act* (oddziaływanie na środowisko – modyfikowanie go, lub komunikacja z innymi agentami).

Sekwencja tych funkcji stanowi krok (funkcja *step*) działania agenta.

Związek agenta ze środowiskiem ogranicza się do tych właśnie dwóch funkcji, które odnoszą się do środowiska danego agenta.

Agent może być przypisany do określonego węzła środowiska (np. agent wymiany ciepła – klasa **AgentT**) lub do grupy węzłów środowiska (np. agent ruchu – klasa **AgentR**). Natomiast agenty niektórych typów (np. agent sterujący - klasa **AgentC**) nie są przypisywane do węzłów środowiska. Dlatego agent uogólniony nie zawiera referencji do węzła środowiska. Referencja pojawia się w wybranych typach agentów dziedziczących z klasy **Agent**.

Implementacja klasy **Agent** w systemie MAFES-2 zawiera abstrakcyjne definicje funkcji *look* i *act*. Klasy dziedziczące z niej dostarczają właściwych dla siebie implementacji tych dwóch metod.

Agent grupujący (klasa AgentGroup dziedzicząca z klasy Agent)

Agent grupujący jest podtypem klasy **Agent** co oznacza, że może być traktowany jako pojedynczy agent z funkcjami *look* i *act*. Służy do ukrywania złożoności przestrzennej, czyli

występowanie jednego zjawiska w pewnym obszarze, grupie węzłów. Pozwala również na modelowanie współistnienia różnych typów zjawisk w tym samym miejscu.

Agent ten udostępnia funkcję *add*, która dodaje agenta klasy *Agent* i automatycznie wszystkich jego podtypów do swojej grupy. Możliwe są dwa rodzaje zastosowania tego typu agenta. Agent może stanowić grupę innych agentów:

a) działających w różnych węzłach środowiska, reprezentujących ten sam typ zjawiska, rozłożonego w przestrzeni,
b) działających w tych samych lub różnych węzłach środowiska, reprezentujących różne typy zjawisk czyli składowe modelowanej przestrzeni zjawisk.

Dopuszczalne jest rekursywne zastosowanie tego typu agentów, czyli tworzenie grup agentów, które są również grupami innych agentów (klasa **AgentGroup** dziedziczy z klasy **Agent**, zatem może być elementem innej grupy). Dla przykładu grupa agentów zjawiska wymiany ciepła (**AgentT**) może wraz z grupą agentów krystalizacji (**AgentCr**) tworzyć grupę wyższego poziomu reprezentującą zjawisko złożone wymiany ciepła i krystalizacji.

Implementacja funkcji **look** w klasie **AgentGroup** polega na wywołaniu funkcji **look** we wszystkich składowych agentach (algorytm 6.4).

```
1.  function AgentGroup.look
2.  begin
3.      for each agent in group do
4.          agent.look
5.      end for
6.  end function
```

Algorytm 6.4. Algorytm funkcji **look** w klasie **AgentGroup**

Działanie funkcji **look** w klasie **AgentGroup** polega na przekazywaniu (rozgłaszaniu) swoim składowym agentom komunikatu o przeprowadzenie operacji obserwacji (linie 3-5). Nie jest tu istotna kolejność, w której ten komunikat zostanie przekazany. Zatem realizacja komputerowa, wykonanie tych funkcji w agentach składowych może odbywać się asynchronicznie, niezależnie. Może to stanowić inspirację do badań nad równoległą, rozproszoną realizacją działań agentów.

Analogicznie do implementacji funkcji **look** wygląda implementacja funkcji **act** (algorytm 6.5). Pierwsza pętla (linie 3-5) wywołuje funkcję **act** we wszystkich składowych agentach. Agent grupujący może zawierać również grupę *syncGroup*. Druga pętla (linie 6-8)

wywołuje we wszystkich składowych agentach należących do grupy *syncGroup* operację **step**.

```
1.    function AgentGroup.act
2.    begin
3.        for each agent in group do
4.            agent.act
5.        end for
6.        for each agent in syncGroup do
7.            agent.step
8.        end for
9.    end function
```

Algorytm 6.5. Algorytm funkcji **act** w klasie AgentGroup

Pewne zjawiska nie mogą być realizowane przez asynchroniczne wywoływanie operacji obserwacji (funkcja **look**), czy oddziaływania (funkcja **act**) w agentach tych zjawisk. Krok ich działania (funkcja **step**), będący pojedynczym sekwencją funkcji **look** i **act**, dla niektórych zjawisk musi być przeprowadzona z zapewnieniem wyłączności dostępu do środowiska. Działania agentów innych zjawisk, muszą czekać na zakończenie tego cyklu, ich działanie musi być synchronizowane.

W tym celu w klasie **AgentGroup** dodano listę *syncGroup*, która zawiera agentów, których działania muszą być synchronizowane. Została w tej klasie dodana również funkcja **addSync** dodająca agentów do tej grupy. Działania agentów tej grupy, ich funkcje **step** są wywoływane w funkcji **act** po wywołaniu funkcji **act** we wszystkich agentach w grupie agentów asynchronicznych – zmienna *group*.

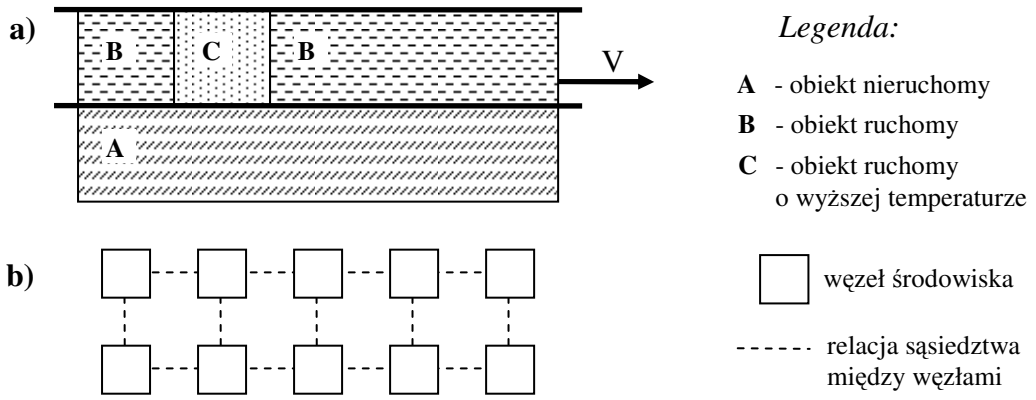
Przykładem zjawiska wśród badanych w pracy zjawisk fizycznych, które musi być realizowane synchronicznie jest zjawisko ruchu wymuszonego, nakładające się na zjawisko wymiany ciepła.

6.5. Agenty zjawisk fizycznych

W systemie MAFES-2 występują następujące typy agentów reprezentujące zjawiska fizyczne:

- agent zjawiska wymiany ciepła (klasa **AgentT**),
- agent zjawiska krystalizacji (klasa **AgentCr**),
- agent zjawiska ruchu (klasa **AgentR**).

W celu dokonania opisu tej kategorii agentów rozważamy następujący model fizyczny (rys. 6.5.a) i jego reprezentację w agentowym środowisku (rys. 6.5.b).



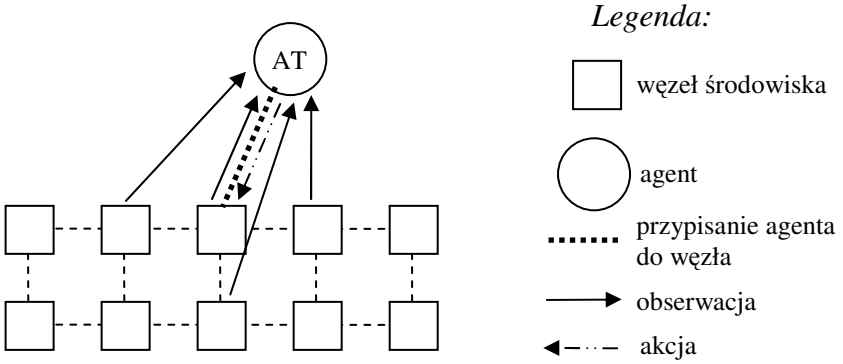
Rys. 6.5. Model fizyczny – a) i jego reprezentacja w dyskretnym środowisku – b)

W przedstawionym modelu przyjęto, że: $T_A = T_B$, $T_C > T_A$, $V_B = V_C > 0$ (T_A , T_B , T_C – temperatury obiektów A, B, C ; V_B , V_C – prędkość przemieszczania się obiektów B i C).

Agent wymiany ciepła (klasa AgentT dziedzicząca z klasy Agent)

Agenty typu **AgentT** realizują swoje zadanie w określonym miejscu środowiska, zostają ulokowane w tych węzłach siatki w których ma miejsce zjawisko wymiany ciepła. Każdy agent tego typu dysponuje referencją do węzła, któremu został przypisany.

Celem agentów typu **AgentT** jest wyznaczanie kolejnych (w dyskretnym czasie symulacji) wartości temperatury (jej zmian) w swoich węzłach zgodnie z prawem wymiany ciepła.



Rys. 6.6. Agent wymiany ciepła i jego relacje ze środowiskiem

Na rys. 6.6 pokazano pojedynczego agenta wymiany ciepła, jego relacje ze środowiskiem w postaci przypisania do węzła i działań: obserwacji (funkcja **look**) i akcji (funkcja **act**). Operacja obserwacji agenta rozważanego typu związana jest z węzłem

środowiska, do którego jest on przypisany oraz wszystkich węzłów sąsiadujących z nim. Operacja oddziaływania (zmian w środowisku) dotyczy tylko węzła, do którego agent jest przypisany.

Funkcja **look** w tym agencie (algorytm 6.6) dokonuje obserwacji różnic temperatur pomiędzy danym węzłem a jego sąsiadami (sześcioma) we wszystkich trzech kierunkach (linie 4-6). Wyznacza zmiany temperatury w tym węźle, pochodzące z oddziaływania z każdym z jego sąsiadów. Następnie oblicza sumaryczną zmianę temperatury w węźle spowodowaną zjawiskiem wymiany ciepła. Uwzględniany jest przy tym kontakt między różnymi ośrodkami, o różnych parametrach termicznych. Realizuje to funkcja **deltaT** (linia 5).

```
1.    function AgentT.look
2.    begin
3.        dT := 0
4.        for each nn in node.neighbours do
5.            dT := dT + deltaT ( nn )
6.        end for
7.    end function
```

Algorytm 6.6. Algorytm funkcji **look** w klasie **AgentT**

```
1.    function AgentT.act
2.    begin
3.        dT := dT * node.material.a
4.        node.T := node.T + dT
5.    end function
```

Algorytm 6.7. Algorytm funkcji **act** w klasie **AgentT**

Agent typu **AgentT**, dysponując wyznaczoną w operacji **look** zmianą temperatury, dokonuje aktualizacji stanu (temperatury) odpowiedniego węzła w funkcji **act** (algorytm 6.7, linia 4). W linii 3 następuje przemnożenie wartości *dT* wyznaczonej w funkcji **look** przez współczynnik *node.material.a*, który reprezentuje parametry termofizyczne materiału, krok przestrzenny siatki węzłów oraz krok czasowy symulacji.

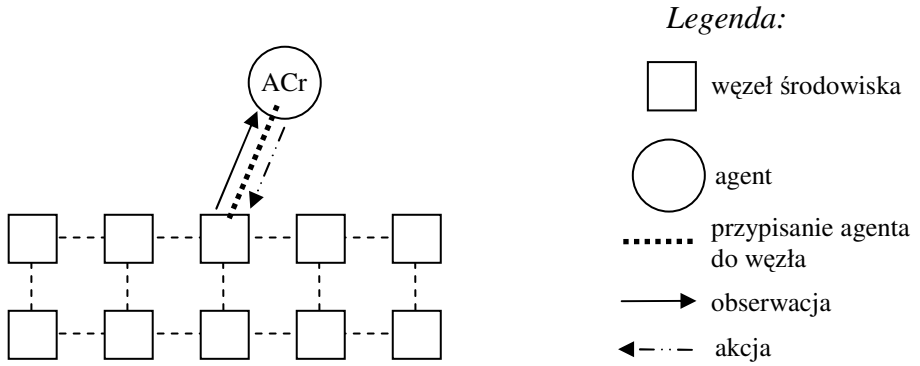
Agenty typu **AgentT** w systemie MAFES-2 działają w grupie (**AgentGroup**) odpowiedzialnej za zjawisko wymiany ciepła w całym modelu. Dzięki odpowiedniej realizacji funkcji **look** i **act** w **AgentGroup** najpierw następuje wyznaczenie nowych wartości dla wszystkich składowych agentów. Później dokonywana jest aktualizacja środowiska przez każdego z nich. Dlatego obserwacje wszystkich agentów wykorzystują ten sam stan środowiska.

Agent krystalizacji (klasa AgentCr dziedzicząca z klasy Agent)

Agenty krystalizacji podobnie jak agenty wymiany ciepła są przypisywane do tych węzłów siatki środowiska, w których występuje rozeważane zjawisko.

Celem tych agentów jest wyznaczanie przyrostu temperatury spowodowanej przemianą fazową, oraz obliczanie parametrów krystalizacyjnych związanych z tą przemianą (średnia ilość kryształów, średni promień kryształów, udział fazy stałej).

Na rys. 6.7 przedstawiono schematycznie agenta krystalizacji, jego relacje ze środowiskiem i działania w postaci operacji obserwacji i akcji.



Rys. 6.7. Agent krystalizacji i jego relacje ze środowiskiem

Operacja obserwacji (funkcja **look**) oraz operacja akcji, oddziaływania (funkcja **act**) dotyczą tylko tego węzła środowiska, do którego dany agent jest przypisany.

Funkcja **look** zapamiętuje bieżącą wartość temperatury w węźle i na jej podstawie obliczane są zmiany parametrów krystalizacyjnych. Zmiana wartości udziału fazy stałej (F_S) służy do wyznaczenia ilości wydzielonego ciepła co z kolei pozwala otrzymać przyrost temperatury związany z zjawiskiem krystalizacji.

```
1.    function AgentCr.look
2.    begin
3.        if node.CGN and node.T > this.T then
4.            node.CGN := false
5.        end if
6.        this.T := node.T
7.    end function
```

Algorytm 6.8. Algorytm funkcji **look** w klasie AgentCr

Algorytm funkcji **look** (algorytm 6.8) rozpoczyna się (linia 3) od sprawdzenia czy nie nastąpiło już przechłodzenie (CGN – ang. Can Growth N). Przechłodzenie występuje w

sytuacji gdy temperatura w danym węźle ulegnie wzrostowi spowodowanemu przewagą ciepła wydzielanego wskutek krystalizacji nad ciepłem oddawanym otoczeniu w efekcie zjawiska wymiany ciepła.

Jeśli nie wystąpił jeszcze efekt przechłodzenia to sprawdzany jest warunek (linia 3) czy bieżąca temperatura jest większa od jej wartości w poprzednim kroku symulacji (zapamiętanej w zmiennej *this.T*). Spełnienie tego warunku (*CGN = false*) oznacza koniec możliwości wzrastania liczby kryształów (linia 4).

W linii 6 dokonywane jest zapamiętanie bieżącej wartości temperatury w zmiennej *this.T*, która będzie potrzebna w kolejnym kroku symulacji. Tak więc agenty tego typu przechowują historię zmian stanu (temperatury) węzła do którego są przypisane.

Funkcja **act** podobnie jak w dla agenta typu **AgentT** powoduje aktualizację temperatury w odpowiadającym węźle środowiska, dodatkowo aktualizowane są nowe wartości parametrów krystalizacyjnych.

Algorytm funkcji **act** agenta typu **AgentCr** (algorytm 6.9) wykorzystuje zmienne lokalne (linia 2) *ltmtp2* oznaczającą kwadrat różnicy temperatury *liquidus* (eutektyki) i temperatury bieżącej w węźle. Druga zmienna lokalna (linia 2) F_S reprezentuje nową wartość współczynnika udziału fazy stałej.

Na początku algorytmu (linie 4-6) sprawdzane jest czy są spełnione odpowiednie warunki, przy których zjawisko krystalizacji zachodzi. Algorytm ten opiera się na modelu krystalizacji równowagi eutektycznej. Czynniki *nf* i *rf* biorące udział przy wyznaczaniu wartości *node.N* (linia 9) i przyrostu *node.R* (linia 11), to stałe wyznaczone doświadczalnie. Zmienna *node.material.ldivs* (linia 13) to iloraz ciepła krystalizacji i ciepła właściwego materiału.

```
1.   function AgentCr.act
2.   var ltmtp2 , F_S : Real
3.   begin
4.       if this.T > node.material.liquidusT or node.F_S >= 1 then
5.           return
6.       end if
7.       ltmtp2 := ( node.material.liquidusT – this.T ) ^ 2
8.       if node.CGN then
9.           node.N := nf * ltmtp2
10.      end if
11.      node.R := node.R + rf * ltmtp2 * dt
12.      F_S := node.N * ( Pi * 4.0 / 3.0 ) * node.R ^ 3
13.      node.T := node.T + node.material.ldivs * ( F_S – node.F_S )
14.      node.F_S := F_S
15.  end function
```

Algorytm 6.9. Algorytm funkcji **act** w klasie AgentCr

Działanie tego typu agenta przebiega w określonym zakresie wyznaczonym przez temperaturę *liquidus* i kończy się z chwilą osiągnięcia przez F_S wartości równej 1 (linia 4). Możliwe byłoby tworzenie agentów typu **AgentCr** w momencie osiągnięcia przez temperaturę w węźle wartości równej wartości temperatury *liquidus*, oraz usuwanie tych agentów w chwili osiągnięcia przez F_S wartości równej 1. Jednak ze względu na uproszczenie algorytmu symulacji agenty tego typu są tworzone na początku symulacji i są utrzymywane do jej końca. Odpowiedni warunek na początku funkcji **act** (linie 4-6) sprawdza czy przemiana fazowa powinna być realizowana w danym węźle.

Analogicznie jak agenty typu **AgentT**, agenty typu **AgentCr** występują w grupie (**AgentGroup**) realizującej zjawisko przemiany fazowej dla całego modelu. Grupa ta wraz z grupą agentów typu **AgentT** jest połączona w grupę wyższego poziomu reprezentującą zjawisko połączone wymiany ciepła i krystalizacji. Zapewnia to, że agenty obu grup swoją obserwację dokonują na podstawie tego samego stanu środowiska, natomiast aktualizacja jest sumą zmian wprowadzonych przez oba typy agentów (w tych węzłach gdzie te dwa zjawiska współdziałają).

Agent ruchu (klasa AgentR dziedzicząca z klasy Agent)

Działanie tego typu agentów polega przenoszeniu własności środowiska w węzłach objętych trajektorią ruchu zgodnie z dynamiką tego ruchu. Dynamika ta ma stałą trajektorię podczas symulacji, aczkolwiek w jej trakcie może być zmieniana prędkość ruchu oraz parametry wejściowe (temperatura wpływającego medium).

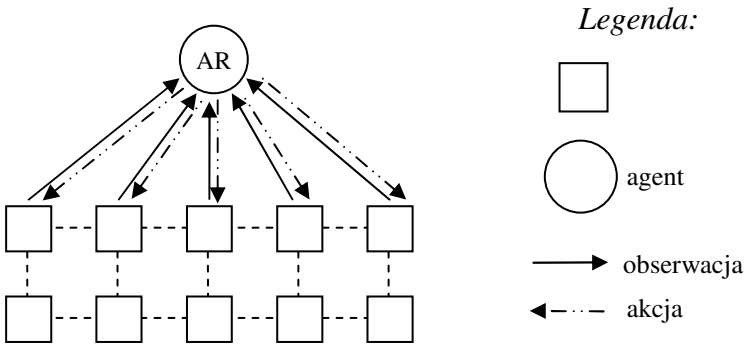
Agenty tego typu służą w szczególności do symulacji przewodów z medium chłodzącym lub przemieszczających się obiektów modelu pod wpływem zewnętrznych sił.

Pierwsze z tych zastosowań jest związane ze sterowaniem w procesie odlewniczym, wówczas agent ruchu pełni rolę regulatora. Funkcja **look** takiego regulatora sprawdza czy nie przyszły nowe decyzje sterujące od agenta sterującego. Funkcja **act** tych agentów realizuje przepływ w przewodzie z medium chłodzącym.

Drugie zastosowanie nie wykorzystuje funkcji **look** do odczytu informacji przesyłanych przez agenta sterującego, natomiast działanie funkcji **act** jest analogiczne – przeniesienie własności węzłów po trajektorii ruchu.

Związek agenta ruchu ze środowiskiem to nie pojedynczy węzeł tak jak to miało miejsce dla poprzednich typów agentów, tylko grupa węzłów – podzbiór siatki środowiska, który jest obejmowany przez trajektorię ruchu tego agenta.

Na rys. 6.8 pokazano agenta ruchu, jego relacje ze środowiskiem w postaci działań obserwacji i akcji.

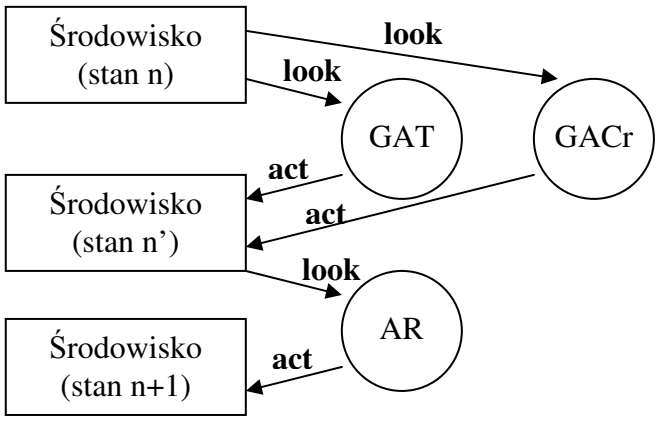


Rys. 6.9. Agent ruchu (AR) i jego relacje ze środowiskiem

Wartość kroku przestrzennego przesunięcia agenta ruchu wyznaczana jest na podstawie prędkości jego ruchu oraz kroku czasowego symulacji.

Zarówno odczyt wartości parametrów środowiska (funkcja **look**) jak i aktualizacja ich (funkcja **act**) poprzez przesunięcie nie może być dokonywana łącznie z analogicznymi operacjami agentów typu **AgentT** i **AgentCr**. Gdyby tak uczyniono to efekt działania agenta typu **AgentR** niwelowałby efekt działania na przykład agentów typu **AgentT** w miejscach środowiska obejmowanych przez agenta ruchu. Wynika to z faktu, że agent ruchu nie wyznacza zmiany temperatury wynikłej z realizacji zjawiska ruchu, tylko dokonuje przeniesienia temperatury (własności) z jednych miejsc środowiska w drugie.

Z tej przyczyny grupa agentów typu AgentR (*aRG*) dodawana jest do grupy wszystkich grup agentów w operacji *addSync* (algorytm 6.3, linia 19), co powoduje, że przy działaniu modelu agentowego funkcje **look** i **act** agentów typu **AgentR** są wywoływane po wszystkich wywołań funkcji **look** i **act** wszystkich pozostałych typów agentów (algorytm 6.5, linie 6-8).



Rys. 6.8. Sekwencja działań agentów zjawisk (GAT – grupa agentów **AgentT**, GACr – grupa agentów **AgentCr**, AR – agent typu **AgentR**)

Zmiana stanu środowiska wynikła z realizacji trzech zjawisk (wymiana ciepła, krystalizacja, ruch) przebiega dwuetapowo. Stan pośredni wynika z zastosowania dwóch

pierwszych zjawisk, natomiast zjawisko ruchu działa na tym stanie pośrednim i wyznacza nowy stan. Tak więc występuje tu sekwencja (iteracja) zjawisk, działania ich agentów na środowisku. Mechanizm ten przedstawiono na rys. 6.8.

6.6. Agenty dodatkowych zadań symulacyjnych

W systemie MAFES-2 występują następujące typy agentów zadań, związanych z samą symulacją:
- agent rejestracji (zapisywania) wyników symulacji (klasa **AgentS**),
- agent wizualizacji wyników symulacji (klasa **AgentV**),
- agent predykcji przyszłego stanu symulacji (klasa **AgentP**),
- agent sterowania (klasa **AgentC**).

Agenty tej kategorii odpowiadają za realizację określonego typu zadania wymaganego od przeprowadzanej symulacji. Zostaną one przedstawione w skróconym opisie, ponieważ zastosowanie agentów do realizowania zadań symulacyjnych zostało szerzej omówione w rozdziale 5.

Agent zapisywania wyników symulacji (klasa AgentS dziedzicząca z klasy Agent)

Operacja obserwacji **look** tego typu agentów pobiera informacje o stanie odpowiednich własności środowiska, jego wybranych obszarów. Funkcja **act** dokonuje zapisu tych danych w plikach wynikowych symulacji.

Ilość agentów tego typu występujących w symulacji jest zależna od danych z pliku wejściowego symulacji. Dodatkowo w pliku tym konfiguracji podlegają zapisywane parametry i miejsca obserwacji.

Agent wizualizacji wyników symulacji (klasa AgentV dziedzicząca z klasy Agent)

Podobnie jak agenty typu **AgentS** agenty typu **AgentV** dokonują obserwacji stanu środowiska w operacji **look**. Funkcja **act** w tym przypadku przenosi te informacje do odpowiednich okien wizualizacji.

Konfiguracja agentów typu **AgentV** jest dokonywana przez użytkownika poprzez interfejs graficzny. Konfigurację tę można przeprowadzić przed rozpoczęciem symulacji i podczas jej trwania.

Agent predykcyjny (klasa AgentP dziedzicząca z klasy Agent)

Agenty typu **AgentP** wykorzystywane są przez agenty sterujące (typu **AgentC**) do dostarczania informacji o przybliżonym przyszłym stanie modelu. Przewidywanie przyszłego stanu modelu można zrealizować na dwa sposoby:

a) Oprócz podstawowego środowiska i podstawowego modelu zjawisk tworzy się nowe środowisko i nowy agentowy model zjawisk dla tego samego modelu wejściowego. Nowy model i nowe środowisko tworzone są na identycznych zasadach jak podstawowe środowisko i podstawowy agentowy model zjawisk. Są jakby ich obrazem (uproszczonym widokiem, modelem modelu). Nowy model ma większy krok czasowy a nowe środowisko większy rozmiar siatki (rzadziej rozmieszczone węzły). Nowy model i środowisko służą do predykcji zachowania się modelu podstawowego. Są źródłem informacji dla sterowania. Po wyznaczeniu przewidywanego przyszłego stanu modelu, model podstawowy wykonuje krok obliczeń. Po tym kroku nowe środowisko aktualizuje swój stan na bazie środowiska podstawowego i nowy model na podstawie tego stanu środowiska wyznacza kolejny przewidywany stan.
b) Agent predykcyjny służy do wyznaczania przyszłego stanu pewnego miejsca (punktu) w środowisku podstawowym. Tworzy on sobie wewnętrzny model zjawisk i wewnętrzne środowisko (obraz środowiska podstawowego, otoczenia badanego punktu). Następnie wykorzystuje ten wewnętrzny model do predykcji.

Agent sterujący (klasa AgentC dziedzicząca z klasy Agent)

Agent sterujący to przykład agenta, który nie jest związany z jednym, określonym miejscem środowiska. Natomiast dokonuje on obserwacji wybranych miejsc środowiska, węzłów, w których odpowiedni przebieg wartości sterowanych własności jest jego celem.

Na podstawie obserwacji podejmuje on decyzje sterujące, które przekazuje agentom ruchu (regulatorom). Może on dodatkowo wykorzystać agenty predykcyjne w celu przewidywania przyszłych stanów modelu i podjęcia odpowiednich decyzji we wcześniejszym, odpowiednim czasie.

W niniejszym rozdziale nie opisano szerzej zadanie sterowania, gdyż zostało to uczynione przy opisie systemu MAFES-1.

Perspektywicznie przewiduje się wprowadzenie do systemu MAFES-2 jeszcze innych typów agentów.

6.7. Eksperymenty symulacyjne

Zostaną przedstawione dwa eksperymenty symulacyjne. Pierwszy z nich ma na celu pokazanie realizacji przez system MAFES-2 współdziałania zjawiska wymiany ciepła i ruchu.

W drugim eksperymencie prezentowana jest symulacja połączonych zjawisk wymiany ciepła i krystalizacji.

Pierwszy eksperyment

Rys. 6.10 przedstawia wyniki uzyskane w systemie MAFES-2 dla modelu fizycznego zaprezentowanego na rys. 6.5, w postaci rozkładów temperatur w różnych momentach czasowych. Kolumna a) pokazuje wyniki dla prędkości ruchu górnej warstwy równej 0.002 m/s, kolumna b) dla prędkości pięciokrotnie większej 0.01 m/s. Kolejne wiersze pokazują wyniki w kolejnych momentach czasowych symulacji. Czasy te zostały tak wybrane by w jednym wierszu znajdowały się wyniki dla tej samej wartości przemieszczenia górnej warstwy.

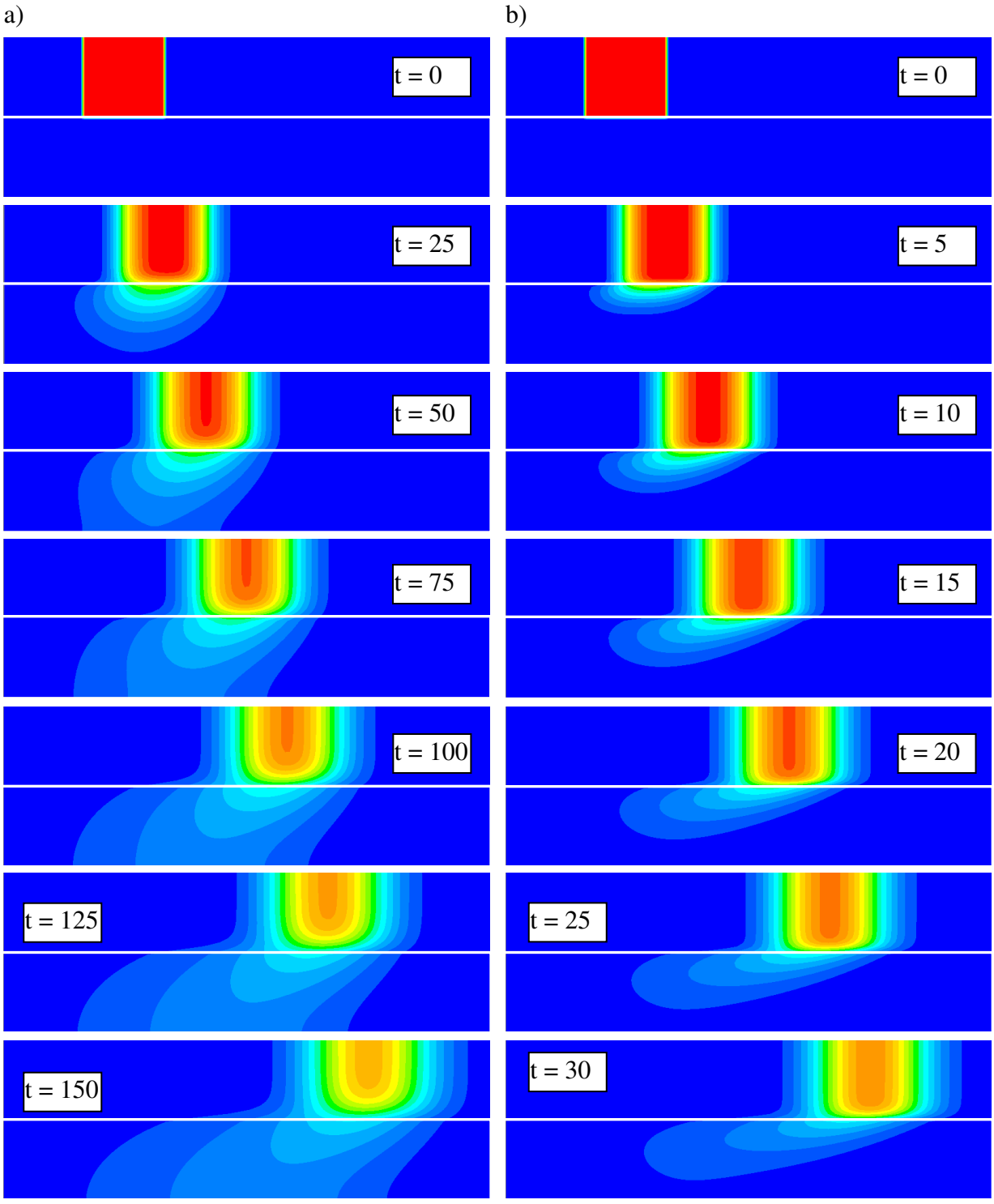
W badanym modelu występują dwa zjawiska nakładające się - wymiana ciepła w całym modelu i ruch górnej warstwy.

Wartości współczynników termofizycznych materiałów symulowanych obiektów (dolnej i górnej warstwy) w pierwszym eksperymencie zostały podane w tabeli 6.1.

Przewodność cieplna materiału górnej warstwy jest czterokrotnie mniejsza od jej wartości dla dolnej warstwy. Dzięki temu przekazywanie ciepła w dolnej warstwie odbywa się znacznie szybciej, co widoczne jest na rys. 6.10.

Tabela 6.1. Parametry termofizyczne materiałów zastosowanych w eksperymentach

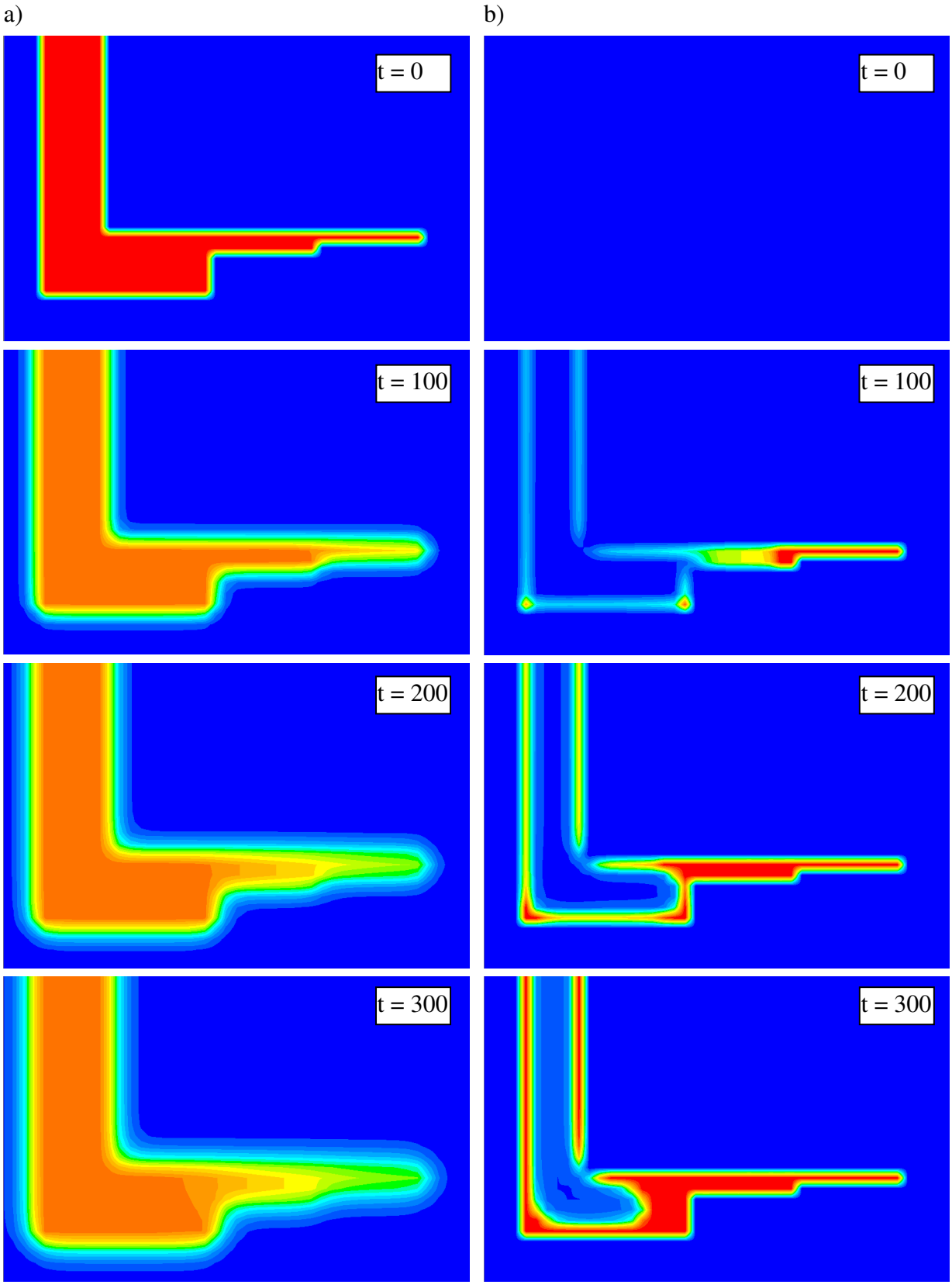
Parametr	Pierwszy eksperyment		Drugi eksperyment	
	Dolna warstwa	Górna warstwa	Odlew	Forma
Przewodność cieplna [W/K]	20	5	210	1
Ciepło właściwe [J/(kg K)]	300	1180	1180	1333
Gęstość [kg/m^3]	1200	2550	2550	1500
Ciepło krystalizacji [J/kg]	---	---	3.73E+5	---
Temperatura eutektyki [ºC]	---	---	610	---

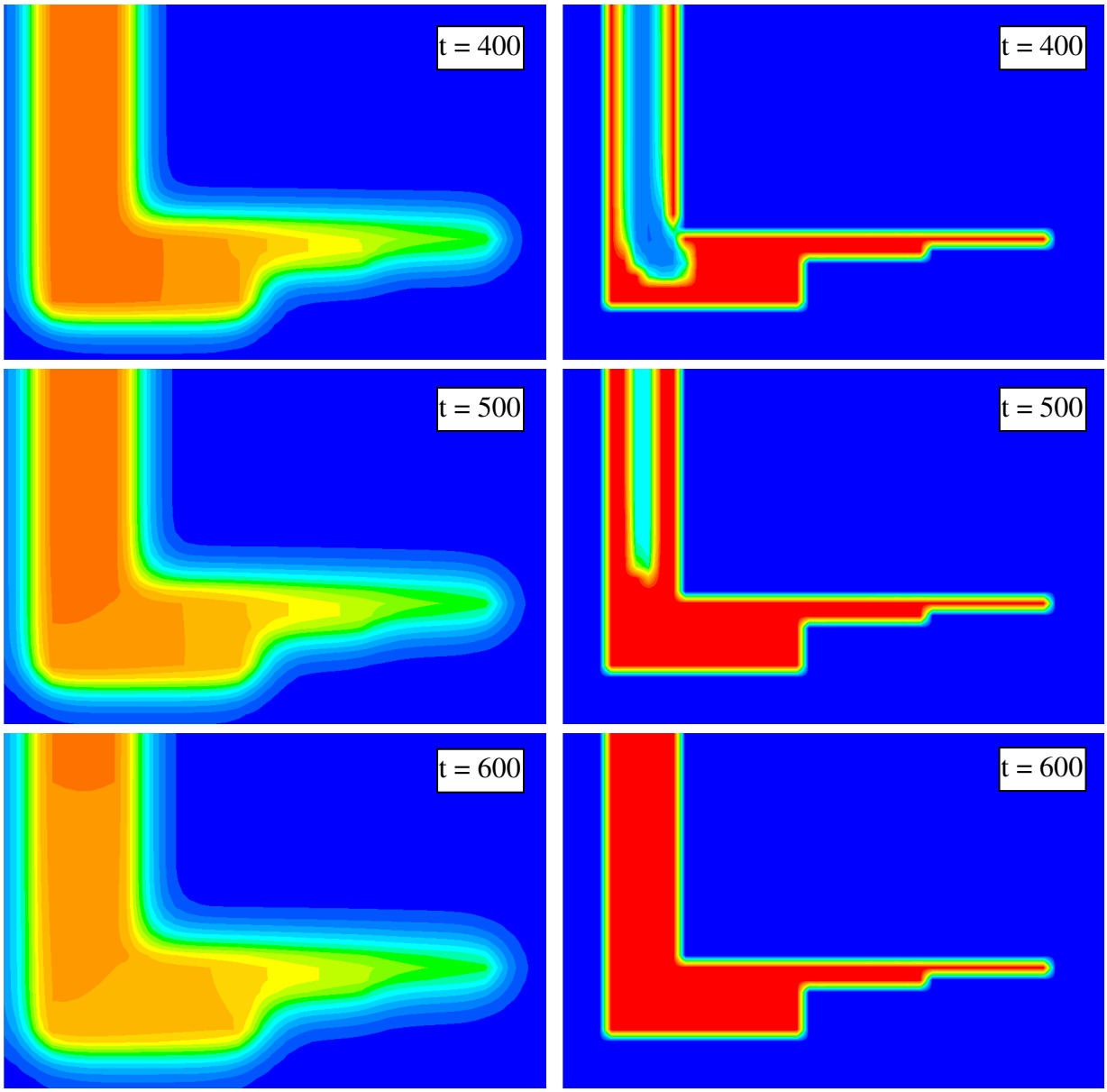


Rys. 6.10. Rozkłady temperatury: a) V = 0.002 m/s, b) V = 0.01 m/s

Drugi eksperyment

Drugi eksperyment dotyczył procesu dla odlewu schodkowego umieszczonego w prostopadłościennej formie.


a)
b)
t = 0
t = 0
t = 100
t = 100
t = 200
t = 200
t = 300
t = 300



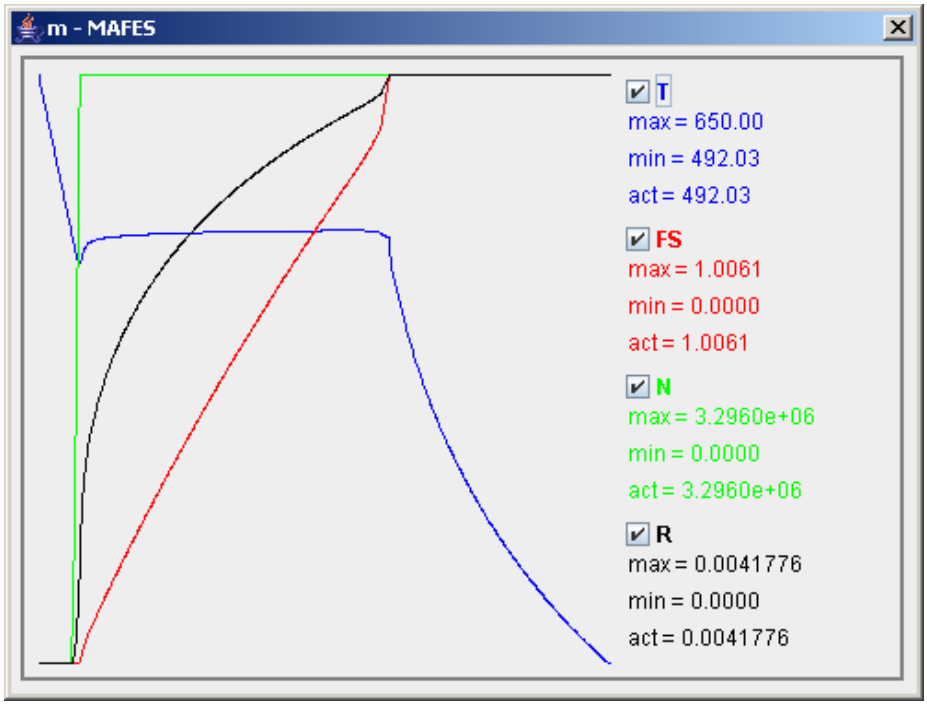
Rys. 6.11. Rozkłady dla t = 0, 100, 200, 300, 400, 500, 600
a) temperatury, b) współczynnika udziału fazy stałej F_S

Wartości współczynników termofizycznych materiałów symulowanych obiektów (odlewu i formy) w drugim eksperymencie zostały podane w tabeli 6.1. Wartości te odpowiadają odlewowi wykonanemu ze stopu aluminium i formie piaskowej.

Rys. 6.11 – kolumna a) przedstawia rozkład przestrzenny temperatury dla tego modelu (odlewu umieszczonego w formie). Widoczne jest zróżnicowanie szybkości stygnięcia odlewu wynikające z jego kształtu. Efektem tego jest nierównomierna struktura mikrokrystaliczna, której kształtowanie w wybranych momentach czasowych widoczne jest na rys. 6.11 – kolumna b), przedstawiającym rozkład przestrzenny współczynnika F_S – udziału fazy stałej.

Rys. 6.12 prezentuje charakterystyki czasowe w małym schodku (temperatura - T, udział fazy stałej – F_S, ilość kryształów w jednostce objętości – N, średni promień kryształów – R). Wykres temperatury uwidacznia efekt przechłodzenia, który ma miejsce w modelu krystalizacji równowagi eutektycznej.

Wartość N osiąga wartość maksymalną w momencie przechłodzenia, od tego momentu już nie wzrasta. Natomiast do momentu osiągnięciu przez F_S wartości 1 wzrasta wartość R. W momencie osiągnięcia przez F_S wartości 1 proces krystalizacji ulega zakończeniu i nie ma już miejsca wydzielanie ciepła związanego z przemiana fazową. Objawia się to szybkim maleniem temperatury T od momentu zakończenia procesu krystalizacji.



Rys. 6.12. Charakterystyki czasowe parametrów krystalizacyjnych i temperatury w małym schodku (t = 125)

6.8. Podsumowanie

Porównanie systemów MAFES-1 (rozdział 5) i MAFES-2 (rozdział 6) nie może dać jednoznacznej odpowiedzi na pytanie, który z tych systemów jest lepszy. Mniejsza ilość typów agentów, w przypadku systemu MAFES-1, związana jest z większą ich złożonością w porównaniu do typów agentów systemu MAFES-2. Jednak z punktu widzenia możliwości

realizacji sterowania w modelu symulacyjnym, nie ma potrzeby wydzielania większej liczby typów agentów niż to uczyniono w systemie MAFES-1.

W systemie MAFES-2 występuje duża ilość typów agentów, w związku z czym celowe stało się podzielenie ich na dwie kategorie: agenty zadań i agenty zjawisk. Dzięki takiemu szczegółowemu podziałowi typów agentów, w systemie MAFES-2 udało się doprowadzić algorytmy ich działania do małej złożoności. Dodatkowo, w systemie MAFES-2 wydzielono typ uogólniony agenta, na którym oparto wszystkie typy szczegółowe. W systemie MAFES-2 wprowadzono również agenta grupującego pozwalającego tworzyć hierarchiczne struktury agentów (zjawisk i zadań) odpowiadające dekompozycji przestrzennej zjawisk i dzieleniu zjawiska złożonego na zjawiska elementarne. Agent grupujący umożliwiał również koordynację działań agentów zjawisk i agentów zadań.

Ważną cechą systemu MAFES-2 jest także wyróżnienie w modelu symulacyjnym środowiska w postaci siatki węzłów. Dzięki temu znacznie zredukowano ilość wzajemnych interakcji między agentami, zastępując je operacjami obserwacji i oddziaływania środowiska. Środowisko w tym przypadku przechowuje stan modelu.

Zaprezentowana koncepcja agentowego systemu symulacyjnego może znaleźć zastosowania dla szerokiej klasy problemów praktycznych, w których uwzględnić trzeba współdziałanie nakładających się na siebie zjawisk zachodzących we wspólnym środowisku.

W niniejszym rozdziale przedstawiono wariant tego systemu dostosowany do specyfiki procesów termo-fizycznych zachodzących w stygnącym odlewie.

Przeprowadzone eksperymenty potwierdziły użyteczność proponowanego podejścia. W perspektywie dalszych badań przewiduje się zwiększenie uniwersalności systemu, poprzez wprowadzenie nowych typów agentów oraz dostosowanie do innych obszarów zastosowań.

7. Model symulacyjny wykorzystujący koncepcję programowania aspektowego

7.1. Motywacja

Trudno jest znaleźć jedną, uniwersalną metodę, najbardziej przydatną w zastosowaniu do złożonych problemów, a dana konkretna metoda służy najlepiej do wąskiej kategorii problemów, dla których została opracowana. Dlatego w rozważanym w pracy problemie o dużej złożoności, wydaje się, że zaproponowanie i przebadanie kilku koncepcji projektowania może prowadzić do interesujących rezultatów.

Niniejszy rozdział jest próbą wykazania przydatności połączenia nowej metodologii budowania systemów komputerowych – *programowania zorientowanego aspektowo* z podejściem agentowym w modelowaniu złożonych zjawisk fizycznych.

W dotychczasowej części pracy zostały przedstawione trzy podejścia do symulacji takich zjawisk, stosujące według kolejności coraz większy stopień wykorzystania koncepcji agentowych.

Pierwsza z prezentowanych w pracy koncepcji (rozdział 4) dotyczyła realizacji systemu ABAG współpracującego z systemem ABAQUS.

Drugie przedstawione podejście (system MAFES-1, rozdział 5) stanowiło samodzielny agentowy system symulujący proces stygnięcia i krystalizacji z dodatkowym zjawiskiem przemieszczającego się medium chłodzącego. Zadaniami spełnianymi przez agentów w tym podejściu były wydzielone role w modelu symulacji i sterowania (agent obliczeniowy, agent regulator, agent sterujący).

W rozwiązaniu tym występowała duża złożoność agenta obliczeniowego, wynikająca z realizacji wszystkich zjawisk występujących w reprezentowanym elemencie przestrzeni fizycznej. Poza tym, utrudniona jest rozbudowa takiego systemu, gdyż wprowadzanie nowych typów zjawisk wymagałoby dalszej rozbudowy agenta obliczeniowego, prowadzącej do jeszcze większej jego złożoności.

W systemie MAFES-1 miała również miejsce duża ilość interakcji wzajemnych między agentami obliczeniowymi, wynikająca z przechowywania przez agenty stanu modelu. Dodatkowo w podejściu tym, większość zadań symulacyjnych typu wizualizacja czy rejestracja wyników realizowana była w sposób nieagentowy (strukturalny, obiektowy).

Kolejne, trzecie przedstawione w pracy podejście (system MAFES-2, rozdział 6) wprowadziło podział ról spełnianych przez agenty na dwie kategorie: agenty zjawisk fizycznych i agenty zadań.

Pierwsza kategoria pozwoliła na wydzielenie osobnych typów agentów dla każdego zjawiska, każdy typ reprezentował poszczególne zjawisko (dokładniej samą dynamikę zjawiska – zmiany wprowadzane w środowisku przez dane zjawisko). Druga kategoria (agenty zadań) pozwoliła na agentową realizację zadań często wymaganych w symulacji, typu wizualizacja czy sterowanie.

W systemie MAFES-2 wyróżniono w modelu symulacyjnym środowisko, przechowujące stan modelu fizycznego. Zastosowano uniwersalny wzorzec agenta, którego działanie sprowadzało się do obserwacji i modyfikacji środowiska (oddziaływania na środowisko).

Dzięki powyższym rozwiązaniom udało się osiągnąć małą złożoność agentów i ich funkcji oraz wspólny wzorzec działania w przypadku wszystkich, występujących typów agentów.

Pewną niedoskonałością systemu MAFES-2 jest występowanie w nim dużej ilości typów agentów. Poza tym dodanie nowego typu analizowanych zjawisk wymaga nie tylko dodania nowego typu agenta i modyfikacji modelu agentowego, lecz również związane jest najczęściej z koniecznością wprowadzenia zmian w węzłach środowiska – dodaniem odpowiednich atrybutów.

Proponowane w niniejszym rozdziale rozwiązanie w postaci systemu MAFES-3 polega na próbie uzyskania połączenia zalet systemów MAFES-1 i MAFES-2. Podejście to dąży do zachowania minimalnej liczby typów agentów (tak jak to miało miejsce w przypadku systemu MAFES-1) oraz wykorzystuje dekompozycję zjawisk (podobnie jak w podejściu MAFES-2), co staje się możliwe dzięki zastosowaniu koncepcji programowania aspektowego – zjawiska są definiowane w osobnych jednostkach kompilacji (aspektach).

Na początku rozdziału przedstawiona jest koncepcja programowania aspektowego i głównej jego implementacji w postaci języka AspectJ. Następnie przedstawiony jest system symulacyjny zjawisk złożonych, przy budowie którego zastosowano tylko podejście aspektowe, bez wykorzystania koncepcji agentowych.

Główną, końcową częścią tego rozdziału jest prezentacja systemu MAFES-3. Zrealizowany on został przy połączeniu podejścia agentowego i programowania aspektowego.

7.2. Elementy programowania zorientowanego aspektowo

Do powstania podejścia aspektowego przyczyniło się szereg prac szukających nowych metod programowania, które wynikały z ograniczeń pojawiających się przy stosowaniu analizy, projektowania i programowania zorientowanego obiektowo do złożonych problemów.

Programowanie zorientowane aspektowo (ang. *Aspect Oriented Programming* – AOP) wraz z koncepcją komputerowej refleksji (ang. *computational reflection*) [55] należy do metod separacji zagadnień (ang. *separation of concerns*).

Głównym twórcą AOP jest G. Kiczales, którego prace w końcowych latach dziewięćdziesiątych [48] stały się podstawą dla powstania tej metody i sformułowania jej zasad. Metoda ta spotyka się z dużym zainteresowaniem. Zostało rozpoczętych wiele badań nad jej zastosowaniem w różnych dziedzinach, powstały konferencje jej poświęcone.

O wielu istniejących językach programowania, włączając języki obiektowe, proceduralne i funkcjonalne (ang. *functional*), można powiedzieć, że posiadają wspólny korzeń w tym znaczeniu, że ich kluczowe mechanizmy abstrakcji i kompozycji mają formę pewnej uogólnionej procedury. Metody projektowania, które powstały by współpracować z tymi językami, zazwyczaj dokonują podziału systemu na jednostki działań czy funkcje. Określane jest to jako dekompozycja funkcjonalna [62].

W kontekście tym AOP dotyczy głownie jednostek dekompozycji systemu, które nie mają charakteru funkcjonalnego.

Celem programowania aspektowego jest dostarczenie twórcom oprogramowania mechanizmów jasnego, klarownego obejmowania wszystkich ważnych elementów działania systemów, zawierających nie tylko ich funkcjonalność, ale również takie kwestie jak strategia obsługi błędów, strategia komunikacji czy strategia koordynacji [48].

Trudności uwzględnienia wspomnianych *aspektów* wynikają z tego iż w pewnym sensie przecinają one (ang. *cross - cut*) podstawową funkcjonalność systemu. AOP daje możliwość czytelnego wyrażania struktury programów i systemów zawierających aspekty oraz właściwego wyizolowania, kompozycji i ponownego użycia kodu aspektowego.

Według twórców AOP, języki programowania bazujące na pojedynczym modelu abstrakcji (procedury, ograniczenia, obiekty, itp.) są w większości przypadków nieadekwatne dla wielu złożonych systemów. Wynika to z tego, że różne aspekty zachowania systemu, które muszą zostać oprogramowane, dążą do zachowania swojej własnej *naturalnej formy*, tak więc podczas gdy jeden model abstrakcji może się dobrze sprawdzać w obejmowaniu jednego aspektu, to dla innych aspektów może on utrudniać pracę

W AOP różne aspekty zachowania systemu są oprogramowane w ich najbardziej naturalnej postaci, następnie te oddzielne moduły są tkane (ang. *weave*) razem tworząc kod wynikowy. Oczekuje się, że podejście to wniesie innowację zarówno w językach programowania ogólnego przeznaczenia jak i językach zorientowanych problemowo.

Przykładem problemu z użyciem pojedynczego modelu abstrakcji jest zasada zachowania relacji między obiektami. Podczas gdy języki obiektowe dobrze sobie radzą z opisaniem zachowania obiektów, napotykają na problemy przy opisie zasad strukturalnych i zachowań, takich jak np. „jeśli obiekt kwadrat otrzymuje komunikat przesuń, przesyła

komunikat odśwież obiektowi wyświetlacz". Przy użyciu obecnych języków programowania trudno jest modelować drugorzędne, aczkolwiek istotne aspekty złożonych systemów.

Niekiedy problem ten można rozwiązać izolując jedno zagadnienie w jednej części kodu i drugie zagadnienie w drugiej części kodu. Jest to często dokonywane w pewnej formie abstrakcji proceduralnej. Na przykład szczegóły alokacji pamięci mogą być ukryte za interfejsem *malloc/free*, kod klienta może zająć się operacjami na zaalokowanej pamięci.

Jednak występują sytuacje, w których dwa aspekty działania systemu niezmiennie pozostają „przeplatane" w kodzie. W takich przypadkach można powiedzieć, że dwa aspekty „przecinają" się wzajemnie w odniesieniu do kodu programu. Problem ten wynika często z zależności pewnych operacji od konkretnego kontekstu, w którym są przeprowadzane.

Zagadnienia przecinające się, pojęcie aspektu

Przecinanie się zagadnień jest bezpośrednio odpowiedzialne za przeplatanie się kodu z nimi związanego w implementacji systemów. Kiedy dwie własności będące programowane muszą się łączyć na różne sposoby i w dodatku być koordynowane można powiedzieć, że przecinają się wzajemnie. Ponieważ języki funkcjonalne dostarczają tylko jednego mechanizmu kompozycji, programista musi dokonywać dodatkową kompozycję samodzielnie, co prowadzi do znacznej złożoności i przeplatania się kodu.

Dla celów określenia AOP zostały zdefiniowane dwa następujące pojęcia [48].

- ***Komponent*** – to coś co może być czytelnie wydzielone jako uogólnienie procedury (obiekt, funkcja, procedura, itp.), dobrze zlokalizowane, łatwo dostępne i mogące być komponowane (łączone) w razie potrzeby. Komponenty w tym rozumieniu mają tendencję być jednostkami dekompozycji funkcjonalnej systemu, tak jak konto bankowe, czy element interfejsu graficznego.
- ***Aspekt*** – to coś co nie można czytelnie wydzielić jako uogólnienie procedury. Aspekty nie da się traktować jako jednostki funkcjonalnej dekompozycji systemu, tylko jako właściwości, które oddziałują na działanie czy semantykę (wzorce dostępu do pamięci, synchronizacja współbieżnych obiektów).

Używając tak zdefiniowanych terminów można powiedzieć, że rolą AOP jest wsparcie programisty w czytelnym oddzielaniu aspektów i komponentów wzajemnie od siebie, poprzez dostarczenie mechanizmów , które umożliwiają ich abstrakcję i kompozycję w celu budowy systemów. Natomiast języki bazujące na pojęciu procedury wspierają programistę tylko w identyfikowaniu i łączeniu komponentów.

Niektóre aspekty są zależne od dziedziny problemu, natomiast inne są tak powszechne, że można je rozważać bez związku z jakąkolwiek dziedziną. Przykładem takiego

powszechnego aspektu jest obsługa błędów. Znane jest zjawisko, że dodanie dobrego wsparcia dla obsługi błędów do prostego prototypu systemu kończy się potrzebą uwzględnienia wielu mniejszych lub większych dodatków i zmian w całym systemie. Wynika to z faktu iż różne dynamiczne konteksty, które mogą prowadzić do sytuacji awaryjnych lub związane są ze sposobem w jaki obsługuje się błędy, przecinają funkcjonalność systemu.

Wiele kwestii związanych z wydajnością oprogramowania można traktować jako pewien rodzaj aspektów, gdyż optymalizacja wydajności często wykorzystuje informację o kontekście wykonania, która obejmuje wiele komponentów.

Zastosowania podejścia aspektowego

Programowanie aspektowe może być stosowane w różnych fazach analizy, projektowania i programowania systemów komputerowych.

Poniżej przedstawiono przykłady użycia aspektów, z których korzysta się wyłącznie w fazie tworzenia oprogramowania, które nie są dołączane go gotowych systemów:

- śledzenie działania programów (ang. *tracing*),
- profilowanie i logowanie,
- realizacja warunków wstępnych i końcowych w metodzie projektowania przez kontrakt (ang. *design by contract*) [57]
- zarządzanie konfiguracją w trakcie kompilacji.

Oczywiście aspekty można stosować również do modularyzacji tych części oprogramowania, które znajdą się w gotowym produkcie, Przykładami tego typu zastosowań mogą być:

- monitorowanie zmian (gromadzenie informacji o wywoływaniach wybranych funkcji),
- przekazywanie kontekstu wybranym funkcjom,
- realizacja spójnego zachowania w grupie funkcji.

Wśród dziedzin zastosowań AOP ważne miejsce odgrywa tworzenie systemów agentowych. Prace łączące podejście aspektowe i agentowe znajdują wyraz w publikacjach następujących autorów: E. Kendall [46, 47], C. Lucena i jego współpracownicy [21, 28-32, 52, 53, 66, 67], A. Helleboogh, T. Holvoet, D. Weyns [19, 39, 40], R. Robbes, N. Bouraqadi, S. Stinckwich [64].

W szczególności określone zostały obszary, w których AOP może być użyteczne dla systemów MAS [64]:

1. Wprowadzenie koncepcji AOP do modeli systemów MAS i ich projektowania. Koncepcje AOP mogą być użyte na poziomie koncepcyjnym systemów MAS.

2. Użycie AOP do implementacji infrastruktury systemu MAS. System MAS jest złożoną aplikacją z wieloma powiązanymi, a nawet czasami będących w konflikcie funkcjonalności, dlatego łatwo zrozumieć, że AOP znajduje wiele zastosowań na tym polu. W pracach rozwiązujących ten problem [30] koncepcja agenta nie jest pierwszą klasą, tylko jest on zbudowany na obiekcie dziedzinowym utkanym z agentowym aspektem.

Język AspectJ

AspectJ [3, 49] jest prostym, jednolitym i praktycznym rozszerzeniem języka programowania Java. Dodając kilka nowych konstrukcji do tego języka, zapewnia wsparcie dla modularnej implementacji przecinających się zagadnień.

Podstawowymi pojęciami języka AspectJ są:

- *punkt złączenia* - dobrze określony punkt wykonywania programu,
- *punkt cięcia* - kolekcja punktów złączeń,
- *rada* - konstrukcja podobna do funckji, która jest dołączana do punktów cięć,
- *wprowadzenie* – pole lub funkcja dodawana do istniejącej klasy,
- *aspekt* – jednostka modularnej implementacji przecinającego zagadnienia, składająca się z powyższych elementów oraz z zwykłych deklaracji pól i funkcji języka Java.

W AspectJ kod aspektowy jest dołączany do standardowego kodu bajtowego Javy. Istnieją rozszerzenia popularnych środowisk programistycznych Javy (np. AJDT dla Eclipse), pozwalające na używanie języka AspectJ, tworzenie aspektów i ich elementów, przeglądanie struktury przecinających się aspektów w sposób podobny do przeglądania hierarchii dziedziczenia klas.

Szersze omówienie języka AspectJ zostało zawarte w dodatku A.

7.3. Koncepcja aspektowego modelu zjawisk fizycznych

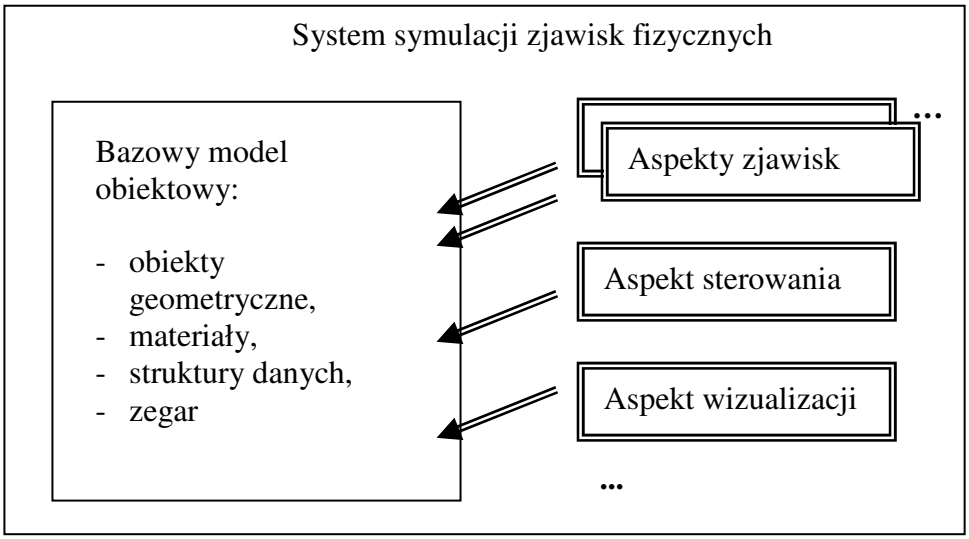
Podrozdział niniejszy opisuje założenia i budowę systemu symulacyjnego APSS zrealizowanego przy pomocy programowania aspektowego (bez koncepcji agentowych). W systemie tym modelowane zjawiska i zadania stawiane w symulacji implementowane są jako oddzielne aspekty, które nakładane są na bazowy model obiektowy.

Rozważania prowadzone są na przykładzie zjawisk cieplnych (wymiana ciepła) z współistniejącym zjawiskiem ruchu. Dodatkowe zadania realizowane w symulacji to wizualizacja i sterowanie.

Koncepcję podzielenia systemu symulacyjnego na bazowy model obiektowy i nakładane na niego aspekty zjawisk i zadań przedstawiono na rys. 7.1.

Bazowy model obiektowy

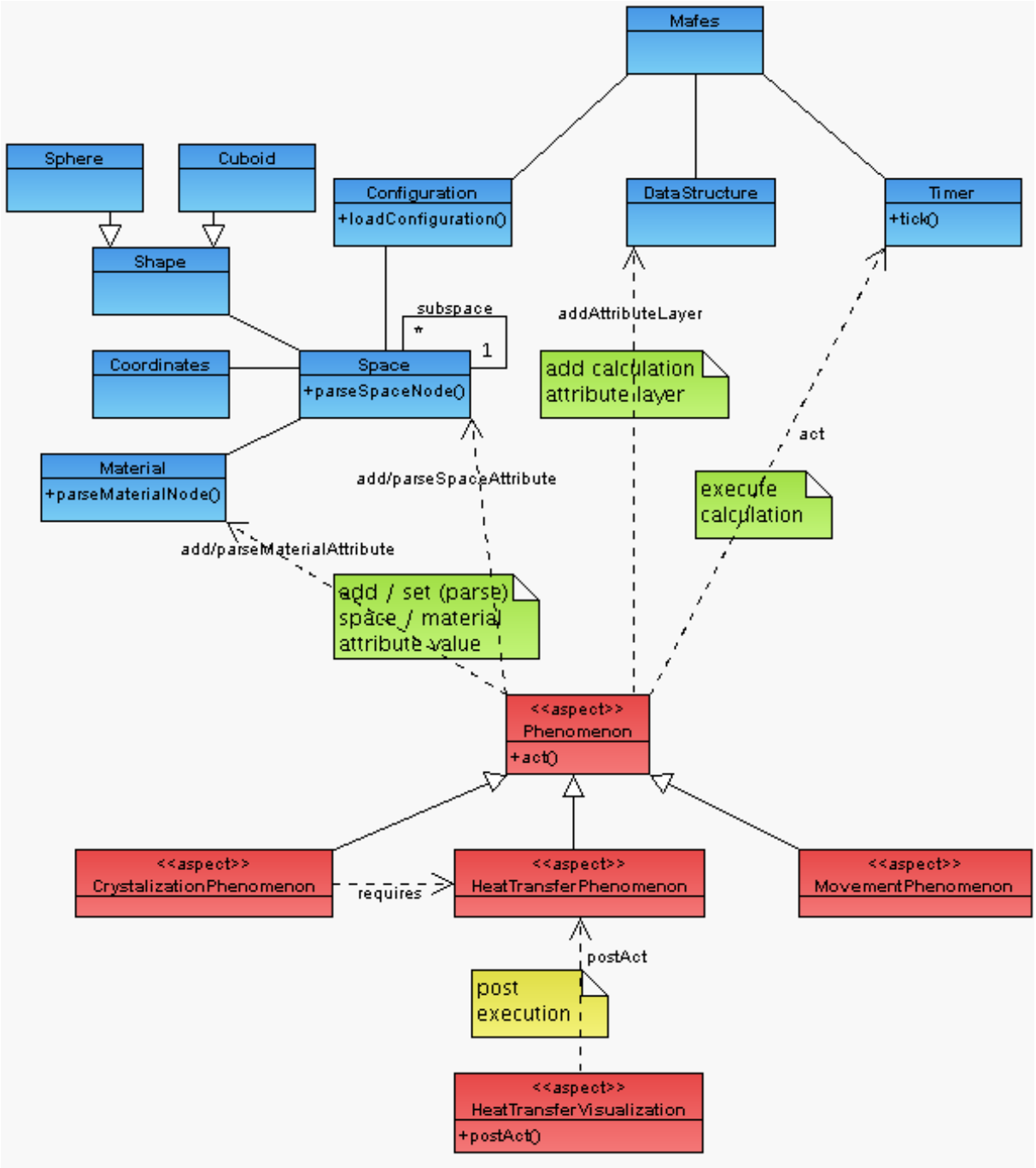
Głównym zadaniem bazowego modelu obiektowego jest wyodrębnienie części wspólnej rozważanych typów zjawisk i zadań symulacyjnych. Powinien on również udostępniać aspektom odpowiednią funkcjonalność do operowania na nim (np. dodanie atrybutu obliczeniowego).



Rys. 7.1. Ogólna architektura systemu symulacyjnego

Najważniejsze klasy modelu bazowego zostały przedstawione w górnej części rys. 7.2. Wczytaniem modelu z pliku (jego konfiguracji) zajmuje się klasa **Configuration.** Badane obiekty geometryczne stanowi drzewiasta struktura oparta na klasie **Space**. Każdy obiekt geometryczny (klasa **Space**) posiada określony kształt (klasa **Shape**). Obiekty geometryczne mogą mieć przypisany określony materiał (klasa **Material**) reprezentujący grupę parametrów termofizycznych.

Po wczytaniu modelu z pliku następuje generacja struktur obliczeniowych (klasa **DataStructure**) dla wszystkich rozważanych obiektów i materiałów. Te struktury obliczeniowe są wykorzystywane przez aspekty w trakcie kolejnych kroków symulacji (klasa **Timer**, funkcja **tick**).



Rys. 7.2. Diagram klas i aspektów systemu symulacyjnego (oddziaływanie aspektów na bazowy model obiektowy zaznaczono liniami przerywanymi)

Aspekty

Na rys. 7.2 (oznaczone przez <<aspect>>) pokazane zostały aspekty występujące w rozważanym systemie symulacyjnym. Bazowym aspektem dla aspektów zjawisk jest aspekt **Phenomenon**. Z aspektu tego dziedziczą aspekty trzech zjawisk:

- wymiany ciepła (realizacja w aspekcie **HeatTransferPhenomenon**),
- krystalizacji (realizacja w aspekcie **CrystalizationPhenomenon**),

- ruchu (realizacja w aspekcie **MovementPhenomenon**).

W wymienionych aspektach zdefiniowane są algorytmy realizujące odpowiednie zjawiska. Algorytmy te są dołączane do cyklu obliczeniowego systemu poprzez odpowiednio zdefiniowane punkty cięć i rady.

Aspekty zjawisk rozbudowują również część konfiguracyjną modelu bazowego, tak by odpowiednie atrybuty mogły być wczytane z pliku modelu. Rolą spełnianą przez aspekty zjawisk jest także rozszerzenie obliczeniowych struktur danych w odpowiednie warstwy atrybutów.

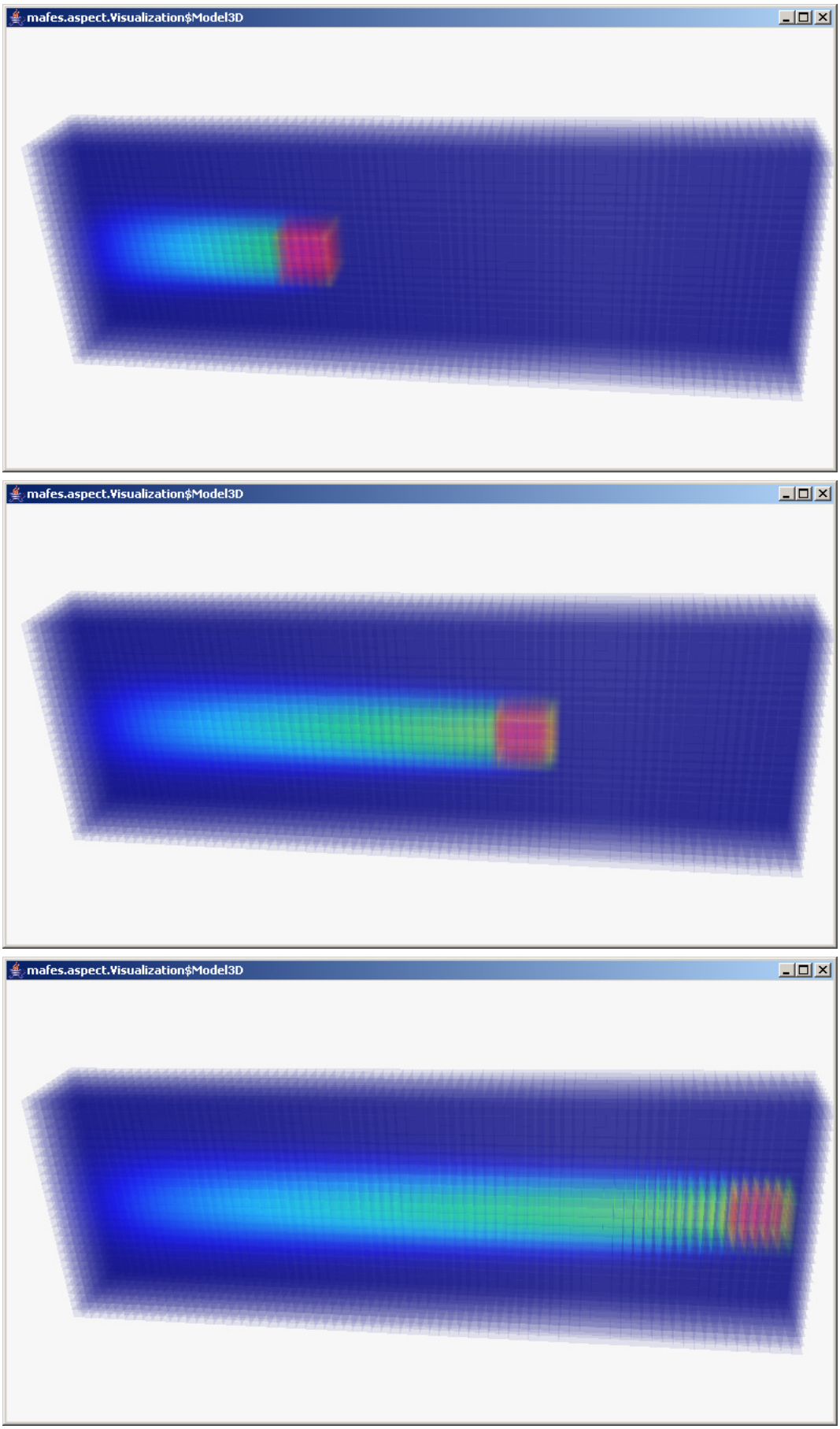
Między niektórymi aspektami zjawisk zachodzi relacja zależności (ang. *requires*) pokazana na rys. 7.2 dla aspektu **CrystalizationPhenomenon**. Mianowicie aspekt tego zjawiska może być nałożony na model bazowy pod warunkiem wcześniejszego nałożenia aspektu wymiany ciepła (**HeatTransferPhenomenon**). Wynika to z faktu, iż nie można rozważać zjawiska krystalizacji bez określenia zjawiska wymiany ciepła. Aspekt wymiany ciepła dodaje do modelu bazowego podstawowe atrybuty procesów cieplnych, jak np. temperatura, czy ciepło właściwe.

W prezentowanym systemie zrealizowano w oparciu o aspekty również takie zadania symulacyjne jak wizualizacja czy sterowanie.

Eksperyment symulacyjny

Celem przeprowadzonego eksperymentu symulacyjnego było pokazanie współdziałania dwóch zjawisk: wymiany ciepła i ruchu. Na rys. 7.3 (wizualizacja trójwymiarowa z zastosowanym efektem przezroczystości) zostały przedstawione wyniki symulacji dla prostego modelu składającego się z prostopadłościanu, wzdłuż którego (wewnątrz) przesuwa się ruchem jednostajnym sześcian o mniejszych rozmiarach lecz o wyższej temperaturze. Pozostawia on za sobą „ślad” ciepła.

W rozważanej symulacji pomijany jest efekt zawirowań, które mogą powstać za ruchomym obiektem. Takie uproszczenie jest możliwe przy małej prędkości ruchu.



Rys. 7.3. Otrzymane na drodze symulacji wyniki w trzech kolejnych chwilach

Podsumowanie

Aspektowa realizacja zjawisk i zadań symulacyjnych pozwala na ich łatwe odłączanie i przyłączanie (np. aspektu sterowania). Dzięki temu konfiguracja systemu symulacyjnego jest zadaniem stosunkowo łatwym. Występuje tutaj wzajemna separacja modelu bazowego, modeli poszczególnych zjawisk i realizowanych zadań. Dzięki temu struktura systemu jest bardziej czytelna.

Do zalet prezentowanego rozwiązania należy zaliczyć:
- występuje tutaj relacja odpowiedniości pomiędzy zjawiskiem fizycznym (etap analizy problemu) a aspektem zjawiska (etap implementacji oprogramowania symulacyjnego),
- kod każdego zjawiska fizycznego oraz kod związany z zadaniami symulacyjnymi (wizualizacja, sterowanie) jest zmodularyzowany i odseparowany (zapisany w osobnej jednostce kompilacji),
- występuje analogia pomiędzy współdziałaniem zjawisk (etap analizy) a przeplataniem się aspektów (etap implementacji),
- tworzenie różnych wersji systemu symulacyjnego dla specyficznych wymagań jest zadaniem łatwym (wystarczy wybrać odpowiednie aspekty),
- implementacja konfiguracji systemu dzięki zastosowania AOP jest prosta w realizacji,
- struktura kodu jest czytelna, łatwa do zrozumienia i dalszej rozbudowy.

Algorytmy aspektów w przypadku rozważanego systemu są jednak dużo bardziej złożone niż algorytmy agentów opisanych w rozdziale 6 (system MAFES-2). Zatem warto rozważyć połączenie podejścia agentowego z programowaniem aspektowym. Propozycję takiego rozwiązania przedstawiono w następnym podrozdziale.

7.4. Model agentowy z zastosowaniem podejścia aspektowego

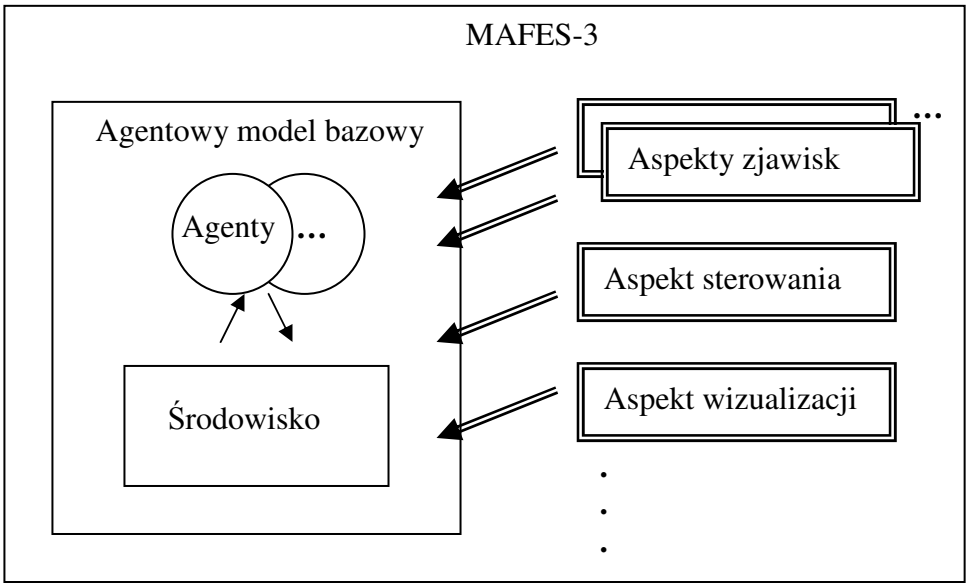
Przedstawione w poprzednim podrozdziale rozwiązanie polegające na zastosowaniu jedynie koncepcji aspektów do modelowania zjawisk i zadań oraz obiektowego modelu bazowego, reprezentującego struktury danych, prowadzi do tego, że postać aspektów staje się złożona. Celem metody prezentowanej w niniejszym podrozdziale jest połączenie zalet programowania aspektowego i podejścia agentowego.

Zaletą systemu MAFES-1 prezentowanego w rozdziale 5 jest występowanie w nim małej ilości typów agentów (obliczeniowy, regulator, sterujący). Powoduje to jednak dużą złożoność agenta obliczeniowego, który realizował wszystkie występujące w modelu fizycznym zjawiska, poza zjawiskiem ruchu (agent regulator).

Rozwiązanie w postaci systemu MAFES-2 (rozdział 6) rozdzielające własności potrzebne do odwzorowania stanu zjawisk w postaci środowiska od dynamiki tych zjawisk reprezentowanych przez grupy agentów oraz dekompozycja przestrzenna zjawiska w postaci lokalnych agentów ma wiele zalet, z których najważniejszą jest prostota i czytelność algorytmów działania tychże agentów.

Prezentowany w niniejszym podrozdziale system MAFES-3 stosuje podział na typy agentów, taki jak występujący w przypadku systemu MAFES-1. Złożoność agenta obliczeniowego została tutaj zredukowana przez zastosowanie aspektów zjawisk (przypominających dekompozycję zjawisk z rozdziału 5 – system MAFES-2). Za pomocą aspektów realizowane są również pozostałe zadania symulacyjne (sterowanie, wizualizacja, rejestracja wyników). Dodatkowo w systemie MAFES-3 wykorzystano reprezentację stanu modelu w postaci wyróżnionego środowiska, tak jak to miało miejsce w systemie MAFES-2.

Koncepcję podzielenia systemu symulacyjnego na bazowy model agentowy i nakładane na niego aspekty zjawisk i zadań przedstawiono na rys. 7.4.



Rys. 7.4. Ogólna architektura systemu MAFES-3 (w stosunku do rys. 7.1 – zamiana bazowego modelu obiektowego na bazowy model agentowy)

Zjawiska fizyczne można rozważać jako pewne reguły, prawa zmian stanu środowiska. Na ogół dotyczą pewnych, wybranych jego własności (masa, ładunek elektryczny, temperatura, itp.). Istnieje bardzo dużo dziedzin fizyki starających się opisać zjawiska występujące w realnym świecie. Każda z tych dziedzin dotyczy pewnego, wąskiego aspektu rzeczywistości i definiuje swoje atrybuty świata, wielkości fizyczne oraz prawa rządzące dynamiką ich zmian, oddziaływań między nimi, ich zachowania.

Pojęcia takie jak przestrzeń i czas są podstawą modelowania wszelkich zjawisk, dlatego zdecydowano się na ich reprezentację w postaci modelu bazowego. Dodatkowo w modelu bazowym zawarta została potencjalna możliwość zmiany stanu w postaci agenta obliczeniowego.

Bazowy model agentowy

W przypadku systemu MAFES-3 bazowym modelem nie jest model obiektowy (rozdział 7.3) lecz model agentowy składający się z środowiska w postaci siatki węzłów oraz agentów obliczeniowych przypisanych do węzłów wspomnianego środowiska. Środowisko w postaci siatki węzłów służy reprezentacji stanu modelowanego obszaru przestrzeni fizycznej.

Agent obliczeniowy stanowi aktywny element modelu. Zadaniem grupy agentów obliczeniowych przypisanych do węzłów środowiska jest wyznaczanie kolejnych stanów modelu (środowiska). Sposób w jaki realizowane są zmiany stanu środowiska opisywany jest w aspektach zjawisk, które nakładane są na agenta obliczeniowego.

Aspekty zjawisk

Podstawowym zastosowaniem aspektów w prezentowanym podejściu jest wydzielenie realizacji różnych typów zjawisk, które mogą być przydzielone do wykonania agentom obliczeniowym.

Aspekty zjawisk spełniają swoją rolę wykorzystując następujące mechanizmy programowania aspektowego (języka AspectJ opisanego w rozdziale 7.3):

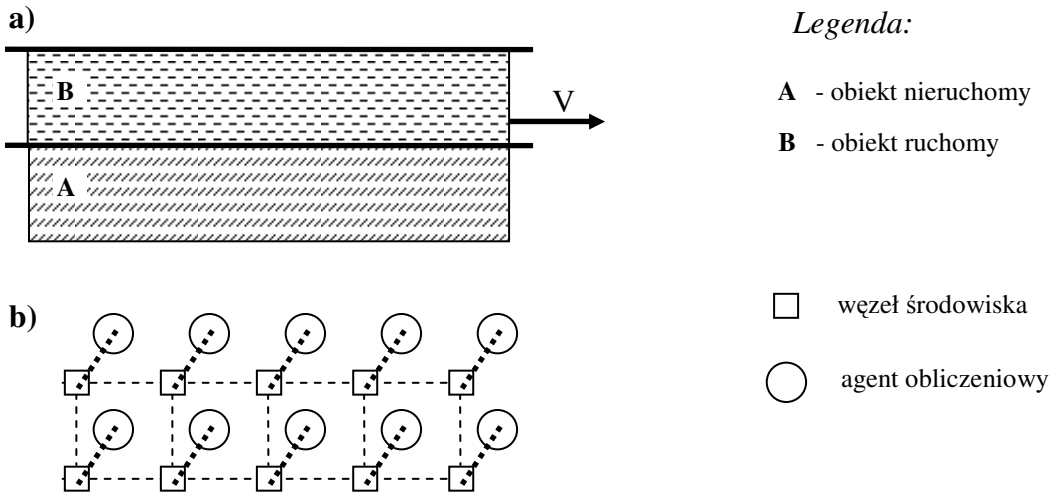
- wprowadzenia, które dodają do węzłów środowiska odpowiednie atrybuty związane z danym zjawiskiem,
- wprowadzenia, które dodają do agenta obliczeniowego zestaw funkcji realizujących dane zjawisko,
- rady, wiążące dodane funkcje z podstawowymi funkcjami agenta obliczeniowego (*look* i *act*).

W systemie MAFES-3 wyróżniono następujące typy aspektów zjawisk (analogiczne do typów agentów zjawisk z systemu MAFES-2):

- aspekt **AspectT** – realizujący zjawisko wymiany ciepła,
- aspekt **AspectCr** – realizujący zjawisko krystalizacji,
- aspekt **AspectR** – realizujący zjawisko ruchu.

W celu obrazowego przedstawienia nakładania aspektów zjawisk na model bazowy na rys. 7.5 zaprezentowano przykładowy model fizyczny i odpowiadający mu agentowy model bazowy składający się z dyskretnego środowiska wraz z umiejscowionymi w jego węzłach

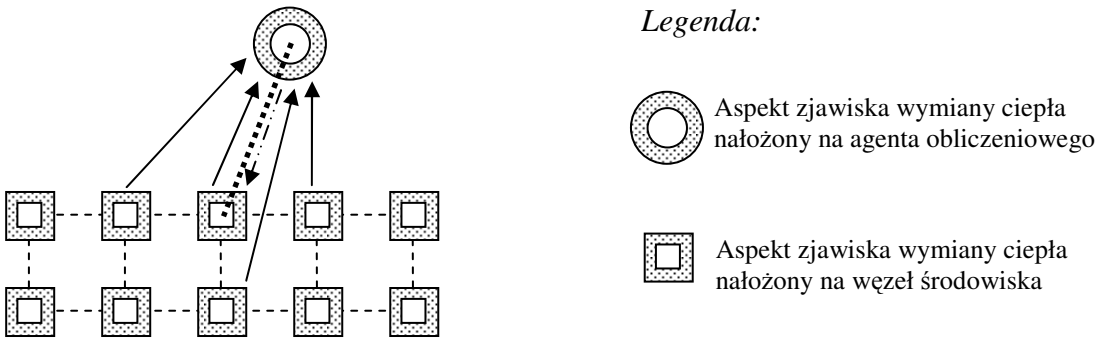
agentami. Zjawisko wymiany ciepła obejmuje obiekty A i B. Natomiast zjawisko ruchu dotyczy obiektu tylko B.



Rys. 7.5. Przykładowy model fizyczny – a)
i jego reprezentacja w postaci bazowego modelu agentowego – b)

Aspekt zjawiska wymiany ciepła dodaje do węzłów środowiska atrybuty takie jak: temperatura, ciepło właściwe, przewodność, gęstość.

Na agenta obliczeniowego aspekt tego zjawiska nakłada funkcje realizujące algorytmy wyznaczające zmiany wartości temperatury w danym węźle środowiska, do którego jest on przypisany. Funkcje te są dołączane do podstawowych funkcji agenta obliczeniowego obserwacji i oddziaływania. Nałożenie aspektu zjawiska wymiany ciepła na model bazowy zostało symbolicznie przedstawione na rys. 7.6.



Rys. 7.6. Aspekt wymiany ciepła nałożony na jednego z agentów i węzły środowiska

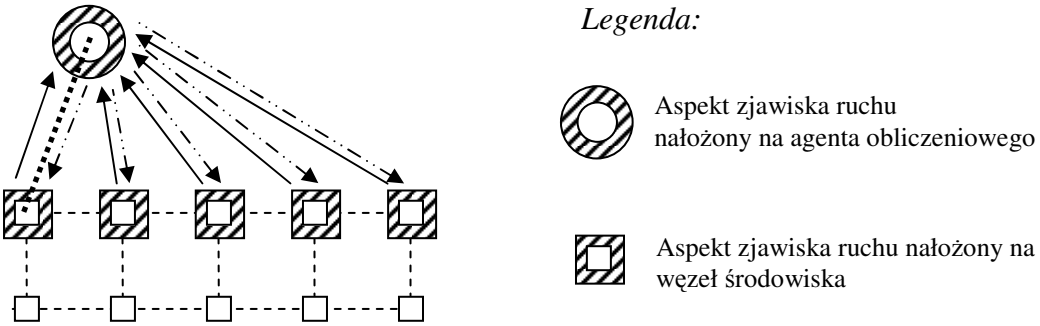
Aspekt zjawiska krystalizacji dodaje do węzłów środowiska atrybuty reprezentujące wszystkie współczynniki potrzebne do realizacji algorytmu tego zjawiska, F_S, N, R, CGN.

Zmiany dodane do agenta obliczeniowego związane z tym zjawiskiem to nie tylko rozszerzenie operacji obserwacji i oddziaływania. Agent ten musi przechowywać informację o ostatniej wartości temperatury, tak by mógł rozpoznać moment przechłodzenia. Obserwacja

agenta pod względem aspektu krystalizacji dotyczy tylko własności węzła, do którego agent jest przypisany. Na podstawie obserwacji wartości temperatury i współczynników krystalizacyjnych wyznaczany jest przyrost temperatury wynikający z tego zjawiska i zmiana parametrów krystalizacyjnych. Otrzymane nowe wartości są aktualizowane w węźle środowiska w wyniku operacji oddziaływania.

Realizacja zjawiska przemieszczania się wymaga by agent obserwował i oddziaływał na całą trajektorię ruchu (węzły środowiska obejmowane przez nie). Operacja oddziaływania uaktualnia atrybuty tych węzłów zgodnie z wartością kroku przemieszczenia dla danej prędkości ruchu i kroku czasowego symulacji.

Nałożenie aspektu ruchu na model bazowy zostało symbolicznie przedstawione na rys. 7.7.



Rys. 7.7. Aspekt ruchu nałożony na agenta i węzły środowiska

Aspekty zadań symulacyjnych

W praktyce symulacyjnej pojawia się wiele wymagań, które muszą być dodatkowo realizowane przez system komputerowy (sterowanie, wizualizacja, zapisywanie wyników). Wkomponowanie ich w podstawowy model wiąże się z koniecznością zmian tego modelu przy zmianie tych wymagań lub chwilowej rezygnacji z nich. Dlatego bardzo użyteczną rolą aspektów może być ich reprezentacja w postaci oddzielnych modułów aspektowych.

Implementacja

Aspekty zjawisk rozszerzają parametry węzłów środowiska poprzez mechanizm wprowadzeń (dodanie atrybutów). Funkcjonalność agentów obliczeniowych wzbogacana jest przez wprowadzenia (dodanie specyficznych funkcji) oraz odpowiednio zdefiniowane punkty cięć i rady.

Aspekty sterowania i wizualizacji są nieco bardziej skomplikowane, lecz używają tych samych środków udostępnianych przez język AspectJ.

7.5. Podsumowanie

Wykorzystanie programowania aspektowego w połączeniu z podejściem agentowym (system MAFES-3) pozwoliło na połączenie zalet tych dwóch metodologii tworzenia systemów komputerowych:

1. Uzyskano podział ról na typy agentów, tak jak to miało miejsce w przypadku systemu MAFES-1.
2. Wyodrębnienie środowiska przechowującego stan modelu ograniczyło ilość koniecznych interakcji bezpośrednich między agentami (tak jak w przypadku systemu MAFES-2).
3. Zastosowanie aspektów zjawisk podzieliło funkcjonalność agenta obliczeniowego na zestawy funkcji związane z odpowiednimi typami zjawisk zawarte w osobnych modułach (aspekty zjawisk realizowane osobno analogicznie jak agenty zjawisk w systemie MAFES-2).
4. Wykorzystując koncepcję programowania agentowego zachowano jednocześnie wszystkie zalety programowania aspektowego przedstawione w podsumowaniu rozdziału 7.3.

Programowanie aspektowe znajduje się w początkowym etapie rozwoju, dlatego system MAFES-3 został opisany w sposób skrótowy.

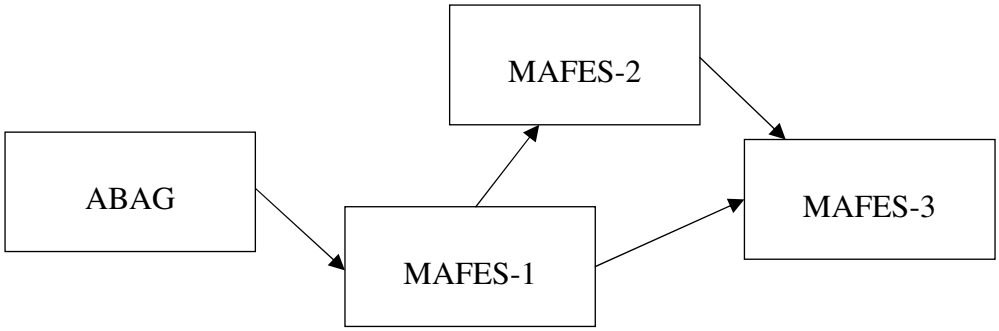
Jednym z problemów występującym przy użyciu języka AspectJ jest brak możliwości dodawania atrybutów do tylko do wybranych instancji (obiektów) danej klasy. Atrybuty można dodawać tylko do samych klas, czyli jednocześnie do wszystkich instancji. Ograniczenie to spowodowało, że atrybuty związane z aspektem zjawiska krystalizacji musiały być dodane do wszystkich węzłów środowiska, a nie tylko do tych, które były obejmowane przez to zjawisko.

Pomimo drobnych ograniczeń programowanie aspektowe rokuje duże perspektywy nie tylko dla implementacji systemów agentowych.

8. Uwagi końcowe

Przyjęta teza pracy wyraża się w stwierdzeniu: „**zastosowanie koncepcji agenta i technologii agentowych pozwala na adekwatną i efektywną reprezentację złożonych zjawisk obejmujących wymianę ciepła w połączeniu z ruchem fizycznych elementów ośrodka**”.

W celu jej udokumentowania, przeanalizowano cztery koncepcje zastosowania metod agentowych w modelowaniu procesów fizycznych poprzez złożenie zjawiska ruchu oraz zjawiska rozpływu ciepła. Opracowane przez autora metody i relacje między nimi przedstawiono schematycznie na rys. 8.1.



Rys. 8.1. Opracowane w rozprawie doktorskiej koncepcje symulacji procesów cieplnych i powiązania między nimi

Dla każdej metody została stworzona przez autora adekwatna implementacja, w postaci działających niezależnie systemów.

- System ABAG wspomaga działanie systemu ABAQUS, pozwalając na zmniejszenie wpływu ograniczeń istniejących w tym systemie i dostosowanie przebiegu symulacji do specyficznych potrzeb użytkownika.

Zastosowanie takiego rozwiązania celowe jest wówczas, gdy użytkownik posiada biegłość w korzystaniu z systemu ABAQUS (lub innego równorzędnego) i chodzi jedynie o usprawnienie przebiegu eksperymentów.

Rozwiązanie to zaproponowano jako pewne nawiązanie do istniejących technik symulacji, posiada ono jednak szereg niedogodności (omówionych szczegółowo w rozdziale 4) i dlatego – w przekonaniu autora rozprawy – celowym jest poszukiwanie bardziej radykalnych modyfikacji metod symulacyjnych.

- System MAFES-1 wykorzystuje w pełni paradygmat agentowości, przypisując poszczególnym agentom działania odpowiadające dynamice procesu stygnięcia metalu,

przepływu medium chłodzącego, a także realizacji funkcji sterowania całym procesem stygnięcia i krystalizacji. Oryginalność takiego podejścia polega na równoczesnym uwzględnieniu przepływu ciepła oraz fizycznego ruchu medium chłodzącego, co przy wykorzystaniu klasycznych (komercyjnych) systemów symulacyjnych jest trudne w realizacji.

Efektem dodatkowym jest w systemie MAFES-1 możliwość uwzględnienia działań sterujących, polegających na zmianie prędkości przepływu i/lub temperatury medium chłodzącego.

Pozostałe własności systemu MAFES-1 zostały omówione w rozdziale 5.

- System MAFES-2 jest rozwiązaniem alternatywnym w stosunku do systemu MAFES-1, opracowanie go miało na celu uzyskanie pewnych odniesień do rozwiązań zastosowanych poprzednio. I tak, zwiększenie liczby typów agentów w MAFES-2 zmniejszyło ich wewnętrzną złożoność, równocześnie jednak wzrosła złożoność struktury całego systemu. Wprowadzenie wyodrębnionego środowiska, w postaci siatki węzłów, zredukowało liczbę wzajemnych interakcji pomiędzy agentami, lecz w zamian pojawiła się potrzeba wprowadzenia obserwacji i uwzględnienia oddziaływania na środowisko. W efekcie, na podstawie przeprowadzonych eksperymentów, trudno jest jednoznacznie wskazać przewagę któregoś z systemów MAFES-1 lub MAFES-2.

Wydaje się, że pełna ocena ich własności (szczególnie z obliczeniowego punktu widzenia) może być sformułowana dopiero po przebadaniu większej liczby zastosowań (szczególnie dla procesów innych niż rozważana w pracy wymiana ciepła).

- System MAFES-3, stanowiący połączenie podejścia agentowego z wykorzystaniem idei programowania aspektowego, uznano za rozwiązanie perspektywiczne, które posiadać może istotne zalety, szczególnie z obliczeniowego punktu widzenia.

Proponowane rozwiązanie systemu MAFES-3 potraktowano jako wersję pilotową, która powinna podlegać doprecyzowaniu – w miarę rozwoju metodologii i narzędzi programowania aspektowego.

Przedstawione powyżej warianty systemu symulacyjnego zostały przetestowane na przykładach modeli rzeczywistych wyrobów odlewniczych, o różnych kształtach i własnościach fizycznych.

Wyniki przeprowadzonych eksperymentów w pełni potwierdzają pożądaną funkcjonalność badanych rozwiązań, co uznać należy za wykazanie tezy przyjętej na wstępie pracy.

Pewien niedosyt budzić może brak możliwości odniesienia uzyskanych rezultatów do danych pomiarowych otrzymanych w eksperymentach fizycznych. Ten aspekt dotyczy jednak głównie precyzji stosowanych modeli matematycznych i wartości ich parametrów, natomiast

przedmiotem prowadzonych w pracy rozważań są rozwiązania dotyczące narzędzi do wykorzystania tych modeli, nie zaś ich precyzji. W tym kontekście, przedstawione rezultaty stanowiące potwierdzenie funkcjonalności opracowanych rozwiązań, uznać można za satysfakcjonujące.

Drugim wątkiem, który w pracy nie został w pełni rozwinięty, jest problem sterowania procesem stygnięcia odlewu. Tutaj również uznano, że sama konstrukcja algorytmu sterowania wychodzi poza zakres prowadzonych rozważań, w związku z czym ograniczono się do wykazania możliwości wprowadzenia takiego algorytmu w ramach danego rozwiązania systemu symulacyjnego.

Reasumując przedstawiony powyżej przegląd opisanych w pracy dokonań, należy stwierdzić, że w odczuciu autora jako najważniejsze osiągnięcie wskazać należy opracowanie koncepcji i konstrukcji rozwiązań systemowych wykorzystujących technologie agentowe do symulacyjnego badania złożonych procesów fizycznych, a w szczególności procesów termofizycznych zachodzących w stygnących odlewach. Koncepcje te obejmują również wstępną fazę rozwiązań opartych na wykorzystaniu programowania aspektowego.

Jako perspektywę dalszych prac w danym zakresie przewiduje się:

- rozwinięcie i uszczegółowienie rozwiązań opartych na agentowiści w połączeniu z programowaniem aspektowym;
- zbadanie opracowanych rozwiązań w zastosowaniu do innych procesów fizycznych, z uwzględnieniem możliwości odniesienia otrzymanych rezultatów do wyników eksperymentów realizowanych na procesach i obiektach rzeczywistych.

Literatura

1. Dokumentacje komercyjnego systemu symulacyjnego ABAQUS

   URL: http://www.abaqus.com/

2. Aksit M., Wakita K., et al.: Abstracting object interactions using composition filters. [w:] *Proc. ECOOP'93 Workshop on Object-Based Distributed Programming*, pages. 152-184, 1993

3. AspectJ Team: The AspectJ Documentation.

   URL: http://www.eclipse.org/aspectj/

4. Batko B., Bieniasz S., Dong B.: *Symulacja procesu stygnięcia i krzepnięcia odlewów metodą elementów skończonych z wykorzystaniem systemu ABAQUS*. Praca dyplomowa, Wydział EAIiE, AGH, Kraków, 1997

5. Bernardi C., Maday Y., Patera A.T.: *A new non conforming approach to domain decomposition: The mortar element method*. [w:] Brezis H., Lions J.L., editors, College de France Seminar, Pitman, 1994.

6. Białobrzeski A., Bieniasz S., Kluska-Nawarecka S., Połcik H.: *Modelowanie agentowe w sterowaniu procesami krzepnięcia odlewów*. [w:] Piela A., Lisok J., Grosman F., editors, *KomPlasTech 2005: informatyka w technologii metali: materiały XII konferencji*, pages 45-50, Wydawnictwo Akapit, Ktraków, 2005

7. Bieniasz S.: *Simulation model of casting control cooling system using FEM and multi-agent architecture.* [w:] *Proc. Of International Workshop Control and Information Technology, IWCIT'99*, pages 31-36, VSB Technical University of Ostrava, 1999

8. Bieniasz S., Cetnarowicz K.: *MA-FE-S System for Simulation Using Agents Acting in Finite Element Environment*. [w:] J. Stefan, editors, *Proc. of the 21st International Workshop: Advances Simulation*, pages 227-233, ASIS 1999, MARQ Ostrava, Czech Republic, 1999

9. Bieniasz S., Cetnarowicz K.: *Model symulacyjny sterowania procesem krystalizacji*. Sprawozdanie z prac badawczych w ramach Grantu KBN Nr 7T08B04814.

10. Bieniasz S., Cetnarowicz K., Kluska-Nawarecka S.: *Agent-Based Simulation in Finite Element Environment*. [w:] Kluska-Nawarecka S., Polcik H., Warmuzek M., Dobrowolski G., editors, *Simulation, Designing and Control of Foundry Processes*, pages 69-75, FOCOMP'99, Instytut Odlewnictwa, Kraków, 1999

11. Bieniasz S., Cetnarowicz K., Nawarecki E., Kluska-Nawarecka S.: *Agent-Based Simulation in Finite Element Environment*. [w:] *Proc. of the Second Conference on Management and Control of Production and Logistics (MCPL'2000)*, Grenoble, 2000

12. Bieniasz S., Kraus P., Śnieżyński B., Połcik H.: *Simulation of physical phenomena by means of aspect programming*. [w:] *System Modelling Control – 2005 : proceedings of the 11th international conference : Zakopane, Poland*, (eds.) Byczkowska-Lipińska L., Szczepaniak P.S., Niedźwiedzińska H., Akademicka Oficyna Wydawnicza EXIT, 2005

13. Bieniasz S., Warmuzek M., Czekaj E., Gazda A., Połcik H.: *Computer simulation of the eutectic Al-Si alloys solidification*. [w:] Kluska-Nawarecka S., Polcik H., Warmuzek M., Dobrowolski G., editors, *Simulation, Designing and Control of Foundry Processes*, pages 77-84, FOCOMP'99, Instytut Odlewnictwa, Kraków, 1999

14. Bigus J. P., Bigus J.: *Constructing Intelligent Agents with Java*. Wiley, 1998

15. Bond A.H., Gasser L.: *An analysis of problems and research in DAI*. [w:] Bond A.H., Gasser L., editors, *Readings in Distributed Artificial Intelligence*, pages 3-35, Morgan Kaufmann, San Mateo, CA, 1988

16. Booch G.: *Object-Oriented Analysis and Design*. Addison-Wesley: Reading, MA, 1994

17. Bratman M.E.: *Intentions, Plans, and Practical Reason*. Harvard University Press, Cambridge, MA, 1994

18. Brugali D., Sycara K.: *A Model for Reusable Agent Systems. Implementing Application Frameworks*, M. Fayad et al. (editors), John Wiley & Sons, 1999

19. Boucke N., Holvoet T.: *StateBased JoinPoints: Motivation and Requirements*. …

20. Cetnarowicz K.: *Problemy projektowania i realizacji systemów wieloagentowych*. AGH Uczelniane Wydawnictwa Naukowo-Dydaktyczne, seria Rozprawy Monograficzne, Kraków, 1999

21. Chavez C.: *Design Support for Aspect-oriented Software Development*. Doctoral Symposium at OOPSLA'2001, Tampa Bay, USA, Outubro, 2001.

22. Ciszewski S., Duras M., Bieniasz S., Kisiel-Dorohinicki M., Nawarecki E.: *Koncepcja stochastycznego mikro--modelowania procesów krystalizacji*. [w:] Kusiak J. et al., editors, *KomPlastTech '2000*, pages 239-246, Wydawnictwo Akapit, Kraków, 2000

23. Coad P, Yourdon E.: *Object Oriented Analysis*. Prentice Hall, 1990

24. El Mahallawy N.A., et. al.: *Solidification Simulation of Equiaxed Grains in Al-Si Eutectic Alloys*, Vols. 217-222, pages 347-352, Materials Science Forum, Switzerland, 1996

25. Ferber J.: *Reactive distributed artificial intelligence*. [w:] O'Hare G.M.P, Jenings N.R., editors, *Foundations of Distributed Artificial Intelligence*, pages 287-317, John Wiley, 1996

26. Fraś E.: *Krystalizacja metali i stopów*. PWN, Warszawa, 1992

27. Gamma E.R., Helm R., Johnson R., Vlissides J.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994

28. Garcia A., Lucena C.: *An Aspect-Based Object-Oriented Model for Multi-Agent Systems*. Advanced Separation of Concerns Workshop at ICSE'2001, 2001

29. Garcia A., Silva V., Chavez C., Lucena C.: *Engineering Multi-Agent Systems with Aspects and Patterns*. Aceito para aparecer no Journal of the Brazilian Computer Society, SBC, 2002

    URL: http://server.teccomm.les.inf.puc-rio.br/Ftp/pub/docs/JSBC2002garcia.pdf

30. Garcia A., Chavez C., Silva O., Silva V., Lucena C.: *Promoting Advanced Separation of Concerns in Intra-Agent and Inter-Agent Software Engineering*, …

    URL: http://www.cs.ubc.ca/~kdvolder/Workshops/OOPSLA2001/submissions/28-Garcia.pdf

31. Garcia A., Kulesza U., Sardinha J., Lucena C., Milidiu R.: *The Learning Aspect Pattern*. [w:] *Proc. of the 11th Conference on Pattern Languages of Programs (PLoP2004)*, USA, 2004

32. Garcia A., Torres V., Lucena C., Milidiu R.: *An Aspect-Based Approach for Developing Multi-Agent Object-Oriented Systems*. Brazilian Symposium on Software Engineering, Rio de Janeiro, Brazil, 2001

33. Genesereth M.R., Nilsson N.: *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, San Mateo, CA, 1987

34. Górny Z., Kluska-Nawarecka S., Bieniasz S.: *Computer Modeling of Composite Casting Crystallization and Solidification Processes*. [w:] *Proc. of II Cast Composite International Conference'98* , Poland, 1998

35. Górny Z., Kluska-Nawarecka S., Połcik H., Bieniasz S.: *Simulation and experimental research of the solidification of some selected cast alloys*. [w:] Pietrzyk M., Mitura Z., Kaczmar J, editors, *ESAFORM: the 5th international conference on Material forming* : Kraków, pages 343-346, Akapit, Kraków, 2002

36. Górny Z., Kluska-Nawarecka S., Połcik H., Bieniasz S.: *Przyczynki do symulacji komputerowej krzepnięcia odlewów stosowanych w przemyśle*. Archiwum Odlewnictwa R. 2 nr 4, pages 113-119, 2002

37. Gutknecht O., Ferber J.: MadKit: *Organizing heterogeneity with groups in a platform for multiple multi-agent systems*. Tech. Rep. 97188, LIRMM, 161, France, 1997

URL: http://citeseer.nj.nec.com/gutknecht97madkit.html

38. Hayes-Roth F., Waterman D.A., Lenat D.B., editors: *Building Expert Systems*. Addison-Wesley: Reading, MA, 1983

39. Helleboogh A., Holvoet T., Weyns D.: *Time Management Support For Simulating Multi-Agent Systems*. [w:] Joint workshop on multi-agent and multi-agent-based simulation (Davidsson, P. and Gasser, L. and Logan, B. and Takadama, K., eds.), pages 31-40, 2004

40. Helleboogh A., Holvoet T., Weyns D., Berbers Y.: *Extending Time Management Support For Multi-Agent Systems*. [w:] Multi-Agent and Multi-Agent-Based Simulation: Joint Workshop MABS 2004, vol 3415 / 2005, Lecture Notes in Computer Science, pages 37-48, 2005

41. Huhns M.N., Singh M.P.: *Agents and multiagent systems: Themes, approaches, and challenges*. [w:] Huhns and Singh [42], pages 1-23

42. Huhns M.N., Singh M.P., editors, *Readings in Agents*. Morgan Kaufmann, San Francisco, CA, 1998

43. Jennings N.R., Sycara K. Wooldridge M.: *A roadmap of agent research and development*. Autonomous Agents and Multi-Agent Systems, 1, pages 7-38, 1998

44. Kapturkiewicz W.: *Model i numeryczna symulacja krystalizacji odlewu*. Zeszyty naukowe AGH, Kraków, 1988

45. Kącki E.: *Równania różniczkowe cząstkowe w zagadnieniach fizyki i techniki*. WNT, Warszawa, 1992

46. Kendall E.A.: *Agent Roles and Aspects*, [w:] *Proc. of The Aspect-Oriented Programming Workshop at ECOOP'98*, 1998

URL: http://trese.cs.utwente.nl/aop-ecoop98/papers/Kendall.pdf

47. Kendall E.A.: *Aspect-Oriented Programming For Role Models*, [w:] *Proc. of The Aspect-Oriented Programming Workshop at ECOOP'99*, 1999

URL: http://trese.cs.utwente.nl/aop-ecoop99/papers/kendall.pdf

48. Kiczales G., Lamping J., Mendhekar A., Maeda C., Lopes C., Loingtier J.-M., and Irwing J.: *Aspect-Oriented Programming*, Proceedings of ECOOP'97, Springer Verlag, pages 220-242, 1997

49. Kiczales G., Hilsdale E., Hugunin J., Kersten M., Palm J., Griswold W.G.: *An Overview of AspectJ*. [w:] *Proc. of the ECOOP 2001*, Lecture Notes in Computer Science, vol. 2072, Springer, 2001

50. Kluska-Nawarecka S., Bieniasz S., Cetnarowicz K.: *Computer Modelling of Casting Cooling Control System*. [w:] University of Coimbra, editors, *Proc. of the conference CONTROLO'98*, APCA, Coimbra, Portugal, 1998

51. Kluska-Nawarecka S., Bieniasz S., Cetnarowicz K.: *Komputerowy model krzepnięcia odlewów z materiałów jednorodnych i kompozytów*. [w:] Pietrzyk M., Kusiak J., Piela A., editors, *Zastosowanie Komputerów w Zakładach Przetwórstwa Metali '98*, pages 43-50. Wydawnictwo Akapit, Kraków, 1998

52. Kulesza U., Garcia A., Lucena C.: *Generating Aspect-Oriented Agent Architectures*. [w:] *Proc. of the 3rd Workshop on Early Aspects - Aspect-Oriented Requirements Engineering and Architecture Design*, 3rd International Conference on Aspect-Oriented Software Development, Lancaster, UK, 2004

53. Kulesza U., Garcia A., Lucena C.: *Towards a Method for the Development of Aspect-Oriented Generative Approaches*. Workshop on Early Aspects - Aspect-Oriented Requirements Engineering and Architecture Design, OOPSLA'04, Vancouver, Canada, 2004

54. Lange D., Oshima M.: *Programming and Developing Java Mobile Agents with Aglets*. Addison-Wesley, 1998

55. Maes P.: *Concepts and Experiments in Computational Reflection*. ACM SIGPLAN Notices, 22(12):147-155, 1987

56. Majchrzak E., Mochnacki B.: *Metody numeryczne – podstawy teoretyczne, aspekty praktyczne i algorytmy*. Wydawnictwo Politechniki Śląskiej, Gliwice, 2004

57. Meyer B.: *Object Oriented Software Construction, Second Edition*. Prentice-Hall, Englewood Cliffs, NJ, 1997.

58. Mochnacki B., Suchy J.S.: *Modelowanie i symulacja krzepnięcia odlewów*. PWN, Warszawa, 1993.

59. Müller J.P, Pischel M., Thiel M.: *Modelling reactive behaviour in vertically layered agent architectures*. [w:] Wooldridge M., Jennings N.R., editors, *Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890)*, pages 261-276, Springer-Verlag, Berlin, 1995

60. Nwana H.S., Ndumu D.T., Lee L.C., Collis J.C.: *ZEUS: a toolkit and approach for building distributed multi-agent systems*. [w:] Etzioni O., Muller J.P., Bradshaw J.M. (eds.), *Proc. of the Third International Conference on Autonomous Agents (Agents'99)*, pages 360–361, ACM Press, Seattle, WA, USA, 1999

61. Object Management Group– Agent Platform Special Interest Group: *Agent Technology – Green Paper*. Version 1.0, 2000

URL: http://www.jamesodell.com/ec2000-08-01.pdf

62. Parnas D.L.: *On the Criteria to be Used in decomposing Systems into Modules*. [w:] *Communications of the ACM*, vol. 15(2), 1972

63. Połcik H., Bieniasz S., Kluska-Nawarecka S., Warmuzek M.: *Simulation and Control of Solidification Processes*. [w:] *Proc. of the Second Conference on Management and Control of Production and Logistics (MCPL'2000)*, Grenoble, 2000

64. Robbes R., Bouraqadi N., Stinckwich S.: *An Aspect-based Multi-Agent System*. ESUG Conference, 2004

65. Russell S.J., Norwig P.: *Artificial Intelligence. A Modern Approach*. Prentice Hall, Englewood Cliffs, New Jersey, 1995

66. Sant'Anna C., Garcia A., Chavez C., Lucena C., von Staa A.: *On the Reuse and Maintenance of Aspect-Oriented Software: An Assessment Framework*. XVII Brazilian Symposium on Software Engineering, Manaus, Brazil, 2003

67. Silva O., Garcia A., Lucena C.: *A Unified Software Architecture for System-Level and Agent-Level Dependability in Multi-Agent Object-Oriented Systems*. Workshop on Mobile Object Systems at ECOOP'2001, Budapest, 2001

68. Warmuzek M., Kluska-Nawarecka S., Bieniasz S., Połcik H.: *Effect of the Kinetic Coefficients Changes on the Process of Control of the Casst Part Microstructure Formation by the Computer Simulation Method*. [w:] *Proc. of the Second Conference on Management and Control of Production and Logistics (MCPL'2000)*, Grenoble, 2000

69. Warmuzek M., Kluska-Nawarecka S., Połcik H., Rabczak K., Bieniasz S.: *Les modeles physiques et la simiuation de la formation de la morphologie de la microstructure d'un alliage Al-Si*, [w:] *Proc. Congres international sur l'aluminium et sa mise en forme a chaud: moulage et forgeage TransAl'2002*, Lyon, 2002

70. Weiss G., editor: *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, London, 1999

71. Weyns D., Helleboogh A., Holvoet T., et al.: *Agents are not part of the problem, agents can solve the problem*. [w:] Proc, of the OOPSLA 2004 Workshop on Agent-oriented Methodologies, 2004

72. Wooldridge M.: *Intelligent Agents*. [w:] Weiss [70], pages 27-78

73. Wooldridge M., Jennings N.R.: *Intelligent agents: Theory and practice*. The Knowledge Engineering Review, 10(2), pages 115-152, 1995

74. Vincent R., Horling B., Lesser V.: *An Agent Infrastructure to Build and Evaluate Multi-Agent Systems: The Java Agent Framework and Multi-Agent System Simulator*. Lecture

Notes in Artificial Intelligence: Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems. Volume 1887, Wagner and Rana (eds.), Springer, 2001

URL: http://mas.cs.umass.edu/paper/200

75. Zeigler B.P.: *Teoria modelowania i symulacji*. PWN, Warszawa, 1984
76. Zienkiewicz 0.C., Taylor R.L: *The Finite Element Method*. Butterworth-Heinemann, Oxford, 2000
77. Zunino A., Amandi A.: *Brainstorm/J: a Java framework for intelligent agents*. [w:] *Proc. of the 2nd Argentine Symposium on Artificial Intelligence* (ASAI 2000 - XXIX JAIIO), SADIO, Tandil, Buenos Aires, Argentina, 2000.

URL: http://www.exa.unicen.edu.ar/~azunino/asai2000.ps.gz

Dodatek A. Język AspectJ

AspectJ [3, 49] jest prostym, jednolitym i praktycznym rozszerzeniem języka programowania Java. Dodając kilka nowych konstrukcji do tego języka, zapewnia wsparcie dla modularnej implementacji przecinających się zagadnień.

Podstawowymi pojęciami języka AspectJ są:

- *punkt złączenia* - dobrze określony punkt wykonywania programu,
- *punkt cięcia* - kolekcja punktów złączeń,
- *rada* - konstrukcja podobna do funckji, która jest dołączana do punktów cięć,
- *wprowadzenie* – pole lub funkcja dodawana do istniejącej klasy,
- *aspekt* – jednostka modularnej implementacji przecinającego zagadnienia, składająca się z powyższych elementów oraz z zwykłych deklaracji pól i funkcji języka Java.

W AspectJ kod aspektowy jest dołączany do standardowego kodu bajtowego Javy. Istnieją rozszerzenia popularnych środowisk programistycznych Javy (np. AJDT dla Eclipse), pozwalające na używanie języka AspectJ, tworzenie aspektów i ich elementów, przeglądanie struktury przecinających się aspektów w sposób podobny do przeglądania hierarchii dziedziczenia klas.

Programy napisane przy użyciu języka AspectJ są łatwe do zrozumienia i analizy.

Punkty złączeń i punkty cięć

Punkt złączenia (ang. *join point*) może być interpretowany jako węzeł w obiektowym grafie wywołań sterowania programu. Przykładami punktów złączeń są:

- wywołanie funkcji (ang. *call*),
- wykonanie funkcji (ang. *execution*),
- odczyt lub zapis pola obiektu lub klasy,
- wywołanie procedury obsługi wyjątku.

Punkty złączeń aspektu mogą dotyczyć zarówno klas jak i innych aspektów. Punkty złączeń są wykorzystywane przy definicji punktów cięć.

Punkt cięcia (ang. *pointcut*) jest zbiorem punktów złączeń i ma bezpośrednie odwzorowanie w języku AspectJ. Słowo *pointcut* jest słowem kluczowym tego języka.

Definicja punktu cięcia składa się z dwóch członów oddzielonych dwukropkiem. Lewy człon stanowi słowo kluczowe *pointcut*, nazwa definiowanego punktu złączenia i lista

ewentualnych parametrów. Prawa strona opisuje definiowany punkt cięcia przy użyciu słów kluczowych takich jak: *call*, *execution*, *handler*, *this*, *target*, *within*, *cflow*, i innych odpowiadającym odpowiednim typom punktów złączeń. Występują w niej również nazwy typów i klas obiektów do których odnoszą się te punkty złączeń.

Przy opisywaniu złożonych punktów cięć można używać operacji logicznych (takich jak: suma, koniunkcja, negacja). Dodatkowo stosuje się często znaki zastępcze (*, ..), które obejmują dowolny typ zwracany przez funkcję, klasę, pole, funkcję, dowolną listę parametrów funkcji.

Rady

Rada (ang. *advice*) to mechanizm służący do definicji kodu, instrukcji, które powinny zostać uruchomione dla każdego z punktów złączeń należących do punktu cięcia, na którym jest oparta dana rada. Punkt cięcia może być anonimowy, opisany w definicji rady lub nazwany, wcześniej zdefiniowany.

Występują trzy podstawowe typy rad, będące słowami kluczowymi służącymi do ich definicji: *before*, *after*, *around*. Rada *before* wykonywana jest przed punktem złączenia, rada *after* po nim. Natomiast rada *around* zastępuje kod punktu złączenia.

Dodatkowo można definiować radę *after returning* i radę *after throwing*. Pierwsza z nich uruchamiana jest po normalnym zakończeniu punktu złączenia, druga w sytuacji gdy jego uruchomienie zakończyło się zgłoszeniem wyjątku.

Rada *around* ma specyficzną właściwość iż może zamienić kod punktów złączeń, z którymi jest związana, zamieniany kod może być wykonany przez specjalne wywołanie *proceed*, wywołanie to może być uzależnione od różnych warunków. Pozostałe typy rad mają charakter addytywny w stosunku do kodu punktów złączeń.

Wprowadzenia

Jest kilka typów wprowadzeń (ang. *inter-type declaration*). Aspekt może deklarować składniki (pola, funkcje, konstruktory), które zostaną dodane do istniejącego typu (klasy, aspektu).

Aspekt może deklarować również, że jakaś klasa dziedziczy z innej klasy lub implementuje jakiś interfejs, czyli może zmieniać zależności w hierarchii dziedziczenia.

Zatem wprowadzenia zmieniają strukturę klas i ich hierarchii.

Aspekty

Aspekt łączy punkty cięć, rady, wprowadzenia w modularną jednostkę przecinającej implementacji. Przypomina definicję klasy i może poza wymienionymi składnikami posiadać również funkcje i pola.

Aspekt może dziedziczyć z innego aspektu, wspiera to abstrakcję i kompozycję przecinających zagadnień. Dziedziczone są nie tylko pola i funkcje ale również punkty cięć. Aspekt jednak może dziedziczyć tylko z aspektu abstrakcyjnego. Dopuszczalne jest również dziedziczenie aspektu z klasy lub implementowanie przez niego interfejsu.

Przykładowy aspekt

Poniżej przedstawiono przykład znajdujący się w dokumentacji języka AspectJ [3], pokazujący prosty aspekt składający się z opisanych wcześniej podstawowych jego elementów.

Pierwsza linia zawiera słowo kluczowe *aspect* i występującą po nim nazwę tworzonego aspektu *FaultHandler*, którego definicja rozpoczyna się klamrą otwierającą w linii 1 i kończy się klamrą zamykającą w linii 23.

```
 1 aspect FaultHandler {
 2
 3   private boolean Server.disabled = false;
 4
 5   private void reportFault() {
 6     System.out.println("Failure! Please fix it.");
 7   }
 8
 9   public static void fixServer(Server s) {
10     s.disabled = false;
11   }
12
13   pointcut services(Server s): target(s) && call(public * *(..));
14
15   before(Server s): services(s) {
16     if (s.disabled) throw new DisabledException();
17   }
18
19   after(Server s) throwing (FaultException e): services(s) {
20     s.disabled = true;
21     reportFault();
22   }
23 }
```

W linii 3 zastosowano mechanizm wprowadzenia. Mianowicie do klasy *Server* dodawane jest pole typu logicznego o nazwie *disabled*. Tak więc wszystkie obiekty klasy *Server* będą posiadały pole *disabled*, pomimo tego iż w definicji klasy Server nie zdefiniowano takiego pola. Na tym polega mechanizm wprowadzeń, rozszerzający strukturę istniejących klas w sposób nie zmieniający definicji klas do których dodawane są pola czy funkcje.

Następnie zdefiniowane są w przykładowym aspekcie dwie funkcje *reportFault* (linie 5-7) i *fixServer* (linie 9-11). Sposób definicji pól i funkcji aspektu jest taki sam jak definiowanie ich w klasach w przy użyciu języka Java.

Kolejnym elementem aspektu jest punkt cięcia o nazwie *services* (linia 13). Parametrem tego punktu cięcia jest obiekt klasy *Server*. Punkt cięcia *services* odnosi się do wywołań (*call*) funkcji o dowolnej nazwie z dowolną listą argumentów o zasięgu publicznym. Pierwszy element koniunkcji (*&&*) prawej strony definicji punktu cięcia (*target(s)*) ogranicza wywołania funkcji do tych, które są wywoływane dla obiektów klasy *Server* (*s*). Punkt cięcia określa miejsca, w które może zostać dodana nowa funkcjonalność przy pomocy rad.

Na punkcie cięcia *services* oparte są dwie rady. Pierwsza z nich (linie 15-17) dołączana jest przed wywołaniem funkcji (*before*) opisanych punktem cięcia, druga (linie 19-22) uruchamiana jest gdy wywołania tych funkcji zakończą się zgłoszeniem wyjątku. Rady te uruchamiane są przy osiągnięciu przez sterowanie programu odpowiedniego punktu złączenia należącego do punktu cięcia, na którym oparta jest definicja danej rady.

Dodatek B. Sposób konfiguracji symulacji w systemach MAFES i APSS

Dla potrzeb przygotowania danych wejściowych dla symulacji w systemach MAFES-1, MAFES-2 oraz MAFES-3 został opracowany format MMD (format pliku tekstowego). Format MMD omówiono w dodatku B.1.

W celu konfiguracji symulacji w systemie APSS (rozdział 7.3) wykorzystano format XML, na bazie którego zdefiniowano odpowiedni zestaw elementów i ich atrybutów. Format ten został przedstawiony w dodatku B.2.

B.1. Format MMD

Format MMD (*ang. Mafes Model Data*) pozwala na opisanie w pliku tekstowym modelu wejściowego dla symulacji w systemie MAFES. Odpowiednie elementy pliku MMD definiują obiekty geometryczne, parametry termofizyczne materiałów, warunki początkowe i brzegowe, gęstość siatki węzłów, warunki realizacji obliczeń (krok czasowy, czas obliczeń) oraz dane wyjściowe symulacji.

Przykładowy plik MMD

Poniżej przedstawiony został przykład pliku MMD zawierającego dane wejściowe dla symulacji schładzania odlewu schodkowego z miedzi umieszczonego w formie piaskowej.

W liniach 1 – 11 zostały określone wymiary i położenia prostopadłościennych (typ figury *PP*) obiektów geometrycznych – schodków, wchodzących w skład odlewu, łącznie z nadlewem. Kolejne linie (13 – 14) opisują geometrię i położenie formy wewnątrz której znajduje się odlew.

```
 1   PP maly(0.1, 0.01, 0.1)
 2   maly.move(0.2, -0.01, 0)
 3
 4   PP sredni(0.1, 0.02, 0.1)
 5   sredni.move(0.1, -0.02, 0)
 6
 7   PP duzy(0.1, 0.06, 0.1)
 8   duzy.move(0, -0.06, 0)
 9
10   PP nadlew(0.06, 0.25, 0.1)
11   nadlew.move(-0.06, -0.06, 0)
12
13   PP forma(0.45, 0.3, 0.2)
14   forma.move(-0.1, -0.11, -0.05)
```

```
15
16   Pipe pipe(0.01, 0.01, +3, 0.2, 0.15, -0.04, -0.05)
17
18   maly.setT(650)
19   sredni.setT(650)
20   duzy.setT(650)
21   nadlew.setT(650)
22   forma.setT(80)
23   pipe.setT(20)
24
25   Material m_cast(210, 1180, 2550)
26   Material m_mould(1, 1333, 1500)
27
28   m_cast.latentHeat(3.73E+5, 610, 577)
29
30   m_cast.applyOn(maly)
31   m_cast.applyOn(sredni)
32   m_cast.applyOn(duzy)
33   m_cast.applyOn(nadlew)
34   m_mould.applyOn(forma)
35
36   pipe.setV(0.01)
37
38   Mesh mesh(0.01)
39
40   Timer time(100, 0.1, 50)
41
42   RaportPoint m(0.25, -0.005, 0.05, 0.01)
43   m.save(T, FS, N, R)
44
45   RaportPoint s(0.15, -0.01, 0.05, 0.01)
46   s.save(T, FS, N, R)
47
48   RaportPoint d(0.05, -0.03, 0.05, 0.01)
49   d.save(T, FS, N, R)
```

W linii 16 wprowadzono do modelu wejściowego dynamiczny element chłodzący umiejscowiony pod średnim schodkiem.

Następnie (linie 18 – 22) przypisują obiektom geometrycznym odlewu i formy temperatury początkowe, natomiast w linii 23 ustalana jest wartość temperatury wpływającego medium chłodzącego.

Parametry termofizyczne odlewu i formy zostały podane w liniach 25 – 28, natomiast w liniach 30 – 34 przypisane zostały one do odpowiednich obiektów geometrycznych. W wierszu 36 określono prędkość przepływu medium chłodzącego.

Warunki realizacji obliczeń obejmują odległość między węzłami siatki węzłów (linia 38) oraz czas symulacji, krok czasowy (linia 40).

W liniach 42, 45, 48 określono punkty umiejscowione w schodkach odlewu, odpowiednio małym, średnim i dużym. Dla punktów tych będą zapisywane wyniki w raporcie symulacji. Wielkości zapisywane (temperatura – T, współczynnik udziału fazy stałej – FS, ilość kryształów – N, promień średni kryształów – R) w tych punktach zostały ustalone w liniach 43, 46, 49.

Geometrie obiektów

Obiekty geometryczne służą do określania obszarów przestrzeni XYZ, które charakteryzują się tymi samymi parametrami termofizycznymi lub tymi samymi warunkami początkowymi.

Definicja obiektu geometrycznego w formacie MMD składa się z następujących elementów:

```
TYP_OG  id_og ( lista parametrów )
```

Typ obiektu geometrycznego (*TYP_OG*) może przyjmować wartości spośród następujących literałów: *PP*, *Cone*, *CC*, *Pyramid*, *Pipe*, które oznaczają odpowiednio:

- *PP* – prostopadłościan, na liście parametrów umieszcza się rozmiary w kierunkach kolejno X, Y i Z;
- *Cone* – stożek ścięty, na liście parametrów kolejno: promień podstawy dolnej, promień podstawy górnej, wysokość. Promień podstawy górnej może przyjmować wartość 0.
- *CC* – figura powstała przez obrót trapezu dookoła osi OY, na liście parametrów kolejno: odległość prawego wierzchołka podstawy dolnej trapezu od osi obrotu, odległość prawego wierzchołka podstawy górnej od osi, wysokość trapezu, odległość lewego wierzchołka podstawy dolnej od osi, odległość lewego wierzchołka podstawy górnej od osi, początkowy kąt obrotu, końcowy kąt obrotu.
- *Pyramid* – ostrosłup ścięty, o wysokości w kierunku X, na liście parametrów: wysokość, rozmiar podstawy dolnej w kierunku Y i Z, rozmiar podstawy górnej w kierunku Y i Z, przesuniecie podstawy górnej od osi OX w kierunku Y i Z.
- *Pipe* – rurka chłodząca o przekroju prostokątnym, na liście parametrów kolejno: szerokość przekroju, wysokość przekroju, kierunek płynięcia medium (+1 w kierunku OX, -1 w kierunku przeciwnym, +2, w kierunku OY, -2 w kierunku przeciwnym, +3 w kierunku OZ, -3 w kierunku przeciwnym), długość, współrzędne X, Y, Z początku rurki.

W systemie MAFES zdefiniowano takie typy obiektów geometrycznych, których potrzeba użycia wynikała z praktyki symulacyjnej.

Identyfikator obiektu geometrycznego (*id_og*) służy nazwaniu danego obiektu, w celu późniejszemu przypisaniu temu obiektowi parametrów termofizycznych i wartości początkowych temperatury.

Istotną rolę odgrywa kolejność definiowanych obiektów. Przy generacji siatki sprawdzana jest przynależność danego węzła do obiektu geometrycznego w takiej kolejności w jakiej są te obiekty zdefiniowane. Zatem pierwszy obiekt zajmuje obszar jemu właściwy.

Natomiast jeśli obszar kolejnych obiektów zachodzi na obszar wcześniej zdefiniowanych, to ten wspólny obszar jest z tego kolejnego obiektu odejmowany.

Tworzone obiekty geometryczne są umiejscawiane w punkcie o zerowych wartościach współrzędnych XYZ. W celu przemieszczenia obiektu w inne miejsce należy zastosować operację *move* dla danego obiektu geometrycznego:

```
id_og . move ( dx ,  dy ,  dz )
```

Parametry materiałów

Parametry termofizyczne wprowadzane są poprzez definiowanie materiałów:

```
Material  id_mat ( cp ,  k ,  g )
```

W powyższym zapisie cp oznacza ciepło właściwe, k – przewodność cieplną, natomiast g – gęstość.

Dla odlewów, w których występuje przemiana fazowa istniej możliwość określenia parametrów krystalizacyjnych poprzez zapis:

```
id_mat . latentHeat ( LH ,  TL ,  TS )
```

Tutaj *LH* oznacza ciepło krystalizacji, *TL* – temperaturę *liquidus*, *TS* – temperaturę *solidus*.

Po zdefiniowaniu materiału, przypisanie go do wybranych obiektów geometrycznych realizuje się przez:

```
id_mat . applyOn ( id_og )
```

Zakłada się jednostki wartości parametrów termofizycznych wyrażone w układzie spójnym (np. SI).

Warunki początkowe i brzegowe

Poniższy zapis danemu obiektowi geometrycznemu (*id_og*), ustala temperaturę początkową równą wartości *temp*.

```
id_og . setT ( temp )
```

Dla rurek chłodzących temperaturę wpływającego medium podaje się w taki sam sposób jak temperaturę początkową obiektów. Natomiast prędkość przepływu medium określa się następująco:

```
id_pipe . setV ( v )
```

Budowa siatki

W symulacji realizowanej przez system MAFES tworzona jest automatycznie regularna, prostopadłościenna siatka węzłów dla obszaru minimalnego obejmującego

wszystkie obiekty geometryczne. Gęstość tej siatki ustala się poprzez odległość między sąsiednimi węzłami siatki (*ds*):

```
Mesh mesh ( ds )
```

Warunki realizacji obliczeń

Symulacja w systemie MAFES realizowana jest przy stałej wartości kroku czasowego. Wartość tego kroku jest przed obliczeniami sprawdzana, ze względu na warunek stabilności. Wartość stabilnego kroku jest odwrotnie proporcjonalna do odległości między węzłami siatki.

Czas całkowity symulacji (*total*), krok czasowy (*dt*), oraz ilość kroków co ile aktualizowane są okna wizualizacyjne (*ni*) opisuje się w następujący sposób:

```
Timer  time ( total ,  dt ,  ni )
```

Dane wyjściowe symulacji

W celu zachowania przebiegu zmian wartości obliczanych w trakcie symulacji, należy zdefiniować punkty raportujące, podając ich współrzędne, oraz odstęp czasowy (*dt*) – co ile mają być zapisywane wyniki.

```
RaportPoint  id_rp ( x ,  y ,  z , dt )
```

Dla określonego punktu (*id_rp*) należy następnie wyszczególnić, które wielkości maja być zapisywane (*w_lista* zawiera listę oddzielonych przecinkami symboli spośród T, FS, N, R).

```
id_rp . save ( w_lista )
```

B.2. Format XML

Konfiguracja systemu APSS (rozdział 7.3) odbywa się przez wczytanie pliku konfiguracyjnego w formacie *XML*. Plik powinien zawierać opis początkowego stanu modelu oraz wszystkich jego atrybutów wymaganych przez symulowane zjawiska.

Konfiguracja została omówiona na przykładzie. Przykładowy plik konfiguracyjny został przedstawiony na rys. B.1.

```
- <mafes>
    <timer start="0" stop="1024" step="1" />
    <mesh grid="1" />
  - <materials>
      <material name="air" density="10.0" specificHeat="0.1" thermalConductivity="50.0" />
      <material name="steel" density="10.0" specificHeat="0.7" thermalConductivity="25.0" />
    </materials>
  - <attributes>
      <attribute name="temperature" description="none" defaultValue="0.0" />
    </attributes>
  - <space>
      <coordinates x="0" y="0" z="0" />
      <shape name="cuboid" width="75" height="25" depth="10" />
      <material name="air" />
    - <attributes>
        <attribute name="temperature" value="0.0" />
      </attributes>
    - <subspaces>
      - <space>
          <coordinates x="2" y="9" z="3" />
          <shape name="cuboid" width="5" height="5" depth="5" />
          <material name="steel" />
          <movementVector x="1" y="0" z="0" />
        - <attributes>
            <attribute name="temperature" value="1000.0" />
          </attributes>
        </space>
      - <space>
          <coordinates x="10" y="20" z="2" />
          <shape name="cuboid" width="17" height="5" depth="7" />
          <material name="steel" />
        - <attributes>
            <attribute name="temperature" value="650.0" />
          </attributes>
        </space>
      </subspaces>
    </space>
  </mafes>
```

Rys. B.1. Przykładowy plik konfiguracyjny

Oto elementy, które powinny znaleźć się w pliku konfiguracyjnym oraz ich znaczenie:

- **mafes**

  Główny element pliku, nie zawiera atrybutów. Oznacza początek konfiguracji.

- **timer**

  Opisuje parametry czasowe symulacji. Zawiera następujące atrybuty:

  - *start* – czas rozpoczęcia symulacji,
  - *stop* – czas zakończenia symulacji,
  - *step* – krok symulacji.

  Dzięki tym atrybutom można regulować czas trwania symulacji (*start*, *stop*) oraz jej szybkość (*step*).

- **mesh**
  Opisuje parametry siatki modelu, na której odbywa się symulacja. W obecnym podejściu zawiera jeden atrybut *grid*, oznaczający gęstość siatki.
- **materials**
  Jest to element, w którym zebrane są dopuszczalne materiały, z których mogą być zbudowane obiekty. Materiały definiowane są w celu uproszczenia opisu obiektów i ujednolicenia ich zachowania. Pojedynczy materiał opisany jest w elemencie *material*.
- **material**
  Element podrzędny w stosunku do elementu *materials*. Opisuje własności fizyczne materiału, wykorzystywane przez symulowane zjawiska, np. przy rozchodzeniu się ciepła. W obecnej wersji zawiera następujące atrybuty:
  - *name* – unikatowa nazwa materiału,
  - *density* – gęstość,
  - *specificHeat* – ciepło właściwe,
  - *thermalConductivity* – przewodność cieplna.
  Własności materiału mogą oczywiście zostać rozszerzone o nowe.
- **attributes**
  Zawiera kolekcje fizycznych własności/atrybutów (np. temperatura, ciśnienie), których wartości są modelowane przez symulowane zjawiska. Głównym przeznaczeniem tych atrybutów jest wykorzystanie ich w wizualizacji.
- **attribute**
  Element opisujący pojedynczą własność/atrybut. Opisany jest przez:
  - *name* – unikatowa nazwa atrybutu,
  - *defaultValue* – wartość domyślna,
  - *description* – dowolny tekstowy opis atrybutu.
- **space**
  Zawiera opis obiektu, który umieszczany jest na scenie tworzonej symulacji. Opis obiektu składa się z następujących elementów podrzędnych:
  - **coordinates**
  Współrzędne obiektu, zazwyczaj określane względem środka ciężkości. Określane przy pomocy trzech wartości: x, y, z w kartezjańskim układzie odniesienia.
  - **shape**
  Kształt obiektu, w obecnej wersji dopuszczalne prostopadłościan lub kula. Definicja kształtu powinna zawierać jego nazwę, atrybut *name* oraz opisujące go wartości numeryczne. W przypadku prostopadłościanu są to wysokość (*height*), szerokość (*width*) oraz głębokość (*depth*). Dla kuli będzie to promień (*radius*).

- **material**

Materiał, z którego zbudowany jest obiekt. Zawiera jeden, jednoznacznie identyfikujący go element *name*, czyli nazwę materiału wybraną z listy materiałów zdefiniowanych w elemencie *materials*.

- **attributes**

Początkowe wartości fizycznych atrybutów obiektu. Opisany przez nazwę atrybutu (*name*) oraz jego wartość (*value*).

- **subspaces**

Ponieważ obiekty w modelu tworzą hierarchiczne drzewo z główna sceną symulacji jako korzeniem, dlatego dla każdego obiektu definiowane są jego „podobiekty", czyli obiekty, które są w nim zawarte. Lista podobiektów jest umiejscowiona w elemencie *subspaces* i składa się z elementów *space*, które są identyczne w opisie i znaczeniu jak *space*.

- **movementVector**

Element opisujący wektor ruchu, przypisany do danego obiektu. Opisany jest przez trzy atrybuty *x*, *y*, *z*, definiujące ruch w kierunku odpowiednich współrzędnych.

Dodatek C. Narzędzia wykorzystane przy implementacji systemów

W niniejszym dodatku dokonano wyszczególnienia narzędzi i języków programowania wykorzystanych przy implementacji systemów prezentowanych w pracy. Zostało to przedstawione w tabeli C.1.

Tabela C.1. Języki programowania, biblioteki, narzędzia programistyczne, środowiska systemowe wykorzystane w implementacji systemów omawianych w pracy

System	Języki i biblioteki	Narzędzia programistyczne	System operacyjny
ABAG (rozdział 4)	C++	GCC, Make	Linux
MAFES-1 (rozdział 5)	C++	GCC, Make, Cygwin	Linux, Microsoft Windows
MAFES-2 (rozdział 6)	Java	Eclipse, Sun JDK	Linux, Microsoft Windows
APSS (rozdział 7.3)	Java, AspectJ, Java3D	Eclipse, AJDT, Sun JDK	Linux, Microsoft Windows
MAFES-3 (rozdział 7.4)	Java, AspectJ	Eclipse, AJDT, Sun JDK	Linux, Microsoft Windows

Wymienione w tabeli skróty oznaczają:

- GCC – darmowe kompilatory języków C, C++, … (*ang. GNU Compiler Collection*), adres WWW: http://gcc.gnu.org/
- Eclipse – darmowe, profesjonalne środowisko programistyczne (IDE) języka Java, dla którego dostępnych jest bardzo duża ilość modułów rozszerzających, adres WWW: http://www.eclipse.org/
- Sun JDK – pakiet darmowy udostępniany przez firmę Sun, zawierający podstawowy zestaw programów narzędziowych, bibliotek i środowisko wykonawcze dla aplikacji tworzonych w języku Java (*ang. Java Development Kit*), adres WWW: http://java.sun.com/j2se/

- AJDT – moduł rozszerzający dla Eclipse, umożliwiający stosowanie w projektach języka AspectJ (*ang. AspectJ Development Tools*),
  adres WWW: http://www.eclipse.org/ajdt/
- Java3D – biblioteka rozszerzająca standardowe biblioteki języka Java o możliwość tworzenia grafiki trójwymiarowej,
  adres WWW: https://java3d.dev.java.net/

Dodatek D. Instrukcja użytkowania systemu MAFES-2

W dodatku tym przedstawiono opis instalacji i użytkowania systemu MAFES-2, którego założenia projektowe omówiono w rozdziale 6. Wersję binarną tego systemu umieszczono na płycie CD dołączonej do pracy.

Instalacja programu

Wersję wykonywalną programu można pobrać:

a) z płyty CD, ścieżka: „mafes/mafes_2_bin.zip”
b) z Internetu, adres: „http://home.agh.edu.pl/bieniasz/mafes/mafes_2_bin.zip”

Instalacja programu polega na rozpakowaniu pliku „mafes_2_bin.zip” do wybranego katalogu na dysku twardym. Plik ten zawiera katalog „mafes_2” z plikami: „mafes.jar”, „run.bat” oraz dwoma przykładowymi plikami MMD (dane wejściowe dla symulacji).

Aplikacja może być uruchomiona pod systemem operacyjnym, w którym zainstalowano wirtualną maszynę Javy 5.0 (*ang. Java Runtime Environment* – JRE 5.0).

JRE 5.0 (dla systemów Microsoft Windows, Sun Solaris, Linux) można pobrać ze strony: „http://java.sun.com/j2se/1.5.0/download.jsp”.

Symulator MAFES-2 był testowany na kilku platformach sprzętowych, z których sprzęt o najmniejszej mocy obliczeniowej charakteryzował się:
- rozmiarem pamięci operacyjnej: 256 MB,
- procesorem: Intel Pentium III 800 MHZ.

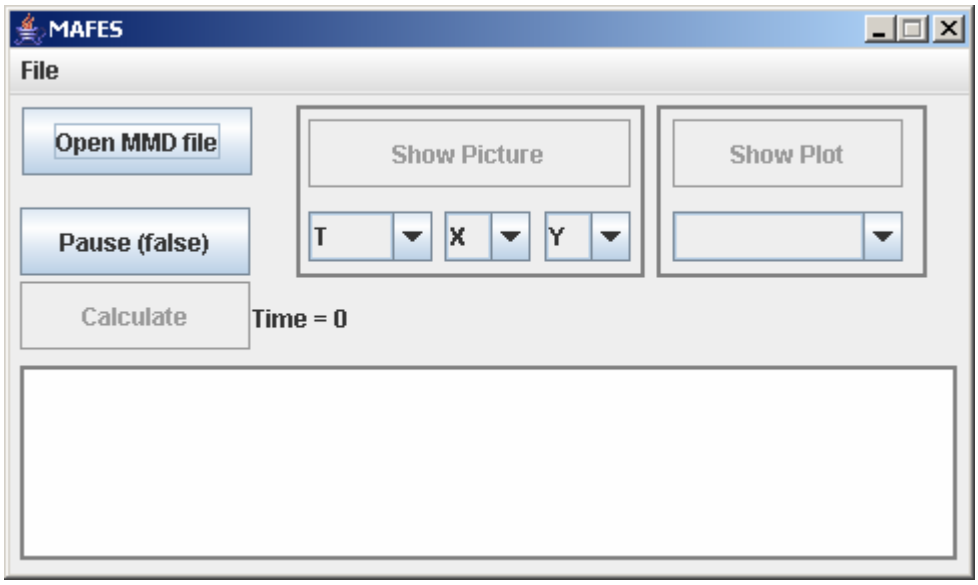
Należy zaznaczyć, że czas obliczeń i zużycie pamięci zależne jest od ilości węzłów siatki obliczeniowej symulowanego modelu.

Uruchomienie programu

Program można uruchomić w jeden z następujących wariantów:

a) w okienku interpretera poleceń systemu poleceniem: „`java -jar mafes.jar`”, wydanym w katalogu „mafes_2”, w którym powinien być plik „mafes.jar”,
b) w systemie Microsoft Windows przez podwójne kliknięcie na pliku „mafes.jar”,
c) w systemie Microsoft Windows przez podwójne kliknięcie na pliku „run.bat”,
d) w systemie Linux/Unix, przez nadanie prawa wykonywania plikowi „run.bat” i jego uruchomienie.

Po uruchomieniu programu, powinno w systemie graficznym pokazać się okno, przedstawione na rys. D.1.



Rys. D.1. Okno główne programu MAFES-2, wygląd początkowy (po uruchomieniu)

W interfejsie użytkownika tego programu dostępne są następujące operacje:

a) wczytanie modelu wejściowego z pliku MMD (dodatek B.1) – przycisk „Open MMD file”, menu „File – Open”,
b) wstrzymanie (wznowienie) obliczeń – przycisk „Pause”,
c) rozpoczęcie obliczeń – przycisk „Calculate”,
d) otwarcie okna pokazującego podgląd pola wielkości obliczanej w symulacji (jednej spośród T, FS, N, R) dla wybranego przekroju (XY, YZ, ZX) – przycisk „Show Picture”,
e) otwarcie okien z charakterystykami czasowymi obliczanych wielkości dla punktów raportowanych zdefiniowanych pliku MMD – przycisk „Show Plot”.

Operacje c), d), e) dostępne są po pomyślnym wykonaniu operacji a). Okien obserwacyjnych – operacje d) i e) można utworzyć kilka, przed rozpoczęciem obliczeń lub przy ich wstrzymaniu, uaktualniane są one co zadaną w pliku MMD ilość kroków obliczeniowych (dodatek B.1).

Przygotowanie modelu

Konfigurację przeprowadzanej symulacji w programie MAFES-2 należy przygotować w pliku tekstowym w formacie MMD opisanym w dodatku B.1.

W ramach opisu modelu wejściowego dla symulacji należy określić: geometrie obiektów, materiały, warunki początkowe i brzegowe, gęstość siatki oraz warunki realizacji obliczeń (krok czasowy, czas symulacji, zapisywane wyniki).

Dwa przykładowe pliki „ruch.mmd" i „schodki.mmd" zostały dołączone do wersji binarnej programu.

Realizacja obliczeń

Po przygotowaniu pliku MMD i uruchomieniu programu MAFES-2, należy wczytać ten plik poprzez kliknięcie przycisku „Open MMD file". W oknie dialogowym wyboru pliku „Open" trzeba wskazać właściwy katalog, a w nim dany plik.

Przy źle zdefiniowanym pliku MMD, błędy pojawiające się podczas jego wczytywaniu są generowane na standardowe wyjście błędów. Dlatego, w celu zaobserwowania tych błędów należy program uruchomić w terminalu (nie poprzez podwójne kliknięcie na pliku „mafes.jar").

Przy pomyślnej operacji wczytania pliku, raport to potwierdzający zostanie wypisany w polu tekstowym w dolnej części okna. Zostaną wówczas również uaktywnione przyciski „Calculate", „Show Picture", „Show Plot".

W przypadku, gdyby krok czasowy nie spełniał warunków stabilności obliczeń, zostanie on dostosowany do odpowiedniej wartości, informacja o tym pojawi się we wspomnianym wyżej raporcie.

Przed rozpoczęciem obliczeń i przy ich wstrzymaniu można otworzyć okna wizualizacyjne opisane w dalszej części dodatku.

Rozpoczęcie obliczeń następuje poprzez kliknięcie na przycisku „Calculate". Bieżący czas, dla którego dane są prezentowane w oknach wizualizacyjnych jest pokazywany w oknie programu.

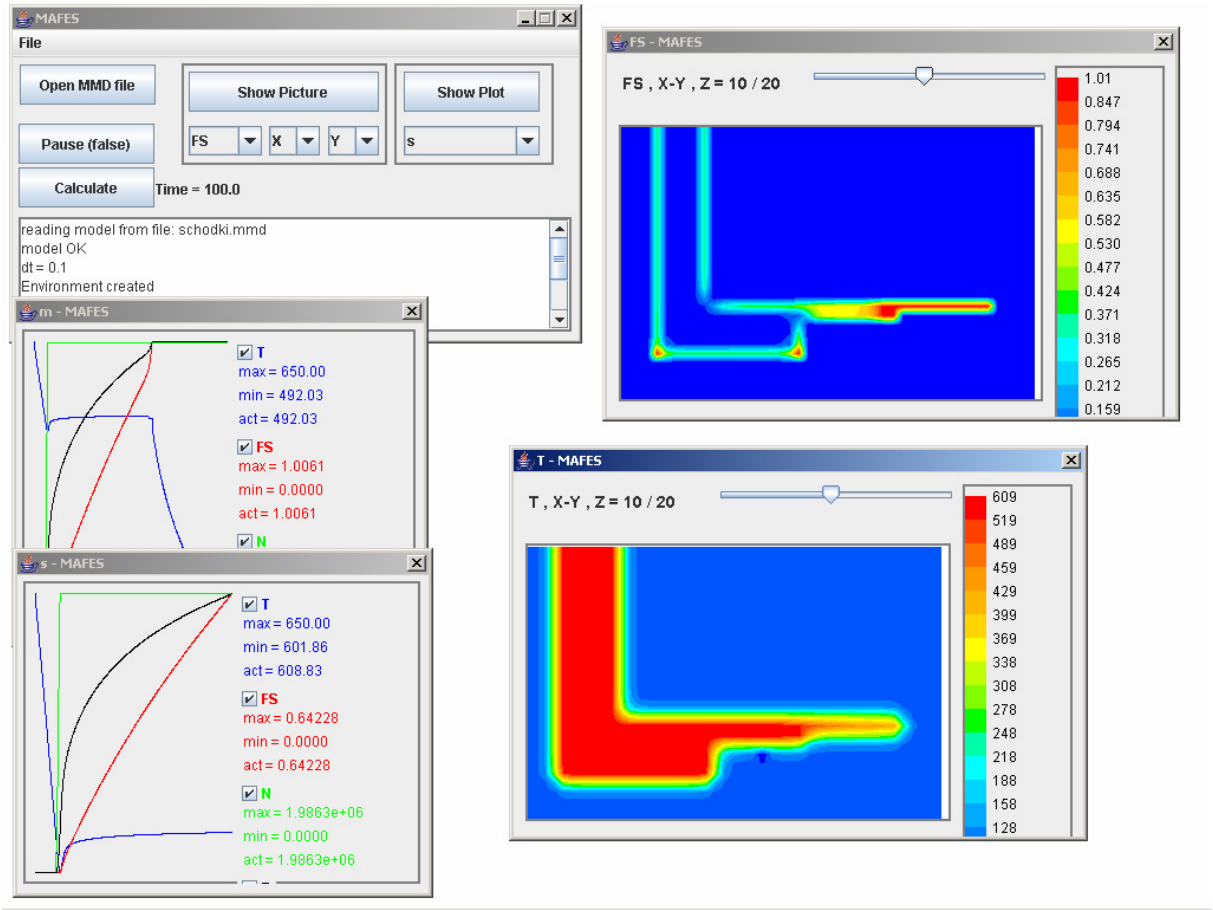
W trakcie obliczeń możliwe jest wstrzymanie obliczeń. W tym celu należy kliknąć na przycisku „Pause". Ponowne kliknięcie na tym przycisku wznawia obliczenia.

Obliczenia kończą się w chwili osiągnięcia całkowitego czasu symulacji, zdefiniowanego w pliku MMD. Aby zwolnić zasoby systemowe (pamięć operacyjną) i zakończyć działanie programu należy zamknąć okno programu, co kończy działanie całej aplikacji.

Podczas obliczeń generowane są dane z punktów raportowania do pliku „rp.txt". Plik ten tworzony jest w katalogu, w którym został uruchomiony program. Dane te mogą być wykorzystane w innych programach (np. wizualizacyjnych).

Prezentacja wyników

Na rys. D.2 zaprezentowano wygląd ekranu dla uruchomionego programu MAFES-2. Poza oknem głównym widoczne są cztery dodatkowe okna wizualizacyjne. W lewej, dolnej części rysunku pokazano dwa okna z charakterystykami czasowymi, natomiast z prawej strony otwarte są dwa okna z przekrojami. Górne okno przedstawia rozkład pola współczynnika FS, natomiast dolne temperatury. Prezentowane wyniki dotyczą czasu symulacji równego 100 sekund, dla odlewu schodkowego (plik „schodki.mmd").

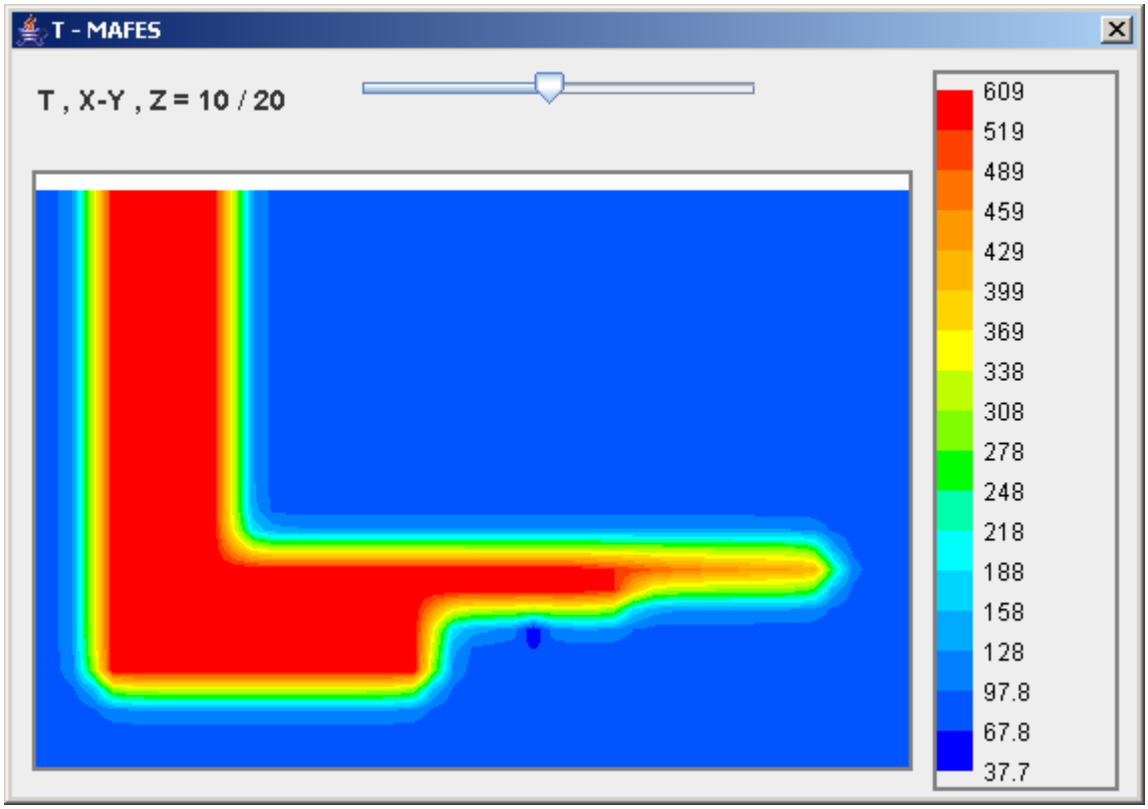


Rys. D.2. Przykładowy wygląd ekranu z oknem programu MAFES-2
i czterema dodatkowymi oknami wizualizacyjnymi

Okna wizualizacyjne w omawianym programie są aktualizowane w trakcie obliczeń co zadaną w pliku MMD ilość kroków.

Prezentacja wyników w symulatorze MAFES-2 pozwala na dwa typy obserwacji. Pierwszy typ ilustruje dynamikę symulowanego procesu poprzez prezentowanie rozkładu przestrzennego wybranej wielkości (T – temperatura, FS – udział fazy stałej, N- ilość

kryształów, R – średni promień kryształów). Okna tego typu otwierane są przez kliknięcie na przycisku „Show Picture”, po wcześniejszym wybraniu żądanej wielkości i współrzędnych przekroju z list wyboru umiejscowionych poniżej tego przycisku. Przykładowe okno omawianego typu pokazano na rys. D.3.

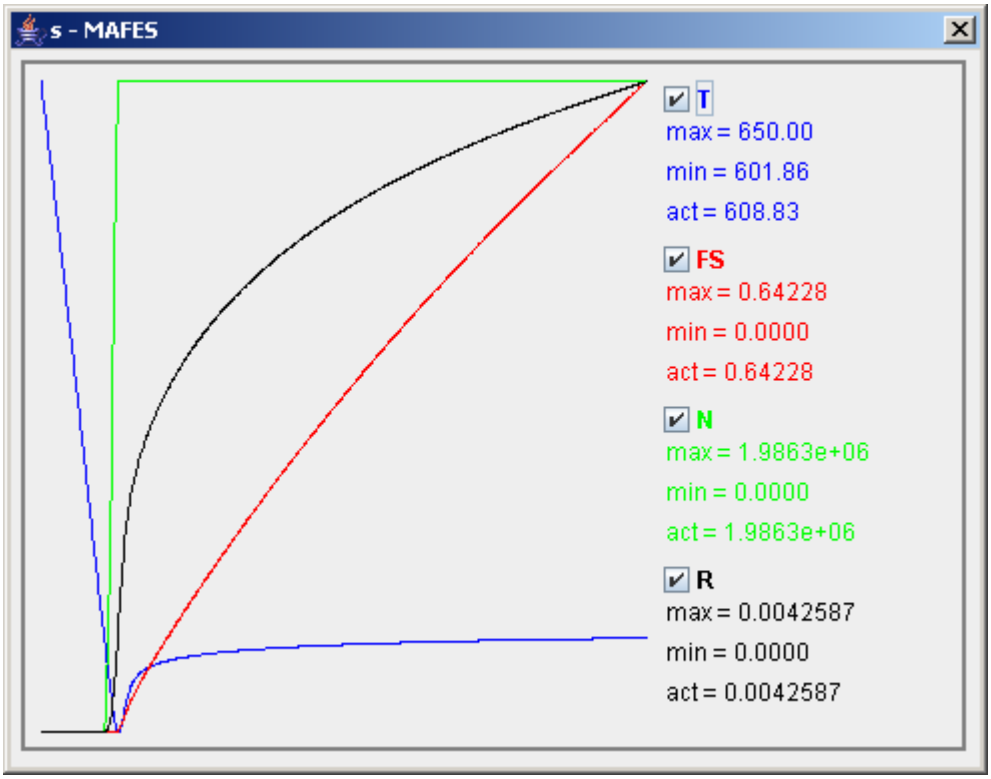


Rys. D.3. Okno wizualizacyjne z przestrzennym rozkładem pola temperatury (T), dla przekroju X-Y, warstwa nr 10 (wszystkich 20) węzłów w kierunku Z

W oknie tego typu można zmieniać warstwę w trzeciej współrzędnej przez przesuwanie suwaka. Kolorem niebieskim oznaczona jest najmniejsza wartość prezentowanej wielkości, natomiast czerwonym wartość największa. Skala wartości pośrednich i odpowiadającym im kolorów umiejscowiona jest w prawej części okna. Wartość maksymalna i minimalna skali oraz wartości pośrednie dostosowują się do wartości aktualnych w symulacji i mogą ulegać zmianom w trakcie obliczeń.

Drugi typ obserwacji pozwala na prezentację charakterystyk czasowych w oknach otwieranych przyciskiem „Show Plot”, po wcześniejszym wybraniu punktu raportowania z listy umiejscowionej pod tym przyciskiem. Na wspomnianej liście dostępne są punkty raportowania zdefiniowane w pliku MMD (dodatek B.1). Przykładowe okno tego typu pokazano na rys. D.4.

Wykresy tego typu rozszerzane są w pionie i poziomie od wartości minimalnych do maksymalnych. Wartości maksymalna, minimalna i aktualna, dla każdej prezentowanej wielkości są opisywane w prawej części okna. Wartość maksymalnego a zarazem bieżącego czasu symulacji dla tych charakterystyk jest zgodna z wartością widoczną w oknie głównym programu.



Rys. D.4. Okno wizualizacyjne z przestrzennym rozkładem pola temperatury (T), dla przekroju X-Y, warstwa nr 10 (wszystkich 20) węzłów w kierunku Z

Możliwości wykorzystania wyników w innych programach

W celu dokonania dokładniejszej analizy wyników symulacji wprowadzono generację pliku wynikowego o nazwie „rp.txt", który jest tworzony w katalogu, w którym został uruchomiony program. Wybór danych do zapisu w tym pliku dokonuje się na podstawie informacji o punktach raportowania zawartych w pliku MMD. Zapisywane są wszystkie wielkości dla wszystkich zdefiniowanych punktów raportowania.

Każdy wiersz zapisu rozpoczyna się od informacji o czasie kolejnego kroku, natomiast w kolejnych kolumnach, oddzielonych znakiem tabulacji są przedstawione poszczególne wartości (kolejnych wielkości, kolejnych punktów). Dzięki takiemu formatowi, dane z pliku wynikowego można w prosty sposób importować w programach arkuszy kalkulacyjnych (np. Microsoft Excel) celem tworzenia specyficznych charakterystyk.