We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

4,800

Open access books available

122,000

International authors and editors

135M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities

CLARIVATE ANALYTICS

BOOK CITATION INDEX

INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

A Framework Providing a Basis for Data Integration in Virtual Production

Rudolf Reinhard, Tobias Meisen, Daniel Schilberg, Sabina Jeschke
Institute of Information Management in Mechanical Engineering
RWTH Aachen University
Germany

1. Introduction

Complexity in modern production processes increases continuously. Therefore, the virtual planning of these processes simplifies their realisation extensively and decreases their implementation costs. So far, several institutions have implemented their own simulation tools, which differ in the simulated production technique and in the examined problem domain. On the one hand, there are specialized simulation tools available simulating a specific production technique with exactness close to the real object. On the other hand, there are simulations which comprise production processes as a whole. The latter do not achieve prediction accuracy comparable to the one of specialized tools.

However, both types are commonly applied in university research.

Furthermore most of the applied algorithms in these tools are not yet implemented in commercial tools. Hence, the simulation of a whole production process using these tools is often not realisable due to an insufficient prediction accuracy or the missing support of the asked production techniques. In solving the problem, it is necessary to interconnect different specialized simulation tools and to exchange their resulting data. However, the interconnection is often not achievable because of incompatible file formats, mark-up languages and models used to describe the simulated objects. Therefore, the simulation of a production process as a whole using different simulation tools is hard to realise because of the missing consistency of data and interfaces.

Therefore, results received within a simulation can only be integrated into another one after being checked manually and being adapted to the needs of following simulations, which is both tedious and fault-prone. On the one hand, the huge data volumes being characteristic for simulation processes are not supported by current solutions. On the other hand, the possibilities to adapt a simulation process as a consequence of changes (e.g. integration of a new application, modification of a simulated object) are poorly supported.

In this paper, the architecture of a framework for adaptive data integration is presented, which enables the interconnection of simulation tools of a specified domain. The framework provides generic functionality which, if customised to the needs for a specified domain (e.g. by transformation rules or data interfaces), supports the system to integrate any domain specific application in the process by making use of adaptive integration.

For this purpose, this chapter focus on the integration of data generated during the applications' usage, whereas the applications' link-up technique, which can be handled with

the help of modern middleware techniques, will not be stressed. The framework is getting developed within the project "Integrated Platform for Distributed Numerical Simulation", which is a part of the Cluster of Excellence "Integrative Production Technology for High-Wage Countries" at RWTH Aachen University.

2. State of the art

Since the eighties, but at least since the nineties, data integration as well as Enterprise Application Integration (EAI) belongs to the most frequented topics across application boundaries [cf. Halevy et al. (2006)]. Today, a multitude of data integration products can be found which are used in different fields of application. In general, the functionality of those products can be sub-divided into three categories [cf. White (2005)] (cf. figure 1):

- data propagation
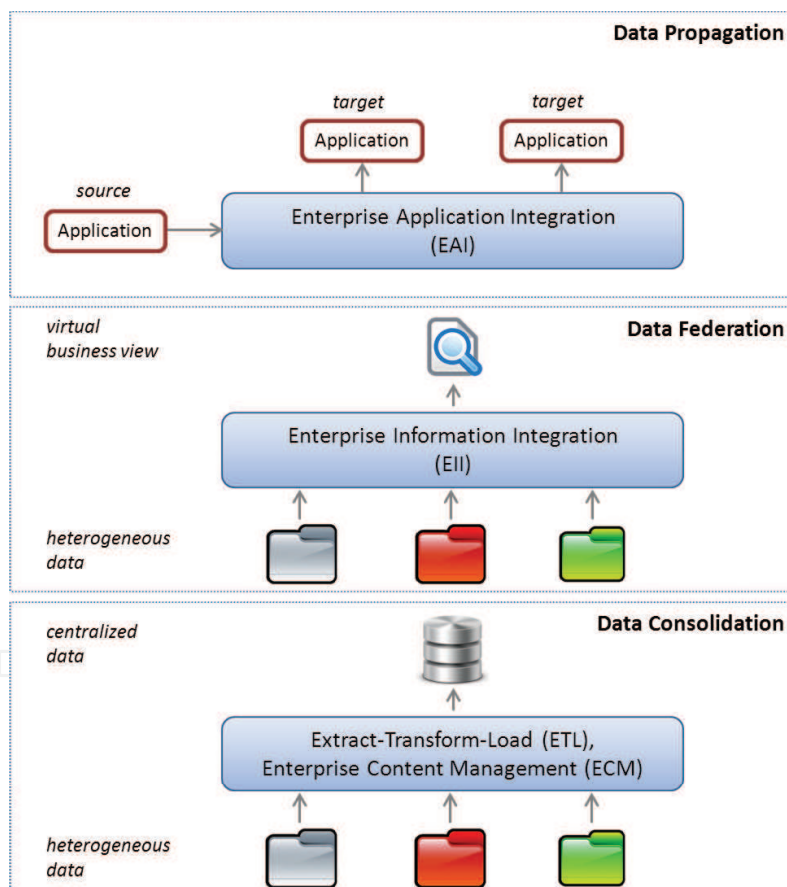- data federation
- data consolidation

Fig. 1. Main areas of data integration

With regard to the operational section, Data Propagation is applied in order to make use of data on a cross-application basis, which is often realised via data propagation. As already presented in [White (2005)], data propagation mainly focuses on small data volumes like messages and business transactions that are exchanged between different applications. In order to realize EAI, a contemporary architecture concept is used, which was developed in

connection with service-based approaches Chappell (2004) and which will be emphasized within this contribution - the so called Enterprise Service Bus (ESB). The basic idea of ESB, which can be compared to the usage of Integration Brokers, comprises the provision of services within a system [Schulte (2002)]. Each service provides a technical or technological functionality with the help of which business processes are supported. All services are connected with each other via the Integration Bus. Transformation services provide general functions in order to transfer data from one format and model into another one. Against that, routing services are used to submit data to other services. Both transformation and routing services are used by adaptors in order to transfer data provided by the Service Bus into the format and the model of an application. Consequently, transformation services support the reuse of implemented data transformations. The advantage of a solution based on the ESB pattern is to be seen in the loose interconnection of several services, whereas the missing physical data interconnection can be regarded as a disadvantage [cf. Rademakers et al. (2008)]: If recorded data has to be evaluated or to be analysed subsequently (e.g. with the help of data exploration techniques like OLAP or Data Mining), it will have to be read out and to be transformed once again. According to this fact, a historic or at least long-term oriented evaluation of data is inconvertible. In order to realize a unified examination on a cross-data basis, other sections belonging to the field of data integration need to be taken into consideration (cf. figure 1). Data Federation, which is examined within the field of Enterprise Information Integration (EII), might serve as one possible solution to enable a unified examination. With the aid of EII, data, which is stored in different data sources, can be unified in one single view [cf. White (2005) and Bernstein et al. (2008)]. This single view is employed by the user to query this virtual, unified data source. The query itself is processed by the EII system by interrogating the underlying, differing data sources. Because of the fact that most EII do not support advanced data consolidation techniques, the implementation will only be successful if the data of the different data sources can be unified and if access to this data is granted (e.g. via query interfaces). Otherwise, techniques belonging to the field of data consolidation, which comprises the integration of differing data into a common, unified data structure, need to be utilised. Extract-Transform-Load (ETL) - a current process with regard to data integration - can be seen as one example of data consolidation [Vassiliadis et al. (2002)]. ETL consists of the following aspects: The extraction of data from one or several - mostly operational - data sources, the transformation of the data format as well as of the data model into a final schema and, finally, the uploading of the final schema to the target data base. The presented sections of data integration (and not just those) have in common that, independent of the type of integration, the heterogeneity of data has to be overcome. In literature, different kinds of heterogeneity are distinguished [cf. Kim et al. (1991) and Goh (1991)]. In this chapter, the types of heterogeneity listed in Leser (2007) will be stressed:

- Technical heterogeneity

- Syntactic heterogeneity

- Data model heterogeneity

- Structural or schema heterogeneity

- Semantic heterogeneity

The problem of technical heterogeneity, which addresses the problem of accessing data, can be handled with the help of modern middleware techniques Myerson (2002). Syntactic heterogeneity, a problem arising as a result of the representation of data (e.g. number formats,

character encoding), is solved by converting the existing representation into the required one; in most cases, the conversion is carried out automatically. The handling of data model heterogeneity is more complex, as this kind of heterogeneity can be traced back to data using different data models (e.g. relational database, XML data model, structured text file). Nevertheless, modern data integration solutions provide readers and writers to access data from popular data models like relational databases or XML. Besides that, the support of other data models can be implemented. The combination of both structural and semantic heterogeneity is the most complex form of heterogeneity. Structural heterogeneity addresses the problem of representing data in one data model in different ways, for instance the usage of element attributes versus nested elements in a XML document. Semantic heterogeneity comprises differences in meaning, interpretation and in the type of usage of schema elements or data. Schema and ontology matching as well as mapping methods can be used to find alignments between data schemas as well as to process these alignments. Thereby, an alignment is a set of correspondences between entities of schemas that have to be matched. In the past years, several matching and mapping algorithms have been published [cf. Euzenat et al. (2007)]. However, these methods often focus on database schemas, XML schemas and ontologies without taking into account the background domain specific information [cf. Giunchiglia et al. (2006)]. This chapter will not take a closer look at the last point mentioned. The restriction to a kind of heterogeneity that is predictable via a set of simulation tools restricted beforehand implies a low flexibility that is provided by the corresponding architecture. The user may not employ the specialization of a single tool for a special purpose and is thus forced to disclaim qualified results in a special case.

3. Use case

During procedures like the manufacture of a line pipe, different production techniques are put to use. Within the use case, these techniques are simulated via specialised tools. The use case starts with a simulation of the annealing, the hot rolling as well as the controlled cooling of the components via CASTs, an application developed by Access e.V.. Within a further step, the cutting and the casting will be represented with the help of Abaqus (Dassault Systems), whereas the welding and the expanding of the line pipe will be simulated via SimWeld, a tool which was developed by the Welding and Joining Institute of RWTH Aachen University, and via SysWeld, a software product contrived by the ESI-Group [cf. Rossiter et al. (2007)]. Furthermore, the simulation of modifications in the micro structure of the assembly will be realized by making use of Micress [cf. Laschet et al. (1998)] and Homat [cf. Laschet (2002)], which were both developed by Access e.V.. All in all, the use case contains six different kinds of simulations, each of them based on different formats and models. Apart from that, the project "Integrated Platform for Distributed Numerical Simulation" comprises four additional use cases, on which the requirements directed to the framework are based. Nevertheless, these use cases will not be stressed in the following. The requirements aforementioned were first examined and described in [cf. Schilberg et al. (2009)] and more detailed in [cf. Schilberg (2010)]. Two requirements, which turned out to be central with reference to the framework presented in this paper, are the possibility of Data Propagation and the necessity of a process-oriented Data Consolidation (cf. figure 1). Both of them are used to facilitate a subsequent visualization and analysis of data collected within the process. Another important demand concerns the implementation of the following aspects without having to adapt the application significantly: the illustration of new simulation processes as well as the integration of new simulation tools.

4. The framework's architecture

The framework's architecture is based on the Enterprise Service Bus' (ESB) architectural concept and thus follows the requirements described in section 3. The architecture is illustrated in figure 2 (as described in Chappell (2004)). In order to realise a communication
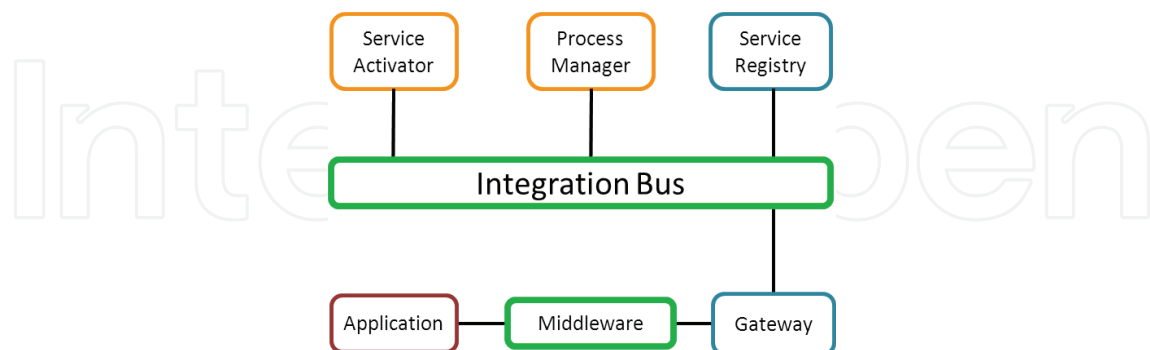

Fig. 2. The system-architecture of the framework

process between the integration server and the applications, a middleware is used that encapsulates the functionality of routing services, which are typical of those ones used in ESB concepts (e.g. within the use case mentioned in section 3, the application-oriented middleware Condor [cf. Thain et al. (2005)] is employed).

Since a service does not provide any capability to communicate over messages it needs a further instance to undertake this task. This instance is the Service Activator. Each Service has its own Service Activator, whereas a Service Activator might also handle several Services. The Service Activator listens to the Integration Bus with the intention to identify any messages containing queries, which could be executed by one of the services the Service Activator cares about. Beside the query itself, the message contains also information about the requirements that need to be fulfilled by the service. In the case there is a query matching the capability of one of the services entrusted to the Service Activator's care, it locks one of its services to process this message and marks it as "in work", so that there is no other service processing this query. The procession's result is getting packed into a message by the Service Activator and is sent to the specified reply queue.

Each process within a simulated production process is managed by the Process Manager. It writes messages containing queries into the Integration Bus' Queue, so that processes can be executed by a service and it cares about the process initiation and eradication. The Integration Bus consists in particular of a queue containing the different queries the Process Manager writes into. The messages are read by at least one Service Activator.

Hence, routing services are not considered in this framework because the integration of standard middleware is straight forward. The framework is employed with the intention of realising an integration level at which service providers, which are directly linked to the Integration Bus, make different services available. Due to the fact that the integration architecture needs to allow the easy substitution of one application by another one, the choice of a service-oriented architecture was helpful, to obtain an adaptable solution. In the following, there will be a concise explanation of the architecture's components.

The services considered in this architecture comprise the following tasks: integration, extraction (both of them act as translators), analysis, transformation and planning. The Integration Services care about the processing of data for the further employment by making use of a particular application. That's why the Service Integration interface needs an own

specialised implementation for each integration purpose. The Analysis Service checks the data that has been inserted into the database concerning their current structure as well as its semantics and the structure as well as its semantics requested by the next simulation tool within the simulated production process. Thereby it determines how the current data have to be transformed for the next step. To define the transformation steps needed to prepare the data, in a way that they can be processed by the next simulation tool within the simulated production process, the Analysis Service has to parse the message in order to know which processes are necessary to fulfil the requirements written into the message. Each implementation of the Transformation Service cares about exactly one special aspect in the existing data. This could be for example the indexing of nodes within a geometry. A necessary data model requirement for this purpose is the link between the node objects and "their" geometry object. There are applications starting the indexing for nodes with 0, whereas other applications start it with 1. Furthermore a random number might be determined during the creation of the geometry. By trying to interconnect two of those tools with each other the necessity is obvious to change the index of the existing data such that it can be understood by the next tool. Another example is the conversion of the temperature of a work piece from °C to °F.

In most cases, it is not sufficient to make more than one step to modify output data of one application such that they can be processed by the next application. An important constraint is the order in which these transformations have to be executed as a request exists to obtain a fully automated interconnection of applications on the one hand and the determining of the kind of transformations and their execution order on the other hand. At this point, Planning Services come into consideration.

They determine the kind of Services needed to perform the required operations and how these services have to interact. After their preparation by the appropriate, transformed data they get extracted by an Extraction Service. The Extraction Service cares about the extraction of data, which got recently processed by an application and is meant to get used by another one. In turn, the simulation results are stored within a file with a particular format. In certain cases it might be necessary to modify the input data. This step is called Enrichment.

Since the communication between all components is message driven the question arises, how to activate the adequate service for a certain task. The Process Manager controls the realisation of the current step by an appropriate service instance within a running integration or extraction process. It does not know which functionality can be provided by any service, not about the data a service needs to run. Thus there is the need for an instance having exactly this knowledge. This instance is called Service Registry. It contains information about available services, the functionality each service provides and which input data is required by each service to run properly.

A Gateway always belongs to a single application, which does not possess any capability to communicate with other architecture components over the Message Bus. The Gateway provides access for the architecture components to the application it belongs to and vice versa [Hohpe et al. (2004)]. The described components, in particular the service oriented architecture allow to implement the concept of data integration in an adaptive way. This point will be considered in the following section.

5. Adaptive data integration

The main goal of the adaptive data integration is to overcome the problems of structural and semantic heterogeneity. The adaptive data integration is part of the enrichment process

step (cf. section 4), which can be assigned to the extended ETL process being used during the extraction of data. The objective of the extraction process consists in the generation of data in a given data format, taking into account the data model and structure as well as the semantics of this format. Therefore, the implemented enrichment allows the discovery and exploitation of background-specific information. The concept is based upon ontologies and planning algorithms that are usually applied in artificial intelligence. The underlying enrichment process is depicted in figure 3 . In the first instance, the existing data is analysed. The goal of the analysis is the determination of so-called features that are fulfilled by the data. A feature is domain specific, which means that it is expressing a structural or semantic property of the domain. Besides, the analysis step determines features that have to be fulfilled by the data to satisfy the requirements of the specific output format of the extraction process. Subsequent to the analysis, planning algorithms are used to find a data translation that transforms and enriches data in a way that allows for the fulfilment of features needed by the output format. After the planning is finished, the data translation, which is part of the executed step, is processed. The domain-specific data transformation algorithms are stored in transformation services following the ESB architectural concept, whereas the information about existing transformations and features is stored within an ontology. According to Gruber (1993), an ontology is an explicit specification of a conceptualization. In this chapter, the ontology-driven data integration will not be focused due to the limited space, which will not suffice to describe it in a proper way.
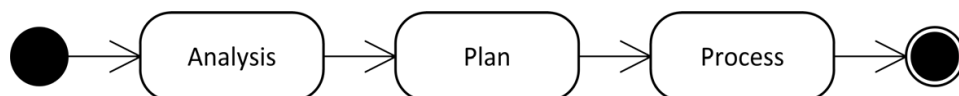

Fig. 3. The Enrichment Process

6. Application of the framework in the use case

Within the domain of the use case described in section 3 and the requirements resulting from the examination of four additional use cases in the domain of FE-simulations, an application has been implemented in parallel to the realisation of the framework. The regarded applications are simulations that use the finite-element-method [cf. Zienkiewicz et al. (2005)]. With regard to the implementation of an application, which is based upon the framework, a domain specific data schema, adaptor services for the integration and extraction process, the transformation service, the data model and the domain ontology have to be provided. An extract of the implementation is presented in figure 4. The domain-specific data schema has been determined by analysing the different input and output formats of the simulations that were employed in the use case. Within this data schema, the geometry of the assembly can be regarded as the central entity. It consists of nodes, cells and attributes. The latter ones exhibit attribute values, which are assigned to individual cells or nodes depending on the class of attributes available in the whole geometry. The integration services, which were specified within the use case, read the geometrical data provided by the simulation, transform it into the central data model and upload the results into the database. In contrast, the extraction services proceed as follows: The geometrical data is read out from the central database and is transformed into the required format. Finally, the data is uploaded into the destination file or into the target database. Because of the prior enrichment, all structural and semantic data transformations have been carried out. Hence, most of the data transformations formerly performed by the adaptors' integration and extraction services are omitted. Integration
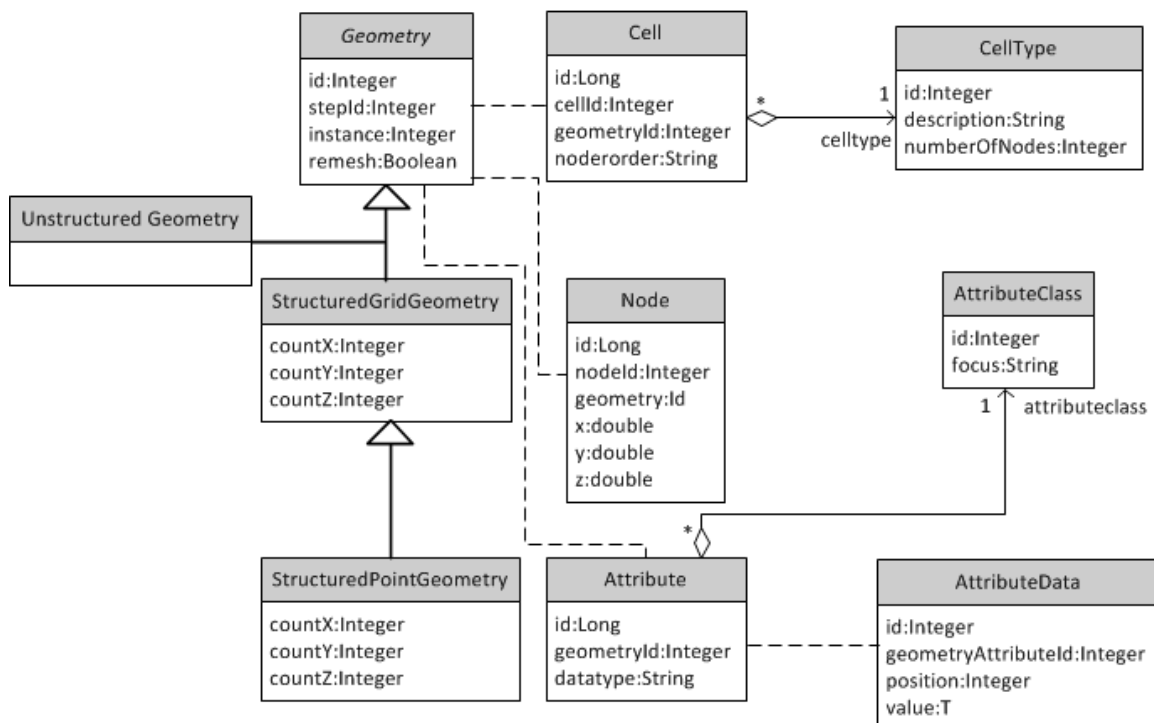
Fig. 4. Extract of the data schema used in the domain of FE-simulation

and extraction service: Most of these service adaptors have been implemented using the Pentaho Data Integrator (PDI). In case that more complex data or binary formats have been given, which can only be read by programming interfaces of the manufacturer, either the PDI functionality have been extended using the provided plug-in architecture or the needed functionality has been implemented using Java or C++. For example, the simulation results generated within the simulation tool CASTS are stored in the Visualization Toolkit (VTK) format [cf. Schroeder et al. (2004)]. Hence, an integration service was implemented, which is based on the programming interface supported by the developers of VTK using the provided functionality of the framework. Furthermore, an extraction service was developed with regard to the Abaqus input format, whereby, in this case, the aforementioned ETL tool PDI was used. Transformation library service: In order to realize the data integration, different sorts of data transformations for FE data were implemented into the application as services, for example the conversion of attribute units, the deduction of attributes from those ones that are already available, the relocating of the component's geometry within space, the modification of cell types within a geometry (e.g. from a hexahedron to a tetrahedron) or the aforementioned re-enumeration of nodes and cells.

7. Conclusion

The development of the framework presented in this chapter can be regarded as an important step in the establishment of digital production, as the framework allows a holistic, step-by-step simulation of a production process by making use of specialized tools. Both, data losses as well as manual, time-consuming data transmissions from one tool to another are excluded by this approach. The suggested framework facilitates the linking of simulation tools, which were, "until now", developed independently from each other and which are specialized for certain production processes or methods, too. Furthermore, the integration of

data generated in the course of the simulation is realized in a unified and process-oriented way. Apart from the integration of further simulation tools into an application that was already established, it is essential to extend the domain of simulations reflected upon with additional simulations covering the fields of machines and production. In this way, a holistic simulation of production processes is provided. Thereby, a major challenge consists in generating a central data model, which supports the possibility of illustrating data uniformly and in consideration of its significance in the overall context, which, in turn, comprises the levels of process, machines as well as materials. Due to the methodology presented in this article, it is not necessary to adapt applications to the data model aforementioned. On the contrary, this step is realized via the integration application, which is to be developed on the basis of the framework. Because of the unified data view and the particular logging of data at the process level, the framework facilitates a comparison between the results of different simulation processes and simulation tools. Furthermore, conclusions can be drawn much easier from potential sources of error. This is a procedure, which used to be characterized by an immense expenditure of time and costs. The realization of this procedure requires the identification of Performance Indicators, which are provided subsequently within the application. In this context, the development of essential data exploration techniques on the one side and of visualization techniques on the other side turns out to be a further challenge.

8. Acknowledgement

9. References

J. M. Myerson, The Complete Book of Middleware. Boston, MA, USA: Auerbach Publications, 2002.

A. Halevy, A. Rajaraman, and J. Ordille, "Data integration: the teenage years" in VLDB'2006: Proceedings of the 32nd international conference on Very large data bases, pp. 9-16, VLDB Endowment, 2006.

C. White, "Data Integration: Using ETL, EAI, and EII Tools to Create an Integrated Enterprise" tech. rep., The Data Warehousing Institute, 2005.

D. Chappell, Enterprise Service Bus: Theory in Practice. O'Reilly Media, 2004.

R. W. Schulte, "Predicts 2003: Enterprise service buses emerge" tech. rep., Gartner, 2002.

T. Rademakers and J. Dirksen, Open-Source ESBs in Action. Greenwich, CT, USA: Manning Publications Co., 2008.

P. A. Bernstein and L. M. Haas, "Information integration in the enterprise" Commun. ACM, vol. 51, no. 9, pp. 72-79, 2008.

P. Vassiliadis, A. Simitsis, and S. Skiadopoulos, "Conceptual modeling for etl processes," in DOLAP '02: Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP, (New York, NY, USA), pp. 14-21, ACM, 2002.

W. Kim and J. Seo, "Classifying schematic and data heterogeneity in multidatabase systems" Computer, vol. 24, no. 12, pp. 12-18, 1991.

C. H. Goh, Representing and reasoning about semantic conflicts in heterogeneous information systems. PhD thesis, Massachusetts Institute of Technology, 1997. Supervisor-Madnick, Stuart E.

U. Leser, Informationsintegration : Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen. Heidelberg: Dpunkt-Verl., 1. Auflage, 2007.

J. Euzenat and P. Shvaiko, Ontology matching. Berlin/New York: Springer, 2007.

F. Giunchiglia, P. Shvaiko, and M. Yatskevich, "Discovering missing background knowledge in ontology matching" in Proceeding of the 2006 conference on ECAI 2006, (Amsterdam, The Netherlands, The Netherlands), pp. 382-386, IOS Press, 2006.

U. D. E. Rossiter, O. Mokrov, "Integration des Simulationspaketes SimWeld in FEM-Analyseprogramme zur Modellierung von Schweißprozessen" Sysweld Forum 2007, 2007.

G. Laschet, J. Neises, and I. Steinbach, "Micro-Macrosimulation of casting processes" 4iéeme école d'été de "Modélisation numérique en thermique", vol. C8, pp. 1-42, 1998.

G. Laschet, "Homogenization of the thermal properties of transpiration cooled multi-layer plates" Computer Methods in Applied Mechanics and Engineering, vol. 191, no. 41-42, pp. 4535-4554, 2002.

D. Schilberg and T. Meisen, "Ontology based semantic interconnection of distributed numerical simulations for virtual production" in Industrial Engineering and Engineering Management, 2009. IE EM '09. 16th International Conference on, pp. 1789 -1793, 21-23 2009.

D. Schilberg, Architektur eines Datenintegrators zur durchg"angigen Kopplung von verteilten numerischen Simulationen. PhD thesis, RWTH Aachen University, 2010.

D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: The Condor Experience" Concurrency and Computation: Practice and Experience, vol. 17, pp. 2-4, 2005.

P. Cerfontaine, T. Beer, T. Kuhlen, and C. Bischof, "Towards a flexible and distributed simulation platform" in ICCSA '08: Proceedings of the international conference on Computational Science and Its Applications, Part I, (Berlin, Heidelberg), pp. 867-882, Springer-Verlag, 2008.

G. Hohpe, B. Woolf, Enterprise Integration Patterns, Addisson-Wesley, 2004

T. R. Gruber, "A translation approach to portable ontology specifications" Knowledge Acquisition, vol. 5, pp. 199-220, 1993.

O. C. Zienkiewicz and R. L. Taylor, The Finite Element Method. Basis and Fundamentals. Butterworth Heinemann, 6th ed., 2005.

W. Schroeder, K. Martin, and B. Lorensen, The Visualization Toolkit, Third Edition. Kitware Inc., 2004.
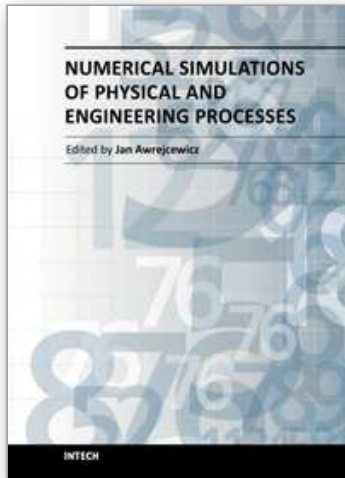
Numerical Simulations of Physical and Engineering Processes

Edited by Prof. Jan Awrejcewicz

Numerical Simulations of Physical and Engineering Process is an edited book divided into two parts. Part I devoted to Physical Processes contains 14 chapters, whereas Part II titled Engineering Processes has 13 contributions. The book handles the recent research devoted to numerical simulations of physical and engineering systems. It can be treated as a bridge linking various numerical approaches of two closely inter-related branches of science, i.e. physics and engineering. Since the numerical simulations play a key role in both theoretical and application oriented research, professional reference books are highly needed by pure research scientists, applied mathematicians, engineers as well post-graduate students. In other words, it is expected that the book will serve as an effective tool in training the mentioned groups of researchers and beyond.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Rudolf Reinhard, Tobias Meisen, Daniel Schilberg and Sabina Jeschke (2011). A Framework Providing a Basis for Data Integration in Virtual Production, Numerical Simulations of Physical and Engineering Processes, Prof. Jan Awrejcewicz (Ed.), ISBN: 978-953-307-620-1, InTech, Available from:
http://www.intechopen.com/books/numerical-simulations-of-physical-and-engineering-processes/a-framework-providing-a-basis-for-data-integration-in-virtual-production

INTECH
open science | open minds