Iterative Software Agent Based Solution of Multiphysics Problems

Matthias Jüttner, André Buchau, Desirée Vögeli, Wolfgang M. Rucker, and Peter Göhner

Abstract A novel approach is presented using software agents for an iterative and distributed solution of multiphysics problems. Overall convergence is achieved by using the individual capabilities of interworking agents. Every agent solves a partial single physics problem based on specialized, commercial or in-house code. The autonomy of each agent allows a physics adapted solution process without the need of a predefined solver sequence. The applied software agents are described in detail. Here, we focus on weak uni- and bidirectional field coupled multiphysics problems. This framework can also be used for node or boundary coupling as well as for optimising partial physics simulation. A coupled 3D electromagnetic wave propagation and heat transfer problem inside a waveguide is examined as numerical example.

1 Introduction

Methods for simulating single physics problems on high-performance computers were state of the art for many years. During the last years, tools were extended to cluster, cloud and graphical processing unit (GPU) computing to achieve further parallelism [1]. Recent developments combine different single physics implementations to a multiphysics framework by considering them as black boxes [2]. Improvements on software maintenance and functionality were achieved on costs of performance and memory usage [3]. For a practical usage, expert knowledge is needed in the fields of physics, their coupling and the numerical solution. However, engineers as users are experts within one or maybe a few physics. Therefore,

---

M. Jüttner (✉) • A. Buchau • W.M. Rucker
University of Stuttgart, Institute for Theory of Electrical Engineering, Pfaffenwaldring 47, 70569 Stuttgart, Germany
e-mail: matthias.juettner@ite.uni-stuttgart.de; andre.buchau@ite.uni-stuttgart.de; rucker@ite.uni-stuttgart.de

D. Vögeli • P. Göhner
University of Stuttgart, Institute of Industrial Automation and Software Engineering, Pfaffenwaldring 47, 70569 Stuttgart, Germany
e-mail: desiree.voegeli@ias.uni-stuttgart.de; peter.goehner@ias.uni-stuttgart.de

A. Bartel et al. (eds.), *Scientific Computing in Electrical Engineering*, Mathematics in Industry 23, DOI 10.1007/978-3-319-30399-4_13

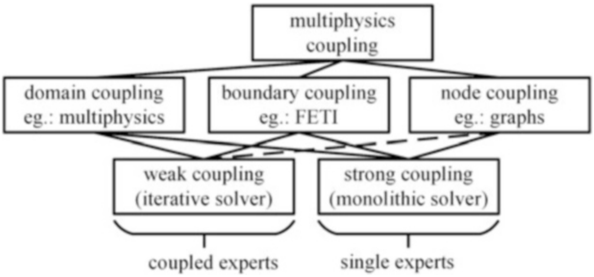

Fig. 1 Different methods of multiphysics coupling shown for two physics. The segregation of a monolithic multiphysics problem also represents a common way for parallelization. A central unit combines the partial results for an overall solution

an initial partitioning of a multiphysics problem (as in Fig. 1) seems odd at the beginning.

In practice, models are step-wise extended to consider multiphysics effects. This step-wise development starts from multiple independent physical models and uses shared variables to couple independent models to a multiphysics system. For solving several multiphysics problems, a monolithic as well as a segregated approach lead in practice to a valid solution [4]. For parallelizing the monolithic approach, the problem must be partitioned, while the segregated approach is natively parallel. Only connections of former independent problems lead to sequential dependencies.

Here, the work flow of distributed interacting single physics experts is projected into a multiphysics simulation environment. This system handles different physics with new encapsulated software agents and automatically coupling the physics. The agents autonomously interact with each other and share collective values. With this, a 3D coupled electromagnetic wave propagation and heat transfer problem inside a waveguide is solved exemplary. The hereinafter presented framework also promotes a physics based parallel calculation. In Sect. 2 an overview about software agents and their design is given. An explanation how that system is used for solving multiphysics problems is given in Sect. 3. In Sect. 4, the solver systems capabilities are demonstrated by a numerical example. A conclusion is given in Sect. 5.

2 Software Agent System

Software agents are encapsulated (software) entities with individual goals [5]. They are well tested in automation technologies for solving complex and distributed problems. A software agent tries to reach its goal by acting autonomously. It interacts with other agents of the system and its environment, while keeping a persistent state. The following list presents the main concepts of agents.

- Encapsulation: An agent encapsulates information. It has a certain knowledge of its environment and of its own capabilities.
- Persistence: An agent has its own control flow and keeps its internal state during lifetime. It is independent of an external activation.
- Autonomy: An agent is able to act autonomously and make decisions by itself.

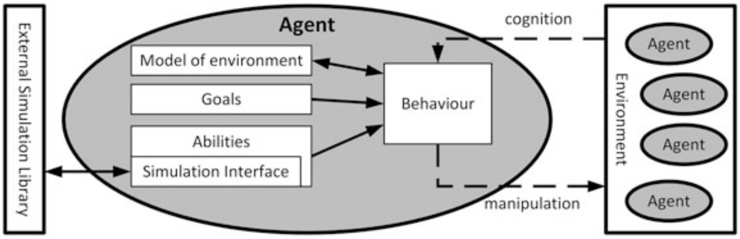

Fig. 2 Internal structure of a software agent. This allows the agent to act autonomously based on its abilities, and interact with other agents and its environment to solve complex tasks in a very flexible way

- Interaction: An agent can interact with other agents of the system. By doing this, agents are able to combine their knowledge and collaborate.
- Activity: An agent reacts to changes in its environment and can evoke changes.
- Goal-oriented: An agent has own goals that may change during lifetime. It is able to plan and execute activities by itself and react to situations by changing its plan.

If several agents work together, the system is called multi-agent system (MAS). Its setup can change during runtime. The internal structure [6] is shown in Fig. 2.

In the following, software agents are used as physics experts. They couple single physics simulations to a multiphysics problem. An interface to an external simulation library enables the agent to manipulate the model, couple it with other physics and control and supervise the attached solver within the simulation library [7]. An early attempt for 2D boundary coupled systems is given in [8]. Here, the presented work handles weakly coupled systems with different experts. Problems solvable with monolithic solvers only, are handled by a single expert (see Fig. 1). For establishing a coupling between the agents, the agents share information about coupling and calculation capabilities. This description provides information about calculation resources, numerical methods, solvable equations, possible boundary conditions, provided results, and derived values as a list. Implementing the agents was done using corresponding design rules [5]. The programming language must handle the complexity of agents' communication, provide the agents itself, manage the attached simulation interface, and handle exchanged numerical data in a powerful and parallel way. To use state of the art software development techniques, Java was chosen [9] together with the Java Agent DEvelopment framework (JADE) [10].

3 Solver System

For practical reasons, two types of software agents are required. A coordination agent (CO) splits the XML-file based multiphysics problem, created with nowadays computer aided design (CAD) tools into multiple single physics problems. Multiple calculation agents (CA) cooperatively solve the coupled sub-problems. For the finite element method (FEM), the problem is given as

$$\mathbf{K}\mathbf{u} = \mathbf{b}, \tag{1}$$

$\mathbf{K}$ represents the stiffness matrix, $\mathbf{u}$ the solution and $\mathbf{b}$ the load. For a multiphysics problem $\mathbf{K}$ is usually not symmetric due to different influences between the physics. For a problem with two physics, u can be grouped and the problem reformulated as

$$(C \circ K)\,u = \begin{bmatrix} \mathbf{C}_{11}\mathbf{K}_1 & \mathbf{C}_{12}\mathbf{K}_{12} \\ \mathbf{C}_{21}\mathbf{K}_{21} & \mathbf{C}_{22}\mathbf{K}_2 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}. \tag{2}$$

$\circ$ representing the Hadamard product and $\mathbf{C}$ an activation matrix for the coupling. An uncoupled problem has a $\mathbf{C}$ equal to the identity matrix $\mathbf{I}$. For a fully coupled system all non-diagonal matrices (e.g. $\mathbf{C}_{12}, \mathbf{C}_{21}$) become $\mathbf{I}$. In a loop wise coupled system, the main and upper diagonal matrices become $\mathbf{I}$, including the element of the first column and last row. For more than two physics, this fits best for an iterative sequential solution. If $\mathbf{K}$ includes further couplings (eg. $\mathbf{K}_{24}$), a parallelization is possible and automatically applied with this approach. Initially, no coupling is considered $\mathbf{C}_{12} = \mathbf{C}_{21} = 0$ and two CAs are used for this problem.

$$\begin{matrix} \textit{Agent 1} \\ \textit{Agent 2} \end{matrix} \quad \text{solves} \quad \begin{bmatrix} \mathbf{u}_1^0 \\ \mathbf{u}_2^0 \end{bmatrix} = \begin{bmatrix} \mathbf{K}_1^{-1}\mathbf{b}_1^0 \\ \mathbf{K}_2^{-1}\mathbf{b}_2^0 \end{bmatrix} \tag{3}$$

in parallel. Each agent uses its own backbox simulation environment for its partial problem. Tests with different environments or solvers can be performed simultaneously by additional agents. The fastest agent for a partial problem survives. The fastest agent for a partial problem currently survives. As soon as any agent finished its calculation (e.g. *agent 1*), all agents get informed about an available result and derived values. Conditions are a first time calculated result or changes in the result $\mathbf{u}_1$ compared to a previous calculation cycle $\mathbf{u}_1^*$. Based on its own features list, each agent decides whether to couple or to ignore and continue calculating. In case of coupling material dependent parameters, $\mathbf{K}_2$ is reassembled. If new sources gets available, the coupling matrix $\mathbf{C}_{21}$ changes to $\mathbf{I}$. The new problem

$$\begin{bmatrix} \mathbf{u}_1^1 \\ \mathbf{u}_2^1 \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1^0 \\ \mathbf{K}_2^{-1}\underbrace{(\mathbf{b}_2 - \mathbf{K}_{21}\mathbf{u}_1^0)}_{\mathbf{b}_2^1} \end{bmatrix} \tag{4}$$

is solved, while calculated intermediate results are used as initial values for further calculations. Equation (4) can be seen as a first iterative step solving Eq. (2) using Jacobi method. The new $\mathbf{b}_2^1$ handles non-linear coupling between the physics. The strength of coupling changes during an iterative process [11]. Stabilising the system should be possible with relaxation methods like Aitken Δ^2 or gradient based ones [12]. Obviously, at least one partial problem must converge during the iterative solution process. The iterative method ends, if the relative changes for $\mathbf{u}_i$ or derived values are below a limit ε_i, i representing the agent number. In Fig. 3 the unidirectional result propagation implementation for two CAs is shown. If more than one expert with the same knowledge works on a problem, methods like the Finite Element Tearing and Interconnection (FETI) domain decomposition approaches allows to engage the agents [13]. As more agents dealing with a problem, as further the parallelisation will be, limited by the communication overhead that is not considered here. Solver selection algorithms [14] as well as learning algorithms are imaginable. Adapted meshes for the different physics have been already tested [7]. Another application of this approach comes together with co-simulation and different time-steps [15] of the agents. In all cases, the individuality of the agents allows to optimize the process.

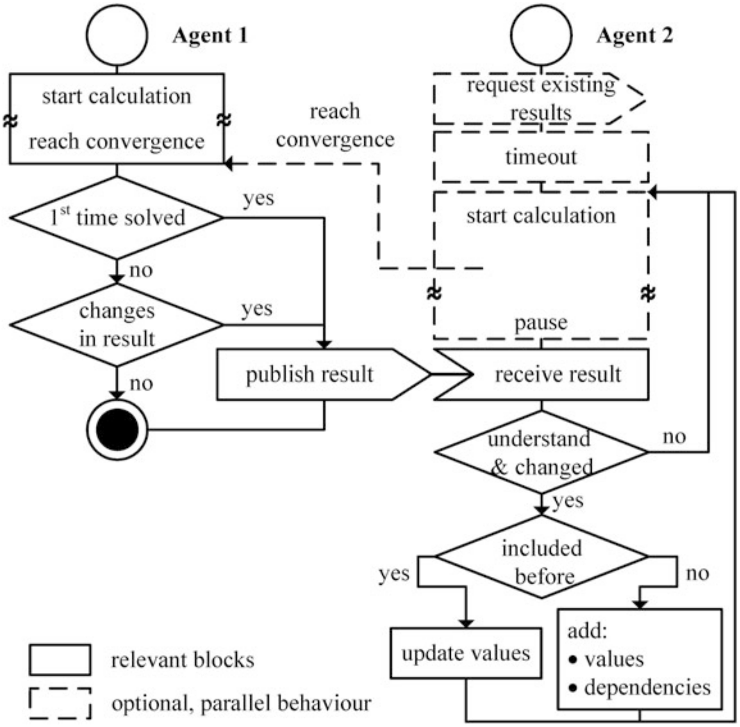

Fig. 3 Unidirectional result propagation process for two agents. *Agent 1* starts computing Eq. (3). *Agent 2* notices another agent working on the same problem and asks for existing results. If no results are provided, *agent 2* starts computing Eq. (3) in parallel. *Agent 1* finishes its calculation first and publishes the results to *agent 2*. This pauses its iterative solver to integrate the offered results, if it's possible. Afterwards, the calculation is continued until *agent 2* is ready to publish its results

4 Numerical Example

The solution process of a coupled electromagnetic wave propagation problem and a heat transfer problem is shown for a lossy dielectric within a waveguide surrounded by air. It demonstrates the principle of the iterative agent based solution of multiphysics problems. Here, three agents are needed. *Agent 0* represents a CO, *agent 1* and *agent 2* CAs. In Fig. 4 the MAS setup is shown.

The agents run on an Intel(R) Core(TM) i7-2600 CPU with 4 cores, max. 3.4 GHz, 16 GB (1333 MHz) RAM and Microsoft Windows 8.1 Enterprise 64-bit. *Agent 1* handles the electromagnetic wave problem according to

$$\varDelta \mathbf{E} + \mu_r k_0^2 (\varepsilon_r - \frac{\mathrm{j}\sigma}{\omega \varepsilon_0}) \mathbf{E} = 0. \tag{5}$$

Here μ_r is the relative permeability, k_0 the wave number of free space, ε_r the relative permittivity, σ the electrical conductivity, ω the angular frequency, and ε_0 the free space permittivity. Eq. (5) is solved in the frequency domain within the waveguide. All over the model, the thermal problem is considered. It is defined by

$$\kappa \varDelta T + Q = 0. \tag{6}$$

and solved by *agent 2* for a stationary case. κ represents the thermal conductivity and Q is a heat source. According to the FEM approach, the electric field strength $\mathbf{E}$ and the temperature T are the dependent variables $\mathbf{u}_1$ and $\mathbf{u}_2$ in Eq. 2. A convective

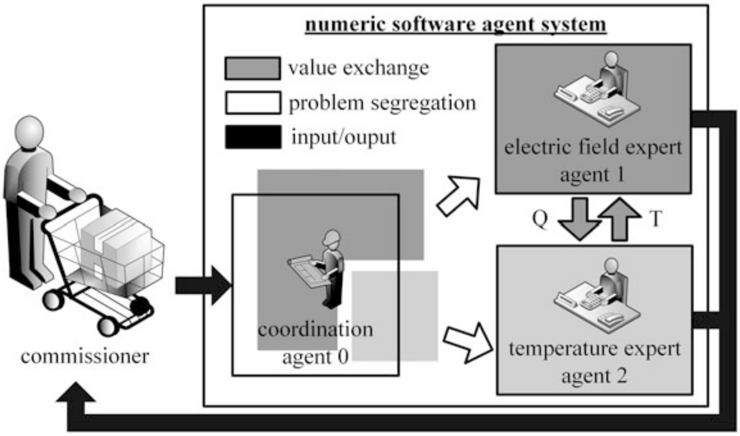

Fig. 4 Setup of the MAS for a coupled two physics problem. The commissioner hands over the multiphysics problem and receives the simulation results. The coordination agent distributes the problem and the calculation agents solve parts of the problem, they are versed to do. Exchanging value allows a coupled iterative solution

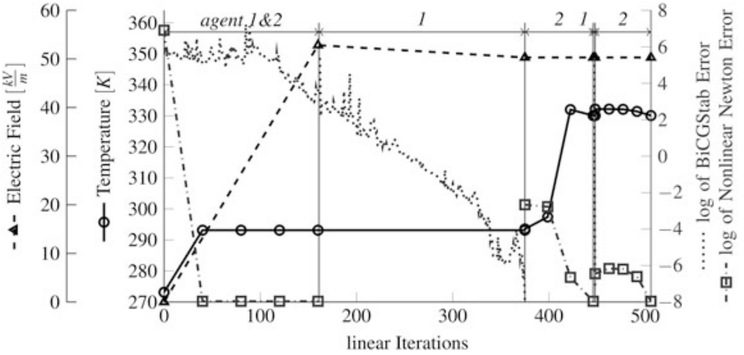

Fig. 5 Solver sequence for the coupled problem including the dependent variables (*left*), the problem dependent errors (*right*), the global iteration counter (*bottom*) and the active agents (*top*)

heat flux with the heat flux coefficient h at the boundaries given as

$$\mathbf{n} \cdot \kappa \nabla T = h(T_{ext} - T) \tag{7}$$

makes a stationary simulation possible. Coupling is dynamically established by a heat source Q representing the total power dissipation density in *agent 2* and the temperature dependent electric conductivity $\sigma(T)$ in *agent 1*. The slow heating process (within seconds) compared to the high frequency wave propagation (10 GHz) allows to consider the heat source Q as constant over time. The numerical solver is chosen from *agent 1* to be BICGStab and from *agent 2* to be a non-linear Newton method combined with a FGMRES. *Agent 0* segregates the multiphysics problem into two single physics problems and distributes them to *agent 1* and *agent 2*. After receiving the problems, *agent 1* and *agent 2* start computing in parallel (Fig. 5).

Values between the marked points for temperature T and the electric field $\mathbf{E}$ are linear interpolated. Here, Eq. (6) is successfully solved first. Due to the temperature dependent electric conductivity $\sigma(T)$ at *agent 1*, results of *agent 2* have to be considered in *agent 1*. Once a solution for *agent 1* is found, *agent 2* is informed about the results. Now, the total power dissipation density of the electromagnetic wave is available and can used as heat source Q in Eq. (6). The bidirectional coupling leads to a loop. Table 1 shows the maximum node wise difference of the exchanged values compared to the previous values. Due to the small changes ε_2 for the temperature, the loop ends. Additionally, a comparison between the agent based solver system and a segregated solver for a given iterative sequence is given. Identical meshes and a BiCGStab solver for both agents are used. The error is computed as maximum node wise difference of the solution vectors.

306 linear iterations were necessary to solve the electric field problem in a purely sequential process. A computation time advantage of the agent based solver is gained by solving the initially uncoupled problems in parallel. The computation of *agent 1* is interrupted when the results of *agent 2* get available (see Fig. 5). Here

Table 1 Solver sequence for the waveguide

Agent	Variable	Integrate	Max. difference	Lin. iterations	Max. error	Rel. error %
2	T	None	First	36	5×10^{-14}	2×10^{-14}
1	$\mathbf{E}$	New Source(T)	First	306	5×10^{-4}	6×10^{-4}
2	T	New Source(Q)	25 K	77	0.71	0.21
1	$\mathbf{E}$	Update(T)	$7.64\,\frac{\mathrm{W}}{\mathrm{m}^3}$	1	5×10^{-4}	7×10^{-4}
2	T	Update(Q)	6×10^{-6} K	43	0.72	0.22

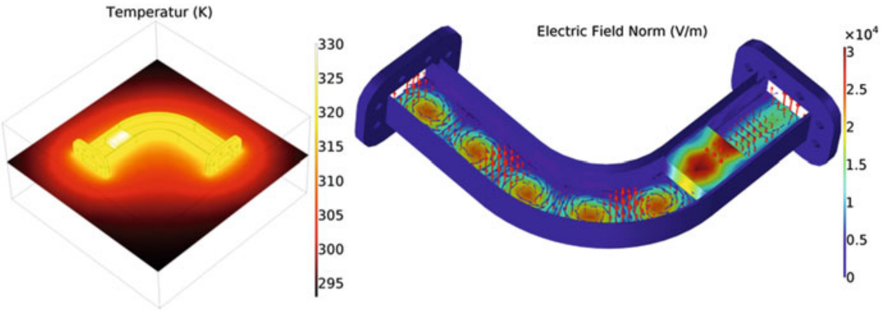

Fig. 6 Visualisation of results from *agent 2* and *agent 1*

agent 1 was interrupted after 160 linear iterations and only 212 additional iterations were needed to solve Eq. (4) after integrating results of *agent 2*. This shows, that iterations are spared, if partial results with final values are integrated before finishing the calculation, and more than two agents are working at a problem. The results of the solved waveguide problem for a mode 10 transverse electromagnetic wave (TE10) at 10 GHz and a convective heat flux at the boundaries of $1\,\frac{\mathrm{W}}{\mathrm{m}^2 \cdot \mathrm{K}}$ are shown in Fig. 6.

5 Conclusion

The step-wise development of multiphysics problems enables a parallelized way of solving coupled multiphysics problems. Based on the idea of interworking experts, several requirements were discussed for implementing this software system. Motivated by the affinity of multi-agent systems to the expert system, an algorithm for uni- and bidirectional coupling was presented. Details about their implementations as well as advantages of the system were given. The solution of a practical example finally demonstrates the performance of the presented expert system. Engaging more agents to further parallelize and optimize the solution process is a future task. Same holds for the selection mechanism of the numerical solver used in each agent. Using the system to solve strongly coupled problems with attached weakly coupled physics is now possible.

References

1. Griebel, M., Zumbusch, G.: Parallel multigrid in an adaptive PDE solver based on hashing and space-filling curves. Parallel Comput. **25**(7), 827–843 (1999)
2. Uekermann, B., Bungartz, H., Gatzhammer, B., Mehl, M.: A parallel, black-box coupling algorithm for fluid–structure interaction, pp. 1–12. CIMNE (2013)
3. Drashansky, T.T., Joshi, A., Rice, J.R., Houstis E.N., Weerawarana, S.: A multiagent environment for MPSEs. In: Conference on Parallel Processing for Scientific Computing (1997)
4. Heil, M., Hazel, A.L., Boyle, J.: Solvers for large-displacement fluid–structure interaction problems: segregated versus monolithic approaches. Comput. Mech. **43**, 91–101 (2008)
5. Jennings, N.R.: Agent-oriented software engineering. in Multiple Approaches to Intelligent Systems, pp. 4–10. Springer, Berlin/Heidelberg (1999).
6. Maurmaier, M., Wagner, T., Development of Embedded Software Systems with Structured Components and Active Composition Support. In: OMER 3 (2005)
7. Jüttner, M., Buchau, A., Rauscher, M. , Rucker, W.M., Göhner, P.: Iterative solution of multiphysics problems using software agents designed as physics experts. In: ISTET'13 (2013)
8. Drashansky, T.T., Joshi, A., Rice, J. R.: SciAgents-an agent based environment for distributed, cooperative scientific computing. In: ICTAI'95, pp. 452–459 (1995)
9. Villacis, J.: A note on the use of Java in scientific computing. ACM SIGAPP Appl. Comput. Rev. **7**(1), 14–17 (1999)
10. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley , Chichester (2007)
11. Markert, B.: Weak or strong: on coupled problems in continuum mechanics. Habilitation, Universität Stuttgart (2010)
12. Küttler, U., Wall, W.A.: Fixed-point fluid–structure interaction solvers with dynamic relaxation. Comput. Mech. **43**(1), 61–72 (2008)
13. Mandel, J., Dohrmann, C.R., Tezaur, R.: An algebraic theory for primal and dual substructuring methods by constraints. Appl. Numer. Math. **54**(2), 167–193 (2005)
14. Ewald, R., Himmelspach, J., Uhrmacher, A.M.: An algorithm selection approach for simulation systems. 22nd Workshop on Principles of Advanced and Distributed Simulation (2008)
15. Schöps, S.: Multiscale modeling and multirate time-integration of field/circuit coupled problems. PhD thesis, Bergische UniversitätWuppertal & Katholieke Universiteit Leuven (2011)