



Nr.: FIN-010-2011

IWDE 2011 - Proceedings of the Second International
Workshop on Digital Engineering

Veit Köppen, Gunter Saake

Arbeitsgruppe Datenbanken



Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg

Technical report

Nr.: FIN-010-2011

IWDE 2011 - Proceedings of the Second International
Workshop on Digital Engineering

Veit Köppen, Gunter Saake

Arbeitsgruppe Datenbanken

Technical report (Internet)
Elektronische Zeitschriftenreihe
der Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg
ISSN 1869-5078



Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg

Impressum (§ 5 TMG)

Herausgeber:

Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Der Dekan

Verantwortlich für diese Ausgabe:

Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Veit Köppen
Postfach 4120
39016 Magdeburg
E-Mail: veit.koeppen@ovgu.de

http://www.cs.uni-magdeburg.de/Technical_reports.html

Technical report (Internet)
ISSN 1869-5078

Redaktionsschluss: 17.11.2011

Bezug: Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Dekanat



IWD E
International Workshop on Digital Engineering

Second International Workshop on Digital Engineering

Veit Köppen, Gunter Saake



VIRTUAL AND
AUGMENTED
REALITY

Veit Köppen
Gunter Saake
Editors

IWDE 2011

Proceedings of the 2nd International Workshop on
Digital Engineering 2011.

Otto-von-Guericke-Universität Magdeburg
November 2011

This workshop is supported by:



Association for Computing Machinery – German Chapter



Gesellschaft für Informatik – Regionalgruppe Sachsen-Anhalt



Otto-von-Guericke-Universität Magdeburg



**CENTER FOR DIGITAL
ENGINEERING**

Center for Digital Engineering

Preface

Last year, we started with the workshop series on Digital Engineering addressing academicals and industrial researchers from heterogeneous domains combining methods and techniques for classical engineering and software engineering. Due to the success of the first workshop and the fruitful and ongoing discussion, we continued the series with the Second International Workshop on Digital Engineering in 2011.

Software-intensive systems are becoming more and more important in an increasing number of traditional engineering domains. Digital Engineering is a new emerging trend which meets the challenge to bring together traditional engineering and modern approaches in Software- and Systems-Engineering. Engineers in the traditional domains will be confronted with both the usage of software systems in a growing amount and also with the development of software-intensive systems. Therefore, Software- and Systems- Engineering play a growing role in many engineering domains.

The usage of virtual and augmented reality has increasingly grown in the last decade. This is not only true for the development in the automotive and aviation domains but also for many other application scenarios. However, between the different phases within the product life cycle (e.g., design, planning, and production) there exists a gap or even a technological break. This complicates the necessary discussion between involved departments and experts. An early adaptation of digital data, methods, and tools can overcome this gap. This includes a close cooperation of models and the different modeling experts.

We see Digital Engineering as a continuous use of digital methods and tools in the complete product life cycle. This increases quality of process and product and makes it possible to control the processes of design, planning, and production more easily. Digital Engineering is the logical continuation of virtual engineering. All relevant process or product properties are not only respected from the classical engineering viewpoint (e.g., physical, geometrical, or electrical) but also from the software perspective. This is respected by the fact that an increasing proportion of software systems is implemented in new products.

Interoperability between models, data, and applications is a requirement to ensure Digital Engineering. In a problem-oriented consideration of product and process properties (on functional and non-functional level) interoperability of tools and data enables interactive models. These models address all human senses that are used in engineering decisions.

Due to heterogeneous challenges and involved domains, we set the focus of the second workshop on new methods and applications in the domain of Digital-, Software- and Systems-Engineering. For this workshop, we received 13 submissions. Due to time restrictions and to

guarantee quality of our workshop, we could only select a few of them. We chose the papers that, in our opinion, are evidently interrelated to start a lively discussion during the workshop.

The paper **On the Interactive Visualization of a Logistics Scenario: Requirements and Possible Solutions** by Heydekorn et al. describes design and implementation of interactive visualizations in a logistic application. The tool provides visual analysis and browsing of complex data to enhance different steps in Digital Engineering. The human factor is respected by different requirements and the tool allows a simplification of complex information models in a comfortable way.

Kellermann et al. present in their paper **Improved Spine Surgery and Intervention with Virtual Training and Augmented Reality** a case-based surgical training system called SpineSurgery-Trainer. This tool combines data and results, videos and diagnostic information for computer-aided therapy planning of spinal disorders. In a realistic workflow, therapeutic decisions and surgical planning can be trained. Augmented reality is used for interoperative visualizations and the segmentation of operation specific important structures raises patient's safety and accuracy of the intervention.

The paper by Mönch et al. with the title **Optimization of Vascular Surface Models for Computational Fluid Dynamics and Rapid Prototyping** presents a procedure to achieve an artifact-free vascular surface model. In practice, data and visualization of surface models for computational fluid dynamics contain artifacts. A differentiation of artifacts and pathological malformations requires medical expert knowledge and different software tools are used in this process.

Physics-Based Simulation of Vascular Trees for Surgery Simulations by Adler et al. deals with a method for interactive physic-based deformation of vascular systems. Minimal-invasive surgery is a complex task for surgeons. Virtual reality enables training for these interventions and the requirement analysis for new medical instruments at same time. Within simulation environments, performance issues are crucial. Optimization of graphical representations of surfaces is a challenging task. The authors present a technique for deformation of vascular structures based on the skeleton and a surface representation.

The modeling framework SAML enables the combination of qualitative and quantitative model-based safety analysis. The analysis results heavily depend on the model accuracy. The paper **Practical Experiences in Model-Based Safety Analysis** by Gudemann et al. discusses dependencies of model parameters and the accuracy of the model. For optimization processes of safety critical systems fast computation is required and therefore, a compromise between accuracy and performance has to be found. The authors' experiments show the influence of parameters on accuracy and safety analysis.

Fruth et al. propose in their paper **Sensitising to security risks in manufacturing engineering: An exemplary VR prototype** a VR simulation environment to visualize potential security risks. Due to the increasing usage of software in automated production systems, security issues have to be considered, too. Attacks can be hazardous for production systems or safety of workers might be unacceptable. Oral instructions on safety information are often not sufficient and the virtual reality improves the understanding on potential risks.

Customized and optimized visualizations for software systems are necessary to enhance software development and maintenance. **Generative Software Visualization: Automatic Genera-**

tion of User-Specific Visualisations by Müller et al. presents an approach to combine generative and model driven paradigm to apply this for software visualization.

Mory et al. in **Evaluation of Techniques for the Instrumentation and Extension of Proprietary OpenGL Applications** give an overview of instrumentation techniques of interfaces. OpenGL serves as a platform for interactive 3D graphic environments. This topic targets at interoperability of different applications and simulation environments in the engineering domain. Thus, four capabilities are qualitatively evaluated.

The paper by Frey et al. with the title **Hypermodelling - Introduction Multi-dimensional Concern Reverse Engineering** introduces a new approach in software reverse engineering. The application of data warehouse methods to software concerns in a program enables developers to search and identify code in a multi-dimensional view. The paper presents a transformation from code fragments to enables the usage of data warehousing.

In the virtual product development phase, exchange formats between different applications and the applications themselves can change. **Management of Flexible Data Exchange Processes in Virtual Product Development** by Stoye et al. designs a data model for the support of traceability of digital models. This data model is especially designed for system and process management in the virtual engineering context. It includes data description, CAD systems, exchange file formats, and process information.

The editors would like to thank the program committee members for doing such a great job regarding their paper reviews, which ensures a high quality of the contributions. The editors would also like to thank the session chairs regarding the organization of their sessions. Cordial thanks go to Uwe Freiherr von Lukas for giving an inspiring talk in the domain of maritime industry and their challenges for virtual and augmented reality. The organization of this workshop could not have been possible without the support of people mainly working backstage. Representative for the whole team we would like to particularly thank Annika Küchenhoff, Matthias Güdemann, and Maik Mory for their efforts with respect to preparation, organization, and post-processing of the workshop. Furthermore, we would like to thank Dennis Kischke for providing support regarding the workshop website and management of the paper processing. Finally, we thank all contributors and participants of the workshop, who helped us in achieving our goals and to make the Second International Workshop on Digital Engineering a great success.

Magdeburg, November 2011

*Gunter Saake
Veit Köppen*

Program Committee

Andreas Brenke, University of Applied Science Niederrhein, Germany

Jan de Meer, German Chapter of the ACM, Germany

Hele-Mai Haav, Tallinn University of Technology, Estonia

Jean-Bernard Martens, Technical University Eindhoven, Netherlands

Richi Nayak, Queensland University of Technology, Australia

Frank Ortmeier, Otto-von-Guericke University Magdeburg, Germany

Vinesh Raja, University of Warwick, UK

Gunter Saake, Otto-von-Guericke University Magdeburg, Germany

Abdel-Badeeh M. Salem, Ain Shams University, Egypt

Michael Schenk, Otto-von-Guericke University & Fraunhofer IFF, Germany

Jules White, Virginia Tech Blacksburg, USA

Contents

On the Interactive Visualization of a Logistics Scenario: Requirements and Possible Solutions <i>Jens Heydekorn, Marcus Nitsche, Raimund Dachsel, and Andreas Nürnberger</i>	1
Improved Spine Surgery and Intervention with Virtual Training and Augmented Reality <i>Kerstin Kellermann, Zein Salah, Jeanette Mönch, Jörg Franke, Georg Rose, and Bernhard Preim</i>	8
Optimization of Vascular Surface Models for Computational Fluid Dynamics and Rapid Prototyping <i>Tobias Moench, Mathias Neugebauer, and Bernhard Preim</i>	16
Physics-Based Simulation of Vascular Trees for Surgery Simulations <i>Simon Adler, Tobias Moench, and Rüdiger Mecke</i>	24
Practical Experiences in Model-Based Safety Analysis <i>Matthias Güdemann, Michael Lipaczewski, and Frank Ortmeier</i>	31
Sensitising to security risks in manufacturing engineering: An exemplary VR prototype <i>Jana Fruth, Richard Münder, Hannes Gruschinski, Jana Dittmann, Bernhard Karpuschewski, and Rolf Findeisen</i>	39
Generative Software Visualization: Automatic Generation of User-Specific Visualizations <i>Richard Müller, Pascal Kovacs, Jan Schilbach, and Ulrich Eisenecker</i>	45
Evaluation of Techniques for the Instrumentation and Extension of Proprietary OpenGL Applications <i>Maik Mory, Mario Pukall, Veit Köppen, and Gunter Saake</i>	50
Hypermodelling: Introducing Multi-dimensional Concern Reverse Engineering <i>Tim Frey, Veit Köppen, and Gunter Saake</i>	58
Management of Flexible Data Exchange Processes in Virtual Product Development <i>Michael Stoye, Stephan Vornholt, and Ingolf Geist</i>	67

On the Interactive Visualization of a Logistics Scenario: Requirements and Possible Solutions

Jens Heydekorn,
Raimund Dachsel
User Interface & Software Engineering
Otto-von-Guericke University
Magdeburg, Germany

{first name}.{surname}@ovgu.de

Marcus Nitsche,
Andreas Nürnberger
Data & Knowledge Engineering
Otto-von-Guericke University
Magdeburg, Germany

{first name}.{surname}@ovgu.de

ABSTRACT

In this paper, we describe the design and prototypical implementation of an interactive visualization in a logistics scenario for the needs of digital engineering. We applied the goals of digital engineering to realize an interactive visual browsing and analysis tool by a seamless integration of different data-sets. Furthermore, we take user requirements into consideration, which also come out of the virtual logistics scenario. Therefore complex information models were simplified in order to fulfill user goals in a comfortable way. Finally, we evaluated the implemented prototype by interviewing experts. The result contains valuable feedback to the suitability for typical logistic applications nowadays.

Categories and Subject Descriptors

H.5.m [Information Systems]: Information Interfaces and Presentation (I.7) – *miscellaneous*, H.4.2 [Information Systems]: Information System Applications – *types of systems, subject: logistics*.

General Terms

Human Factors, Design, Logistics, Visualization, Digital Engineering

Keywords

Design, Visualization, Logistics, Digital Engineering

1. INTRODUCTION

There are many very specialized information management tools that are used within the domain of logistics and goods transportation respectively. In most cases, these tools are presenting abstract and detailed information, e.g. the state of all active transfer orders in form of a textual list. Such lists are not efficient for tasks like monitoring and analyzing order states, for example. This work describes the development and implementation of such an information system, which provides easy access to complex information spaces by simplifying the content and providing an interactive and visual access. To do this we are following a human centered design process with respect to digital engineering recommendations. Therefore, we identified requirements such as application aspects in our data, elements to visualize, and search & filter methods. The concept was prototypically implemented using a fictional logistics scenario. Furthermore, we used the early stage prototype to interview experts in order to get first evaluation results. These results are used for a discussion, which is finally the base for improvements in our future work.

1.1 Motivation

In many domains, nowadays much more data and information is available than before. Beyond that, the goals for work, such as safety related decisions, cannot be found in a single value or data-set. Ordinarily, these kinds of goals are related to many values, which are only making sense in specific combinations. In such situations rather direct and simple data visualization is neither effective nor efficient. The reason for this is that either the visualization itself is difficult to read or various single visualizations are needed for displaying all required values. The concept of digital engineering tries to overcome such restrictions by many isolated applications and by the development of solutions, which are integrating complex digital environments.

Challenges for digital engineering. The concept of digital engineering includes a broad range of topics, such as embedded technology, efficiency in production, safety and security. However, in our view this includes the requirement for an improved transparency and access across various information spaces. This includes for example the support of digital systems over as many phases as possible of the product lifecycle. The system should support a seamless access to related information with the possibility for manipulating, searching and analyzing relevant information. Furthermore, in complex production environments the role of a user should be considered in application design. This includes the domain and role dependent information visualization, flexible interaction techniques and the support for digital communication of and interaction with work results.

Application: A Logistics Scenario. Our main application domain is the logistics for goods transportation. Basically, the data space contains information of payloads in a specific scope, e.g. orders to a logistic transport company. In doing so we are focusing on a future scenario that enhances the data space by various sensor data.

The technical part of the scenario basically includes sensors and the ensured transmission and storage of these data. In our scenario sensors are attached to the containers and register global position, temperature, airpressure, acceleration and the position in space. Also a contact-free identification of the container is possible, e.g. by RFID [9] or by NFC [10]. Furthermore, we take for granted that the containers have a radio link to a central computer system for data storage and logging at times. Most of these features are realized in research environments [13] and will possibly become standard in the near future (see also chapter 3). However, we are using synthetic and simulated data respectively for our research due to the high expense for a full-featured assembly.

As mentioned before we are focusing on the application domain of a transport company or similar. In this domain, we mainly identified search and analysis tasks. One possible usecase is the quality inspection of specific orders and payloads, respectively. Here, the access to a definable selection of payloads as well as the free browsing within a list of orders should be possible. A definable selection could be typically the state of an order, a time range or a location. Furthermore, the free search for textual expressions is an often used method, especially if there is just a part of a product name known. Beyond that, the inspection of orders can serve the purpose of ensuring safety and security aspects. In this case, a user should have special privileges to access such sensitive information. These privileges must differ from normal user rights.

2. REQUIREMENTS

The mentioned scenario was developed and discussed with domain experts. We considered possible requirements of future end users and logistic operators as well. Based on interviews with logistic experts the following functional requirements have been identified:

The overall goal of development by digital engineering in the logistics domain is to provide customers and service providers with a high degree of transparency within the heterogeneous environment. With transparency we mean to cover complex information and integrate these data sources into goal oriented views. Such user groups are not primarily interested in textual details of transported goods and their states. They rather want to see answers for everyday problems, such as to clarify responsibilities in case of intermodal transport chains. Here, different service providers are responsible for a specific payload.

Furthermore, different transportation goods vary in quality, quantity and other specific characteristics. Usually, users need to deal with a high amount of heterogeneous data, which need to be managed. An ergonomic possibility of direct manipulation in order to navigate, browse, search, filter and explore these data-sets is needed.

Since this system is supposed to be used for monitoring purposes also a single user should be able to use the application.

3. RELATED WORK

The used scenario is often content of research work. For example Jedermann et al. [13] described an autonomous sensor system in logistics. In technical manner, this situation is similar to ours. However, the article describes less visualization work than an agent environment for intelligent freight control. In this work, the presentation of the state of each container for an interactive application is in focus. Therefore the visualization of complex information is important to preserve an easy access for interactive task achievement in means of browsing, searching and filtering. Another example for a browsing application on tracked transportation goods, which are identifiable by RFID, is described in by Siror et al. [26].

3.1 Data Description and Exchange

The storage of data in the domain of logistics depends on the used management system, such as [3]. Therefore, various storage formats can be found in practice. However the EDIFACT [28] standard is commonly used for data exchange for administration, commerce and transport. This standard regularizes the electronic

communication for economy applications. There are also subsets of messages defined, which are domain-specific. The subset for transportation and logistics is the EDIFOR standard, which is also defined in XML [11].

3.2 Visualization

To our knowledge, the information visualization as well as the interactive visualization is sparsely applied in the domain of logistics. However, the work of Wenzel et al. [29] shows a fundamental taxonomy of visualization techniques for simulation in production and logistics, which also considers different application fields and target groups. For this work, solutions are of interest, which are in particular using geo-referenced visualization.

Geo-referenced Visualization in Logistics Systems for the visualization of geographic data can be found within the domain of geographic information systems (GIS). Such information systems often work solely with geographic data, such as ground elevation, soil composition or climatic and weather data (e.g. Schreier et al [24] and Nikolaou [17]). Beyond that, information and statistics of populations are connected with locations and areas. In this work, used information has no geological origin. Furthermore the information is referenced to items like streets and similar infrastructure.

An overview for the geo-referenced visualization of a complex information space gives Chen et al. [4]. In this work, the application domain is public health surveillance. However, many examples provided in [4] are showing the importance for providing many different references for visualization, such as time or abstract hierarchical structures of organizations, to supply task- or goal-oriented work in complex information spaces

An application for routing simulation for city logistics can be found in Baccelò et al. [1], which mainly uses street map visualizations. Furthermore, georeferenced visualizations for logistics using three-dimensional content can be found in [14]. An interactive visualization application on mobile devices can be found in [11].

Process Visualization Process visualization is mainly applied on business, software processes and other engineering domains [2][22][23]. Thereby, graphs are often used to show sequences, procedures and correlations [22]. Furthermore, existing visualizations are visually encoding rules and tests, such as are known from traditional UML diagrams [19].

3.3 Searching and Filtering

Current logistics monitoring systems like [3][16][27] allow navigating through item lists. When users search for a specific item this is the most inefficient method because of its sequential character. Open text search on the other side is rarely used since complex and dynamic index structures, which will be updated every time the data is changing, are needed. Therefore in most cases the data is organized in a database. A huge disadvantage of database queries is that they return just perfectly matching results to users [25]. They do not build ranked lists like in information retrieval. Therefore users do not get similar results to their query and typing errors, similar terms and specific use contexts will not be considered. But this can be extremely helpful to solve complex problem analyzing tasks as focused on in our scenario (also mentioned in [15]).

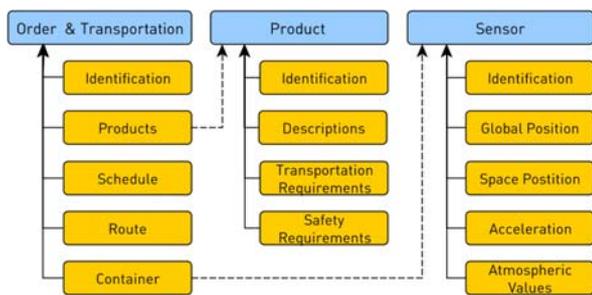


Figure 1. Schematic illustration of the information categories.

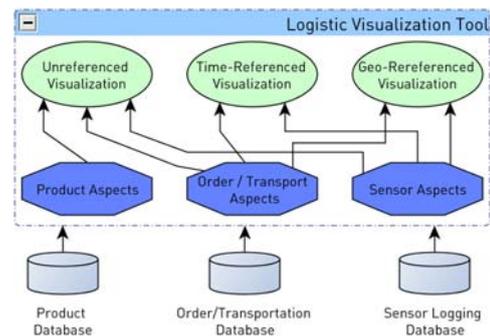


Figure 2. Database management.

4. CONCEPTUAL DESIGN

Based on the requirements we designed the whole system environment for our research. We started with the data and information design as foundation for all further work. After this, we choose visualization methods to provide the contextual access to the information. Finally, we designed the linkage of visualizations by user interaction.

4.1 Data Model

Basically, there are three important data spaces that we identified in our scenario: product items, information and processes of transportation, and the data of sensor logging. A simplified overview of the spaces and their main content can be found in Figure 1 and Figure 2. All three data spaces are integrated while working on typical tasks. Similar to general information visualization tasks we are focusing on the tasks browsing, searching and analyzing.

Therefore, a central role plays the data space containing information regarding order and transportation because it semantically connects all available spaces. As illustrated in Figure 1, this space stores data to handle transportation orders, which includes the item type, container for transportation and the processing state. Furthermore the route and scheduling information are stored within this space. We assume that the route is already defined, for example using criteria like in [1]. However, we include types for the order like “express” or “security” in order to differentiate types of processing from logistic processes. We strongly simplified the possible status of order and payload, respectively.

The data space of products basically contains information regarding special needs for transportation. All related requirements, such as the safety requirements of an order, are derived by descriptions here. Beyond that, the interpretation of sensor data is also related to the declared requirements for transportation in this data space.

As mentioned before, we assume that every container for product transportation is equipped with sensors. These sensors are logging relevant values over the whole transportation process. Based on product requirements, atmospheric conditions (e.g. temperature, humidity, and air pressure), slope of the underground and accelerations by shocks or collisions are needed. Furthermore the global position is logged so that all other measurements have a geographic reference. By using signals like from GPS [7], all values also have a globally unique timestamp.

4.2 Support for multiple views

We intend to support multiple views on the used information model for logistic purposes simultaneously. As mentioned in the section before, the data model allows us to express various aspects within different visualizations.

View on global positions. Every item of our data space contains or can be linked to a global position. The data-set of a transportation order (including the process status) as well as sensor-events contains a global position, whereas the product items can be manually linked with a position, for example by retrieving the approximate position of a postal address. However, the omnipresence of global positions allows us to display all items of our data space within geo-referenced visualization.

View on time. All information in the described data space, except for the product descriptions, contains both: a position and a point in time. Therefore, we can show all these information also in the context of time, e.g. by using a timeline containing representations of the sensor events or transportation statuses.

Views on processes. The transportation of goods must fulfill various requirements from the client, which include special work steps and inspections. The resulting process model is often quite complex. Furthermore, the process oriented order management is a criterion for quality rating within this domain. Therefore, the logistic of goods transportation uses process plans to define the handling of all possible orders. The described processes are important for monitoring and analysis of current transportations and completed orders. Basically, the display of a process status can be different. On the one hand, it is possible to display the *actual* status of an order. On the other hand, it is possible to display the *change* of a process status, for example as an event. A combination of both cases is also possible, to highlight the phase transition of an item. However, the number of phases and statuses depends on supported transportation options and their regarding processes. To keep the scenario simple at the beginning, we designed a simplified status model, which includes the states: start, pause and end. Such events can be visualized in a geographical context, within the timeline and in a specialized status log.

Views on signal data. We do not want the support of a direct and abstract view to logged signal data, because we try to keep up the concept of visualization by context. Basically, we show abnormal signals in form of events, which are finally displayed. An abnormal signal is a sensor value that is out of a defined interval of normal values. Furthermore each event implies a reference to the sensor type, which allows to find events by sensor type. A signal event can be seen and accessed in one of the provided visualizations and within the search results. The

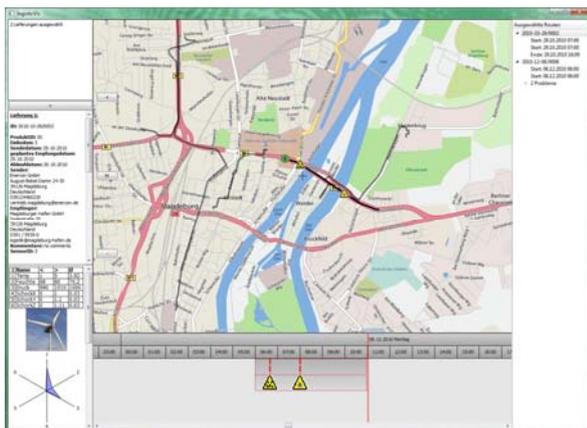


Figure 3. The main window of the prototype.

visualizations are displaying a type-describing icon (see Figure 3), whereas items at payload lists get a general visual mark to show conspicuity.

Interaction and view linkage. Basically we provide almost exclusively a visual access to the displayed transport and ordering information. This means that all views on the data do not use abstract lists or descriptive selection or navigation functions. Our environment for interaction primarily includes mouse and keyboard, because this is commonly used in business environments. However, selection and navigation activities should not result in the change of a single view, but rather all views should fit to these activities. For example, if a user selects a geographic area containing many routes of transportation orders by the mouse, the timeline also changes the view to show the selected orders (and vice versa). This allows an easy to use brushing technique by different contextual visualizations.

5. PROTOTYPICAL IMPLEMENTATION

We implemented a prototype to proof our concepts (see Figure 4). This prototype is implemented in C++ using the Qt Framework [18]. During the development of the prototype we were mindful to keep up a modular concept that allows us a flexible combination and rearranging of interface and software components.

5.1 Data Handling

The logistic data used by our prototype is locally stored in files, which follow a self-defined XML scheme. The used XML scheme is also useful for basic safety and verification checks. We do not adopt existing standards to maximize flexibility for this prototype. However, this type of data description allows us to easily adapt for potential future requirements, such as transformation in other XML dialects, the enhancement of the data space or the use of databases over a computer network.

We have two stages of data-sets within the application. The first data-set includes the loaded data and is the base for all further selection and filtering functions. The second data-set includes the result after a filter action. All views are displaying and highlighting a single selected payload or the filter results if available.

5.2 Views on data space

We realized the following views to provide a complete visual access to the available information: overview on all payloads, a detailed view of selected payloads, a geo-referenced view, a time-related view, and a visualization of the status log.



Figure 4. Geo-referenced visualization of route data and sensor events.

The detail view contains basic details of the order and the transported products. Therefore, the view shows general information of the product, such as size, weight and a photo. Furthermore, the condition of the container is displayed by a table as well as a star graph, which shows averaged sensor values.

5.3 Linking logistic data to geoinformation

As mentioned in section 4.2, we are able to use logistic data from transportation scheduling plans and logs as well as sensory data to visualize positional information.

Basically transportation routes like streets and railways are suitable geographic references for our scenario. Therefore we use public services in the World-Wide-Web to visualize street map information. Such maps are provided for example by services like OpenStreetMap [21], Google Maps [6] or Yahoo Maps [30]. In particular we are able to use web map services (WMS) [20], which provide more specialized and configurable maps.

Within the map view routes are visualized by line paths. Additionally, process states such as start, end and waiting positions are displayed by icons. Beyond that additional icons show events of critical sensor values. The icons are corresponding to the type of the critical sensor event, such as a strong shock or high humidity.

5.4 Providing time references

Our prototype uses the commonly used timeline metaphor to provide a time related view on the data space (see Figure 5). Following this method all transportation orders can be displayed by their temporal order from left to right. A vertical line indicates the current position in time. Concurrent orders will be displayed with a vertical shift. This view contains all orders in scope. Therefore finished orders as well as planned transportation in future are also visible. The timeline shows all sensor events by icons. The icons are the same as in all other views for better recognition.

The timeline provides an interactive access for the user. The view automatically zooms and pans to items, which have been selected outside of this view. Furthermore every displayed item can be selected separately or in a group. For group selection the user can freely mark a time interval. Of course, after a selection, all other

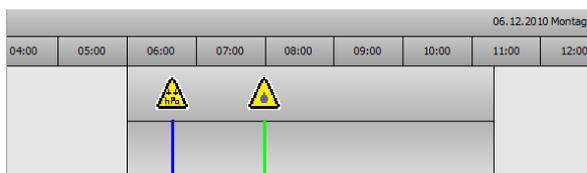


Figure 5. Screenshot of the interactive timeline view.

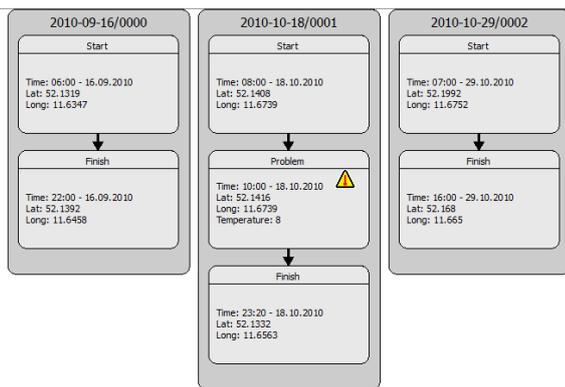


Figure 6. Visualization of the status log including state and sensor events.

views are updating their focused items.

5.5 View on process information

As mentioned before logistic processes can become complex to cover the broad range of requirements on goods transportation. Because of this complexity we simplified the representation of processes in the system. We decided to present process events only in order to display the transition from one process phase to another. The view on processes shows these events in form of a status log for first tests (see Figure 6). The simplified view shows a box for each reached process phase and all statuses of a transportation order. Furthermore, we add sensor events as special status items. Future orders are displayed with the planned schedule information.

Basically, the horizontal order of the displayed payloads is based on the chronological order if no payload was selected or no filter is defined. Selected items are always displayed on the left. The result set of a filter operation uses a combination of both sorting methods. Every change of the order happens in an animated way to keep up the orientation for the user.

5.6 Information Management

The complete content within the information space, which is described in section 4.1, is used to create a search index. We used CLucene [5], an open-source text search engine, in order to index all the information items used in this logistics scenario. Furthermore, we used this search engine for realizing the search and filtering features.

Searching and Filtering Approach. The proposed tool features a basic text search with auto complete functionality shown on top of Figure 7. Presented results can be filtered in a next step. Setting filters gives users the opportunity to refine their information need. In contrast to other filtering approaches, as mentioned in [8], our tool enables users to choose from hard and vague filters. Unlike hard filters, vague filters offer fuzziness functionality. This is also supported by the free text search. To create filter criteria, the user can drag single items like a product, a payload containing several products or single sensor events from the result listing to a drop area at the bottom of the screen (see Figure 7). Here, the user chooses from location-based, time-based or process-oriented filters (from left to right). For example, a user might be interested in a timeframe, which covers three specific payloads. To filter for this time interval the user needs to drag those payloads onto the

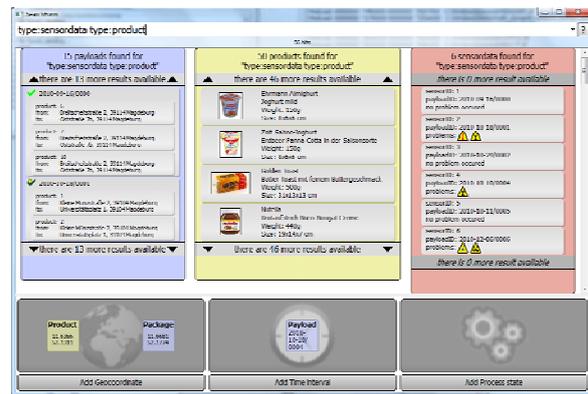


Figure 7. Screenshot of the text and keyword search interface. On top is the text input line, below the retrieved results from the databases and at the bottom the area to define the filter.

drop area showing the time symbol. The user is able to refine the filter by dropping further items such as product items onto the area. Furthermore it is possible to define filter criteria by pressing one of the buttons that are placed below each filter area (see at bottom of Figure 7). Each time a dialog window asks for filter parameters and after confirmation a filter element will appear. Every filter element can be removed and exchanged by simple drag-and-drop. The filter result will be immediately shown in the geo-referenced and time-oriented visualizations.

Result Presentation. Unlike ordinary text listings, the user of our tool gets results in multiple ways. The search results within the database are grouped in three result categories following the database management: payload results, product results and abnormal sensor-event results. Furthermore, each result is highlighted in each type of visualization. Related information will be shown in a map by route highlighting, in the timeline and within the process visualization.

6. EVALUATION

We evaluated the prototype by using the method of *free interviewing*. We asked three experts, who covered the domains information retrieval, interaction design and logistics. We presented and explained all features of the system mentioned in this work. All experts saw the same presentation. After this presentation, all experts were asked to name problems and possible solutions. Furthermore, all experts were interviewed alone to get more detailed feedback. The following list shows the summarized feedback.

Search The expert emphasized the interactivity between searching, filtering and item selection in different areas as a good concept. However, he wishes a more integrated view between the two windows, searching & filtering and browsing & navigating, and no separation into two single windows. This might improve the interactive and context sensitive character of this application. Furthermore, he asked for some missing functionality: Saving past searches and their corresponding results, backtracking functionality (undo), possibility to identify comparable problems or familiar events in the past and representation of limited resources (e.g. the limited number of trucks owned by a logistic company).

Result presentation The expert liked the possibility of switching between different visualizations like from map view to

process view and back. This supports different types of users and use contexts. But he critically mentioned the missing function keep a certain view as frozen in order to browse another aspect without directly changing the view. This might be helpful for users to fulfill subtasks or to compare certain information to each other.

Domain application The expert rated the general functionality of the prototype as very good. The prototype might be helpful especially for controllers and departmental managers. But the expert identified a need for improvement with regard to the support of an internal logistics scenario, which considers stockpiling time, place and mode. For example, the support for storage protocols is currently missing. Also the initial map view was found to be more likely a second level view mode. The first view mode for logistics should be the process-oriented view. Furthermore, he stated that there might be intermodal transportation chains, which require the support of different roles of users since different users might not be allowed to see each single step of a more complex transportation process. An exemplary use case mentioned by the expert is the transport of temperature sensitive products which need to be cooled or even stick to a certain temperature during delivery, such as pharmaceutical products like vaccines.

7. DISCUSSION AND FUTURE WORK

For this prototype we started with a rather simple scenario and combined it with the needs for digital engineering. With these requirements we developed a concept to provide a user with visual access to the complete data space. However, the feedback of the experts shows that the approach for the concept seems to be valid. Otherwise, we need to enhance the scenario by more user roles and requirements from logistics. With these enhancements this prototype could be a more valuable example for real applications.

In order to overcome problems by the separate filter interface, we plan to create a more integrated application screen. Therefore, we enhance the browsing feature through payloads by a slot for searching & filtering. Furthermore, we want to enhance the logistic environment in our scenario by the support for intermodal transportation chains, which is also useful to enhance our process and status model. Finally, we need to take different user roles into consideration to underline security aspects. Although the differentiation of user rights was originally included in the scenario, the actual prototype does not support this feature. The expert feedback underlines the need for rights management. Therefore, we will create an enhanced scenario, which takes more safety and security aspects into consideration, for example when delivering sensible drugs.

Moreover, the source can switch from files to XML databases without changing the description scheme to enhance performance and flexibility of data storage. Since the presented work is still in progress, we are currently focusing on system enhancement and stabilization following the results. Within the next improvement steps we will also include end user participation and usability tests to proof our prototype.

8. CONCLUSION

In this paper, we presented an information system for a logistics scenario that includes geo-referenced visualizations as well as time-referenced visualizations. We considered the requirements for digital engineering as well as for user centered development. The realized system seamlessly integrates various data sources and provides visual access to the large and heterogeneous data space. The resulting data space is visualized by different

referenced views. Furthermore, we realized an interface, which supports users by search and filter tasks. Thereby, all provided views are immediately reacting on a user's selection or a changing filter result. Furthermore, we evaluated the prototype by interviewing experts. The results of the evaluation provided us with valuable feedback for the improvement and further development of the system.

9. ACKNOWLEDGMENTS

The authors are funded by the German Ministry of Education and Science (BMBF) within the ViERforES II project (no. 01IM10002B). We thank Patrick Saalfeld for his implementation work.

10. REFERENCES

- [1] Barceló, J., Grzybowska, H., and Pardo, S. 2007. Vehicle Routing And Scheduling Models, Simulation And City Logistics. In: *Dynamic Fleet Management*, V. Zeimpekis, C. D. Tarantilis, G. M. Giaglis and I. Minis, Eds. Springer US, pp. 163–195.
- [2] Bobrik, R., Reichert, R. and Bauer, T. View-Based Process Visualization. *Lecture Notes in Computer Science*, 4714/2007, Springer, 2007, 88-95.
- [3] CargoWise, <http://www.cargowise.com>
- [4] Chen, H., Yan, P., Zeng, D., Chen, H., Zeng, D., and Yan, P. 2010. Data Visualization, Information Dissemination, and Alerting. In: *Infectious Disease Informatics*. Springer US, 73–87.
- [5] CLucene: <http://clucene.sourceforge.net/>
- [6] Google Maps: <http://maps.google.com/>
- [7] National Coordination Office for Space-Based Positioning, Navigation, and Timing, Global Positioning System, <http://www.gps.gov/technical/icwg/>
- [8] Hanani, U., Shapira, B., Shoval, P. 2001. Information filtering: Overview of issues, research and systems. *User Modeling and User-Adapted Interaction*, 11, pp. 203-259.
- [9] ISO/IEC 18000-1:2008: Information technology – Radio frequency identification for item management – Part 1: Reference architecture and definition of parameters to be standardized.
- [10] ISO/IEC 22536:2005: Information technology -- Telecommunications and information exchange between systems -- Near Field Communication Interface and Protocol (NFCIP-1) -- RF interface test methods.
- [11] ISO/TS 20625:2002: Electronic data interchange for administration, commerce and transport (EDIFACT) -- Rules for generation of XML scheme files (XSD) on the basis of EDI(FACT) implementation guidelines.
- [12] Jansson, H., Bjærum, R., Hellman, R., and Morka, S. 2009. Accessible User Interfaces in a Mobile Logistics System. In: *Universal Access in Human-Computer Interaction. Intelligent and Ubiquitous Interaction Environments*, C. Stephanidis, Ed. Springer Berlin / Heidelberg, pp.317–326.
- [13] Jedermann, R., Antúnez Congil, L. J., Lorenz, M., Gehrke, J. D., Lang, W. & Herzog, O. (2007). Dynamic Decision Making on Embedded Platforms in Transport Logistics - A Case Study. *International Conference on Dynamics in Logistics*, H.-D. Haasis, H.-J. Kreowski and B. Scholz-Reiter, Eds., Bremen, Germany, Aug. 28-30, pp. 191-198.

- [14] Jessen, U.; Wenzel, S., 2001. The Integration of 3-D Visualization into the Simulation-based Planning Process of Logistics Systems. *Simulation* 77 (2001) 3-4, pp. 114-127.
- [15] Kumpulainen, S., Järvelin, K., Serola, S., Doherty, A., Byrne, D., Smeaton, A. F., Jones, G. F. J. 2009. Data collection methods for analyzing task-based information access in molecular medicine, *MobiHealthInf 2009 - 1st International Workshop on Mobilizing Health Information to Support Healthcare-related Knowledge Work*, Porto, Portugal, 16 January 2009.
- [16] Logmo GmbH & Co. KG, FUMO, <http://www.logmo.biz/fumo/>
- [17] Nikolaou, S. 2005. Geographic information systems for ground motion evaluation in seismic bridge analysis. *Bridge Structures: Assessment, Design and Construction*, Volume 1, Issue 3, Issue 3, 2005, pp. 293-306 .
- [18] Nokia, Qt Framework: <http://qt.nokia.com>
- [19] Object Management Group, OMG Unified Modeling Language, <http://www.omg.org/spec/UML/2.2/>
- [20] Open Geospatial Consortium: Web map service - Implementation Specification, <http://www.opengeospatial.org/standards/wms>
- [21] Open Street Map: <http://www.openstreetmap.org/>
- [22] Rinderle, S., Bobrik, R., Reichert, M. and Bauer, T. Business Process Visualization – Use Cases, Challenges, Solutions. In: Proc. ICEIS'06, Springer, 2006, pp. 204-211.
- [23] Schönhage, B., van Ballegooij, A., and Elliëns, A. *3D gadgets for business process visualization—a case study*. In: Proc. VRML '00. ACM, 2000, pp. 131-138.
- [24] Schreier, H, Brown, S., Schmidt, M., Shah, P., Shrestha, B., Nakarmi, G., Nakarmi, G., Subba, K., Wymann, S. 1994. Gaining forests but losing ground: A GIS evaluation in a Himalayan watershed, *Environmental Management*, Volume 18, Issue 1, pp.139-150.
- [25] Silberschatz, A., Korth, H.F., and Sudarshan, S. 2006. Database system concepts. McGraw-Hill Higher Education, Boston.
- [26] Siror, J. K., Guangun, L., Kaifang, P., Huanye, S., Dong, W., 2010. Impact of RFID Technology on Tracking of Export Goods in Kenya, *Journal of Convergence Information Technology*, Vol. 5, no. 9, pp. 190-199.
- [27] Transpo Data AG, TRANSPO, <http://www.transpodata.ch/de/produkte/transpo/uebersicht-transpo.html>
- [28] United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport, UN/EDIFACT Standard Directories, http://www.unece.org/trade/untdid/down_index.htm
- [29] Wenzel, S.; Bernhard, J.; Jessen, U., 2003. A Taxonomy of Visualization Techniques for Simulation in Production and Logistics. In: Proceedings of the 2003 Winter Simulation Conference. SCS, San Diego, pp. 729-736.
- [30] Yahoo Maps: <http://maps.yahoo.com/>

Improved Spine Surgery and Intervention with Virtual Training and Augmented Reality

Kerstin Kellermann
Department of Simulation and Graphics
Universitätsplatz 2
39106 Magdeburg, Germany
kerstin@isg.cs.uni-magdeburg.de

Jörg Franke
Department of Orthopedic Surgery, University Hospital of Magdeburg
Leipziger Str. 44
39120 Magdeburg, Germany
joerg.franke@med.ovgu.de

Zein Salah
Department of Electronics, Signal Processing and Communications
Universitätsplatz 2
39106 Magdeburg, Germany
zein.salah@ovgu.de

Georg Rose
Department of Electronics, Signal Processing and Communications
Universitätsplatz 2
39106 Magdeburg, Germany
georg.rose@ovgu.de

Jeanette Mönch
Department of Simulation and Graphics
Universitätsplatz 2
39106 Magdeburg, Germany
jeanette@isg.cs.uni-magdeburg.de

Bernhard Preim
Department of Simulation and Graphics
Universitätsplatz 2
39106 Magdeburg, Germany
bernhard@isg.cs.uni-magdeburg.de

ABSTRACT

Computer-aided diagnostic and therapy systems are increasingly becoming established tools for several medical procedures. In this paper, we introduce a case-based surgical training system prototype for imparting anatomical knowledge, cognitive strategies of treatment and interaction techniques for computer-aided therapy planning of spinal disorders. Patient data, image data, image analysis results and 3D models as well as annotated surgery videos and diagnostic information underlay the SpineSurgeryTrainer. Neurosurgeons or orthopedic surgeons who want to specialize in spine surgery can, in a realistic workflow, train therapeutic decisions and surgical planning. Expert opinions and their case-specific approaches support the learning process.

In addition, we integrate a prototypic add-on for enhancing the intraoperative visualization within a navigated spine surgery utilizing an augmented reality approach. In essence, operation-specific important anatomical structures are segmented from preoperative patient data and superimposed on the video stream of the operation field. In addition, slices of the anatomy data, as well as shape and depth information of targeted structures, like spinal nerves or herniated discs, can be blended, which allows for a better protection of risk anatomy and accurate identification of the structures under consideration, and thus raises the safety and accuracy factors of the intervention.

Categories and Subject Descriptors

I.3.7 [COMPUTER GRAPHICS]: Three Dimensional Graphics and Realism—*Virtual reality*; J.3 [LIFE AND MEDICAL SCIENCES]: *Training*

Keywords

Spine surgery, surgery training, augmented reality, intraoperative visualizations

1. INTRODUCTION

In recent years, complex surgical procedures mostly rely on extensive analysis and preoperative processes, including exploration and interpretation of patient image data. However, one of the challenging tasks in a surgeon's work is to transfer the information displayed in 2D diagnostic images to the 3D situation in the real world of the patient's anatomy. This challenge is mastered in a continuous learning process, still with significant obstacles. Moreover, the specific treatment decision is often based only on the clinical experience and the intervention spectrum of the surgeon.

In spine surgery, e.g., the spatial relations between neural and spinal structures and the position of the spine to the surrounding muscles, vessels and other tissues must be known. In this regard, image-guided surgery is a versatile and effective technology to improve the intraoperative orientation to the unexposed anatomy [7]. This is highly advantageous for surgeries in regions with high density of critical and vital structures. Introducing augmented reality facilitates the transfer of the diagnostic imaging to the individual patient anatomy in a straightforward fashion. Furthermore, surgical training systems enhance the experience and skills of the surgeon and related physicians.

In this paper, we extend our SpineSurgeryTrainer prototype, introduced in [3], with AR functionalities. In essence, we present two tightly-coupled prototypes: The augmented reality-based prototype improves intraoperative visualizations, whereas the training prototype supports future surgeons to develop anatomical knowledge and strategies regarding therapy decisions and planning for various diseases (e.g., disc herniation or stenosis of the spinal canal) based on actual case data. We first present the prototype for improving intraoperative visualization by augmenting the video stream of the operation field with relevant patient data from different diagnostic and intraoperative imaging modalities. This will facilitate the identification of vital structures like blood vessels and nerves or landmarks like bony structures. In section 4, we present our training system pro-

tototype. Apart from the interactive training of treatment decisions and planning of minimally invasive surgery, open surgical procedures and access trajectories, we aim to familiarize the trainee with virtual and augmented reality based tools for later therapy planning or intraoperative navigation.

2. MEDICAL BACKGROUND AND RELATED WORK

The education and training in surgery is highly dependent on the available experts and the existing case spectrum. For effective training the case spectrum must include many variants of treatment methods and their different indications. Casus [6], e.g., provides cases of chosen medical specialties for different education levels in order to train therapy findings.

Practical training includes the training with very expensive artificial models of organs, as well as mannequins, cadavers, or real patients with expert supervision. Drawbacks of mannequins include limitations in their replication of physiology and that, at best, they have a limited range of anatomical variability.

An alternative approach that makes an impact on the medical community are computer simulation systems, which can train practitioners on a virtual patient whilst critically analyzing skills and providing feedback on the performed procedure without the presence of an expert trainer. Of the human sensory modalities, the two cues most frequently used in virtual simulators are *vision* and *touch*. Combining 3D display devices and haptic devices is commonly used, such as in the prototype of a telerobotic surgical simulation system for training spine surgery [21]. PalpSim is another example of a visio-haptic medical training environment based on augmented reality technology [2]. Training systems, such as the KISMET-based Virtual Endoscopic Surgery Training system VSOne [9], provide simulation techniques which allow the modeling of “virtual tissue” based on a data-model which reflects the physical characteristics like mass, stiffness and damping of real tissue. The degree of reality is associated with a specialization of a simulator for a certain training task. This process requires customized hardware and time-consuming and costly preparations (e.g., force measurements carried out in vivo) or deformation simulations in terms of the tissue or the instrument movements (e.g., [4]). Training or simulation of several training tasks with one system bears a challenge. For the training of planning surgical and interventional procedures, the anatomical knowledge and development of planning strategies based on anatomical landmarks are crucial. A simulation of realistic device steering is not required.

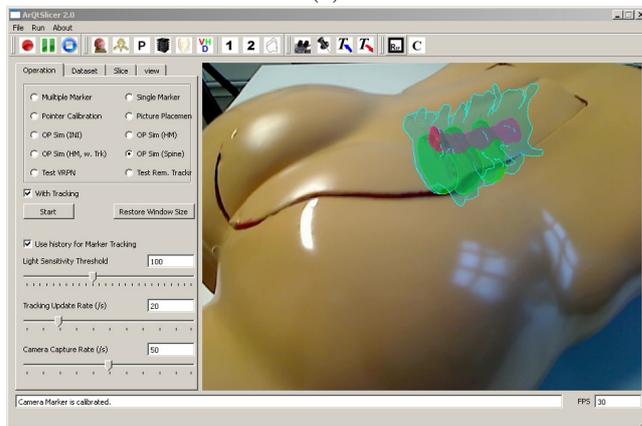
In order to support the surgeon during surgery, medical navigation systems are used to establish correspondences between locations in an acquired patient dataset and the patient’s physical body. In the last two decades, many research works aimed at enhancing image-guided surgery with AR technologies. Some works provided enhanced endoscopic views that are paired with synthesized virtual renderings generated from the same view, e.g. [10]. Other systems tried to modify the design of operating binoculars [1] and microscopes [5] to allow for data augmentation. Augmented reality has also been introduced as a training tool for surgical procedures [14, 13].

3. AR-ASSISTED SPINE SURGERY

In this section, we adapt and extend our intraoperative visualization method [18] for the use in navigated spine surgery. Figure 1(a) depicts the hardware setup of our prototype, in which we use a model of the upper body with an integrated model of the lower part of the spine. We also adapt a tablet PC with a high resolution built-in camera to simulate a surgical microscope or endoscope. In a first step, we compute the camera intrinsic parameters using MATLAB’s Camera Calibration Toolbox.



(a)



(b)

Figure 1: Prototype of the intraoperative extended reality visualization system: (a) Hardware setup; (b) visualization module.

For tracking, we implement a marker-based optical tracking server, which continuously captures a video stream of the trackable region with a calibrated high resolution camera. At each frame, the system searches for predefined markers and, for each detected marker, computes its pose in the camera coordinate system. Pose information of all detected markers are transmitted to the tablet PC over a WLAN con-

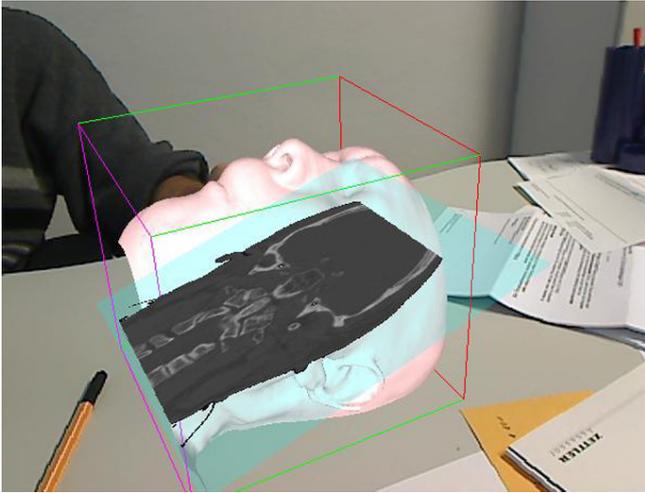


Figure 2: Interactively selected tomographical slices.

nection. However, the design allows for connecting to other commercial tracking systems.

Prior to the operation, a one-time hand-eye calibration step is performed. In essence, the tablet PC camera is calibrated with respect to the tracker using a common reference marker that is visible (only at the calibration step) to both cameras. A second tracked marker is fixed to the tablet PC, where the goal of the calibration is to compute the spatial transformation between the tracked marker and the video camera of the tablet PC. After calibration, the transformation remains valid as long as the marker does not move with respect to the video camera.

To register the patient/phantom with the scanned dataset, and hence with the anatomical models reconstructed from it, the reference marker is attached to a rectangular plastic board that is scanned with the patient or phantom model. The corners of the marker are interactively selected from the scanned dataset, and their absolute positions are calculated considering image spacing, and defined as registration points. 3D positions of the corresponding points in the patient coordinate system are precisely defined using the tracking system. The two sets of points are finally registered adapting a paired-point rigid registration scheme, applying a least square fitting approach.

The rendering module (Figure 1b) running on the tablet PC continuously captures a video stream and renders it as a background. At each frame, relevant virtual objects are rendered/overlaid using a two-pass rendering algorithm that highlights object silhouettes for better shape perception. Several objects can be overlaid according to the current operation conditions. These include 3D reconstructions of segmented structures from the anatomy dataset.

Additionally, tomographical slices can be superimposed. For this purpose, we adapt an optimized slicing algorithm [17] to compute tomographical slices at the desired position and orientation. The generated slice image is then blended in the real scene with the correct pose (see Figure 2). Here, the dimension and spacing of the 3D dataset and the generated cross section are considered to correctly adjust the physical proportion with the patient and environment. For certain structures, such as the yellow colored vertebral discs

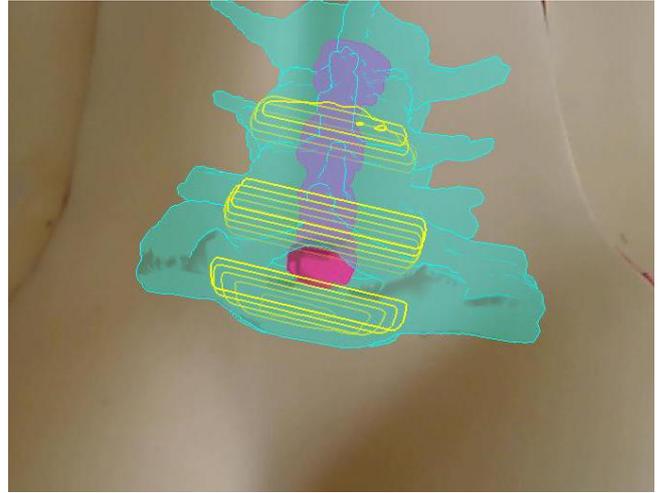


Figure 3: Shape information of intervertebral discs is emphasized by a set of contour lines.

in Figure 3, an enhanced visualization of shape and depth information can also be provided. This is achieved by extracting the planar contours of the (herniated) discs at successive depths perpendicular to the viewing direction. Depth information is conveyed via depth cueing by defining the transparency of a contour as a linear function of its depth. In minimally-invasive endoscopic or microscope-based spine surgery, augmentation should be performed on the video stream of the endoscope/microscope camera. However, this requires tracking these devices and a more complicated calibration of their cameras.

4. VIRTUAL TRAINING SYSTEM

To develop a successful training system, the integration of potential users in the design is crucial. As described in [3], we used the informal specification method of scenarios to involve them actively into the conceptual process. Furthermore, the conceptual design of the SpineSurgeryTrainer is based on the Four-Component Instructional Design Model (4C/ID-Modell) [11]. According to this model, four components are necessary to realize complex learning: learning tasks, supportive information, procedural information, and part-task practice. Initially, the prototype provides some cases of various degenerative diseases of the neck and lumbar spine. Based on these cases, the process of treatment decision and the relevant planning of conventional therapy (e.g., physiotherapy), epidural injection or access planning in minimally invasive surgery can be trained.

4.1 Workflow

The training procedure of the SpineSurgeryTrainers, depicted in Figure 4, is based on a high-level clinical workflow. After selecting the training case, the process of diagnosis follows. At first, the relevant patient data is provided including age, sex, weight, and family anamnesis, the assessment of the professional environment, case history and previous treatments. The results of clinical and physical examinations and medical images are then presented and have to be assessed just as the anamnesis. Afterwards, the trainee interactively explores the MRI image slices and a polygonal

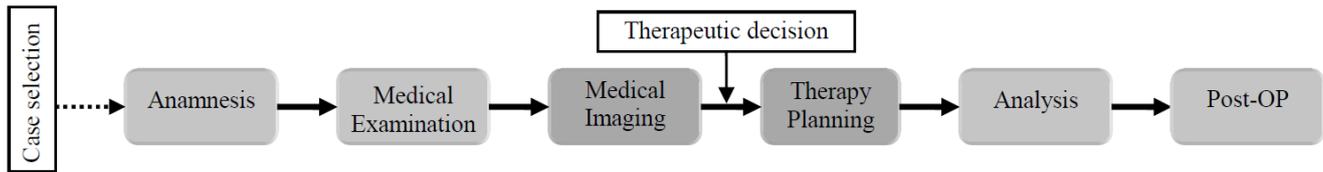


Figure 4: Training workflow of the SpineSurgeryTrainer (based on clinical workflow).

3D model reconstructed based on those image data, in order to finalize a therapy decision. The provided data and exploration tools and interaction techniques are described in more detail in the subsequent sections. After the treatment decision, the trainee can plan the selected therapy virtually. The focus is on the interactive planning of interventions and surgical procedures based on the image data and the 3D model. After completing the therapy planning, the result will be compared with those of experts in the analysis step. Finally, the user can learn about the progress of the real operation and the follow-up. Subsequently, he/she has the option to train another variant of this case.

4.2 Data

For the generation of polygonal 3D models, the slice data (mostly MRI) ideally need to exhibit a standard resolution of minimally 1mm and a slice thickness of maximally 3mm. In general, a training case contains the following 3D models: vertebrae, intervertebral discs, spinal cord, large vessels and nerves, muscles, and the skin, which is relevant for access planning. Neck cases additionally contain tissues of the respiratory tract and the esophagus.

4.3 Exploration of Medical Image Data

To explore medical image data, the user can switch between two views, as shown in Figure 5. One view allows for displaying data in a traditional slice-oriented manner, where the grey value window can be defined. Zooming through individual slices is also possible to allow for more exact inspection of the data. Moreover, semitransparent colored overlays of segmented structures can also be blended, whereas 3D representations of these structures are displayed in a second 3D viewing widget, which supports the cognitive connection between the two views as well as the spatial perception of the anatomy. For this purpose, the 2D slice can also be integrated into the 3D view, as Figure 5 shows.

The system is intended to be used with a six degree of freedom (6DOF) input device to slice through the data, analogous to the marker-based augmented reality approach described in Section 3 [17]. Instead of the tracked marker (see Figure 2), a 6DOF input device from the SensAble Technologies PHANTOM product line of haptic devices¹ is used to manipulate the slice within the 3D scene. We used a PHANTOM Omni[®] and PHANTOM Desktop[®] from SensAble for the implementation and evaluation of our 6DOF interaction methods.

Using a force feedback Phantom device, differences in haptic material properties (e.g., stiffness) help to distinguish between critical and less critical tissues for minimally invasive procedures at the spine. This classification is performed by our medical experts during the examination of the 3D recon-

structions. An injury of vulnerable risk structures such as nerves and large vessels must be avoided, while impenetrable structures (e.g., vertebrae) may serve as landmarks. Injury of fat and muscle tissue is unavoidable. Principally, the haptic material properties are used to detect collisions of the 3D cursor with surfaces instantly (by force feedback), even if the view is blocked. Thus, the trainee can quickly identify and correct errors during, e.g., a trajectory planning (see Section 4.4). Indirectly, this extends the trainee's knowledge of the underlying three-dimensional spinal anatomy.

Free navigation in 3D space by rotation (with one or all three degrees of freedom), translation and zooming is difficult for untrained persons. Thus, in addition to the traditional views (axial, coronal, and sagittal) predefined favorable views are provided in order to support the user in navigation to crucial views of the 3D scene. Interfering structures are already masked out in those selectable views. Moreover, automatically generated camera paths guide the user through the 3D scene and therefore simplify the exploration task [12]. The camera tour can be arbitrarily interrupted by the user to interactively explore a certain region, and thereafter the animation can be resumed.

4.4 Virtual Therapy Planning

In addition to the exploration task, the 2D medical image slices (with overlays) and the 3D model are provided for the interactive treatment planning. On request, an introductory animation (illustrated in Figure 6) shows only the spine with the highlighted pathology and transforms the view from the initial coronary view into the operation view, showing all segmented structures.

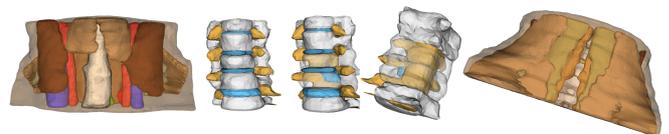


Figure 6: Intro sequence from coronary view to one operation view

In the following subsections the proceeding of trajectory planning, and tools to enhance visibility of important shapes or to verify the planing result are explained.

4.4.1 Trajectory Planning

Planning of a trajectory to the operation field in a minimally invasive procedure, a target area of injections, or setting of screws for spinal fusions are, by default, marker-based. That is to say, the user has to place a marker for the puncture and the target point of the virtual needle, dilator (tubular retractor) or screw. Those two markers can be defined via 2D mouse on the 2D image slices (CT or MRI)

¹<http://www.sensable.com/products-haptic-devices.htm>

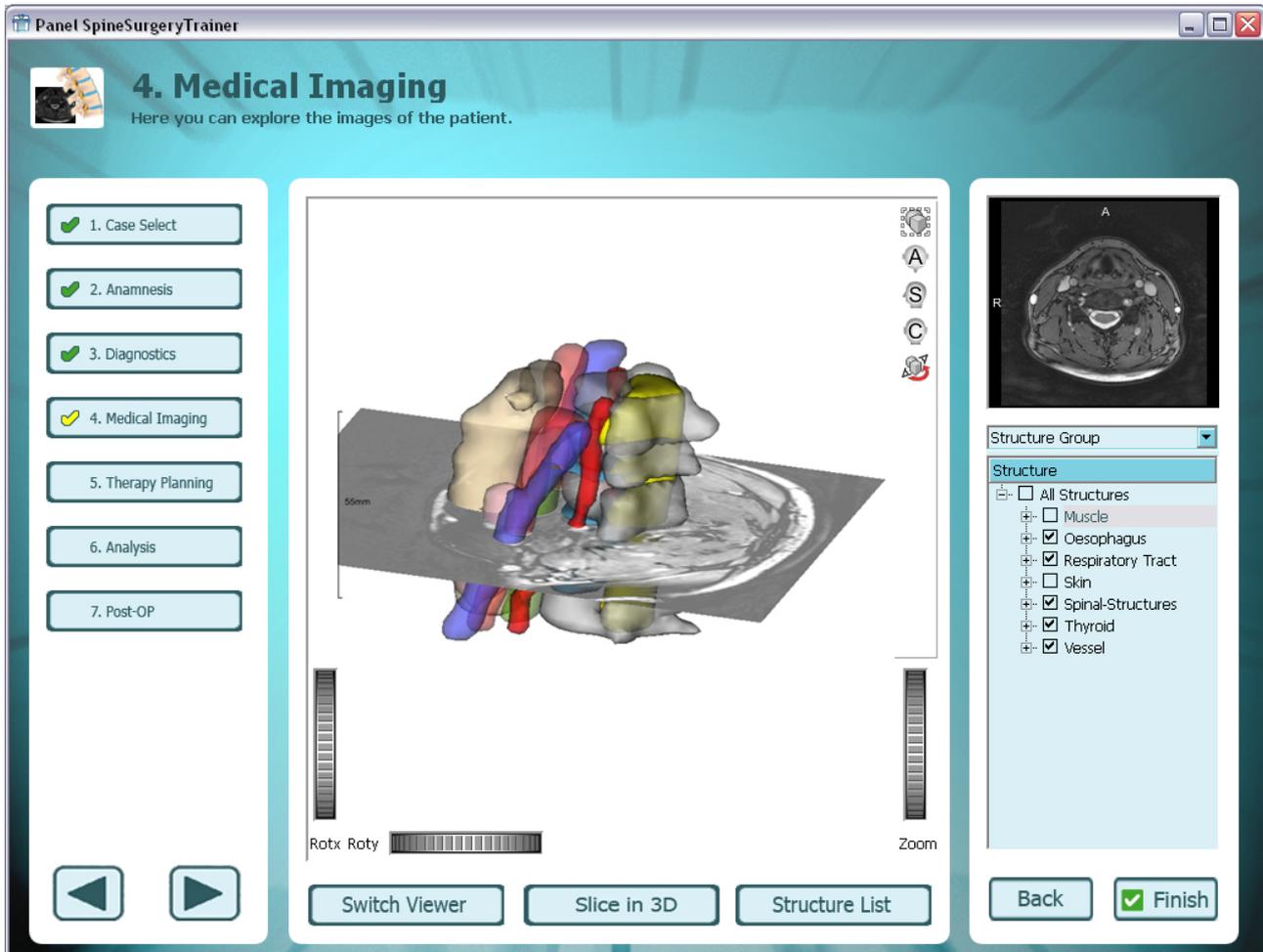


Figure 5: User interface for the exploration of medical image data and 3D model of the patient’s anatomy. The currently displayed slice (top-right) is integrated in the 3D scene of a cervical vertebrae dataset. The colored overlays in the slice view correspond to the 3D representations of segmented structures. Structures (or groups of structures) can be selected by the structure tree on the right side. The current training workflow is displayed on the left side, where jumping to individual steps is possible.

or on the reconstructed 3D model of the patient anatomy. An animation visualizes the puncture of the virtual puncture device in 3D along the resulting trajectory between the markers. Since it is only implicitly defined by the two markers, the estimation of this trajectory bears a high mental effort. Therefore, we provide a more intuitive alternative taking advantage of the haptic 6DOF input device (depicted in Figure 7) which is specified in [8]. Here, the puncture task is separated into three individual phases (placement of the device tip, orientation, and insertion), whereas each phase contains a specific set of haptic constraints to simplify their respective performance. While working on the image slices corresponds to the clinical routine, the 3D models provide a better perception of spatial relations.

4.4.2 Enhancement of Visual Perception

For better visibility of the target area or to check for possible injury risks, structures can be masked out or shown interactively via a tree structure (see the right side of Figure

5a). Additionally, we integrated the isoline-based method of our augmented reality module described at the end of Section 3 in order to optionally support the trainee with more depth cues without occluding important structures. In Figure 3, e.g., the shape of intervertebral discs is emphasized by a set of yellow contour lines. Thus, the shape and therefore the lumbar disk herniation is clearly visible within the virtual vertebrae and spinal canal and the video captured skin of the mannequin phantom.

4.4.3 Verification of User Planning

To verify the planning, the 3D scene including the virtual surgical instruments can be explored. An optional animation can also visualize the planned process. In case of a minimally invasive access planning, the animation conveys the whole process of extending the aperture with dilators to the point of the insertion of tabular retractors. For the purpose of verification as well as initial assistance, it is possible to automatically superimpose predefined insertion regions

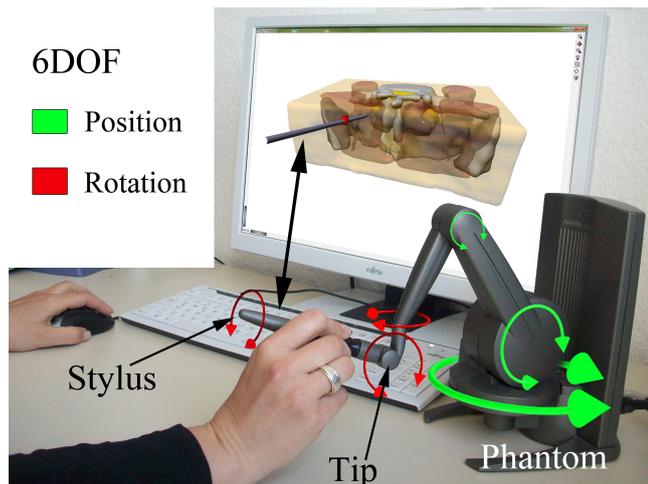


Figure 7: Manipulating the puncture device with a 6DOF PHANTOM Desktop[®] device (using force feedback).

on the skin, similar to the method in [19]. Furthermore, this gives an impression of the very limited operation field in spine surgery. Such insertion regions also designate regions in which risk structures like blood vessels might be injured, where impenetrable bony structures might hamper the opening, or regions with too long instrument paths or bad angles.

4.5 Utilizing Augmented Reality

In order to take advantage of and familiarize the trainee with our AR-based approach (recall Section 3), we combine both the intraoperative prototype and our virtual training prototype.

As stated in Section 4.3, we adapt the optimized slicing algorithm as an exploration tool, utilizing advantages of a 6DOF input device. A further step is to use it (as originally intended) with a video captured physical marker, as shown in the upper part of Figure 5b, using the prototype setup.

In the advanced stage of training, the trainee can familiarize with AR-aided therapy planning methods, like the one described in [15]. For this purpose, the case data of the training system is expandable by new but incomplete cases (e.g., without anamnesis). Thus, the MRI or CT data of locally available mannequins, cadavers or real patients with the relevant reconstructed polygonal 3D models can be explored and used to make a virtual therapy planning.

Using our AR-based prototype, the virtual planning results (e.g., insertion point and incidence angle of a puncture device) will be overlaid on the real patient body, mannequin or cadaver. Afterwards, the trainee performs the usual training on the real body object supported by the AR technique. Of course, any other virtual object can be mapped on the body as well. Similar to [15], our training system can also be used for real therapy planning, and using the augmented reality technique the physician will be supported in real surgery. In this way, the trainee can first develop planning strategies at a home PC/workstation before he/she starts to train the procedure with a training object. Supported by his/her own virtual planning results mapped on the real training object, the trainee learns not

only performing an intervention, but also the use of emerging intraoperative medical visualization technology.

4.6 User Guidance and Support

The user is (optionally) directed through the whole training process by offering information for each step on functionality and interaction possibilities (textually, and sometimes by videos). The user may also request expert support for both the treatment decision as well as for therapy planning. Thus, the trainee learns possible treatment suggestions and their respective indications from different experts. These are presented textually and by displaying the relevant 2D image slices as well as by a static or animated 3D visualization of the patient anatomy. During the therapy planning process, proposals of several experts are available in the form of videos of the experts' planning processes and the interactive explorability resulting 3D scenes. In the analysis step, the comparison with the planning of the experts is firmly integrated. The simultaneous presentation of the learner's therapy planning and processing and the recommendations of several experts enable the learner to check his/her results and get an impression of the variety of surgical and therapeutic strategies for a particular case. It is also planned to run the planning process of the experts as an interactive animation in order to communicate solution strategies and approaches.

5. IMPLEMENTATION

The system has been developed on an Intel[®] Xeon[®] quad-core Processor with 3.06 GHz, 8GB RAM and an NVIDIA GeForce GTX 460 graphics card with 768MB memory supporting 3D graphics. For the development of the virtual training system, the prototyping environment MeVisLab [16] was used, incorporating the visualization toolkit (VTK) for geometry handling and graphics rendering, and the Open Haptics Toolkit [20] for accessing the phantom device and haptics rendering.

We use a modular software design that fits to the framework of MeVisLab. Each step of the workflow, outlined in Figure 4, is implemented in one macro module with python script. With this design, the relevant software modules of the AR-assisted spine surgery system could be well incorporated into the training prototype.

The software modules (tracking, calibration, registration, slicing, and rendering modules) of the prototype for AR-assisted spine surgery have been implemented with C++, OpenGL, VTK, and Qt. For the current implementation, the tracking module relies on marker-based tracking provided by the ARToolkit, which allows for multiple marker tracking in real time. However, due to the inaccurate calibration, we calibrate camera parameters using MATLAB. As a result, the new calibration was significantly more accurate regarding marker detection and pose estimation.

6. RESULTS

Figure 5 depicts a snapshot of the GUI of the medical imaging module within the training prototype. With a few buttons the user can define the view that is represented in the main window and which exploration tools are used. In the screenshot, e.g., a plane widget is added to the 3D scene that represents the relevant medical image slice. The GUI is reduced to the suitable control elements or widgets in

respective to the content of the current main view and to the current training task (here exploration). Additionally to the current module interface, the training workflow is displayed continuously on the left side. Thus, the user keeps track of the current interaction possibilities. With the optional use of a 6DOF input device with force feedback, interaction tasks, such as the control of a virtual puncture device, are simplified, and the training is therewith improved.

The training system has been developed in cooperation with orthopedic surgeons, and it includes six cervical and eight lumbar vertebrae cases with indication for peridural injection and operative interventions. Modeling of single cases partially prerequisites clinical expertise and is extensive regarding data/material acquisition and processing, which includes anonymization, segmentation, generation of case variants, and editing of operation video clips. It is often necessary during the real operation of a case to decide whether it is suitable for the surgery training system.

As stated before, the hand-eye calibration requires minimal user interaction and is performed only once prior to the operation. Registration is similarly a one-time step that is performed during setting up the system. The slicing algorithm allows for on-the-fly computation and rendering of slices at a near real-time rate.

For the OP training scenario, a phantom model of the upper body is scanned with the mounted reference marker. After extrinsic calibration of the video camera, the mannequin phantom model is registered to the scanned data, using the corners of the reference marker as the set of correspondence point pairs. From a co-registered patient dataset, three vertebrae, intervertebral discs, and the spinal canal are segmented and 3D models are reconstructed. Finally, the visualization module starts the video stream augmentation.

Figure 1b depicts a snapshot of the GUI of the visualization module in a simulated spine surgery scenario. The widget shows a right posterior oblique (RPO) view, with augmented models of the lumbar vertebrae L2-L4 (cyan), intervertebral discs (green), and spinal canal (pink). Object silhouettes are slightly highlighted for enhanced shape perception. In Figure 8, a side view with an additional transparent overlay of a tomographical slice from the patient data is shown.

7. CONCLUSION AND FUTURE WORK

The SpineSurgeryTrainer is a case-based interactive training system, which provides training for the complete clinical workflow in spine surgery. With the current prototype future surgeons are able to acquire and improve their anatomical knowledge concerning spinal and surrounding structures, and they can train diagnosis and therapy decisions on the basis of several cases and case variations. Available expert recommendations and explanations as well as the analysis of the planning results help the user to plan the therapy self and to understand the planning processes of the experts. The difficulty varies according to the optionally provided assistance and expert proposals. An AR extension does not only support the trainee performing a planned intervention on a real training object, but also the use of AR technology in a real surgery. We aim at raising the safety and accuracy factors of the intervention by superimposing relevant operation-specific anatomical structures on the video stream of the operation field, which obviously allows for the protection of risk structures like spinal nerves, blood ves-

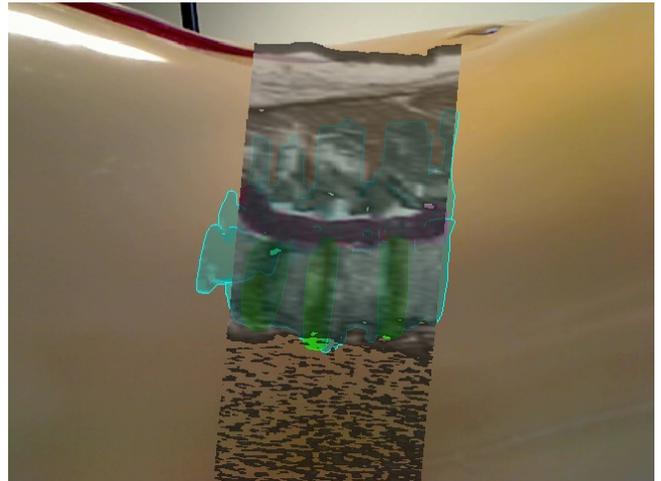


Figure 8: Side view with augmented vertebrae (cyan), discs (green), and spinal canal (pink) in addition to a blended MRI cross section.

sels, and the spinal canal. Thus, we introduced a prototypic tablet PC based add-on for enhancing intraoperative visualization, with the focus on spine surgeries. In addition, targeted structures like tumors or herniated discs are better localized and information about the shape and extent of such structures can be conveyed at different depth levels. From a surgical point of view, we have received positive feedback regarding the relevance and applicability of the presented approach.

Our future work concerning the intraoperative visualizations will focus on evaluating the applicability of the concept to other surgery scenarios. In a following development phase of the Spine Surgery Trainer, the effectiveness of further interaction tools, e.g., distance measurements or virtual removal of bony structures, will be examined. In addition, the system will be extended with further scenarios (e.g., placement of artificial discs or implants) and new operation cases. Therewith, a representative set of different planning processes will be provided and the trainee would obtain an overview of the possible disease patterns and treatments. Otherwise, a continuous evaluation of the systems regarding the usability and learning success will be conducted.

Acknowledgments

The intraoperative prototype utilizing augmented reality visualization is funded by the German Ministry of Education and Science (BMBF) within the ViERforES project (no. 01IM10002B). We also want to thank MeVis Medical Solutions for providing MeVisLab.

8. REFERENCES

- [1] W. Birkfellner, M. Figl, K. Huber, F. Watzinger, F. Wanschitz, J. Hummel, R. Hanel, W. Greimel, P. Homolka, R. Ewers, and H. Bergmann. A Head-Mounted Operating Binocular for Augmented Reality Visualization in Medicine-Design and Initial Evaluation. *IEEE Transactions on Medical Imaging*, 21(8):991–997, 2002.
- [2] T. Coles, N. John, D. Gould, and D. Caldwell. Integrating haptics with augmented reality in a

- femoral palpation and needle insertion training simulation. *IEEE Transactions on Haptics*, 4(3):199–209, 2011.
- [3] J. Cordes, K. Hintz, J. Franke, C. Bochwitz, and B. Preim. Conceptual Design and Prototyping Implementation of a Case-based Training System for Spine Surgery. In S. Hambach, A. Martens, and B. Urban, editors, *Proc. of the 1st International eLBA Science Conference (e-Learning Baltics)*, pages 169–178, 2008.
- [4] S. DiMaio and S. Salcudean. Interactive simulation of needle insertion models. *IEEE Transactions on Biomedical Engineering*, 52(7):1167–1179, 2005.
- [5] P. Edwards, A. King, C. Maurer, D. de Cunha1, D. Hawkes, D. Hill, R. Gaston, M. Fenlon, S. Chandra, A. Strong, C. Chandler, A. Richards, and M. Gleeson. Design and Evaluation of a System for Microscope-Assisted Guided Interventions (MAGI). *IEEE Transactions on Medical Imaging*, 19(11):1082–1093, 2000.
- [6] M. Fischer. CASUS: An Authoring and Learning Tool Supporting Diagnostic Reasoning. *Zeitschrift für Hochschuldidaktik*, pages 87–98, 2000.
- [7] I. Kalfas. *Minimally Invasive Spine Surgery: A Practical Guide to Anatomy and Techniques*, chapter Image-Guided Spinal Navigation: Principles and Clinical Applications, pages 7–22. Springer New York, 2009.
- [8] K. Kellermann, M. Neugebauer, and B. Preim. A 6DOF Interaction Method for The Virtual Training of Minimally Invasive Access to the Spine. In *Proc. of CURAC*, 2011. Accepted for publication.
- [9] U. Kühnapfel, C. Kuhn, M. Hübner, H. Krumm, H. MaaSS, and B. Neisius. The Karlsruhe Endoscopic Surgery Trainer as an Example for Virtual Reality in Medical Education. *Informa Healthcare: Minimally Invasive Therapy & Allied Technologies*, 6(2):122–125, 1997.
- [10] R. Lapeer, M. Chen, G. Gonzalez, A. Linney, and G. Alusi. Image-Enhanced Surgical Navigation for Endoscopic Sinus Surgery: Evaluating Calibration, Registration and Tracking. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 4(1):32–452, 2008.
- [11] J. Merriënboer, R. Clark, and M. DeCroock. Blueprints for complex learning: The 4C/ID-model. *Educational Technology, Research and Development*, 50:39–64, 2002.
- [12] K. Mühler and B. Preim. Günstige Kamerapfade für medizinische Animationen. In *Bildverarbeitung für die Medizin (BVM)*, pages 46–50, 2010.
- [13] N. Navab, M. Feuerstein, and C. Bichlmeier. Laparoscopic virtual mirror - new interaction paradigm for monitor based augmented reality. In *Virtual Reality*, pages 43–50, 2007.
- [14] B. Ockert, C. Bichlmeier, S. Heining, O. Kutter, N. Navab, and E. Euler. Development of an Augmented Reality (AR) training environment for orthopedic surgery procedures. In *Proc. of CAOS*, 2009.
- [15] L. D. Paolis, G. Aloisio, and M. Pulimeno. An Augmented Reality Application for the Enhancement of Surgical Decisions. In *Proc. of the The Fourth International Conference on Advances in Computer-Human Interactions*, 2011.
- [16] M. Research. MeVisLab Home Page. URL: <http://www.mevislab.de>, 2011.
- [17] Z. Salah, B. Preim, and G. Rose. An Approach for Enhanced Slice Visualization utilizing Augmented Reality: Algorithms and Applications. In *Proc. of PICCIT*, 2010.
- [18] Z. Salah, B. Preim, A. Samii, R. Fahlbusch, and G. Rose. Enhanced Interoperative Visualization for Brain Surgery: A Prototypic Simulation Scenario. In *Proc. of CURAC*, pages 125–130, 2010.
- [19] A. Seitel, M. Engel, M. Fangerau, K. Fritzsche, K. Yung, H. Meinzer, and L. Maier-Hein. Computer-Assisted Trajectory Planning for Percutaneous Needle Insertions. *Medical Physics*, 38(6):3246–3259, 2011.
- [20] SensAble. *OpenHaptics® Toolkit 3.0 Programmer's guide*. SensaAble Technologies, Inc., 2008.
- [21] X. Yongming, P. Heng, Z. Jishuai, L. Ping, Z. Tanchao, and L. Jianying. A Simulation System for Training Telerobotic Spine Surgery. In *International Conference on Electric Information and Control Engineering (ICEICE)*, pages 102–105, 2011.

Optimization of Vascular Surface Models for Computational Fluid Dynamics and Rapid Prototyping

Tobias Moench

Mathias Neugebauer

Bernhard Preim

Otto-von-Guericke University Magdeburg
 Faculty of Computer Science, Department of Simulation and Graphics
 Universitaetsplatz 2, 39016 Magdeburg, Germany
 {tobias.moench, mathias.neugebauer, bernhard.preim}@ovgu.de

ABSTRACT

The generation of surface models for computational fluid dynamics and rapid prototyping implies several steps to remove artifacts caused by image acquisition, segmentation, and mesh extraction. Moreover, specific requirements, such as minimum diameters and distances to neighboring structures are essential for rapid prototyping. For the simulation of blood flow, model accuracy and mesh quality are important. Medical expert knowledge is often required to reliably differentiate artifacts and pathological malformations. Currently, a number of software tools needs to be employed to manually solve the different artifact removal and mesh editing tasks. Within this paper, we identify the related tasks and describe the procedure, which is used to receive artifact-free vascular surface models for these applications.

Categories and Subject Descriptors

I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—*Curve, surface, solid, and object representations*

Keywords

Model Generation, CFD, Simulation, Rapid Prototyping, Optimization, Reconstruction

1. INTRODUCTION

Vascular diseases, such as the coronary heart disease or cerebral and abdominal aneurysms are severe, often life-threatening diseases. Initiation and progress of these diseases are not fully understood. Many research projects clearly show, that diseases primarily occur at regions of complex and instable blood flow. This gives rise to blood flow simulations of specific patient data and eventually also including implants to predict treatment effects. Such simulations are based on many assumptions. Experimental validation based on a large variety of physical phantoms is essential to investigate the reliability of such simulations. Applications, such as rapid prototyping (RP) or computational fluid dynamics (CFD) for blood flow simulation require faithfully reconstructed surface models as input (see Fig. 1). Vascular surface models are generated from tomographic medical image data (e.g., computed tomography (CT) or magnetic resonance imaging (MRI)). Depending on the employed segmentation and mesh generation methods, an initial vascular surface model may contain several artifacts. The causes of these artifacts are various: During image ac-

quisition, low resolution, partial volume, and beam hardening effects as well as inhomogeneous or insufficient contrast agent distribution disturb the representation of the vessel lumen. As a consequence, vessels might occur locally narrowed or interrupted. Furthermore, beam hardening can cause small vessels to visually blend with closely located, bigger, high contrast vessels. These artifacts vary to a certain degree, e.g., such that adjacent or outgoing vessels are represented as high frequency noise on the surface of a bigger vessel. Other artifacts, like staircases, may occur during mesh generation. Thus, e.g., noise, staircases, blended vessels, or abruptly changing vessel diameters need to be removed faithfully.

These tasks are, considered separately, not very complicated and can often be handled with specific tools and algorithms. However, the overall process to achieve appropriate models for CFD or RP requires a lot of manual effort in different software tools. This, however, may alter patient-specific properties of the target structure. Thus, medical experts need to be involved to validate the intermediate model adjustments as well as the final surface model. In this paper, we identify the problems and tasks occurring during generation of vascular surface models in the different steps of the model generation pipeline. Furthermore, we discuss current solutions, which involve multiple software tools to overcome the model generation difficulties.

2. PRIOR AND RELATED WORK

Surface models are usually derived from tomographic image data, acquired via MRI or CT. For vascular structures, special acquisition techniques (e.g., magnetic resonance angiography (MRA), computed tomography angiography (CTA), rotation angiography, time-of-flight MR angiography) are employed to receive a high contrast to the surrounding tissue. This eases a direct mesh extraction from the intensity data, but may still yield several artifacts in the resulting surface meshes.

2.1 Generation of Medical Surface Models

By employing manually, semi-automatic (e.g., thresholding, region-growing, ...), or fully automated segmentation methods (e.g., [7]), the target structures can be delineated and finally transformed into a surface mesh via, e.g., Marching Cubes (MC) algorithm [19], or level-set methods [31]. Specialized vessel surface reconstruction methods, such as MPU implicits [27, 28] or Convolution Surfaces [23], are available. Convolution Surfaces, however, are model-based

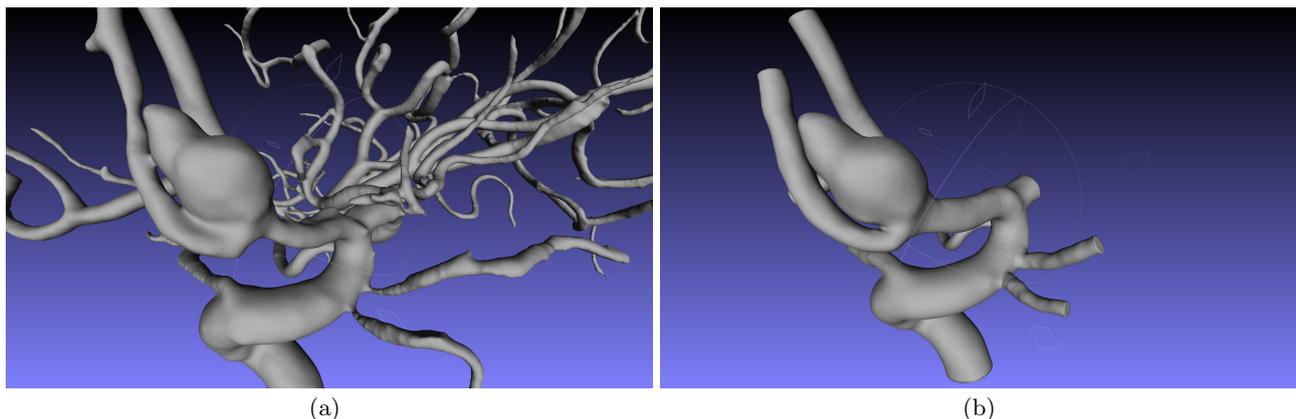


Figure 1: (a) Example of an initial vascular surface model with an aneurysm and several artifacts at the branches. (b) Result of the complex model generation and artifact correction pipeline.

and require a vessel skeleton and diameter as input. Thus, they are not suitable to represent pathological vascular structure, such as aneurysms. In contrast, MPU implicits can be used to describe vessels with pathological deviations but use point clouds from binary segmentation masks as input. The intensity information of the original image data is ignored, which makes MPU implicits less accurate and partially sensitive to artifacts from image inhomogeneities. Similarly, Wu et al. [32] employ point clouds from binary masks for generating a 3d implicit indicator function, which is subsequently used as input for polygonization. Such specialized model generation methods for vascular structures may help to reduce some artifacts, as they guarantee a certain smoothness and vascular shape regarding the vessel profile, but also at branching points. Some issues, such as incomplete contrast agent dispersal or touching vessels, may also not be removed. There are several specific methods available, e.g., that detect features and adjust sampling of the data [17], that apply an additional trilinear interpolation and subdivision to the surface elements (Precise MC [1]), or that relax an initial surface iteratively with additional position constraints (e.g., Dual MC [22], Constrained Elastic Surface Nets (CESN) [9, 14]). These methods are, however, only specific solutions, especially to the staircase problem, but can not remove all potential artifacts.

2.2 Artifact Reduction

The usage of segmentation masks for mesh generation yields strong staircase and terracing artifacts [6, 21], especially for image data with anisotropic voxel dimensions. By applying segmentation masks to the intensity data to remove irrelevant neighboring structures, local staircases may emerge. For the reduction of staircases, several methods are available, which, e.g., interpolate intermediate slices via shape-based interpolation [25] or apply smoothing filters on the mesh level (e.g., Laplace, Laplace+ HC [30], LowPass [29], Mean Curvature Flow [13] filtering). Even local staircases can be removed systematically [20]. Especially vascular surface models may contain artifacts occurring, e.g., from incomplete contrast agent dispersal resulting in local narrowing or frayed parts. Moreover, beam hardening artifacts and closely located vessels may yield unwanted blend-

ing of separated structures (see Fig. 5) [20]. Such artifacts are usually not treated for pure visualization tasks. Thus, there are no specialized methods available to remove, e.g., blending artifacts or narrowed vessels faithfully.

The VASCULAR MODELING TOOLKIT (VMTK) [2, 3, 24] is a framework for the reconstruction and geometric analysis of vascular structures. However, methods for systematic removal of specific artifacts are not included. To receive an appropriate surface mesh for CFD or RP, several tools and expert feedback need to be employed (see Fig. 1) to manually remove artifacts.

2.3 Application in CFD and Rapid Prototyping

For the patient-specific simulation of blood flow, further requirements, such as mesh resolution and triangle quality play an important role [11, 12]. The shape of the mesh's triangles is required to be almost equilateral and homogeneous over the mesh, since the surface models are used as input for volume mesh generation. Thus, triangle quality and size influence convergence and accuracy of the simulation computations. An optimization can be achieved, e.g., by employing an advanced front remeshing algorithm [26]. The surface generation method by Wu et al. [32] does also take care of mesh quality during polygonization of their implicit function, which may reduce the number of steps to be taken within the model generation pipeline. Augsburg et al. [5] have shown, that the mesh extraction procedure may yield strongly varying simulation results depending on the involved segmentation and artifact reduction methods. Similarly, Cebra and Löhner [10] describe the strong dependence of blood flow characteristics on the vessel geometry. RP is employed for different applications, such as treatment planning, simulation, measurement tasks, or even teaching of interventional techniques. Knox et al. [16] presented several examples of RP applied to pathological vascular structures for such purposes. Available literature, however, focuses more on the fabrication process [18, 4] than surface model generation. Thus, requirements for generating surface models are rarely specified.

The above mentioned artifacts and tasks are well-known. There are several methods available to account for individual

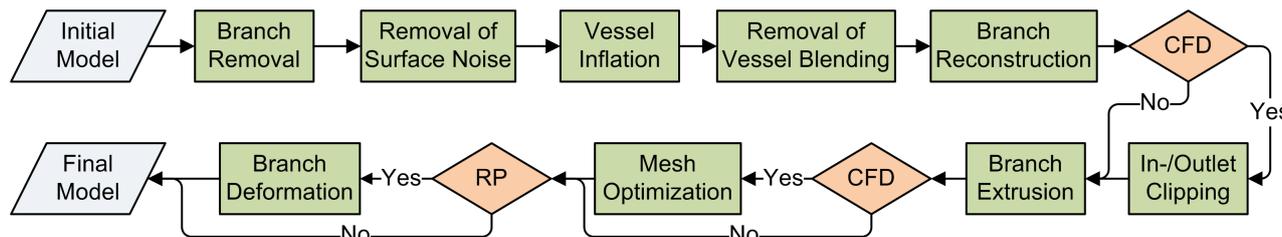


Figure 2: The employed surface optimization pipeline.

artifacts, but, in several cases, only medical experts can distinguish between an artifact and a pathology. There exists, however, no unified solution that allows users to generate an artifact-free surface model for the desired application.

3. SURFACE OPTIMIZATION PIPELINE

The reduction and removal of artifacts within the model generation procedure for CFD or RP raises several tasks. We will discuss the individual tasks for exemplary data of a cerebral aneurysm, acquired via CT angiography. The initial surface model has been extracted after thresholding of the contrast-enhanced image data via MC. Subsequently, after applying a connected component analysis, we receive the initial surface model of the target vessel with all connected vessel branches and several artifacts. In order to identify all artifacts reliably and distinguish them from real anatomical malformations, an expert-driven, iterative manual optimization process is necessary. Additional steps are required to make the vascular surface model suitable for application in CFD and RP.

3.1 Requirements

The generation of vascular surface models for visualization has extensively been studied, even under consideration of model assumptions. The requirements of pure visualization tasks, however, differ from those of CFD and RP. The simulation of blood flow in CFD primarily requires accuracy and quality of the surface (smoothness, triangle quality). To achieve reliable simulation results, even small surface artifacts need to be removed and size and shape of the mesh triangles usually need to be improved. Mesh generation for CFD has also been done extensively, in particular regarding mesh triangle quality and mesh size. However, no typical artifacts have been considered.

Physical vessel models created by RP can be used for various tasks, from teaching to treatment planning. A major application is to build phantoms. For instance, inverse transparent silicon models are used to experimentally simulate blood flow. Optical velocimetry methods are applied to gain information about the complex flow patterns within the different vessel configurations [8]. A surface reconstruction of the vessel serves as input for most of the RP techniques. However, some of the surface features cannot be physically reproduced due to procedural- and material-related constraints.

First, a mold needs to be constructed and subsequently be filled with a low melting material. The resulting cast is enclosed by silicon and finally the cast is removed from the resulting silicon block by melting it. Depending on the specific material, the final inverse silicon model and all intermediates of this process can represent surface details only

to a certain degree. Additionally, it must be ensured that the mold can be opened without destroying the cast. In turn, it must be possible to remove the cast, without melted parts remaining in the silicon block. Thus, in order to reconstruct a physical model from the reconstructed surface, it must satisfy certain constraints, e.g., adequate distances between adjacent surface parts, no strong bending angles of vessels and the possibility to define a more or less planar cutting plane through the whole vessel reconstruction. Therefore, in some cases, it is necessary to deviate from the patient-specific vessel representation and to perform local adaptations. These adaptations, however, need to be performed carefully to ensure that the results are plausible from a medical point of view. Otherwise, the results of phantom tests and measurements might be useless. This delicate task needs to be performed with the help of an expert, in an iterative process balancing between anatomical correctness and producibility of the physical model.

These requirements lead to the following pipeline (see Fig. 2): starting with a mesh of the vascular tree and the aneurysm, unnecessary distant vessel branches or branches that are - in terms of flow direction - located behind the aneurysm, are removed. Subsequently, vessel blending artifacts, underestimated vessel diameters and surface noise are corrected. If necessary, insufficiently represented branches that are necessary for CFD or RP are reconstructed. After faithfully reconstructing the vessel surface, optimization for CFD and RP has to be performed. In-/outlets need to be cut perpendicular to the vessel centerline. In some cases, the in-/outlets have to be elongated artificially. The mesh quality has to be improved by employing a remeshing algorithm. After this procedure, the model is suitable for CFD.

For usage as RP input, the model may need to be altered, e.g., if surfaces are too close, if the vessel bending is too strong, or if it is not possible to apply a more or less planar cut through the complete model. These alterations are done by locally changing vessel diameters or bending vessels.

Details of each step are presented in the following sections, where we focus on artifact reduction purely on the mesh level. Alternatively, artifacts could be reduced by modifying the segmentation mask and the image data, which, however, may be more complicated and might introduce further artifacts [20].

3.2 Removing Branches

In CFD, each additional branch with in-/outlets increases computational effort. Thus, branches that do not directly affect the blood flow behavior in the target area (e.g., the aneurysm), need to be removed. Currently, branch removal is achieved by a combination of different software tools. At first, the vessel is clipped with 3d modeling software (e.g.,

BLENDER¹ or 3D STUDIO MAX²). Such 3d modeling software usually provides a large set of tools to modify 3d models. However, to remove the branch, a clipping geometry (e.g., a cube, a plane, ...) needs to be specified. After adjusting position and orientation of the clipping geometry, Boolean operations are applied to remove the negligible branches, resulting in a slight, flat bump on the main vessel and a closed surface.

Afterwards, this bump can be reduced by using SCULPTRIS³ to iteratively and locally smooth the surface (see Sec. 3.3). Alternatively, branch removal can be performed in a more offensive way by cutting the thin vessel directly at the branching point on the larger main vessel. Cutting out this branching area leaves a hole in the main vessel which needs to be closed afterwards. This is again achieved via BLENDER. Depending on the size and profile of the cutting area, the closed hole may be subject to manual local deformation via SCULPTRIS.

3.3 Noise and Bump Removal

After initial mesh extraction and branch removal, the surface may contain vessel rudiments (see Fig. 3(a)) due to incomplete contrast agent dispersal or beam hardening. A possible solution are smoothing filters. However, typical uniform smoothing will cause strong volume shrinkage of the whole model and removes relevant details. Especially for vascular structures, locally adaptive filters are necessary, to focus smoothing to the artifact areas only. Since the artifacts being target for smoothing operations may vary in their shape and size, an automated approach detecting the artifacts reliably is very complicated. Thus, the most useful solution is an interactive approach where the user brushes over the artifact area. During brushing, all vertices in a defined neighborhood (topological or Euclidean distance) are smoothed appropriately. To achieve this, we employed SCULPTRIS, which provides several mesh brushing tools for dilation, extrusion and smoothing of surface meshes. For removal of bumps, the local smoothing operator can be used, which is parametrized by the operator size and strength.

3.4 Vessel Inflation

Locally narrowed vessels (see Fig. 4(a)) need to be adjusted for two reasons:

1. The geometry of the structure is incorrect and may thus yield wrong conclusions during visual inspection or during exploration of resulting RP models.
2. For usage in CFD, non-converging simulations or wrong simulation results have to be expected for, e.g., wall-shear-stress and flow velocity, but also vortices may occur which, altogether, may influence the flow behavior within the whole model.

To resolve this, the vessel needs to be inflated locally (see Fig. 4(b)). The artifact areas can basically be detected automatically by generating the vessel centerlines and comparing the behavior of vessel diameters along the centerlines. This implies a circular vessel shape, which must not be true in all cases. Vessels can be slightly flattened due to pressure

¹<http://www.blender.org/>

²<http://usa.autodesk.com/3ds-max/>

³<http://www.sculptris.com/>

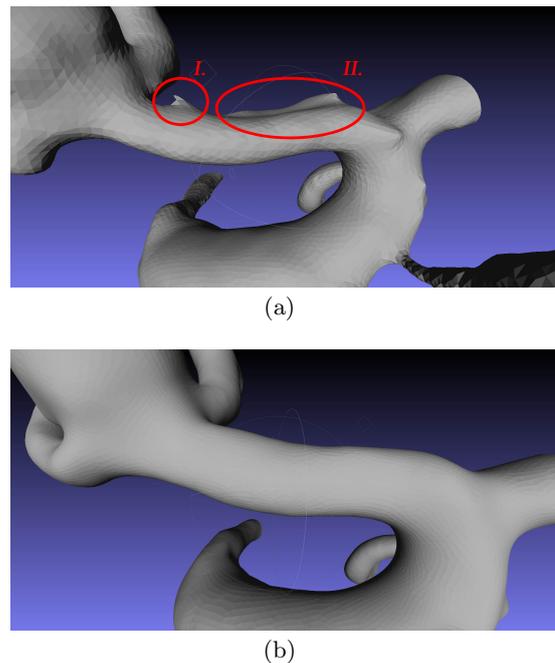


Figure 3: Example of rudimentary vessel branches and slight vessel blending (see Labels I. and II.) (a) before and (b) after local smoothing. Screenshots of MeshLab.

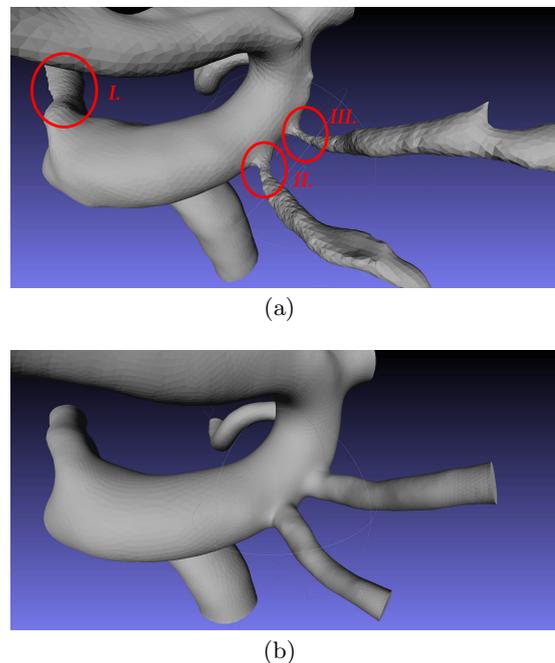


Figure 4: (a) Thin, anatomically incorrect vessel branches (see Labels I., II., and III.). (b) The vessels after local inflation and branch clipping. Screenshots of MeshLab.

from surrounding structures or they may contain pathological variations. Thus, an automated procedure may be error-prone, but could be used to support the user to identify these areas faster. After identification of possibly narrowed vessels, it is essential to refer to the image data again to validate the narrowing before further correction.

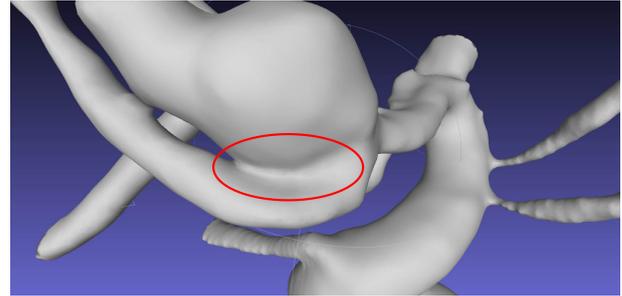
The narrowing artifact can again be corrected by using SCULPTRIS with a combination of the provided “Inflate” and “Smooth” tools. For both tools, operator size and strength need to be adjusted to fit to the size and diameter of the target vessel.

3.5 Removal of Vessel Blending

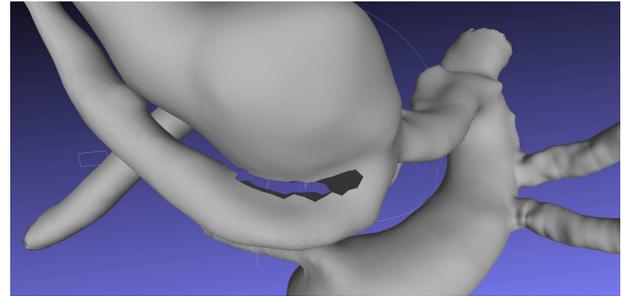
Vessel blending artifacts (see Fig. 5(a)) may arise locally restricted at touching vessels, but also very expanded, if a vessel passes another vessel over a long range. In particular, the latter is a critical situation, since the involved vessels are visually hard to distinguish and thus hard to divide faithfully.

The separation of blended structures is a complex problem whose specific solution depends on the data and the extent and shape of the artifact. Once more, 3d modeling software is employed to perform the mesh editing tasks. Via BLENDER, the mesh can be cut along the desired path to split the blended vessels. This process may be tedious, since the cutting path can be complex and needs to be drawn precisely on all sides of the artifact (see Fig. 5(b)). Especially finding an appropriate alignment of the cuts on the front and back side may be very complicated. Since the resulting hole needs to be closed, we added support triangles manually. The support triangles are added at critical points, where the “shape” of the hole changes significantly. By adding those triangles, the hole may be divided into several less complex parts, which eases final hole filling. Hole filling is done via MESH LAB⁴, which detects and closes the holes automatically. For complex artifact shapes, this cutting and hole filling procedure may not be efficient anymore. As an alternative, we perform a stamping-like procedure in BLENDER, where we generate a stamping geometry (e.g., a cylinder or cuboid), which is then aligned with the artifact. This gives, however, a good preview of the resulting holes in the target model. After correct placement and slight adjustment of the stamping geometry, the artifact is cut using constructive solid geometry (CSG). CSG employs Boolean operations and yields correctly closed surface meshes. Depending on the artifact shape, this needs to be repeated several times but does still save a lot of effort, since the manual specification/drawing of support triangles and subsequent hole filling can be neglected.

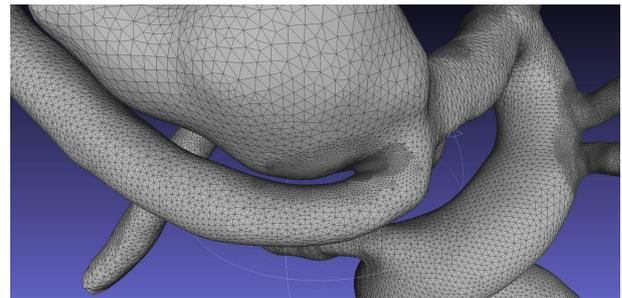
After applying one of the above described artifact removal operations, local adjustments via SCULPTRIS may still be necessary. As a prerequisite for usage in SCULPTRIS, local remeshing and subdivision may be necessary (e.g., via BLENDER). Applying local, interactive smoothing or inflation, the artificial vessel surface is modified to achieve a plausible vessel shape and profile. Furthermore, for usage in RP, the distance between the separated vessels needs to be considered to prevent an anew blending during physical model building (see Sec. 3.10).



(a)



(b)



(c)

Figure 5: Example of vessel blending and removal of the artifact. (a) Initial model with blending artifacts, (b) after manual cutting, and (c) after hole filling and smoothing. Screenshot of MeshLab.

3.6 Branch Reconstruction

Incomplete contrast agent dispersal and segmentation may also yield detached vessels. If such detached parts are essential for further evaluation of, e.g., the flow behavior, they need to be reconstructed to allow for a faithful virtual representation of the specific patient anatomy. The reconstruction of single branches requires a lot of manual effort, since two vessel rudiments need to be reconnected and thus be modified manually. For each vessel rudiment, cutting operations (and possibly local deformations) are necessary in order to reconstruct a valid vessel profile. These open vessel profiles may then be connected in different ways:

1. They may be extruded until they match each other, whereas it is unlikely that both endings will perfectly match after a linear extrusion and extremely stretched triangles may occur (see also Sec. 3.8). Thus, a plausible deformation (see Sec. 3.10) and local remeshing are also required to approximate a vessel geometry.

⁴<http://meshlab.sourceforge.net/>

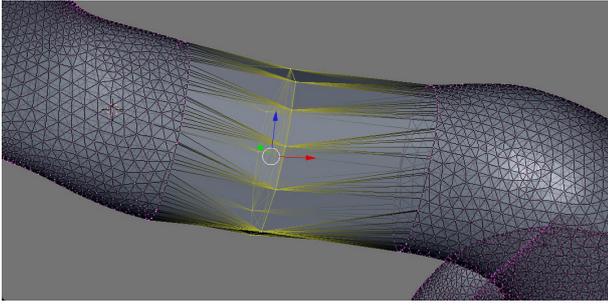


Figure 6: Screenshot of Blender during reconstruction of a formerly disconnected vessel branch.

- Both open vessel rudiments may be connected via an artificial vessel geometry, e.g., a tube. The endings of the tube have to be fitted to the open vessel profiles and their vertices require correct merging. Again, the tube needs to be deformed to fit the centerline of the real vessel geometry.

We employ BLENDER to perform these operations (see Fig. 6). The procedure involves a lot of interaction to select the correct vertices and to drag them towards their new position. Along the newly reconstructed vessel geometry, the centerline needs to be adjusted to plausibly fit into the centerlines of the former vessel rudiments. At this point, an automated procedure could support the user by interpolating the new vessel centerline. This would allow to extrude the vessel rudiments without necessity of subsequent deformation. For the tube geometry approach, the centerline of the tube could be fitted along the interpolated vessel centerline automatically. Besides a deformation along the vessel centerline, slight deformations of the vessel profile may also be necessary, e.g., if the connected real vessel geometry does not exhibit a perfect circular shape. For that, again, SCULPTIS can easily be used by using the provided brushing tools for inflation, deflation, and smooting.

As a last step, the vertices at the “touching” open vessel profiles need to be merged for a correct triangulation. Currently, this is also done manually in BLENDER.

3.7 In-/Outlet Clipping

For application in CFD, in-/outlets need to be specified (see Fig. 4(b) and 8(c)) to define inflow and outflow behavior. Similar to branch removal in Section 3.2, we use the 3d modeling software BLENDER for generating in-/outlets. We create a clipping geometry and adjust its location and orientation iteratively according to the vessel centerline (see Fig. 8(b)). This process has to be repeated several times for all contained vessel branches. Finally, Boolean operations are performed to clip the vessel perpendicular to the centerline (see Fig.8(c)). It has to be ensured that clipping yields a closed surface mesh.

User interaction could be supported by an automated alignment of clipping volumes perpendicular to the centerline, where the user only needs to drag each clipping volume along the centerline. Such functionality is, however, typically not contained in the available 3d modeling software.

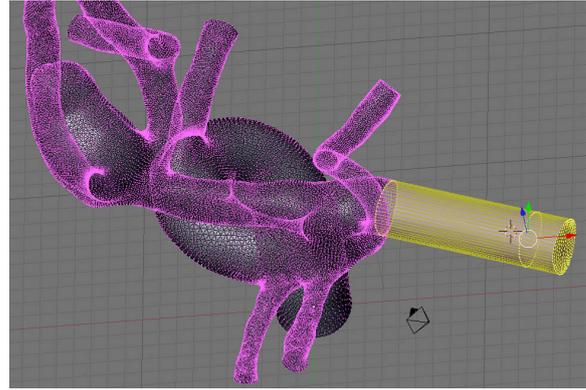


Figure 7: The in-/outlet area is selected and extruded to fit the requirements of CFD for a minimum vessel length at the in-/outflow areas. Screenshot of Blender.

3.8 Branch Extrusion

For CFD, the in-/outlets require a minimum length of the adjacent vessel. The inflow and outflow areas may not be directly adjacent to bent vessel parts to achieve more stability during simulation. This task is currently solved via BLENDER. The surface mesh needs to be edited manually by selecting the in-/outlet area of the vessel and subsequently extruding it (see Fig. 7). This procedure involves a lot of manual effort but could be supported algorithmically, if centerline information is involved. Thus, the user might only drag the in-/outlet along the (automatically extrapolated) centerline or its own average surface normal. During this elongation operation, it is, however, not sufficient to simply move the vertices of the in-/outlet areas, since this will result in extremely stretched triangles along the vessel surface. For the resulting tubular structure, local remeshing is required to guarantee similar mesh properties compared to the initial surface model.

Besides CFD, RP may also require branch extrusion. This requirement arises from the phantom building procedure, where a mold is created and needs to be filled. Thus, the vessel extrusion is used as casting channel. In such a case, the extrusion may not match further anatomical requirements.

3.9 Mesh Optimization

For application in CFD, high mesh quality in terms of a good triangle edge ratio and homogeneous triangle size need to be ensured for stability and convergence of the simulation computations. To remesh the modified surface model, we employ NETGEN [26]. First of all, feature edges, which have to be preserved during remeshing, need to be detected and highlighted. This is essential for the feature edges at the earlier specified in-/outlets. Feature detection is performed semi-automatically by adjusting two parameters until the visual result (edge highlighting) fits the user’s requirements. After user interaction and specification of the desired mesh granularity, NETGEN automatically proceeds with an advancing front algorithm. As a result, the complete surface model has been remeshed, whereas the feature edges at the in-/outlets are have been maintained.

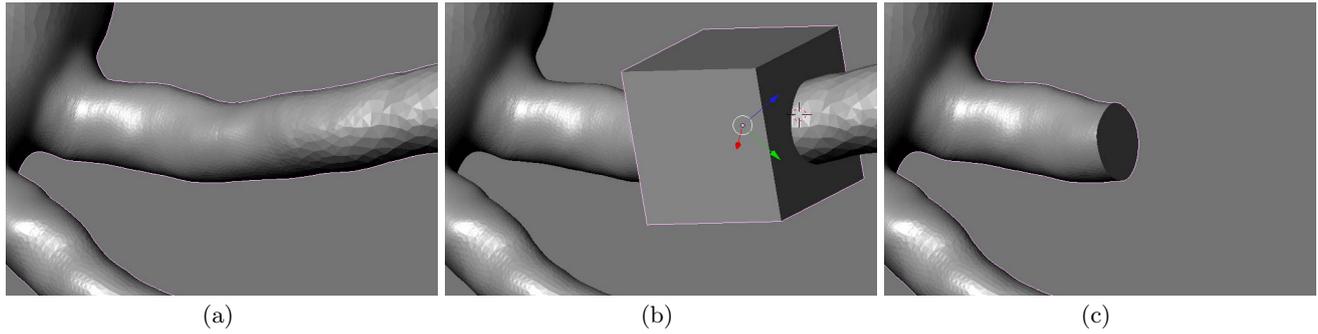


Figure 8: Generation of an in-/outlet: (a) Initial branch; (b) Vessel with clipping box; (c) Result after clipping and removal of the separated part. Screenshot of Blender.

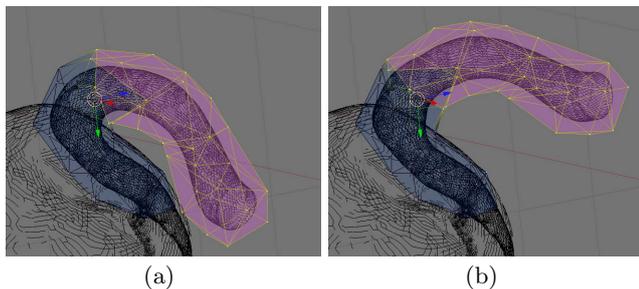


Figure 9: Example of vessel branch deformation via hull volumes. (a) Before deformation and (b) after deformation. Screenshots of Blender.

After performing the tasks described in the previous sections, the surface model is ready for subsequent generation of a volume mesh required for CFD.

3.10 Branch Deformation

Usually, the most important requirement during surface model generation is *accuracy*. Especially for usage in surgical planning, radiation treatment or CFD simulation it is prohibited to alter the shape of the target structure. In contrast, RP may require bending of closely located vessel branches to prevent blending during phantom building. For deformation of vessel branches, we employed the software BLENDER, which provides deformation via hull volumes and harmonic coordinates [15]. Thus, a hull volume is generated which is then used to control the deformation of the inner target structure. In our case, we generate the hull volume for all vessel branches that are located too close to other parts of the model. The structure can then be deformed via mouse interaction until it fits the specific local demands. The underlying algorithm guarantees that volume and local properties of the vessel are preserved while the global shape is modified (see Fig. 9).

4. CONCLUSION

The reconstruction of vascular surface models for applications, such as CFD or RP, involves a number of different and complex steps. For CFD, accuracy and mesh quality play an important role. In contrast, RP may even require local deformations to account for requirements from the physical model building process. Within this paper, we discussed

these requirements and the tasks arising for solving the specific problems. For generating vascular surface models, no automated procedure can be used, since the occurring artifacts require an extensive, often interactive treatment and expert knowledge to distinguish an artifact and a pathology. Thus, we described current solutions to remove the specific artifacts and to prepare the surface model for usage in CFD and RP.

The above described steps may alter the shape of a target structure locally. For several artifacts, the original shape of the model (e.g., the vessel radius) is obvious, even to non-medical employees, and can thus be adjusted appropriately. Finally, the plausibility of these changes as well as the general shape of model still need to be validated by medical experts.

It is obvious, that the described model generation process is complex and may get tedious. The target model needs to be exchanged between different separated software tools. Thus, a unified solution offering the described tools for remeshing, local smoothing, local inflation, and cutting is desirable and could help to guide the user through this process faster and to achieve a better basis for discussion and collaboration with medical experts.

5. ACKNOWLEDGMENTS

Tobias Moench is funded by the German Federal Ministry of Education and Research (BMBF) within the ViERforES-II project (no.01IM10002B).

6. REFERENCES

- [1] F. Allamandri, P. Cignoni, C. Montani, and R. Scopigno. Adaptively adjusting marching cubes output to fit a trilinear reconstruction filter. In *Proc. of EG Workshop on Scientific Visualization*, pages 25–34, 1998.
- [2] L. Antiga, B. Ene-Iordache, and A. Remuzzi. Computational geometry for patient-specific reconstruction and meshing of blood vessels from mr and ct angiography. *IEEE Trans Med Imaging*, 22(5):674–84, 2003.
- [3] L. Antiga, M. Piccinelli, L. Botti, B. Ene-Iordache, A. Remuzzi, and D. Steinman. An image-based modeling framework for patient-specific computational hemodynamics. *Medical and Biological Engineering and Computing*, 46:1097–1112, 2008. 10.1007/s11517-008-0420-1.

- [4] A. Armillotta, P. Bonhoeffer, G. Dubini, S. Ferragina, F. Migliavacca, G. Sala, and S. Schievano. Use of rapid prototyping models in the planning of percutaneous pulmonary valved stent implantation. *Proc Inst Mech Eng H*, 221(4):407–16, 2007.
- [5] L. Augsburger, P. Reymond, R. Ouared, O. Brina, D. A. Rufenacht, N. Stergiopoulos, and V. M. Pereira. Influence of segmentation on hemodynamics in cerebral aneurysms. In *Proc. of Conference on Modelling Fluid Flow*, pages 81–87, 2009.
- [6] R. Bade, O. Konrad, and B. Preim. Reducing artifacts in surface meshes extracted from binary volumes. *Journal of WSCG*, 15(1-3):67–74, 2007.
- [7] H. Bogunović, J. M. Pozo, M. C. Villa-Uriol, C. B. L. M. Majoie, R. van den Berg, H. A. F. Gratama van Anel, J. M. Macho, J. Blasco, L. S. Román, and A. F. Frangi. Automated segmentation of cerebral vasculature with aneurysms in 3dra and tof-mra using geodesic active regions: an evaluation study. *Med Phys*, 38(1):210–22, 2011.
- [8] T. Bolke, S. Seshadhri, O. Gurvit, R. Bade, B. Preim, G. Janiga, M. Skalej, S. Serowy, and G. Rose. Phantom based flow analysis by means of dynamic angiography, cfd and laser-doppler-velocimetry. In *Nuclear Science Symposium Conference Record, 2007. NSS '07. IEEE*, volume 5, pages 3440–3445, 26 2007-nov. 3 2007.
- [9] P. W. D. Bruin, F. M. Vos, F. H. Post, S. F. Frisken-Gibson, and A. M. Vossepoel. Improving triangle mesh quality with surfacenet. In *Proc. of MICCAI*, pages 804–813, 2000.
- [10] J. R. Cebral and R. Löhner. From medical images to anatomically accurate finite element grids. *International Journal for Numerical Methods in Engineering*, 51:985–1008, 2001.
- [11] J. R. Cebral, R. Löhner, O. Soto, P. L. Choyke, and P. J. Yim. Patient-specific simulation of carotid artery stenting using computational fluid dynamics. In *Proc. of MICCAI*, pages 153–160, 2001.
- [12] J. R. Cebral, F. Mut, J. Weir, and C. M. Putman. Association of hemodynamic characteristics and cerebral aneurysm rupture. *AJNR Am J Neuroradiol*, 32(2):264–70, 2011.
- [13] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. of the ACM SIGGRAPH Conference on Computer Graphics*, pages 317–324, 1999.
- [14] S. F. F. Gibson. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In *Proc. of MICCAI*, pages 888–898, 1998.
- [15] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki. Harmonic coordinates for character articulation. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.
- [16] K. Knox, C. W. Kerber, S. Singel, M. Bailey, and S. Imbesi. Rapid prototyping to create vascular replicas from ct scan data: Making tools to teach, rehearse, and choose treatment strategies. *Catheterization and Cardiovascular Interventions*, 65(1):47–53, 2005.
- [17] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *Proc. of the ACM SIGGRAPH Conference on Computer Graphics*, pages 57–66, 2001.
- [18] P. Lermusiaux, C. Leroux, J. C. Tasse, L. Castellani, and R. Martinez. Aortic aneurysm: Construction of a life-size model by rapid prototyping. *Annals of Vascular Surgery*, 15(2):131 – 135, 2001.
- [19] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. of the ACM SIGGRAPH Conference on Computer Graphics*, pages 163–169, 1987.
- [20] T. Moench, R. Gasteiger, G. Janiga, H. Theisel, and B. Preim. Context-aware mesh smoothing for biomedical applications. *Computers & Graphics*, 35(4):755 – 767, 2011.
- [21] T. Moench, M. Neugebauer, P. Hahn, and B. Preim. Generation of Smooth and Accurate Surface Models for Surgical Planning and Simulation. In *Proceedings of SPIE Medical Imaging*, San Diego, California, USA, 13 - 18 February 2010 2010.
- [22] G. M. Nielson. Dual marching cubes. In *Proc. of IEEE Visualization*, pages 489–496, 2004.
- [23] S. Oeltze and B. Preim. Visualization of Vascular Structures: Method, Validation and Evaluation. *IEEE Transactions on Medical Imaging*, 24(4):540–548, April 2005.
- [24] M. Piccinelli, A. Veneziani, D. A. Steinman, A. Remuzzi, and L. Antiga. A Framework for Geometric Analysis of Vascular Structures: Application to Cerebral Aneurysms. *IEEE Transactions on Medical Imaging*, 28(8):1141–1155, Aug. 2009.
- [25] S. Raya and J. Udupa. Shape-based interpolation of multidimensional objects. *IEEE Transactions on Medical Imaging*, 9(3):32–42, 1990.
- [26] J. Schöberl. Netgen - an advancing front 2d/3d-mesh generator based on abstract rules. In *Computing and Visualization in Science*, volume 1, pages 41–52, 1997.
- [27] C. Schumann, M. Neugebauer, R. Bade, B. Preim, and H.-O. Peitgen. Implicit Vessel Surface Reconstruction for Visualization and Simulation. In *International Journal of Computer Assisted Radiology and Surgery (IJCARS)*, volume 2(5), pages 275–286, 2008.
- [28] C. Schumann, S. Oeltze, R. Bade, and B. Preim. Model-free surface visualization of vascular trees. In *IEEE/Eurographics Symposium on Visualization, Eurographics*, pages 283–290, 2007.
- [29] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 351–358, New York, NY, USA, 1995. ACM.
- [30] J. Vollmer, R. Mencl, and H. Müller. Improved laplacian smoothing of noisy surface meshes. *Computer Graphics Forum*, 18(3):131–138, 1999.
- [31] R. T. Whitaker. Reducing aliasing artifacts in iso-surfaces of binary volumes. In *Proc. of the IEEE symposium on volume visualization*, pages 23–32, 2000.
- [32] J. Wu, M. Wei, Y. Li, X. Ma, F. Jia, and Q. Hu. Scale-adaptive surface modeling of vascular structures. *Biomed Eng Online*, 9:75, 2010.

Physics-Based Simulation of Vascular Trees for Surgery Simulations

Simon Adler

Fraunhofer IFF
Magdeburg

adler@iff.fraunhofer.de

Tobias Moench

Dept. of Simulation and Graphics
University of Magdeburg

tobias.moench@ovgu.de

Rüdiger Mecke

Fraunhofer IFF
Magdeburg

mecke@iff.fraunhofer.de

ABSTRACT

In minimal-invasive surgery there is a trend to interventions with even smaller instruments. These procedures are gentle for patients, but in many cases more complex for the surgeon. The new Single-Port Technique (SPT) enables the surgeon to perform an intervention through one incision point with several instruments. Nowadays, the SPT is applied only in selected surgeries. The application depends on the movement the surgeon can perform with the instrument in the region of interest. This depends on the patients specific anatomy and also on the design of the instruments. Virtual environments can support not only the training, but also the development of instruments adapted to new interventions. To estimate the possible movements and the residual degrees of freedom for the instruments, thus are inserted through one incision point, not only the organs are relevant, but also the vessels connecting them.

We present a method for the interactive physic based deformation of vascular systems during interaction. Therefore, the centerline of the vascular system is utilized for the physic based simulation and a surface representation is synchronized. Because the performance is a major issue and depends substantial on the surface resolution, so we also present a approach for the generation of a low resolution surface mesh for vascular systems whose resolution is adaptive to the vessels diameter.

Keywords

Deformation, Surface Generation, Physics-based Simulation, Angular Spring, Surgery Simulation, Medical Simulation

1. INTRODUCTION

In minimal-invasive surgery interventions are performed with incisions of a few centimeters. In endoscopic surgery, three incision points are commonly used. Two for instruments and an additional one for the endoscopic camera. The Single-Port-Technique (SPT) aggregates these points into a single incision point, that is commonly located at the patients navel and called *the portal*. The pivots of the instruments are closely located, so the moving space for each individual instrument is restricted. This method, however, has advantages for the patient, but is quite new and nowadays applied in selected interventions. For a broader application of the SPT, new instruments optimized for more complex interventions are required. Therefore, it has to be verified if the degrees of freedom of an instrument are sufficient to accomplish the surgery. Virtual simulation environments can

support surgeons during training of new methods, but they can also support the development of instruments and the related examination of the achievable moving space and degree of freedom.

The scenario, targeted in our development, is the removal of the spleen (splenectomy) with the SPT. The spleen is difficult to reach via a single port. The location of the port and the operation strategy depend on the patients specific anatomy. Thus, such interventions may be very complex. The spleen is supplied with two main vessels buried within the patients fat. They pass other organs such as the stomach or the pancreas [6] closely and may be difficult to detect. Thus, important tasks during a splenectomy are the extraction and cutting of vessels, while an injury of surrounding anatomical structures should be avoided.

Virtual environments are used for the planning of difficult interventions. There are surgical simulation systems available, that a surgeon can use to train surgical procedures. The main challenge is the simulation of complex tissue deformations in real-time. For the liver, different simulation models are achieved to simulate its complex tissue behavior, but vessels are only considered if they have high relevance during the performed surgery.

However, vessels are important to verify the degrees of freedom and the movement space of instruments at a port, since surgeons need to reach the vessels for coagulation. Furthermore, vessels may interactively be moved by means of deformation if regions located behind have to be reached.

In medical visualization, the location of vascular structures can be identified from tomographic image data acquired via, computed tomography angiography (CTA) or magnetic resonance angiography (MRA). In surgical planning systems and real-time visualization applications, vascular structures are commonly stored by their centerline to reduce complexity. While the surface representation, generated via marching cubes algorithm from the medical image data, is used for visualization, the centerline representation can be used to measure the distances between the vessel and other anatomical structures. Depending on the application (e.g., computational fluid dynamics), the vascular surface mesh may be highly resolved and is required to be very accurate. In contrast, high mesh resolution is critical for systems involving real-time tissue deformation, such as required for surgical simulation and training environments.

Thus, we present methods for real-time deformation of vascular trees. In section 3 we describe a physics-based deformation model utilizing centerline information. The deformation model extends a mass-spring system by additional

angular forces. For visualization, a surface can be synchronized with the centerline deformation.

Highly resolved medical surface models can be deformed in real-time, but in a complex scenario (e.g. surgical simulator), computation time is very limited. In section 4, we present a surface generation algorithm which depends on the vessel centerline and local diameter information. The resulting surface mesh has smooth transitions at the branching points of the vessel-tree. Surface deformation is performed according to the deformation of the centerline, which allows a fast and artifact-free visualization. In section 5, we present some results before we discuss the work and draw conclusions in section 6.

2. RELATED WORK

In medical applications, centerline representations are frequently used. A vascular tree has a complex shape because of multiple branches, changing curvature and diameter of vessels (Fig. 1). A mesh representation of the surface often requires thousands of triangles. The centerline of a vascular system is used as simplified representation for data exchange, to measure the distances and for other time consuming computations. From a geometrical perspective, the centerline is a branching lines sequence where a diameter is associated with each vertex. Additionally, the centerline with the diameter information represents the vessel's volume.

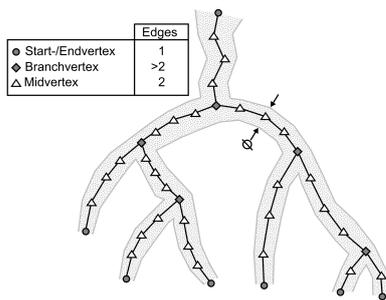


Figure 1: 2-D projection scheme of a vascular tree. The centerline with vertices and additional diameter information represents the vessel's volume. Branch- and end-vertices support connectivity information. Thus, mid-vertices describe the vessels shape.

One application of centerlines is the detection of similarities in shapes by comparing them [25, 14]. The centerlines represent topological connectivity as a graph, so that methods from graph theory can be applied for analysis and optimization.

Centerlines of vascular systems can be generated with various methods. For example, Montanari et al. [12] have introduced an analytical approach to compute the centerline from polygonal shapes [2, 1, 17]. Another approach in image processing is the computation of centerlines by morphological thinning of the shape [8]. Vincent et al. [23] has introduced an approach using a matrix to analyze neighboring pixels in image space. In [16] hierarchic Voronoi skeletons are introduced, which are also applicable to 3D geometries. However, the resultant skeleton is very detailed and requires additional post-processing for reduction and removal of artifacts.

As the name implies, centerlines are line structures, such as an additional surface representation is required for visualization of a vascular tree. The centerline can be achieved from an existing 3D geometry [16], but rather from binary images of anatomical data. A simple approach is to use cylinders for each line segment of the centerline. Such approximations are efficient for visualization, but they lead to surface intersections at the branches. More advanced approaches, similar to MPU Implicits [20, 21, 24] or model-based approaches like convolution surfaces [15], will prevent intersections but result in high resolution surface approximations and often requires fine tuning of several parameters for acceptable results.

For an efficient real time visualization of interactive deformable vascular trees, a low resolution surface is required with soft transitions at the branching points. Intersections between the surface mesh of connected vessels may lead to artifacts during deformation. The subsequently decimation of a high resolution surface would not only preserve surface geometry closely related to the centerline, but also details in shape, which is desired for planning purposes.

For surgical simulation or for the evaluation of residual degrees of freedom and movement space of an instrument during a possibly new intervention by experts, complex scenarios are employed. Thus, computational resources are required, e.g., for physics-based deformation of the vascular tree, tissue deformation, collision detection, or force feedback. As a consequence, high resolution surface meshes often needed to be decimated in order to reduce computational effort and to save rendering performance. It is not trivial to find an adjustments for a mesh decimation, that results in a low resolution mesh without artifacts. Once found it is not very likely that such a setup can be transferred to other vascular trees. On the other hand the centerline with diameter informations describes already the desired shape. Thus, we present a method to generate a surface mesh for visualization inspired by Scharf et al. [22] based on the centerline. The mesh has a low resolution, smooth transitions at branching points and a high quality of the faces.

3. PHYSICS-BASED ANIMATION

The physics-based model which we will present is motivated by the anatomical layers of vessels. The vessels boundary is subdivided in the layers intima, media and externa. The externa is a border between the vessel and surrounding tissue, whereas the intima- and media-layer are relevant for the vessel's deformation. The intima-layer contains a flexible sub-layer for flexibility longitudinal in direction of the vessel. It is distinctive in arterial vessels located close the the heart, and converts the outgoing pulse wave into a continuous blood-stream. In peripheral regions, the pulse-stream is preserved by a pulsative motion of the muscles in the media-layer. Thus, the media-layer is responsible for the perpendicular deformation of the vessel.

The centerline will be used for a physics-based deformation of a vascular tree that is visualized with a synchronized surface mesh. Thus, it is required to connect the vertices of the surface with the centerline representation. For each vertex of the surface the closest centerline segment is determined (Fig. 2) using a Sphere-Tree as hierarchical bounding volume according to Preim et al. [18].

The difference of the i -th surface vertex \vec{p}_i to its projection $\vec{p}_i^c(\alpha)$ on the centerline yields the connection vector \vec{v}_i .

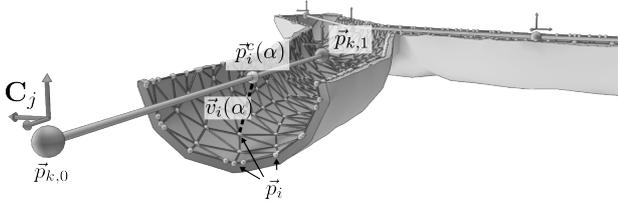


Figure 2: Scheme of a surface and the corresponding centerline

The projected position is represented relative to the k -th line segment of the centerline, where $\vec{p}_{k,[0,1]}$ are the vertices characterizing the segment. A change of length will also change the position of the projection point, because α is defined during initialization.

$$\vec{p}_i^c(\alpha) = \vec{p}_{k,0}(1 - \alpha) + \vec{p}_{k,1}\alpha \quad (1)$$

$$\vec{v}_i^c = \vec{p}_i - \vec{p}_i^c(\alpha) \quad (2)$$

The intima-layer is simulated with a mass-spring system. The segments of the centerline are utilized as springs, while the mass is lumped at the centerline vertices [19]. An external force applied to one of the centerline vertices will result in internal forces until the rest length of each segment is restored. However, the mass spring system wont induce forces if the centerline is bent or the segments are rotated. This would require angular stiffness which can be induced by angular springs. Bourguignon et al. [5] have introduced a method for the computation of damped angular springs between coordinate axes to simulate anisotropic materials with tetrahedrons, which we will utilize.

Initially a local coordinate system is determined for every j -th vertex of the centerline. The orientation of the z -Axis is directed to the next segment vertex (Fig. 3, A). The x - and y -axis are not especially defined, so they can be arbitrary oriented, but the vectors must define an orthonormal base. For each local coordinate system the rotation \mathbf{R}_j to its antecessor is computed [13], such that Eq. 3 is fulfilled in the initial configuration.

$$\mathbf{C}_{j-1} = \mathbf{R}_j \mathbf{C}_j \quad (3)$$

If the vertices are move, e.g., by interaction, the segments affected by these vertices could not only be changed in length, but also rotated. This yields to a violation of Eq. 3, because the rotational matrix \mathbf{R}_j is defined during initialization and the corresponding local coordinate systems are connected by their z -axis with the segments, so that they are also rotated. The angular forces are computed with forces acting on the local coordinate systems until Eq.3 is no longer violated.

First, the coordinate system \mathbf{C}_j is transformed into the coordinate system of \mathbf{C}_{j-1} with the initially computed rotation matrix \mathbf{R}_j (Fig. 3B). Because both local coordinate systems are rotated by interaction they are not equal if \mathbf{C}_j is transformed. A regular spring is defined with a rest length of zero (Fig. 3C) between the corresponding tips of the axes of the coordinate systems. This means that the forces, applied by these regular springs will affect each pair of axes individually. In Fig. 3 forces are just applied to the y - and z -axes. The coordinate system is not rotated but the axes

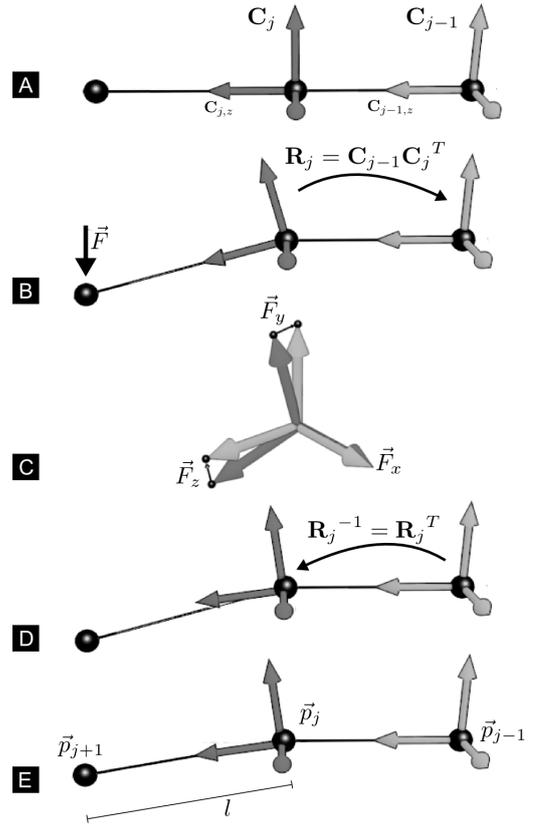


Figure 3: Scheme for the computation of the angular forces in five steps A-E

of the coordinate systems are reoriented by inner forces of the springs.

Until each simulation step, the axes must neither be normalized, nor perpendicular to each other so that the orthonormal base must be reconstructed. Thus, all axes are normalized and the orthogonal system is recovered by Eq.4 - 7. Thereafter, the rotation of the local coordinate system is reverted (Fig. 3, D).

$$\vec{v}_z = \vec{p}_j - \vec{p}_{j-1} \quad (4)$$

$$\mathbf{C}_{j,z} = \frac{1}{2}(\hat{v}_z + \mathbf{C}_{j,z}) \quad (5)$$

$$\mathbf{C}_{j,y} = \mathbf{C}_{j,z} \times \mathbf{C}_{j,x} \quad (6)$$

$$\mathbf{C}_{j,x} = \mathbf{C}_{j,y} \times \mathbf{C}_{j,z} \quad (7)$$

Now the angular forces are applied by the regular springs defined between the tips of the local coordinates systems. Accordingly, the segment controlled by \mathbf{C}_j is re-aligned by Eq. 8, such as $\vec{p}_{j+1} - \vec{p}_j$ is parallel with $\mathbf{C}_{j,z}$ (see Fig. 3E).

$$\vec{p}_{j+1} = \vec{p}_j + l \mathbf{C}_{j,z} \quad (8)$$

After the centerlines deformation is completed, the connected surface must be synchronized to the new configuration. Therefor the vector \vec{v}_i is not stored in world coordinates, but rather in the local coordinates \mathbf{C}_j and \mathbf{C}_{j-1} of the supporting centerline segment, which will yield to the

local vectors $\vec{v}_{i,j}$ and $\vec{v}_{i,j+1}$ (Eq. 9). It must be emphasized, that the regular springs, used to re-align the local coordinate systems are subsequently applied. The force is applied during each step of simulation and the duration of re-alignment depends on the springs stiffness. Thus, if they are not yet completely re-aligned, the world coordinates defined by $\vec{v}_{i,j}$ in \mathbf{C}_j and $\vec{v}_{i,j+1}$ in \mathbf{C}_{j+1} may differ. Therefore, the average between both supporting local vector are used to determine the world coordinates of the surface vertex (see Eq. 10).

$$\vec{v}_{i,j} = (\vec{v}_i \cdot \mathbf{C}_{j,x}, \vec{v}_i \cdot \mathbf{C}_{j,y}, \vec{v}_i \cdot \mathbf{C}_{j,z}) \quad (9)$$

$$\vec{p}_i = \vec{p}_i^c(\alpha) + \frac{1}{2}(\vec{v}_{i,j} + \vec{v}_{i,j+1}) \quad (10)$$

4. SURFACE GENERATION

In surgical treatment planning applications, surface approximations like truncated cones [7], are commonly used, for non-pathological vessels. During deformation and bending of a vascular tree, smooth transitions at the branches are preferred.

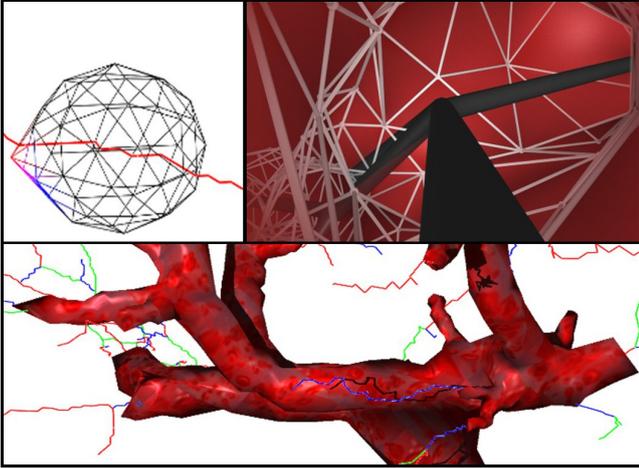


Figure 4: We use a geodesic sphere as seed for the growing process (top, left). The generated surface has smooth transitions at the branches and may also be used for interior visualizations (top, right). Visualization of the vascular tree surface during the generation process (bottom)

To generate a surface model with smooth transitions at branching points and appropriate mesh properties (e.g., number of triangles and triangle shape) based on given centerline and radius information, we use a geodesic sphere geometry as initial seed (Fig. 4, top left). It is placed on the branching vertex with the largest diameter, so that a large initial diameter of the seed geometry could be used. Because we are starting at a branching point, the seed will enter each vessel connected in the branching point which result in multiple branches and could be computed in parallel.

By the iteratively growth of the seed an approximation of the vessel's volume boundary is achieved. Therefore, each vertex \vec{v}_i will be moved by a force \vec{F}_i , which is defined by Eq. 11 (see also Fig. 5). The force is defined by the surface normal \vec{n}_i and an attraction force \vec{a}_i which directs from the vertex position to the closest position at the volume boundary of the vessel.

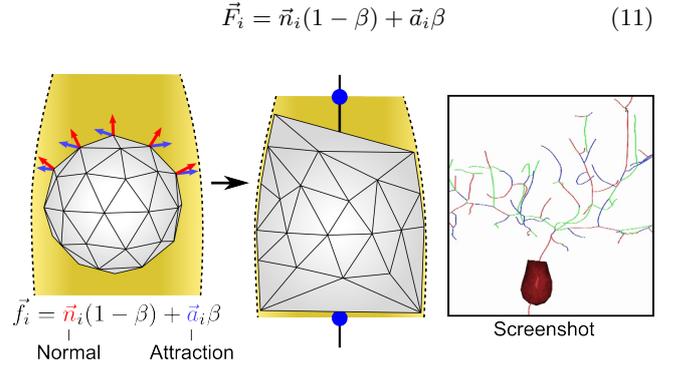


Figure 5: The vertices of the geodesic sphere are moved by a force composed of the vertex normal and an attraction force towards the vessels surface (left). The computation is continued, until each vertex has reached the surface defined by local diameter information (middle).

The normal will lead to an expansion of the seed, whereas the attraction force will direct the vertex movement to the volumes surface. The ratio between expansion speed and surface attraction is controlled by the weight $\beta \in [0, 1]$. For low values of beta the expansion is very fast and low detail is achieved longitudinal to the vessels expansion. For higher values the expansion is low, but will result in a high resolution surface. Nevertheless each vertex position is updated until it has reached the vessels surface. From an implementation point of just a small amount if vertices must be taken into account during computations.

We are using a mesh structure similar to the Winged-Edge representation [3]. Thus, each vertex contains topology information. If a vertex has reached the surface, all adjacent faces are verified. If all vertices of the face have reached the surface, the face is not in motion anymore.

If all vertices have reached the surface, the expansion can not proceed, but at branches with small vessels it could be that such vessels are not considered. Thus, if a vertex has reached the surface it adjacent faces are considered for subdivision. If one of the faces vertices are still in motion, the face is not subdivided. If all its vertices had reached the surface, it is determined if the centerline intersects with the face. If an intersection occurred the face is subdivided edge-centric. This yields to ten new faces and three additional vertices, considered as movable by Eq.11. If all vertices have reached the vessels surface and faces are not intersected by the centerline the approximation is completed.

In an additional post-processing step the generated surface is optimized. Edges are turned, if the shape is preserved and the triangle quality is increased, measured by geometrical edge- and radius-metric. Additionally, the resolution of the mesh can be increased to increase the accuracy. Therefore, the distance of the face's center to the vessels surface is computed and faces are subdivided if the distance is greater than a given threshold.

A high value for β in Eq. 11 will lead to a small step size longitudinal to the centerline and wont have influence on the detail of approximation detail in a cross section. Thus, small we are suggest smaller values, e.g. $\beta = 0.3$ and an additional subdivision step.

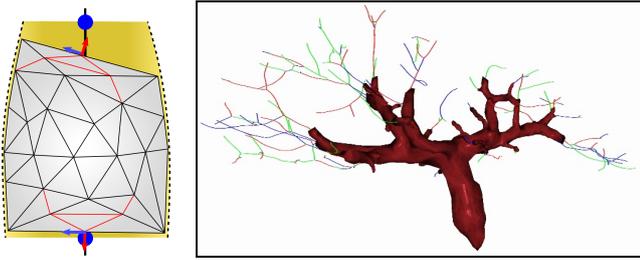


Figure 6: Faces whose vertices have reached the surface are verified. If the deviation is too large, the face is subdivided edge-centric

5. RESULTS

For evaluation of the performance, centerlines and meshes of different complexity were chosen. The centerlines of the vessels were extracted from CT data, segmented by medical experts. The surface in Fig. 7C was also generated by Marching Cubes from these binary images, while the surface in Fig. 7B was generated with the presented approach. Since both surfaces are related to the same centerline, a third vascular system of the liver with a more complex topology has been taken into account (Fig. 7D).

The results are summarized in Tab. 1 and illustrate that the presented approach a plausible physics based simulation of vascular trees could be achieved in real time. The visual result in Fig. 7A shows that a local force applied to a simulated vessel results in a deformation and that sharp kinks could be avoided, like it would appear if changes in rotation would not result in internal forces those are an outcome of the presented angular damped springs. Forces that result in large segment rotation were applied without the appearance of kinks or twists.

The computation performance of the physics-based simulation is adequate for interactive real-time applications, even in case of the more complex surface of the vascular tree in Fig. 7D. In case of the complex centerline in Fig. 7D, the update of the physics simulation is convincing, but the synchronization of the surface requires too much time. Indeed, the required time for synchronization is a multiple of the time required for the physics-based update. The test were made on a Intel Xeon CPU with 2,66GHz but without using possibilities for parallelization via the graphics card (CUDA) or CPU-based multi-threading.

The presented surface generation approach is easy to implement and yields surfaces with a lower mesh resolution, as marching cubes or MPU-Implicits provide. Furthermore, it results in smooth transitions at the centerlines branching points. The resulting low-resolution surface is not intended for surgical planning or computational fluid dynamics (CFD), but in surgical simulation applications, the smooth branching transitions are desired, while the performance of the simulation is essential.

Nevertheless, the surface of the vessel's volume must be approximated, so that evaluations about the required degrees of freedom during intervention can still be considered as meaningful. Thus, we compared the presented approach with respect to the resulting mesh complexity, mesh quality and accuracy of approximation.

For evaluation, we were using a data set from the livers portal vein (see Fig.7D). The centerline is more complex

and the variations in the vessel's diameter are more significant. We generated a high resolution surface using marching cubes and the centerline with diameter information from segmented anatomical data. This high resolution surface is used as reference for accuracy measurements in Tab. 2. Because the marching cubes surface is the reference for evaluation, a surface based on truncated cones is used to compare the quality of approximation.

Table 2: Comparison of our surface generation approach with marching cubes [11, 9] and truncated cones. The marching cubes surface and the centerline are generated from binary data. This centerline was utilized for the generation of surfaces with truncated cones and our approach. The Equiangle Skewness is an indicator for mesh-quality with [0,1] meaning [fine, poor].

	Number of		Equiangle Skewness		Hausdorff Distance (mm)
	Vertices	Faces	mean	max	
Marching Cubes	89.406	178.712	0,48	0.90	-
Truncated Cones	58.070	119.430	0,50	0,99	7,24
Low-Resolution	11.571	23.138	0,23	0,70	7,67
High-Resolution	51.979	103.952	0,17	0,98	7,08

Geometry: Livers portal vein

The quality of the triangles is measured by the equiangle skewness, which is the ratio between the largest and smallest inner angle in a triangle. Thus, the mean quality of the low resolution result is better than truncated cones and marching cubes. At the same time

At the same time, 20% of the vertices are used compared to truncated cones and 13% compared with marching cubes. The Hausdorff distance measures the longest distance between an approximation to the reference surface mesh based on marching cubes. Hence, a similar accuracy could be accomplished with our approach compared with truncated cones.

6. DISCUSSION AND CONCLUSIONS

Surface meshes generated using the presented methods are supposed to be used for visualization of vascular trees within surgical simulation systems, whereas the accuracy is sufficient compared to established methods. If a surface mesh based on truncated cones is deformed, the intersections between cones may lead to further artifacts. This, however, is avoided our method exhibiting smooth transitions at the branching points Fig. 8.

The results confirm that the presented method is suitable for interactive real-time environments and will be used in the specified scenario. The groundwork is established for the interaction with complex vascular structures and will be integrated into an environment for the virtual evaluation of minimal-invasive surgical procedures with a focusing SPT. Currently, the parametrization (stiffness, damping, mass) of the physics-based model has been done heuristically, so that the simulation is stable for update rates of at most 20ms.

For future work a parametrization based on specifically measured values is aimed. For more detailed parametrized

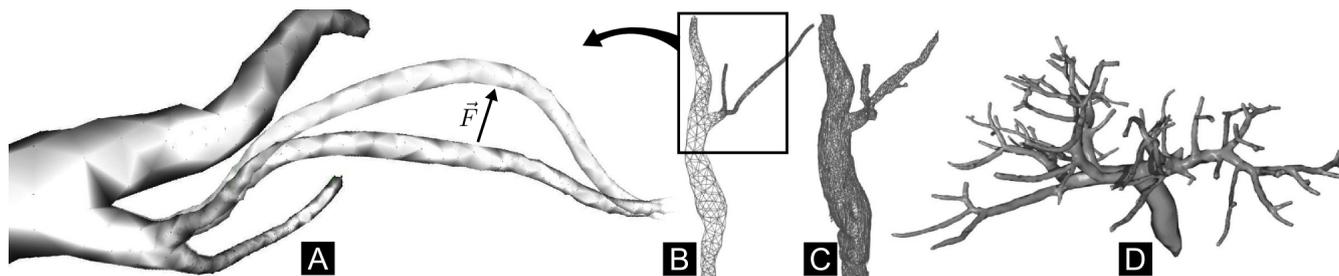


Figure 7: Results of the physics-based simulation. A Force is applied at one vessel of a vascular tree. The force was applied to a centerline connected with surfaces in (B) and (C) with different resolutions. In (A) a crossfade of initial and deformed configuration of the affected region was shown. In (D) the data-set of a vascular tree with a more complex topology is shown.

Table 1: Summarized test results for the vascular systems in Fig. 7. The computation times for the physic based model and the time for the synchronization of the surface are feasible for interactive real-time szenarios.

	Fig. 7, B	Fig. 7, C	Fig. 7, D
Centerline (Segments, Vertices)	118/119	118/119	1559/1560
Surface (Triangles, Vertices)	2426/1215	9799/4907	25730/12867
Comp. Time			
Deformation	0,4ms (2500Hz)	0,4ms (2500Hz)	5,5ms (182Hz)
Surface Sync.	1,5ms (667Hz)	6,1ms (164Hz)	21ms (48Hz)
Total	1,9ms (536Hz)	6,5ms (154Hz)	26,5ms (38Hz)
Surface Sync. (%)	78,95%	93,85%	79,2%

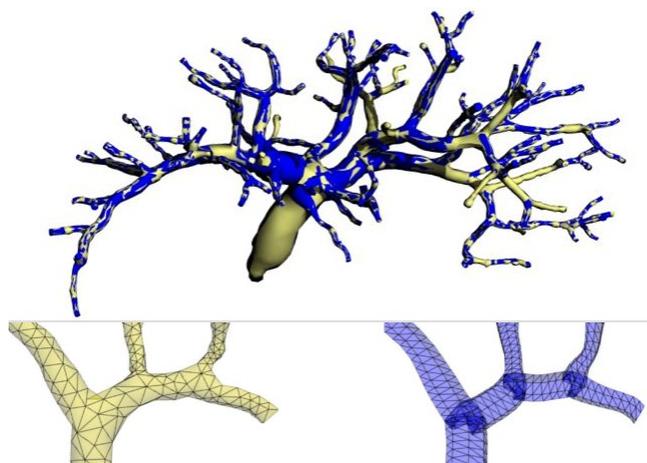


Figure 8: Overlay of surface meshes based on truncated cones (blue, bottom right) and our method (yellow, bottom left). For a static configuration, the intersections of the cones (bottom right) are covered. During movement of vessels, intersections may lead to visualization artifacts.

models, e.g., changing parameters depending on deformation for a simulation of non-linear deformations, the real time performance must be preserved. In mass spring systems, a general problem could be observed. The parameters of mass-spring systems could be achieved from young-modulus, a material parameter that could be measured from sample[4, 10]. The parametrization by measurement will define the stiffness and damping coefficient and of course the mass. Once these values are defined the stability of mass-spring

systems depends only on the update interval and the number of nodes, where the mass is lumped.

Thus, a centerline-based simulation has advantages with regard to the stability, because the centerline has less vertices compared to a synchronized surface (Fig. 7B 9,7%, Fig. 7C 2,4%). At the moment, the physics-based method allows the deformation in terms of bending. The method will be extended in future work, such that the vessel's profile can be deformed. This means that a force will not only affect the rotation of the local coordinates but also the length of the connecting vectors. An evaluation of the geometrical models and the physics-based simulation with surgeons are also preparation.

7. ACKNOWLEDGMENTS

The work in this paper is partially funded by the German Ministry of Education and Research (BMBF¹) within the **VIERforES-II**² project (Grand no. FKZ 01IM10002A).

8. REFERENCES

- [1] L. Antiga, B. Ene-Iordache, and A. Remuzzi. Computational geometry for patient-specific reconstruction and meshing of blood vessels from mr and ct angiography. *IEEE Trans Med Imaging*, 22(5):674–84, 2003.
- [2] L. Antiga, M. Piccinelli, L. Botti, B. Ene-Iordache, A. Remuzzi, and D. Steinman. An image-based modeling framework for patient-specific computational hemodynamics. *Medical and Biological Engineering and Computing*, 46:1097–1112, 2008. 10.1007/s11517-008-0420-1.
- [3] B. G. Baumgart. Winged edge polyhedron representation. Technical report, Stanford University, Stanford, CA, USA, 1972.
- [4] Y. Bhasin and A. Liu. Bounds for damping that guarantee stability in mass-spring systems. *Studies in Health Technology and Informatics*, 119:55–60, 2006.
- [5] D. Bourguignon and M.-P. Cani. Controlling anisotropy in mass-spring systems. In *In Eurographics, Eurographics Assoc.*, pages 113–123. Springer-Verlag, 2000.
- [6] H. Frick, H. Leonard, and D. Starck. *Allgemeine Anatomie, Spezielle Anatomie II: Eingeweide - Nervensystem - Systematik der Muskeln und Leitungsbahnen*, volume 2 of *Taschenlehrbuch der gesamten Anatomie*. Thieme Verlag, 3. überarbeitete aufgabe edition, 1987. ISBN 3-13-356903-1.
- [7] H. Hahn, B. Preim, D. Selle, and H.-O. Peitgen. Visualization and Interaction Techniques for the Exploration of Vascular Structures. In *IEEE Visualization*, pages 395–402, San Diego, Oktober 2001.
- [8] P. Kwok. A thinning algorithm by contour generation. *Commun. ACM*, 31(11):1314–1324, 1988.
- [9] T. Lewiner, H. Lopes, and A. Vieira. Efficient implementation of marching cubes cases with topological guarantees, 2003.
- [10] B. A. Lloyd, S. Kirac, G. Székely, and M. Harders. Identification of dynamic mass spring parameters for deformable body simulation. In *Eurographics 2008 - Short Papers*, pages 131–134, 2008.
- [11] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. of the ACM SIGGRAPH Conference on Computer Graphics*, pages 163–169, 1987.
- [12] U. Montanari. Continuous skeletons from digitized images. *J. ACM*, 16(4):534–549, 1969.
- [13] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable real-time deformations. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 49–54, New York, NY, USA, 2002. ACM.
- [14] M. Náf, G. Székely, R. Kikinis, M. E. Shenton, and O. Kübler. 3d voronoi skeletons and their usage for the characterization and recognition of 3d organ shape. *Comput. Vis. Image Underst.*, 66(2):147–161, 1997.
- [15] S. Oeltze and B. Preim. Visualization of Vascular Structures: Method, Validation and Evaluation. *IEEE Transactions on Medical Imaging*, 24(4):540–548, April 2005.
- [16] R. L. Ogniewicz and O. Kübler. Hierarchic voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.
- [17] M. Piccinelli, A. Veneziani, D. A. Steinman, A. Remuzzi, and L. Antiga. A Framework for Geometric Analysis of Vascular Structures: Application to Cerebral Aneurysms. *IEEE Transactions on Medical Imaging*, 28(8):1141–1155, Aug. 2009.
- [18] Preim, Tietjen, and Hindennach. Integration automatischer abstandsrechnungen in die interventionsplanung. *BVM2003, Informatik aktuell*, pages 343–353, 2003.
- [19] M. Schill. *Biomechanical Soft Tissue Modeling - Techniques, Implementation and Application*. PhD thesis, Fakultät für Mathematik und Informatik, Mannheim, 02 2002.
- [20] C. Schumann, M. Neugebauer, R. Bade, B. Preim, and H.-O. Peitgen. Implicit Vessel Surface Reconstruction for Visualization and Simulation. In *International Journal of Computer Assisted Radiology and Surgery (IJCARs)*, volume 2(5), pages 275–286, 2008.
- [21] C. Schumann, S. Oeltze, R. Bade, and B. Preim. Model-free surface visualization of vascular trees. In *IEEE/Eurographics Symposium on Visualization, Eurographics*, pages 283–290, 2007.
- [22] A. Sharf, T. Lewiner, A. Shamir, and L. Kobbelt. On-the-fly curve-skeleton computation for 3d shapes. In *Eurographics 2007 (Computer Graphics Forum)*, volume 26, pages 323–328, Prague, october 2007. Eurographics.
- [23] L. Vincent. Efficient computation of various types of skeletons. In M. H. Loew, editor, *Medical Imaging V*, volume 1445, pages 297–311. SPIE, 1991.
- [24] J. Wu, M. Wei, Y. Li, X. Ma, F. Jia, and Q. Hu. Scale-adaptive surface modeling of vascular structures. *Biomed Eng Online*, 9:75, 2010.
- [25] H. Zaboli, M. Rahmati, and A. Mirzaei. Shape recognition by clustering and matching of skeletons. *Journal of Computers*, 3(5):24–33, 2008.

¹<http://www.bmbf.de/en/>

²<http://vierfores.de>

Practical Experiences in Model-Based Safety Analysis

Matthias Güdemann, Michael Lipaczewski and Frank Ortmeier

{matthias.guedemann, michael.lipaczewski, frank.ortmeier} @ovgu.de

FIN / ITI, Computer Systems in Engineering, Otto-von-Guericke University Magdeburg

ABSTRACT

The formal modeling framework Safety Analysis and Modeling Language (SAML) allows for a combined specification of qualitative (non-deterministic) and quantitative (probabilistic) aspects. Using semantically founded model transformations, SAML models can be analyzed using current state-of-the-art verification tools while guaranteeing that the analysis results of the different tools are computed on equivalent models.

The significance of the results, in particular of computed quantitative occurrence probabilities heavily depends on the accuracy of the modeling. The most important factors are the temporal resolution of the model, the considered mission time for the analysis and if applicable also factors like spatial resolution. The overall results always have to be interpreted depending on the character of the system, the properties under consideration and choice of modeling parameters. In general, unfortunately, an increase in system accuracy is strongly correlated to a more complex, in particular resource intensive, analysis.

In this paper we discuss qualitatively, how the dependencies between the relevant model parameters of SAML models afflict the accuracy of the quantitative analysis results. We conduct some experiments with different parameters and survey the effects on the overall results. This work is triggered primarily with regard to optimization of such systems, where the analysis of many different system variants is necessary. In this use case, in early phases of optimization, a compromise between accuracy and running time can be used to explore many variants, identify the most promising ones and then analyze these with more accuracy.

1. INTRODUCTION

Software is becoming a major innovation factor in many different application domains. With increasing demands and more features, the complexity of systems growing steadily. At the same time, software-intensive systems are increasingly used in safety-critical systems. This makes accurate safety analysis methods at the same time more important, but also more difficult to execute. Many existing safety analysis methods only consider only a subset of the important aspects of safety and reliability of systems.

The safety analysis and modeling language (SAML) [4] was developed as a tool-independent framework to allow for combined qualitative and quantitative model-based safety analysis. Semantically sound model transformations allow for a provably correct transformation of SAML models into the input languages of different verification tools. This

greatly increases the significance of the computed results, by guaranteeing that qualitative and quantitative properties are analyzed on equivalent models.

First experiments with SAML showed, that an accurate quantitative analysis of models of safety-critical systems is possible [5, 6]. Nevertheless, the relatively long analysis time which is required is disadvantageous, in particular with regard to using quantitative properties to optimize systems, as proposed in [17, 7]. The running time heavily depends on the temporal (and where required the spatial) resolution of the modeled system and also on the “mission time” which is considered. The higher the resolution and the longer the considered interval, the more accurate results are computed. On the other hand, this increases both the state space and the analysis time which is required.

Nevertheless, when using the quantitative analysis results as basis for the optimization of systems, it is not always necessary to use the full accuracy. The most relevant aspect is the pairwise relative comparability of different system design variants. In this paper we examine how different modeling resolutions influence the analysis results and in particular, how analysis results can be extrapolated to longer “mission times”. The rest of the paper is structured as follows: Sect. 2 gives a short overview of the general issues that are encountered in model-based safety analysis. Sect. 3 introduces our quantitative model-based safety analysis technique pDCCA. Sect. 4 introduces two case studies and shows the experiments of the application of pDCCA to them with different parameters. Sect. 5 concludes the paper and discusses some further work.

2. MODEL-BASED SAFETY ANALYSIS

Model-Based safety analysis aims at finding the causal connections between component malfunctioning and overall system hazards based on mathematical deduction techniques. Different approaches for this are possible. One possibility for classification is the choice of deduction tool support. Interactive theorem provers like KIV; PVS or ISABELLE can theoretically cope with infinite state space models, automatic verification tools like model-checkers are very often limited to finite state spaces.

An orthogonal way for classification is whether the safety analysis is qualitative, with two-valued logic or quantitative where probabilities are computed for logical properties. For both, there exist elaborate verification tools and appropriate formalization logics.

A third way for classification is the model of time which is used in safety analysis. The two main different possibilities

are discrete and continuous time. The approach used in this paper is based on automatic verification of finite state spaces with probabilistic model-checkers in a discrete time model.

The decision for this is based on the following considerations: In practical use, quantitative properties are of highest importance, as they are required for certification purposes of safety critical systems. This basically eliminates interactive theorem provers which do not offer support for quantitative reasoning.

The choice of the timing model is more difficult. On the one hand, for safety analysis, it is not possible to consider the system independent of the controlled hardware and surrounding physical environment - which would suggest a continuous time model. Nevertheless, this is well-fitting for interleaved systems [10], while on the other hand, many systems, in particular software-intensive ones, are working synchronously parallel. Such a behavior is not representable in continuous time models¹. Considering this, and the fact that experiments have shown the applicability of discrete time models to relevant case studies, this timing model was chosen for our safety analysis.

2.1 Model Creation Issues

The creation of an accurate (or at least adequate) stochastic model of the system under consideration is one of the most important tasks for all model-based analysis techniques. For safety analysis, the model must include modeling of the software, the hardware, the physical environment and also of the potential erroneous behavior. A model which includes all that is called the *extended system model* [11, 12]. In general the necessary steps for modeling and safety analysis are as follows:

1. Formal modeling of the functional system
2. Identification of the relevant failure modes and system hazards
3. integration of the local failure mode effects to form the *extended system model*
4. computation of critical combinations of failure modes (qualitative analysis)
5. computation of hazard occurrence probabilities (quantitative analysis)

More detailed information about the construction of extended system models can be found in [19], for modeling the effects of failure modes in [18] and for quantitative modeling in [5].

2.2 Important Model Parameters

In this paper we will concentrate on two very important model parameters which greatly influence the quantitative analysis results, especially the accuracy and the calculation time, and which have to be chosen in all discrete time, probabilistic models.

Temporal Resolution The first parameter is the discretization of time, called the temporal resolution. This parameter (Δt) specifies the basic time unit of a system which

¹Hybrid models which can combine both discrete and continuous behavior are currently still of very limited prevalence, still have many limitations [3] and there are no efficient verification tools available.

passes at each discrete time step². The temporal abstraction on which is chosen with this parameter obviously has a big impact on both qualitative and in particular quantitative properties, as the semantics of probabilistic logics is based on this time.

Mission Time The second model parameter is the length of the time interval which is considered in the analysis, called the *mission time*. This specifies the time interval which serves as a reference to compute the hazard occurrence probability. This is necessary, as the naïve approach to analyze an infinite time interval will almost never give usable results. The reason is simply that the analysis to answer the question “What is the probability that the system fails” is computed for a (theoretically) infinite amount of time. Therefore all possible failure modes and therefore also all possible system hazard will eventually occur and the probability would be 1. Because of this, it is very common to find values for mean time to failure (MTTF) or expected values for system failures as results of quantitative safety analysis. In our approach, generally the occurrence probability of the system hazard is computed for a given mission time.

2.3 Restrictions of Model Creation

It is obvious, that a tiny value for the temporal resolution and a long mission time grants the best results. It is also obvious, that this will lead to the problem of state space explosion and longer analysis time. Therefore it is necessary to find compromises between accuracy and required analysis time. A possible solution could be a numeric multi grid-technique, for which the calculation would start with rough approximations and then a stepwise improvement of the approximation rate for “regions of interests”. This could be useful to find optimal sets of parameters for a given safety critical system. Nevertheless, no such efficient method exists at the moment for solving larger models. In this paper another strategy is being used. The question should be answered, if the analysis results can be used for extrapolate failure probabilities of longer mission times without longer time intervals and the temporal resolution.

3. QUANTITATIVE SAFETY ANALYSIS

In accordance with the notion of *low-demand* and *high-demand* failure mode types described in IEC 61508 [16], failure mode modeling in SAML is separated into the basic cases of *per-demand* and *per-time* failure modes. A *per-demand* failure mode can only appear if there is a demand to the safety critical system at the moment. Such a failure mode occurrence is described with a failure probability. In contrast, a *per-time* failure mode can occur at any time and is generally specified using a failure rate.

3.1 Approximation of Per-Time Failure Rates

The failure rates of *per-time* failure modes are specified as a parameter λ of a continuous exponential distribution function as shown in Eq. (1). Informally speaking, this parameter represents the expected number of occurrences of the failure mode in a given time interval $[0; t]$.

In a discrete time model as in SAML, such a continuous distribution can be approximated via the discrete geometric

²In this model we assume the same temporal discretization for all model components, if different discretization should be used in different components, this can be expressed as the greatest common divisor of both discretizations.

distribution [5] as shown in Eq. (2). The parameter p_{step} of the geometric distribution describes the probability that the failure mode occurs within a single time-step of length Δt . For the approximation of an exponentially distributed failure mode occurrence, p_{step} is computed as $\lambda\Delta t$.

$$P'(X \leq t) = \int_0^t e^{-\lambda x} dx = 1 - e^{-\lambda t} \quad (1)$$

$$P(X \leq k) = 1 - P(X > k) = 1 - (1 - p_{step})^k \quad (2)$$

Using the exact computation of the failure probabilities with the exponential distribution and their approximation via the geometric distribution, the relative (and absolute) approximation errors can be computed. The relative approximation error is $\epsilon(t) = |(P'(X \leq t) - P(X \leq k)) / P'(X \leq t)|$. For a temporal resolution of $\Delta t_1 = 10ms$ and $\Delta t_2 = 1s$ the effect on the accuracy is as follows: for Δt_1 the maximal relative error amounts to $1.2710 \cdot 10^{-8}$ which corresponds to a cumulative relative error of $4.4577 \cdot 10^{-4}$. For Δt_2 the maximal relative error is $1.3889 \cdot 10^{-6}$ and the cumulative relative error amounts to 0.048764.

This means that the order of the relative error basically coincides with the order of the relation of the temporal resolution. It also turns out that the higher resolution is slightly better than just explainable from the relation of the order of magnitude of the temporal resolution.

3.2 Safety Analysis Method

The method we used to compute the frequent safety hazard is the probabilistic deductive cause-consequence analysis (pDCCA) [4] which is a quantitative model-based safety analysis method. It is based on probabilistic model checking and the probabilistic temporal logic PCTL [9]. The formalized property for safety analysis states informally: “What is the maximal probability that a hazard occurs?” and is formalized in PCTL as follows:

$$P(H) := P_{max=?}[true \mathbf{U}^{\leq steps} H] \quad (3)$$

Here the predicate H describes the occurrence of the hazard and $steps$ describes the considered mission time as the number of basic time units. For this, $steps$ is the quotient of the “real” mission time and the temporal resolution Δt of the system. Informally speaking, $steps$ describes the maximal length of the observation interval of the discrete time model. The notion of the *maximal* probability is necessary as SAML allows non-determinism. For safety analysis, the worst case behavior of a system, i.e. the worst non-deterministic choices wrt. the hazard probability are of interest which is computed as the maximum probability.

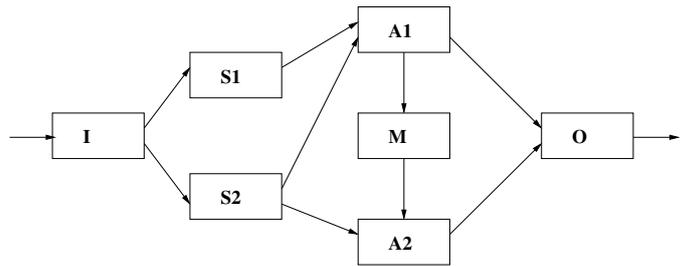


Figure 2: Example System with Hot-Spare Redundancy

4. EXAMPLE CASE STUDIES

For the experiments, two case studies were selected. The first study is an abstract illustration of a hot-spare system, which is a reference case study taken from safety analysis literature using algebraic techniques for dynamic fault trees [20]. It was used to illustrate temporal dependency analysis of failure modes. Hot-spare technique is used to provide a backup system in dormant mode in case of a failure of the main system, in which case the backup system becomes active. This is used widely for example at safety-critical controllers of airplanes or trains. In this case-study, both parameters can be changed and the resulting accuracy can be evaluated.

The second case study is taken from an actual project and represents a real case study of safety-critical engineering. It is an airbag-controller designed for the purpose to rescue life. In this example there is not always the possibility to change the value of the presented important parameters. Because of the design of the controller, the temporal resolution is fixed. The formal model of this case study does not allow for an easy adjustment of the temporal resolution, therefore here only the effect of different mission times were analyzed.

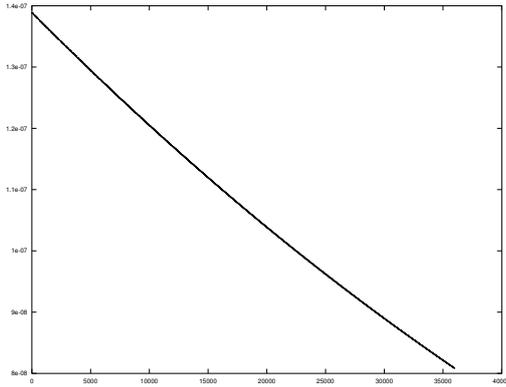
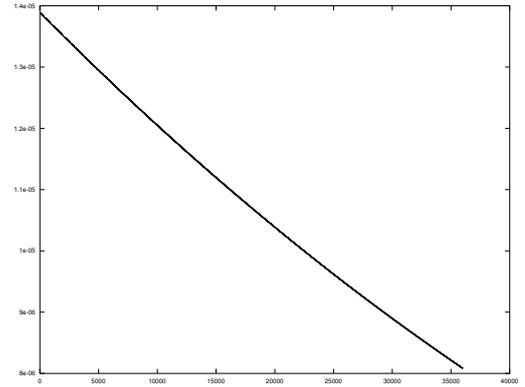
Both models are implemented in SAML, then transformed and afterwards analyzed with the PRISM model checker [14].

4.1 Hot-Spare Redundancy Case Study

The first case study consists of two redundant input sensors (S1 and S2) measuring an input signal (I). This signal is then processed in an arithmetic unit to generate the desired output signal. Two arithmetic units process the signal, a primary unit (A1) and its backup unit (A2). The primary unit gets an input signal from both input sensors, the backup unit only from one of the two sensors. If the primary unit (A1) produces no output signal, then a monitoring unit (M) switches to the backup unit (A2) for the generation of the output signal. The backup unit will only produce an output signal if it has been triggered by the monitor. If the secondary unit is triggered, the primary unit will get switched off. A schematic view of the case study is depicted in Fig. 2.

A much more detailed description of the formal modeling of the case study and the qualitative safety analysis results (combinations of failure modes that can cause the hazard if they occur together) are presented in [8, 5]. Here only the quantitative hazard occurrence probability is considered.

The system is functionally correct if it delivers a correct output signal. It is easy to imagine that if such a system is used in a safety-critical environment, a malfunctioning (omission of values) could be very dangerous and the system

(a) $\Delta t = 1s$ (b) $\Delta t = 10ms$ **Figure 1: Relative Approximation Error for Per-Time Failure Modes**

can become safety-critical. Therefore the redundancy and the degraded mode have been integrated.

A variety of failures modes is possible, it is assumed that a list of the relevant failure modes exists. The ones considered in the modeling and analysis are the following: The sensors can omit a signal (*S1FailsSig*, *S2FailsSig*), making it impossible for one of the arithmetic units to process the data from the sensors correctly. The arithmetic units themselves can omit producing output data (*A1FailsSig*, *A2FailsSig*). The monitor can fail to detect that situation (*MonitorFails*), either switching if not necessary or not switching if necessary. The activation of the A2 unit may fail (*A2FailsActivate*) although the monitor sent the activation signal to A2. The failure occurrence patterns for these six failure modes are integrated into the SAML model of the case study for its safety analysis. In the case study, *A2FailsActivate* is modeled as *per-demand* failure mode, the other failure modes as *per-time*.

The failure rate for the different failure modes of the case study are expected to be as the following: for *S1FailsSig*, *S2FailsSig* and *A2FailsSig* $10^{-2} \frac{1}{h}$, for *A1FailsSig* and *MonitorFails* $1 \cdot 10^{-6} \frac{1}{h}$, as the per-demand failure probability of *A2FailsActivate* $p = 10^{-7}$.

4.1.1 Quantitative Analysis of the Case Study

For the small case study, the occurrence probability of the hazard $H :=$ “no output signal is delivered” is computed for a mission time of $10h$.

The occurrence probability was computed both for $\Delta t_1 = 10ms$ (Eq. (4)) and $\Delta t_2 = 1s$ (Eq. (5)) where the model parameters that are dependent on the temporal resolution were changed accordingly.

$$P_{max=?}[true\mathbf{U}^{\leq 3.600.000}H] = 9.1490 \cdot 10^{-7} \quad (4)$$

$$P_{max=?}[true\mathbf{U}^{\leq 36.000}H] = 9.27072 \cdot 10^{-7} \quad (5)$$

It is very obvious, that the difference in order of approximation quality of the per-time failure modes has only a minor impact on the computed hazard occurrence probabilities. In particular for a more coarser estimation used for

relative comparison this can be a big advantage for the analysis of very complex models, for which the analysis effort can be very large. This is especially relevant, as the number of steps which has to be analyzed (k in this case) translates to a nearly linear increase in computation time. The reason is that the probabilistic model-checking algorithms use k matrix-vector multiplications in this case. Speedups like square and multiply algorithms cannot really be used, as they potentially destroy the structure and therefore the sparsity of the respective matrix which in return would increase the memory requirements enormously.

4.1.2 Extrapolation of the Analysis Results

A second possibility to reduce the runtime is by trying to calculate the results for small temporal resolutions and extrapolate them for higher time intervals.

Even for relative small models the analysis costs for such a long time period is long (ca. $30s$ with $2.6Ghz$ CPU at $\Delta t_1 = 10ms$). The reason for this is the bounded temporal $\mathbf{U}^{\leq k}$ - operator for which k vector-matrix multiplications have to be calculated. This needs a lot of time for the large matrices, which are typical for probabilistic model checking. It is hard to avoid this by other methods because the sparse structure of the matrices is often destroyed by this methods. For this, it is necessary to evaluate how to extrapolate from calculated values of small intervals to larger time periods. Fig. 3 shows the calculated failure probabilities by PRISM [15] (including the time periods until $10h$). The results are measured in steps of $10s$ by $\Delta t = 10ms$. The minimum value at $t = 10s$ is $7.2927 \cdot 10^{-14}$, the maximum is located at $t = 10h$ with $9.1490 \cdot 10^{-7}$.

Comparison of Different Approaches.

There are three different extrapolation methods, which will be compared. This will be used to approximate the failure probabilities for a mission time of $10h$ based on the calculated values of $t = 10s, 20s, 30s, 40s$. Notice that because of the type of probability calculation with the probabilistic model checker for $k = 4000$ all previous values for k are also calculated and therefore they are also available without explicit calculation.

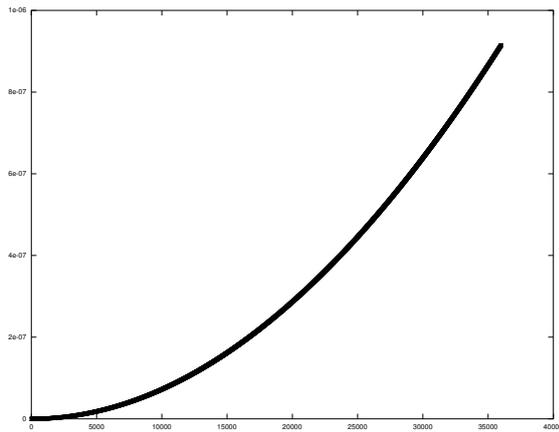


Figure 3: Calculated Failure Probabilities

The obvious method to approximate from a calculated failure probability to a value for a longer mission time is to fit the probabilities proportional to the duration. This leads to a simple linear extrapolation and is nearly similar to the expected rate of failure amount for this longer time interval.

The second approach is a cubic approximation, in which case the numerical derivations are computed from the calculated values. Because the actual characteristics of an (cumulative) distribution function is monotone increasing and changes the curvature between the stable phases 0 (for $t = 0$) and 1 (for $t \rightarrow \infty$), a cubic extrapolation (constant third derivation) is used.

As third approach an approximation with a (cumulative) exponential distribution $1 - b \cdot e^{-ct}$ is chosen. The free parameters b, c of the equation are defined by minimizing the squares of the differences between the calculated values. The values for the parameters were calculated with a minimization function which does not rely on derivative information. The implementation of the Octave Open Source tool for numerical analysis was used.

Results.

The results of the three approaches for the extrapolation are visualized in table 1. The noticeable deviation of the linear extrapolation can be explained by several reasons. In this case study, the fact that the single failures are not completely independent, certainly plays a significant role. As far as the A1 unit runs into a failure or the A2 unit gets activated because of a malfunction of M, obvious the breakdown of S2 becomes more critical. This circumstance leads to a relatively high increase in the distribution function and explains the strong deviation of the plain linear extrapolation.

By exploiting higher (numerical) derivations a much better approximation of the real probabilities can be achieved with cubic extrapolation. As seen in table 1, this leads to a much better extrapolations of the entire probabilities as well as the accumulated deviation compared with the results of the linear extrapolation.

The results of the extrapolation with the use of exponential distribution are very similar to the ones of the linear extrapolation, which is quite surprising, because a combina-

tion of exponential derivation functions is being used and because of that it would be obvious, to assume a exponential distribution for the full probability. But this particular assumption can not be proven by the results of the test. A possible explanation could be the relatively small probabilities, which are influencing the inaccuracy of the parameter definition. If this is the main reason for the big difference has to be checked by further experiments.

Interpretation of the Results.

Because the curvature of differential continuous distribution functions generally changes, linear and square approximations are quite difficult. The cubic approach, which needs the same time as the linear one, is way much better than the linear and even as the exponential approach. If these results are quantifiable is surely dependent on many input parameters. The main factors seems to be:

- **persistent changes of the system:** this can lead to stochastic dependencies in the system, which can not be represented appropriately by a linear model (in the case study the switching to A2 obviously changes the impact of the full model by its failure rate).
- **representativeness of the chosen interval:** only an interval which contains a representative subset of the reachable states should be used for extrapolation (the most obvious restriction is that the hazardous states must be reachable in that time, else the hazard occurrence probability would be estimated to zero).

Whether a system allows persistent changes which are non-revertible can be decided with specific proof obligations. The idea is to prove – after a *warm-up time* for initialization – if it is always possible to go back to a given state. This kind of approval can be formulated in the temporal logic CTL [2] and with the help of the transformation of a SAML model for qualitative model checker easy to prove.

The choice of a representative interval can be reduced to the problem of getting a maximum of state coverage. Perfect would be the examination of *all* possible states which mean the choice of a mission time k for which all possible states are reachable in k steps. This can be done by SAT Solvers analog to bounded model checking [1], by proving, when there is no unreachable state existing any longer. As a result of calculations of probabilistic model checkers it can be assured that even if not all states can be reached the impact to the overall probability with increasing mission times will decrease. Because pDCCA basically is a reachability analysis, this kind of method should be usable for the identification of promising system variations for optimization purposes.

4.2 Airbag Model Case Study

This second case study is taken from the automotive domain, namely an airbag controller. This sort of controller aligns to the safety-critical systems and are made for high-availability. There are several different designs, which all have the same purpose to make the controller more safe and trustable, mostly by adding redundant components. This case study is based on an airbag-model of the Fraunhofer IESE [13]. The main parts of the model as they matter for the analysis and also seen in Fig. 4 are

- **magnetic and acceleration sensor:** redundant sensors with different measure abilities and car-positions. Each sensor includes the errors *false_negative*, which leads to an negative sensor output independent of the

	value $t = 10h$	accumulated deviation	max. rel. error
linear	$1.3120 \cdot 10^{-9}$	0.0011048	0.99857
cubic	$9.1421 \cdot 10^{-7}$	$4.9125 \cdot 10^{-7}$	$7.5364 \cdot 10^{-4}$
exponential	$4.2785 \cdot 10^{-10}$	0.0010835	53.982

Table 1: Summary of Extrapolation Results

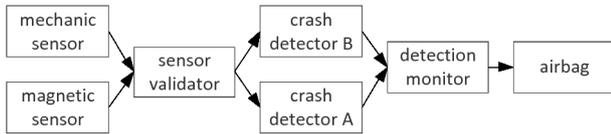


Figure 4: Model of Airbag Controller

expected output, *false_positive* which leads to an positive output and *defect* in which case there is no sensor output anymore.

- **sensor validator:** merges the input of the sensors including the compensation of a defect sensor by using only the signal of the available sensor.
- **two crash detectors:** redundant detectors, which checks the sensor validator every five steps and uses the last 6 values of the validator to check for a crash. If there are more than a specified amount of positive values in this measured valued and additionally the actual value is positive, a crash will be detected. Each detector includes the error *wrong*, which shifts the calculated output to the opposite.
- **crash detection monitor:** merges the input of the detectors. Includes the error *wrong*, which shifts the calculated output to the opposite.
- **airbag:** will be activated if the monitor has recognized a crash i.e when the monitor activates the airbag. Includes the errors *selfIgnition*, which will activate the airbag immediately and *defect*, which is an permanent error and prevent the airbag from activation.

Additionally, there is an environmental model for braking, accelerating and the occurrence of a crash to provide some detailed driving information. As part of the environmental model, the probabilities for some errors depends on the driving situation, especially for breaking.

The system is functionally correct if the airbag activates if and only if an accident happens. A malfunction of the airbag could result in massive injuries. There are two possibilities for such a malfunction. On the one side, there could be an accident without the activation of the airbag, i.e. a false negative, or, on the other side, there could be an airbag activation without an accident, i.e. a false positive. Because the second one will probably result in an accident by hurting the driver, this malfunction is more critical. For this reason, the airbag model contains redundancies at the main components and is designed never to activate the airbag, if there is a chance for a false detection. This false positive activation is the hazard which will be considered in the following.

There are a lot possible errors for this model. The failure modes for this case-study are taken from the described paper of the Fraunhofer IESE. All errors, except *airbag_defect*, are designed as *per-time* failures, as *airbag_defect* is imple-

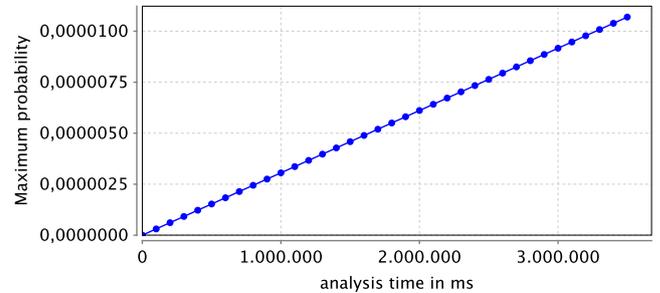


Figure 5: false_pos Probabilities of Airbag Controller

mented as *per-demand* failure. The demand for this error is specified as the activation of the airbag by the detection monitor. The used probabilities for failures are based on the EN ISO 13849-1 c-e. Basically all per-time failures are expected to be $10^{-6} \frac{1}{h}$, which is defined as EN ISO 13849-1 c. Only the airbag failure rates are slightly different. *Airbag_selfIgnition* has a probability of $10^{-8} \frac{1}{h}$ and *airbag_defect* with $p = 10^{-8}$ as it is written in the EN ISO 13849-1 e.

4.2.1 Analysis of the airbag model

As described, the main hazard of the airbag is the activation of the airbag without the occurrence of a crash, which leads in most cases to an accident. So the analyzed hazard *false_pos* is formulated as activation of the airbag without a crash occurred before. To get significant results, it is necessary to calculate the probabilities of this hazard for a long time period with about a minimum of one hour. In the model, the temporal resolution is fixed to 1 ms because of the behavior of the crash detector, which needs a signal of the sensor each ms. This means that a coarser Δt is not easily possible in this model without significant changes of the model structure and therefor of the model behaviour.

The required iterations for the desired time period can be calculated as 3,600,000 steps. This already takes a very long time for the case study. As it is not possible to change the step size of the model, a way to get significant results is to shorten the mission time and extrapolate on longer intervals.

Therefore an important question is which mission time is needed to prove a given extrapolation method. For this, it is necessary to prove the probability behavior along different time intervals to predict the results for long time periods out of short ones.

Results.

Fig. 5 shows the evaluation results of the hazard *false_pos* for different temporal resolutions. As seen, the calculated probabilities are rising linear to the temporal resolution. The results are measured in steps of 10,000ms by $\Delta t = 10ms$. The minimum value at $t = 100ms$ is $3.0222 \cdot 10^{-10}$, the maximum is located at $t = 3,500,100$ with $1.069 \cdot 10^{-5}$. For

	first airbag model	extended airbag model
time for model construction	0.703seconds	7.72seconds
states	528363	499456001
transitions	4527444	16590774822
choices	0	998912002
nodes	60705	345934

Table 2: Comparison of Airbag Models

the calculated 36 measure points in Fig. 5, already 4 days are needed.

Interpretation of the Results.

Compared with the results of the abstract case study with hot-spare redundancy, where the calculated probabilities are increasing asymptotically cubic, these results are different. A possible explanation is, that the airbag-model does not contain permanent changes, which changes the impact of a given probability on the global probability at some time. In this case, the model is more “predictable”.

To explain the difference, an extension of the airbag model can be imagined, where a second airbag can be released after the first one was activated, as it is being used in some car models for preventing injuries by a second crash. The hazard can be formulated as the probability, that the second airbag will be activated without the occurrence of a second crash. As mentioned, these error can only occur, when the first airbag was activated. In this case, the probability of self ignition of the second airbag becomes more impacting on the global probability of the hazard, when the first airbag is activated. Than the probabilities for different mission times would be expected to be also cubic, because of the permanent change of the first airbag state.

5. CONCLUSIONS AND FURTHER WORK

This work was about qualitative evaluation of the impact of discretization of error rates to the results of quantitative model-based safety analysis and the extrapolation of quantitative analysis results over an given time interval. One of the aimed results of the ProMoSA Project is the optimization of systems, at which the failure probabilities should also be used as an target function. Because of the possible long calculation time for an analysis and the need for lots of calculated values, the analysis with an less exact resolution seems to be tolerable.

This kind of proceeding could be used in early states of the the optimization process, to eliminate especially bad system modulations and identify the most promising ones. The analysis resolution can then be raised again for a more precise analysis of potentially good system variations. the first results in this direction leads to the following conclusions:

- discretization errors of per-time failure rates have a lower impact to die results than the approximation error, i.e. a inaccurate approximation can be used for a rough estimation
- a simple proportional (linear) extrapolation can be accurate for systems like the airbag, but it can also be inaccurate even for simple models, therefore if possible, higher order approximations should be used, in particular if persistent changes are possible in a system

For further experiments with the airbag model, current work is an improved airbag model, with a more detailed en-

vironmental model as well as more fine grained behavior of the single modules. As shown in table 2, a first version of this improved model is much larger than the original version. It requires more than ten times the time of the version presented in this paper for the internal model creation by PRISM. Also the number of nodes, states and transitions increased significantly. Because the extended model contains non-deterministic choices, this number was raised too. As a result, the calculation of representative results seems to be hardly possible and not practical in an appropriate time. With the help of the results presented in this paper, a extrapolation of results with small mission times to provide significant results for this improved model is expected to be possible. Also the type of airbag model with second crash protection and therefor permanent changes can become interesting for further experiments.

The results presented in this work will be examined further. Important next steps will be the analysis of a larger case study with where the development of strategies for the usage of the extrapolation and a adaptive adjustment of the resolution for optimization is possible. Furthermore, the localization of proof obligations to quantify the quality of an extrapolation seems to be a desirable result of future work. This kind of criteria could be some sort of structural type or about validation of the extrapolation over additional calculated values of the model.

Acknowledgments

Michael Lipaczewski is sponsored by the Deutschen Ministerium für Bildung und Forschung in the ViERforES project (BMBF, project-Nr.: 01IM08003C).

Matthias Güdemann is sponsored by the German Research Foundation (DFG) within the ProMoSA project.

6. REFERENCES

- [1] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. In *Formal Methods in System Design*, page 2001, 2001.
- [2] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 2000.
- [3] M. Gribaudo and A. Remke. Hybrid petri nets with general one-shot transitions for dependability evaluation of fluid critical infrastructures. In *Proceedings of the 12th High Assurance System Engineering Symposium (HASE 2010)*, pages 84–93. IEEE Computer Society, 2010.
- [4] M. Güdemann and F. Ortmeier. A framework for qualitative and quantitative model-based safety analysis. In *Proceedings of the 12th High Assurance System Engineering Symposium (HASE 2010)*, 2010.
- [5] M. Güdemann and F. Ortmeier. Probabilistic model-based safety analysis. In *Proceedings of the 8th*

- Workshop on Quantitative Aspects of Programming Languages (QAPL 2010)*. EPTCS, 2010.
- [6] M. Gdemann and F. Ortmeier. Quantitative model-based safety analysis: A case study. In *Proceedings of 5th conference for Sicherheit, Schutz und Zuverlaessigkeit (SICHERHEIT 2010)*. Lecture Notes in Informatics (LNI), 2010.
- [7] M. Gdemann and F. Ortmeier. Model-Based Multi-Objective Safety Optimization. In *Proceedings of the 30th International Conference on Computer Safety, Reliability and Security (SAFECOMP 2011)*. Springer LNCS, 2011. to appear: 19.9.2011.
- [8] M. Gdemann, F. Ortmeier, and W. Reif. Computing ordered minimal critical sets. In E. Schnieder and G. Tarnai, editors, *Proceedings of the 7th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems (FORMS/FORMAT 2008)*, 2008.
- [9] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:102–111, 1994.
- [10] H. Hermanns, J.-P. Katoen, J. Meyer-Kayser, and M. Siegle. A markov chain model checker. In *Proceedings of the 6th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS 2000)*, pages 347–362. Springer, 2000.
- [11] A. Joshi and M. P. E. Heimdahl. Behavioral fault modeling for model-based safety analysis. In *Proceedings of the 10th IEEE High Assurance Systems Engineering Symposium (HASE 2007)*, pages 199–208, Washington, DC, USA, 2007. IEEE Computer Society.
- [12] A. Joshi, M. Whalen, and M. P. Heimdahl. Modelbased safety analysis: Final report. Technical Report CR-2006-213953, NASA Contractor Report, 2005.
- [13] J. Kloos, T. Hussain, and R. Eschbach. Failure-based testing of safety-critical embedded systems. 2011.
- [14] M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: A hybrid approach. In *Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2002)*, volume 2280 of LNCS. Springer, 2002.
- [15] M. Kwiatkowska, G. Norman, and D. Parker. Quantitative analysis with the probabilistic model checker PRISM. *Electronic Notes in Theoretical Computer Science*, 153(2):5–31, 2005.
- [16] P. B. Ladkin. An overview of IEC 61508 on E/E/PE functional safety, 2008.
- [17] F. Ortmeier and M. Gdemann. Promosa - probabilistic models for safety analysis. In *Proceedings of 6. Dagstuhl-Workshop on Model-Based Development of Embedded Systems (MBEES 2010)*, 2010.
- [18] F. Ortmeier, M. Gdemann, and W. Reif. Formal failure models. In *Proceedings of the 1st IFAC Workshop on Dependable Control of Discrete Systems (DCDS 2007)*. Elsevier, 2007.
- [19] F. Ortmeier, W. Reif, and G. Schellhorn. Formal safety analysis of a radio-based railroad crossing using deductive cause-consequence analysis (DCCA). In *Proceedings of 5th European Dependable Computing Conference (EDCC 2005)*, volume 3463 of LNCS. Springer, 2005.
- [20] M. Walker, L. Bottaci, and Y. Papadopoulos. Compositional Temporal Fault Tree Analysis. In *Proceedings of the 26th International Conference on Computer Safety, Reliability and Security (SAFECOMP 2007)*, 2007.

Sensitising to security risks in manufacturing engineering: An exemplary VR prototype

Jana Fruth, Richard Münder, Hannes Gruschinski, Jana Dittmann, Bernhard Karpuschewski,
Rolf Findeisen
Otto-von-Guericke University of Magdeburg, Germany
{fruth, muender, hannes.gruschinski, dittmann, karpu, rolf.findeisen}@ovgu.de

ABSTRACT

With the increasing use of standard IT technologies in automated production systems, similar security vulnerabilities and threads known from the desktop IT domain are also introduced in manufacturing engineering. These vulnerabilities could be misused for different attacks employing malicious codes. Derived attacks can be hazardous to production system safety, for example objects and/or people in the vicinity of functioning industrial robots might be in danger. Usually, users are given oral instructions on relevant safety information. Often this is not sufficiently to explain the complexity of security attack scenarios, which could cause safety issues. For this paper we propose the use of a VR simulation environment as a better visualization of potential security risks in the production domain, e.g. the manipulation of robot control programs via malicious codes. We provide an example scenario in a VR simulation, the automated chamfering of large gearings with an industrial robot to outline the idea. In the future, VR simulation can be annotated with multimedia security warnings to facilitate better understanding of the potential security and safety risks. The problem of detecting and avoiding malware infections is not addressed in this paper.

Keywords

security-safety interdependencies, production engineering, virtual reality (VR), e-learning, malware

1. INTRODUCTION

Today, in production systems standard desktop IT network technologies, e.g. Ethernet, are increasingly used. Implementing state-of-the-art desktop IT security measures may pose the same potential risks in production domains known and increases the probability of known threats and vulnerabilities, like eavesdropping on classified communication, manipulating, or theft of sensitive data and denial-of-service (DoS) attacks. Attackers often use malicious codes to carry out attacks. Industrial and production networks today are also susceptible to malicious attacks as seen in the recent incident caused by the computer worm Stuxnet [16]. In the production domain, safety mechanisms are used, along with other mechanisms, to prevent the damage of objects and the injury of humans. Security incidents, like malicious code attacks, can also have an indirect impact to safety; as shown in [6].

Production engineers often have a low awareness of potential security risks in production systems and they are not adequately prepared to cope with them. Furthermore, instal-

ling security protection mechanisms, e.g. intrusion detection systems (IDS), in production domains is still a challenge for industrial manufacturers. This is a consequence of special demands on production systems, as they often need to meet critical real-time requirements [4, 12]. In this paper we propose a concept and a virtual prototype environment to raise production systems safety awareness among current and future users, engineers, and manufacturers of industrial robots. We illustrate potential security risks using new production application examples such as the chamfering of large gears. It is our intention to sensitise and educate trainees, employees, robot operators, workers, and technicians on potential security incidents that could be caused by malicious code infections of robot control systems. The potential dangers to functional and indirect safety of production systems are simulated in a virtual reality (VR) to decrease the possibility of serious injuries in the actual work environment. In our opinion, the effects and interdependencies between security and safety issues caused by malware infections have to be taken into account, especially when considering future interaction scenarios between humans and robots [10]. We use a VR simulation of an industrial robot to depict potential security risks and indirect safety incidents, which could be potentially caused by a malware infected robot control program. Note that this paper does not address detection and avoidance of malware infections.

In Section 2, the state-of-the-art safety and security employed in production domains is described. In Section 3, an example production scenario, the automatic chamfering of large gears using an industrial robot, is described. Potential security risks and safety incidents caused by malicious codes on robot control systems are illustrated in Section 4. The paper closes in Section 5 with our example VR studies of potential security risks and indirect safety incidents.

2. STATE-OF-THE-ART: SAFETY AND SECURITY SENSITISING IN PRODUCTION DOMAIN

Today the process of educating and sensitising production engineers, employees, robot operators, workers, and/or technicians on production systems for potential safety risks of industrial robots is often done through oral instructions. However, this method of training is clearly insufficient. A better method is to carry out the training by using VR techniques. Currently VR training programs are often used in domains, where the training in real scenarios are too dangerous and/or too expensive. Examples of e-learning applications for safety

instructions are the safety at work for the military [17] and an advanced on-the-job training system for employees in the automotive production field [7]. To the best of our knowledge, today there are no VR training systems for the production domain that consider security incidents and/or safety and security interdependencies.

Potential security risks and indirect impacts on safety, are difficult to understand for people who are not security and/or safety experts. Learning by means of virtual reality simulations can be used to demonstrate difficult industrial processes, like chemical, manufacturing, or logistic processes. The VR world enables interaction with the user, so learning by doing is possible; which is the desired method of learning. In Section 5 we discuss our approach of using VR technologies to illustrate interdependencies between non-functional properties of production systems. Disadvantages of VR technologies for teaching purposes are the financial and personal costs and the high degree of difficulty to create virtual models, scenes and content. These facts have to be weighed against the benefits of using VR.

There are different approaches to illustrate non-functional properties by augmented reality (AR) technologies of robotic systems. Some approaches are developed in the German interdisciplinary research project ViERforES¹. For example, in [5] an AR application prototype of a Katana manipulator is described. People interacting with this robot are informed about its next motion. Another approach is published in [18], where AR safety zones are depicted with different colours. These zones may be dangerous for people that are in close proximity of the mobile assistant robot in the laboratory, called LiSA [8]. The use of VR technologies to sensitise production engineers and manufacturers of production systems to security risks, resulting dysfunctions, and hazards is, however, novel.

3. APPLICATION SCENARIO

In this section the chamfering of large gearings is discussed as a scenario in which the machining process can be subject to infiltration of malicious code. Prior work with an industrial robot system has already been introduced in [6]. Increasing the range and the quantity of production by adjusting different machine tools and modifying the setup of the manufacturing plants is a common tendency among manufacturing companies. These measures decrease production costs and labour tremendously. They also make manufacturers more competitive in a market with increasingly specialized requirements and rising manufacturing costs.

Several questions have been raised regarding the milling operation of large gearings with an industrial robot system. The first question is to investigate to what extent the geometrical relationships of a gearing can be exploited for contour generation and path planning for the robot. In a gearing, the teeth and all the gaps are the same size and are also equally distributed along the circumference. Hence, a simple derivation for the path generation may be calculated. The profile of the tooth flanks is described by an involute and have been done in an auxiliary computation. They have been omitted here for brevity reasons. The tooth parameters nee-

ded for the computations are discussed as follows. Essential dimensions of a gearing are already specified in its blank, for example its central point and its tip circle diameter. Further important reference parameters of involute gearings are the module and the pitch. For calculations beyond the reference diameter, the transverse tooth thickness and the spacewidth are relevant. The high complexity of the geometric properties of a gearing are paired with a symmetric properties that can be used to simplify calculations. The number of teeth of an involute gearing is the same as the number of pitches, which are arranged on the reference diameter. Apart from manufacturing process variations, the teeth and gaps are formed all the same. In gear production at the side of the tooth root an approximate semicircle is formed, the root rounding. This is highly dependent on the gear cutting tool. Thus the tothing contour is limited to the blank side by the tip circle diameter and to the tooth root side by the root diameter.

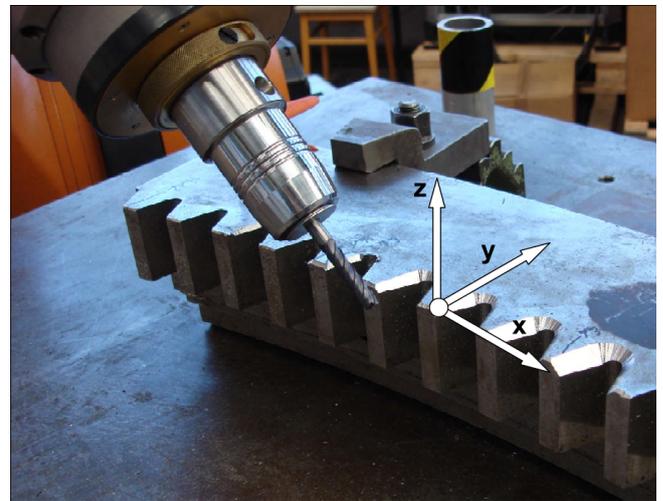


Figure 1: Large gear segment with tothing contour (left: sharp edged, right: chamfered)

These mentioned performance characteristics used to describe an involute gearing are known in the gearing manufacturing industry. The basis for determining all parameters of spur gears with involute tothing is the DIN 3960 [1]. It should be noted that some of the gearing parameters may turn out differently from the example set forth herein. Finally in the canon of the contour lines only the involute tooth flank line is missing. As already mentioned, this is the highest complexity, mathematically. All other contours can be expressed by straight lines or circles. The mathematical description of the involute can be represented by a parametric curve or by the involute function. Which variation in the application of the algorithm is used, is significantly influenced by the capabilities of the used machine control. As delivered, the involute function is not implemented in the KUKA robot controller. The Siemens 840D is a potential controller that masters the involute function [2]. However, the cost of retooling the existing robotic system would be far too expensive for our current financial situation.

The development algorithm is assisted by another characteristic of large gearings. The property referred to is the high

¹<http://www.vierfores.de>, 2011

number of teeth and the relatively small tooth depth and the very low curve of the tooth flank lines. Furthermore, the process of chamfering by using a gear based automation algorithm is not a manufacturing process where the geometric results have to be highly accurate for subsequent processes. As a result, an approximation of the tooth surface profile can be determined, or, if necessary, whether a straight or circular arc is sufficient for the requirements of the process.

The current manufacturing process of large gearings, as supposed to mass production of small gearings, is a smaller customized production process. Manufacturing of large gearings is organized in small batches or designed as specialized production. Because of the described combination of these data that differentiate for nearly every gear case the spur gears profile contour can be mapped. When transmitting this contour into a robot program, a suitable reference coordinate system should be established before configuring. In the static machining of a large gearing, only a segmented processing is possible because of the enormous size of the components. Therefore, it is useful to place a workpiece reference coordinate system over a certain number of teeth. Then the axes of this coordinate system are defined. Regarding the machining path for the robotic process can be set after the determination of the orientation of this coordinate system. This consists of the contour segments of the tooth trace, the root rounding and, in some cases, the tip line. According to the described algorithm, the trajectory is based on a tooth gap on either five or seven points. The latter depends on whether the involute curve is approximated by a straight line or a circle segment. Now the actual machining can begin. For this offset values as corrections for the tool are still to be considered. The tool used in the current scenario is a four-edged solid carbide cutting tool. The industrial robotic system features six degrees of freedom. The process of chamfering requires at least four-axis programming, because the cutting depth is held constant in vertical direction.

The programming of the robot system is done in the KUKA robot language (KRL). The robot is controlled by the KUKA robot controller KRC2 (ed. 05). First, the workpiece is clamped. Then, the workpiece-specific so called BASE coordinate system is measured. For this purpose, a probe tip is clamped in the receptacle of the tool spindle. Then by three points, the coordinate system is defined. First, the coordinate origin is determined. Then the orientation of the x-axis is identified. By setting the xy-plane, the base coordinate system is finally marked. Afterwards the TOOL coordinate system is taught to the robot. In the current scenario the inserted tool is a carbide cutter. Once machining starts a fixed point is approached by the tip of the tool. The teaching process involves manually moving of the robots tool sequentially one by one from four different angles to a fixed point. The tool is ready to be used, now that the geometric dimensions and orientation have been defined in the robot control.

The other record-by-record programming is done with respect to this TOOL and BASE coordinate system. The programmer creates the traversing program for the robot by using standard movement commands. Basically, there are the commands PTP (point to point), LIN (linear), and CIRC (arc). In Figure 3 an excerpt from a robot program written in KRL can be seen.

4. EXEMPLARY SECURITY RISK SCENARIOS AND INDIRECT SAFETY IMPACTS IN PRODUCTION DOMAINS

The protection of production systems has often focused on the aspects of implementation of safety requirements. These could be explained by the specific demands on production systems, e.g. the fulfillment of real-time requirements. However, this demand entails extremely complicated automation solutions with very high costs and expert knowledge is rare and hard to get in daily product design processes. Hence, manual production by humans in co-operation with industrial robots seems to be a promising and attractive approach. However, the closer the interaction between humans and robots, the more vulnerable the human becomes to injury and accidents. According to [15] three safety categories are to be considered: 1) *primary safety* - the system itself induces no damage or harm, e.g. electric shocks; 2) *functional safety* - the system fulfils its intended function, e.g. the chamfering of large gear; 3) *indirect safety* - implications of system actions generate no damages, e.g. harm to humans or damage of objects. In this paper, we focus on scenarios in production systems, where functional and/or indirect safety aspects are potentially affected by malicious codes.

Besides random hazards (safety incidents), intended incidents (security) have to be considered. Production networks are also threatened by malicious software, like the Stuxnet computer worm [16]. Stuxnet was written to infect automation control components called Programmable Logic Controllers (PLCs) produced by Siemens that were employed in Iranian nuclear power plants [9]. Malware infections of control components of production systems can have several implications. Below examples of various risk scenarios are outlined, which can potentially influence the functional and/or indirect safety of production systems.

Attackers, e.g. hackers, professional criminals or economic spies choose different tools to perform their attacks. Malicious codes are used for remote automated attacks, and these security attacks can be categorised as passive and active attacks [3]. **Passive attacks** typically do not involve changes of data, configurations, or programs, which are often in the focus of attackers. An example of a passive attack is the *eavesdropping of information* by means of malware. In the production domain mainly intellectual properties like production programs for industrial robots, configurations, or calibrations are at risk. Malicious codes are often used to transmit eavesdropped information from the victim's system to the attacker [9]. If that information was acquired by the attacker, it could be disastrous for a production company. Possible risks include but are not limited to: high financial damages, the loss of public reputation, the loss of technology and market advantage, see for instance [9].

Compared to passive attacks, **active attacks** have a malevolent impact on data, configurations, or programs and hence impacts on **functional safety** are imaginable. Examples of active attacks are the manipulation or deletion of sensitive data, like control programs or calibration data of control units of industrial robots. Furthermore, the manipulation of robot control programs could have an impact on other components, like the movement of mechanic components,

mechanical robot arm. Imagine the following scenario in which a milling spindle mounted on the flange of a robot machines a workpiece like a large gear. The quality of produced parts may then be reduced through the improper and malicious control of the robot's movement (functional safety). As mentioned earlier, these kind of attacks could lead to high financial losses as well as loss of reputation for the production company.

Furthermore, **indirect implications to safety**, which can be caused by malicious codes infecting control systems of industrial robots have to be taken into consideration. For safety reasons the work space of factory personal and of industrial robots should be strictly separated when the robot is in action. However, there is a large demand for close interaction of robots and humans.

There is a trend to have more interaction between humans and industrial robots in future application scenarios [13]. This follows from an increasing demand for a more differentiated product range, e.g. car and mobile phone and television devices production even at the same plant. This endeavor implicates a much higher cost if implemented in automation processes. Consequently, manual production by humans could become the preferred approach in future production processes. With increasing interaction between humans and industrial robots, the higher the possibility that malicious code manipulation of control software threatens humans [9]. Examples of such interactions are the teach-in process for the online programming of a robot, the manual placement of a work piece as a so called "third hand" and the manual leading of the robot arm for difficult manipulation steps, like the installation of a driver seat in a special variant of a car [9, 14]. One current example is the Daimler project "InCarRob", where a lightweight constructed robot (LBR) interacts with the worker during the production process [19]. In such human-to-robot interaction scenarios, the default user could not recognise unauthorised changes of the robot control program induced by malware.

Security risks and robot programming

A special feature of the KRL programming is presented by the folds in figure 2. These are program structures which enable hiding of not initially required orders and commands for specific user groups (e.g. operators). These folds entail a significant security vulnerability that can be exploited by evil hackers. Thus making it imperative that the source code be clearly arranged for the programmer and especially the operator for easier identification of e.g. malicious code structures. In the robot controller, registration as a standard user is not sufficient enough to be able to edit or manipulate these program structures or the robot control program. These access rights are reserved for users logged on as experts, only. Amongst these security protection mechanisms, the code folding is particularly vulnerable as it may contain malicious code written by un-authorized users. By code folding, several lines of code are combined to one subsection, and cannot be seen by users without expert or administrative access rights (see folded line 46 in Figure 2 and unfolded lines 60-68 in Figure 3). On the KCP default users could not recognise manipulated code, because it may be hidden by code folding. In our case, the malware is changing the original

robot control program by changing the command to shift the base coordinate system (see Figure 3, line 61) and pasting two additional moving commands. The second command may cause a damage of the desk and the third may injure a worker (see the second changed command in Figure 3, line 65).

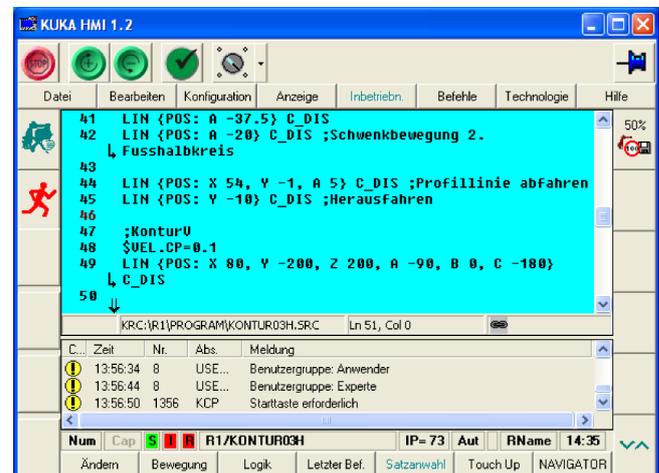


Figure 2: Hidden manipulated code by using folding in line 46 (operator mode)

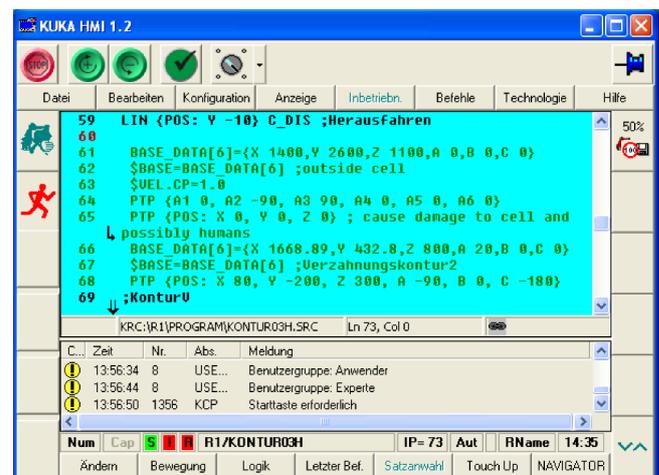


Figure 3: The manipulated code in line 65 without folding (lines 60-68, administrator mode)

As sketched in this section, manipulated robot movements caused by malware infected robot control programs could threaten the operators' safety and health. In the following section we propose a VR prototype with the aim of sensitising users to security risks and potential safety impacts in the manufacturing domain.

5. VR PROTOTYPE FOR SENSITISING TO SECURITY RISKS IN THE MANUFACTURING ENGINEERING

In this section we describe our VR prototype for illustration and simulation of potential security risks and indirect safety

incidents in the production domain, caused by malware infected control systems of industrial robots. The simulation is conducted within the VR software "KUKA.Sim Pro" Version 2.1 and KUKA.OfficeLite KR C Version 5.4. In Figure 5 the simulation of the automatic chamfering of large gears with a KUKA robot KR210-2 [11] is demonstrated as described in Section 3. We seek to sensitise users, e.g. production engineers, technicians, and machinists to potential security risks and safety implications caused by malware infected industrial robot control systems. Our simulation prototype is designed to instruct users of production systems on potential security risks and indirect safety hazard scenarios in the production domain, compare with Figure 4.

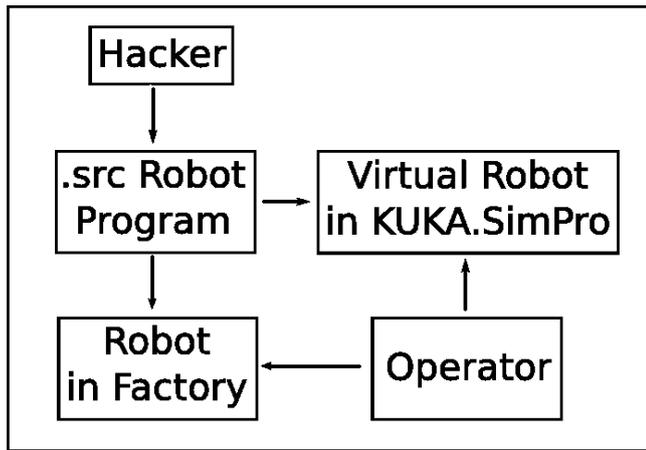


Figure 4: Proposed VR environment including the virtual robot system and its operator

Additionally, with our VR model users can instruct themselves on potential security and indirect safety impacts caused by malicious code infections. Furthermore, robot manufacturers should be motivated to develop and provide adequate security protection solutions as opposed to using sub-standard IT technologies that increase the risks for malicious codes.

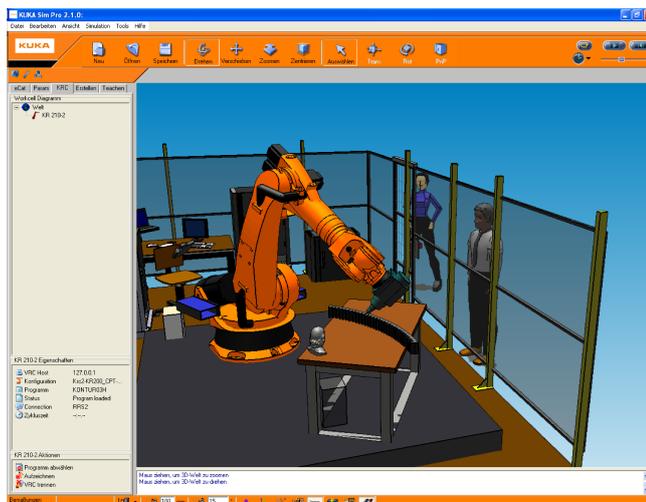


Figure 5: VR prototype: scenario of automatic chamfering of large gear segments with a KUKA robot KR210-2

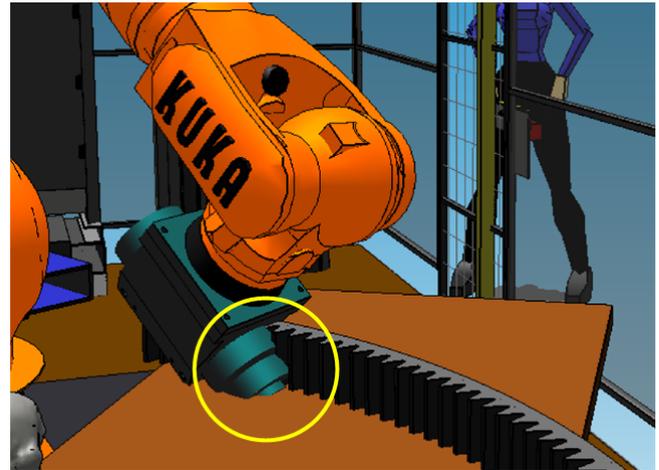


Figure 6: VR prototype: scenario of damage of a desk by misdirection of the KUKA robot arm caused by robot control program malware infection (see encircled space in yellow)

Our VR simulation should demonstrate both potential impacts to the security and impacts to the safety of production systems caused by malware infected robot control systems. An example active attack on security is the manipulating of robot control programs by adding additional moving commands (see also Section 4). This security attack may cause discrepancies in functional safety, e.g. the wrong movement of the robot arm. In Figure 6 a damage of a desk caused by a misdirection of the KUKA robot is illustrated. Furthermore, security attacks could have an impact on indirect safety. These attacks could cause damage to objects and/or cause injury to humans. An example, which illustrates a collision of a misdirected KUKA robot and a human is shown in Figure 7.

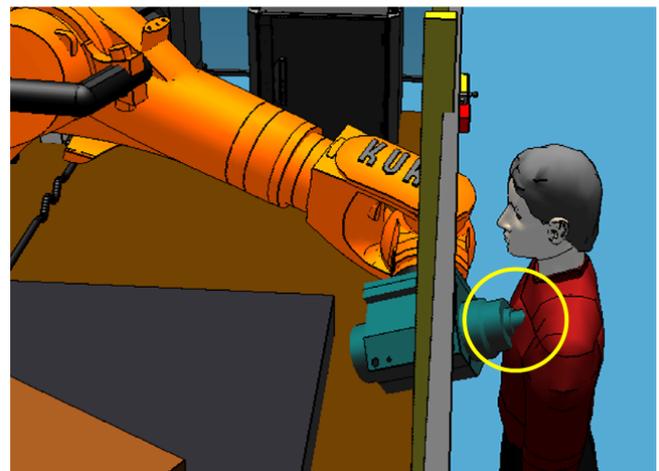


Figure 7: VR prototype: scenario of injury of a human by misdirection of the KUKA robot arm caused by robot control program malware infection (see encircled space in yellow)

As explained before, our current prototype of VR simulation

of the chamfering of large gearings with an industrial robot can be employed for security and safety instruction and teaching scenarios. Furthermore, manufacturers of industrial production systems should be motivated to improve the security mechanisms of their systems to avoid the described security attacks and safety impacts.

6. CONCLUSION AND FUTURE WORK

In this paper, we describe a VR simulation prototype, which demonstrates the automated chamfering of large gearings with a KUKA robot. The VR scene should be used for a better illustration of potential security risks in the production domain, e.g. the manipulation of robot control programs with malicious codes.

In the future, this VR simulation should be extended. Therefore, we plan on endowing our simulation with an annotation that includes multi-media security warnings for a better understanding of the potential security and safety risks. This will allow our VR simulation prototype to be used as a self-education tool for users. Furthermore, a detailed testing and evaluation process of our prototype with users of production systems is ongoing. In the future, the instruction with VR simulation including security risks with potential interdependencies to safety could be applied to other robotic domains, like service robotics.

Acknowledgements

We would like to thank KUKA Roboter GmbH Augsburg, Fischer AG Herzogenbuchsee, and Schunk GmbH & Co. KG Lauffen. Jana Fruth and Richard Mnder are funded by the German Ministry of Education and Science (BMBF), project 01IM10002A. The presented work is part of the ViERforES project. We also wish to thank Ryan Maher for proofreading the paper.

7. REFERENCES

- [1] *DIN 3960 - Begriffe und Bestimmungsgren fr Stirnrder und Stirnradpaare mit Evolventenverzahnungen*, 1987.
- [2] *SIEMENS, SINUMERIK 840D sl / 840Di sl SINAMICS S120, Diagnosehandbuch*, 2009.
- [3] M. Bishop. *Computer Security: Art and Science*. Addison-Wesley Professional, 2003.
- [4] E. Byres and J. Lowe. The Myths and Facts behind Cyber Security Risks for Industrial Control Systems. In *VDE Congress 2004*, Berlin, Germany, 2004.
- [5] A. Dietrich, M. Schulze, S. Zug, and J. Kaiser. Visualization of Robot's Awareness and Perception. In *First International Workshop on Digital Engineering (IWDE)*, 2010. pages 38–44, ACM, Magdeburg, 14. June 2010.
- [6] J. Dittmann, B. Karpuschewski, J. Fruth, M. Petzel, and R. Mnder. An exemplary attack scenario - threats to production engineering inspired by the Conficker worm. In *First International Workshop on Digital Engineering (IWDE)*, 2010. pages 25–32, ACM, Magdeburg, 14. June 2010.
- [7] Forschungsprojekt ALF - Arbeiten und Lernen im Fachbereich. e-Learning in der Production. 2005. http://www.alf-projekt.de/alf_eLearning_production.pdf, last access: 22. Feb. 2011.
- [8] Fraunhofer IFF. Project "LiSA" - Assistant Robot in Life Science Company Labs. 2009. http://www.iff.fraunhofer.de/en/iffdben/Produkt_detail.php?ProduktId=66, last access: 21. Feb. 2011.
- [9] J. Fruth, C. Krtzer, and J. Dittmann. Design and Evaluation of Multi-Media Security Warnings for the Interaction between Humans and Industrial Robots. In *Electronic Imaging Conference 7575: Intelligent Robots and Computer Vision XXVIII: Algorithms and Techniques, IS&T/SPIE*, 2011. San Francisco Airport, CA, USA, 23.–27. Jan. 2011.
- [10] N. J. S. R. D. p. . Haegele, M. From Robots to Robot Assistants. In *Proceedings of the 32nd ISR (International Symposium on Robotics)*, pages 404–409, 2001.
- [11] KUKA Roboter GmbH. KR 210-2 (Series 2000). 2011. http://www.kuka-robotics.com/germany/de/products/industrial_robots/high/kr210_2.2000/, last access: 1. March 2011.
- [12] S. Lueders. (No) Security in Automation!?. In *VGB PowerTech*, volume 88, pages 127–130, Essen, Germany, 2008.
- [13] B. Matthias. Mensch-Roboter-Kooperation in der industriellen Praxis - Stand und Ausblick. In *12. IFF-Wissenschaftstage - Fachtagung: Sicherheit in der Mensch-Roboter-Interaktion*, 2009. 16.–18. Jun. 2009.
- [14] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Handbook of Robotics - Modelling, Planning and Control*. Springer, London, 2009.
- [15] N. Storey. *Safety-Critical Computer Systems*. Addison Wesley Longman Limited, 1996.
- [16] Symantec. Stuxnet Introduces the First Known Rootkit for Industrial Control Systems. *Symantec Official Blog*, Aug. 2010. <http://www.symantec.com/connect/de/blogs/stuxnet-introduces-first-known-rootkit-scada-devices>, last access: 21. Feb. 2011.
- [17] szenaris. Virtual Worlds. *szenaris homepage*. <http://www.szenaris.com/109-1-Virtual+Worlds.html>, last access: 22. Feb. 2011.
- [18] C. Walter and M. Schenk. Flexible Arbeitsraumberwachung mit ortsfesten und mobilen Sensoren fr die Mensch-Roboter-Koexistenz. In *12. IFF-Wissenschaftstage, Fachtagung: Sicherheit in der Mensch-Roboter-Interaktion*, 2009. 16.–18. Jun. 2009.
- [19] L. Westhoff. Aktuelle und zuknftige Nutzung von sicheren Robotern in der automobilen Produktion. In *14. IFF-Wissenschaftstage, Fachtagung: Sichere Mensch-Roboter-Kooperation*, 2011. 30. Jun. 2011.

Generative Software Visualization: Automatic Generation of User-Specific Visualizations

Richard Müller
rmueller@wifa.uni-
leipzig.de

Pascal Kovacs
kovacs@wifa.uni-
leipzig.de

Jan Schilbach
schilbach@wifa.uni-
leipzig.de

Ulrich W. Eisenecker
eisenecker@wifa.uni-
leipzig.de

Information Systems Institute
University of Leipzig
Leipzig, Germany

ABSTRACT

Software visualization provides tools and methods to create role- and task-specific views on software systems to enhance the development and maintenance process. However, the effort to produce customized and optimized visualizations is still high. Hence, we present our approach of combining the generative and the model driven paradigm and applying it to the field of software visualization. Based on this approach we want to implement a generator that allows to automatically generate software visualizations in 2d, 2.5d, 3d, or for virtual reality environments according to user-specific requirements.

Keywords

software visualization, model driven visualization, software visualization families, automation

1. INTRODUCTION

The preconditions for software visualization in 3d and virtual reality (VR) have improved dramatically, because of increased computing power available at low price and new presentation and interaction techniques. Our research tries to explore the resulting potential for software engineering, especially with respect to software development as well as maintenance.

Software visualization has the potential to considerably enhance understanding of software through providing structural, behavioral, evolutionary, or combined views [7]. This understanding is necessary for nearly all stakeholders involved in software development and maintenance, such as developers, project managers, and customers. All of these stakeholders have different tasks in different parts of the software lifecycle and therefore need different information about the software system they are involved in. Software visualizations have to support these users and their tasks, otherwise they are not useful and therefore will not be used. This task-oriented view was first proposed by Maletic et al. [12].

Reiss [16] identified some important issues of software visualization. Of these we address the lack of simplicity to use a visualization technique and the lack of adoption to real user problems. One reason for the lack of simplicity is, that visualization users need to supply exactly the data the visualization tool demands, because many tools require

a special input format which the user has to provide. With our approach we are able to handle multiple software artifacts in multiple input formats, as long as they are in a well-structured format. The second issue, the lack of adoption to real user problems, comes with the mostly general scope of actual visualization tools. Reiss [16] takes the resource usage in the time around a specific event as an example for a real world use case, which is not covered by the available visualization tools. We believe that the adoption of principles, methodology, and techniques of software system families is the basis for developing a generator that addresses these problems. This generator takes one or more software artifacts and an easy-to-create configuration of the desired visualization as input. Furthermore, it provides ready-to-use visualizations optimized to the users requirements without the necessity of additional user intervention. These visualizations optimally support the different user needs and therefore the specific tasks in the process of the software development and maintenance. Moreover, the visualizations produced by the generator support any form of visualization technique be it two-, two-and-a-half- or three-dimensional (2d, 2.5d or 3d), printed on paper, displayed on a monitor, or presented in VR.

In this paper, we will describe the theoretical concepts of our approach and the design of the generator. In Section 2, we summarize theories and publications our work is based on. In Section 3, we explain the generative visualization process. In this context, we will first introduce the basics of Generative Programming, especially the generative domain model, and derive a generative software visualization domain model. Afterwards, we outline a technology projection to show how this model can be instantiated. To explain the characteristics of this approach, an example scenario demonstrates the process of generative software visualization in Section 4. The conclusion gives a brief evaluation of the work described in this article and provides an outlook to future research.

2. RELATED WORK

To respect the task-specific needs of different users, many tools for software visualization, e.g. Mondrian [14], Code-City [17], or sv3d [13], allow to configure some aspects of the visualization. However, these tools are limited with respect

to their configuration options, e. g. the number of metaphors and layouts they offer. Furthermore, the configuration requires a substantial amount of additional manual work, or the tools are restricted to a single type of software artifact. Vizz3D [15] and Model Driven Visualization (MDV) [4] try to overcome these deficits with a more general approach.

Bull [4] describes MDV as an architecture for adapting the concepts of Model Driven Engineering (MDE). The software models used as input have to correspond to a platform independent metamodel, e. g. Dagstuhl Middle Model (DMM) [11]. Such a model can be retrieved by parsing sourcecode. To generate the visualizations, platform independent models called views are used, e. g. tree views or nested views. The definition of the necessary transformations between the input models and the view models have to be programmed by the user in a model transformation language, e. g. Atlas or Xtend¹. After the transformation, a platform specific visualization will be automatically generated for a certain tool, e. g. Zest².

As a weakness of this approach, we identified the necessary creation of platform independent view models from scratch. On the one hand, this creates a high level of freedom. On the other hand, many possible benefits are prevented, such as using a common layout algorithm for different view models. Another drawback is the use of complex multi-purpose model transformation languages which are not easy to understand for non-experts. We show that using a formalized domain specific language (DSL) to describe the mapping from source to view elements will be easier to use while still preserving the automatic generation of a visualization.

Panas et al. [15] describe Vizz3D as a framework for configuring a visualization by using models and transformations. Beginning with a formalized model of software corresponding to a metamodel defined by Vizz3D, the user first configures the mapping to an abstract view. This view has a graph structure with nodes and edges including properties such as color or shape. In a second step, the user configures the mapping of the view to a concrete scene rendered by a tool, including the configuration of a metaphor and an optional layout.

A limitation of this approach is the required transformation of the users data into the Vizz3D source format, which causes additional effort. The user has to define two mappings to configure a visual representation instead of only one mapping. Finally this results in a tight coupling of platform independent metaphors to platform specific visualization tools.

3. GENERATIVE SOFTWARE VISUALIZATION

3.1 Generative Paradigm

Generative Programming aims at the automatized production of software systems based on software system families:

“Generative Programming (GP) is a software engineering paradigm based on modeling software system families such that, given a particular requirements specification, a highly customized and

¹<http://www.eclipse.org/modeling/emf/>

²<http://www.eclipse.org/gef/zest/>

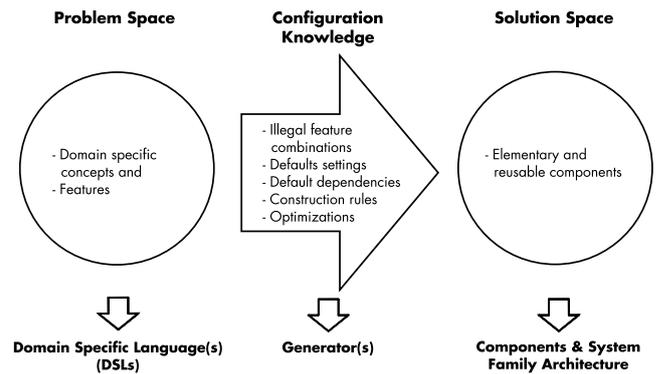


Figure 1: Generative Domain Model (GDM) [6]

optimized intermediate or end-product can be automatically manufactured on demand from elementary, reusable implementation components by means of configuration knowledge.” [6]

Such a family covers a set of systems being similar enough from an architectural point of view to be built from a common set of assets. The requirements of the resulting system are described with a DSL. In this context, a domain is an area of knowledge comprising expert knowledge from stakeholders and technical knowledge of how to build software systems. A DSL is a specialized and problem-oriented language for domain experts to specify concrete members of a system family. It abstracts from technical knowledge and implementation details. This specification is processed by a generator, which automatically assembles the system by combining elementary and reusable components according to configuration knowledge and a flexible system family architecture as well. Components are building blocks for assembling different systems of a family.

The basic terms of the generative paradigm and their relationships are summarized by the generative domain model (GDM, see Fig. 1). It comprises the problem space, the solution space, as well as configuration knowledge for mapping the problem space to the solution space. The problem space covers domain specific concepts as well as their features used by domain experts to specify their requirements. The requirements are expressed in terms of one or more DSLs. The solution space includes elementary and reusable implementation components which can be assembled as defined through the system family architecture. The configuration knowledge encapsulates illegal feature combinations, default settings, construction rules, and optimizations as well as related information.

The parts of a GDM and two or more GDMs can be connected in different ways [5]. One possibility is that the solution space of one GDM is the problem space of another GDM. This is called a chaining of mappings. Furthermore, specifications in a DSL can be processed by different generators which map them to different solution spaces. In this case, there are several alternative solution spaces instead of only one.

3.2 Generative Software Visualization Domain Model

Comparing the generative paradigm with the field of software visualization, especially its visualization process, yields

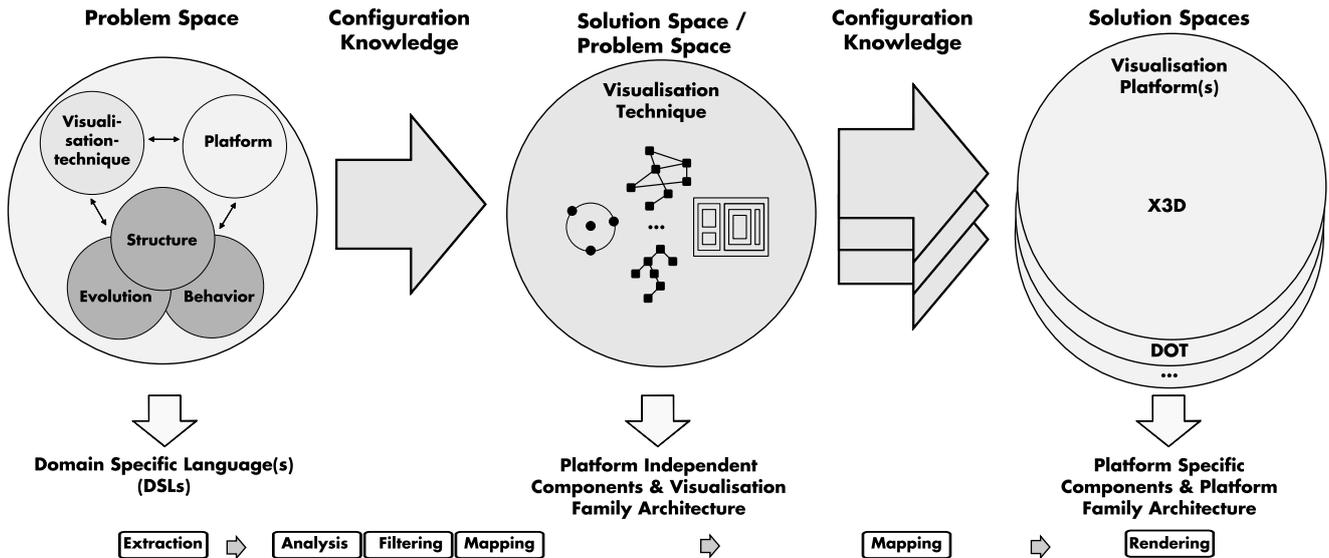


Figure 2: Generative Software Visualization Domain Model (GSVDM)

many remarkable similarities. A visualization should be automatically generated according to user-specific requirements by mapping information of software artifacts to a visual representation. For this reason, we adapt the definition of GP as follows:

“The visualization process should be arranged such that, given a particular requirements specification, a highly customized and optimized visualization can be automatically generated on demand from elementary, reusable implementation components belonging to a visualization family by means of configuration knowledge.”

The difference to the original definition is that the result of the generation process is not a software system but an optimized and ready-to-use visualization representing the structure, the behavior, and/or the evolution of a software system. Instead of assembling each visualization manually, it is created automatically from implementation components on the basis of a visualization family according to a specification in a DSL provided by the user.

This concept can be described in terms of the generative domain model. The chaining of mappings and the alternative solution spaces are used to realize the high variability of platforms for different visualization techniques. Besides that, the separation of the platform independent and the platform specific solution spaces makes it possible to reuse the implementation components. The resulting concept is called generative software visualization domain model (GSVDM, see Fig. 2).

The problem space offers means to specify concrete members of a visualization and a platform family for representing information from software artifacts. Using a DSL, the user can specify on which platform which information of a software artifact has to be visualized with which visualization technique. The DSL corresponds to the requirements specification in the above mentioned definition and abstracts from concrete implementations.

The solution space includes implementation components that can be assembled according to a family architecture.

Due to the chaining of mappings, there are at least two solution spaces. The platform independent space contains abstract visualization techniques, such as trees, graphs, tables, and abstract or real world metaphors. The platform specific spaces provide concrete platforms for these techniques, such as Graphviz [10], Tulip [2], Gephi [1], or X3D [3].

The configuration knowledge, which is implemented as a generator, defines the mapping of problem space to solution space. In this case, the generation process is also the visualization process. This means that the process corresponds to a fully automated visualization process comprising all necessary parts of a visualization pipeline [8]. Thus, the knowledge about illegal feature combinations, default settings, default dependencies, construction rules, and optimizations is augmented with knowledge about extraction, analysis, filtering, mapping, and rendering from the DSL.

3.3 Model Driven Technology Projection

In order to implement this theoretical concept, it is necessary to identify concrete techniques for the elements of the software visualization domain model. Consequently, all spaces are described with structured models, and the configuration knowledge provides mappings between these spaces using model-to-model transformations. The DSL can be implemented as text-only or as a dialogue-based wizard controlling the different steps of the visualization pipeline. By using the XText-Framework³ for implementing the DSL, we will be able to utilize existing functionality, to provide code completion, syntax highlighting and other useful features.

The starting point of the automatic visualization process are structured software artifacts containing information about structure, behavior, or evolution of software systems. Some of those structures – also known as metamodels – are Ecore, UML, XML or the DMM.

The common architecture for the visualization family is provided by a visualization technique meta-metamodel. It consists of a graph with nodes and edges where each element can have additional properties. The basic assumption

³<http://www.eclipse.org/Xtext/>

behind this is that all visualization techniques can be reduced to this abstract structure, so it is sufficient to have only one model that can be flexibly instantiated. Another advantage of using a graph structure is the ability to employ existing 2d and 3d layout algorithms rather than implementing them. The different platform independent visualization techniques are the implementation components.

It is obvious that the common architecture for the visualization platform depends on the used platform. For this reason, there is one model for each platform. Imagine the visualization should be an X3D-scene. Then, the X3D-file is the model, the X3D-schema definition is the metamodel, and the XML schema definition is the meta-metamodel. Further examples for visualization platforms are DOT⁴ from Graphviz, TLP⁵ from Tulip, or GEXF⁶ from Gephi.

The configuration knowledge maps the elements of software artifacts to the elements of visualization techniques and finally to elements of a visualization platform corresponding to the DSL. The steps of the visualization pipeline are implemented by means of the Eclipse Modeling Framework. In this way, formal models can be analyzed and checked with predefined validation rules. For the mapping, i. e. model-to-model transformations, the transformation rules are defined on the meta-level and they are applied for each model conforming to the corresponding metamodel. In order to handle more than one source model the so called model-weaving is used. The rendering is done by the concrete visualization platforms.

4. EXAMPLE SCENARIO

In order to illustrate our approach we want to use a simple fictional scenario. Imagine a project manager who is preparing a meeting. In order to make the development team pay attention to current problems, information of the software system's structure enhanced with metrics is required. The system under development is a banking system implemented in Java and the relevant metrics are McCabe Complexity and LOC. To make it more understandable for all stakeholders, the manager waives source code and complex tables. Instead, the visualization should be 3d and represented by a nested visualization technique, which can be explored interactively in the company's virtual reality environment.

To do this in a generative way, the following steps have to be carried out. As a precondition, the necessary information has to be available in formal models, e.g. an Ecore model for the structure and an XML file for the metrics. Initially, the models are loaded by the generator (*extraction*). If necessary, the user can apply predefined rules to check the models for consistent semantics (*analysis*). Then, the relevant information from the software artifacts is selected (*filtering*), and a visual representation for each piece of information is defined by the user (*mapping*). This mapping comprises two stages: In the first stage, the visualization technique is selected, and in the second stage, the visualization platform is chosen. During these stages, the user sets the mapping rules for packages, classes, methods, attributes, and references to clusters, nodes, and edges as well as the visual appearance including shape, size, and colour of the different types of clusters, nodes, and edges. Clusters are special nodes, that

can contain further clusters or nodes. The mappings are either completely controlled by the user or default mappings are applied. The simplified mapping rules in the banking example are as follows:

- package ↪ cluster ↪ brown cube
- class ↪ cluster ↪ blue sphere
- method ↪ node ↪ green to red cylinder
- attribute ↪ node ↪ blue cone
- reference ↪ edge ↪ blue line
- McCabe ↪ node ↪ green(low) - red(high)
- LOC ↪ cluster/node ↪ size

At this time, only the size and positions of the elements in the 3d space are missing. Here the graph structure from the meta-metamodel comes into play. By applying established layout algorithms the missing information is computed. For the banking example a force directed layout algorithm for clustered graphs is used [9]. Now, the generator has all necessary information to produce the visualization. As a result, the X3D-model in Fig. 3 and 4 is automatically generated

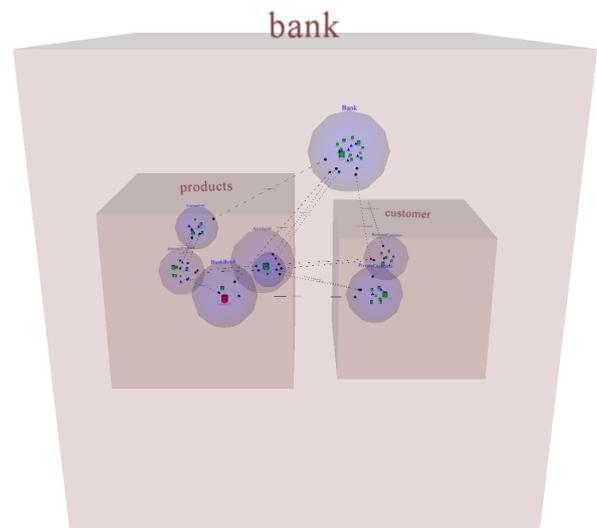


Figure 3: X3D-model of the banking example including structure and metrics (Overview)

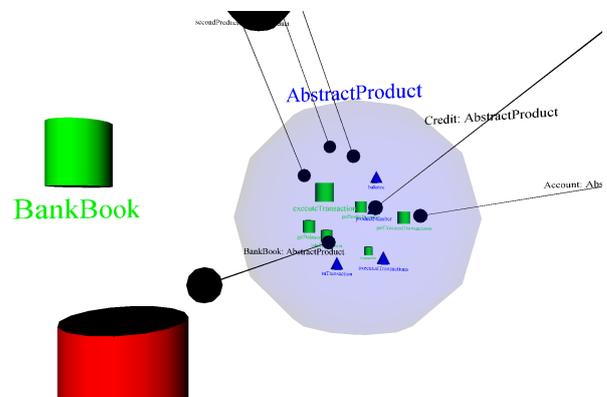


Figure 4: X3D-model of the banking example including structure and metrics (Detail)

⁴<http://www.graphviz.org/content/dot-language>

⁵<http://tulip.labri.fr/TulipDrupal/?q=tlp-file-format>

⁶<http://gexf.net/>

and can be interactively explored in a suitable browser, such as an Eclipse view, a standalone X3D-Browser, or a virtual reality environment. These visualizations have been generated using a predecessor of the planned generator. In this predecessor a box- and solar-system-metaphor as well as X3D as target platform are hard-wired. By this means, it was possible to visualize the structure of a real-world example with several hundred classes.

5. CONCLUSION AND FUTURE WORK

It was explained how the generative paradigm and the model driven paradigm can be adopted to meet the requirements of generating highly customized and ready-to-use software visualizations by the user without writing any glue code by hand. Hence, this promising concept makes it possible to integrate different kinds of software artifacts with different visualization techniques and well-approved visualization tools, not being limited to a specific platform and configured by an easy-to-use DSL.

Our future work will continue the implementation of the generator architecture and infrastructure based on the Eclipse platform, the specification of the grammar of the DSL, the iterative development of the meta-metamodel for visualization techniques including a representative amount of metamodels of visualization techniques as well as the integration of some established visualization tools. Eventually, we plan to use the generator to evaluate different visualization aspects, like three-dimensionality, animation, and interaction for their suitability in different tasks and different stages of the software life cycle. With the resulting findings we want to improve the spread of task- and role-specific software visualization in industrial software development and maintenance.

6. REFERENCES

- [1] *Gephi: An Open Source Software for Exploring and Manipulating Networks*, 2009.
- [2] D. Auber. Tulip : A huge graph visualisation framework. In P. Mutzel and M. Jünger, editors, *Graph Drawing Softwares*, Mathematics and Visualization, pages 105–126. Springer-Verlag, 2003.
- [3] D. Brutzman and L. Daly. *X3D: Extensible 3D Graphics for Web Authors*. Elsevier, 2007.
- [4] R. I. Bull. *Model Driven Visualization: Towards a Model Driven Engineering Approach for Information Visualization*. PhD thesis, University of Victoria, 2008.
- [5] K. Czarnecki. Overview of generative software development. In *Unconventional Programming Paradigms*, number 3566 in Lecture Notes in Computer Science, pages 326–341. Springer, Berlin Heidelberg, 2005.
- [6] K. Czarnecki and U. W. Eisenecker. *Generative Programming. Methods, Tools and Applications: Methods, Techniques and Applications*. Addison-Wesley Longman, Amsterdam, June 2000.
- [7] S. Diehl. *Software Visualization - Visualizing the Structure, Behaviour, and Evolution of Software*. Springer, 2007.
- [8] S. dos Santos and K. Brodlie. Gaining understanding of multivariate and multidimensional data through visualization. *Computers & Graphics*, 28(3):311–325, June 2004.
- [9] T. Dwyer. Extending the WilmaScope 3D graph visualisation system: software demonstration. In *APVis '05: proceedings of the 2005 Asia-Pacific symposium on Information visualisation*, pages 39–45. Australian Computer Society, Inc., 2005.
- [10] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE*, 30(11):1203–1233, 2000.
- [11] T. C. Lethbridge, E. Plödereder, S. Tichelaar, C. Riva, P. Linos, and S. Marchenko. The dagstuhl middle model (DMM). <http://www.site.uottawa.ca/~tcl/dmm/DMMDescriptionV0006.pdf>, 2002.
- [12] J. I. Maletic, A. Marcus, and M. L. Collard. A task oriented view of software visualization. In *VISSOFT 2: Proceedings of the 1st International Workshop on Visualizing Software for Understanding and Analysis*, pages 32–40. IEEE Computer Society, 2002.
- [13] A. Marcus, L. Feng, and J. I. Maletic. Comprehension of software analysis data using 3D visualization. In *Proc. 11th IWPC*. IEEE Computer Society, 2003.
- [14] M. Meyer, T. Girba, and M. Lungu. Mondrian: an agile information visualization framework. In *Proceedings of the 2006 ACM symposium on Software visualization*, SoftVis '06, pages 135–144, Brighton, United Kingdom, 2006. ACM. ACM ID: 1148513.
- [15] T. Panas, R. Lincke, and W. Löwe. Online-configuration of software visualizations with vizz3d. In *Proc. 2nd SoftVis*, pages 173–182, New York, NY, USA, 2005. ACM.
- [16] S. P. Reiss. The paradox of software visualization. In *Proc. 3rd VISSOFT*, pages 59–63, 2005.
- [17] R. Wetzel, M. Lanza, and R. Robbes. Software systems as cities: a controlled experiment. In *Proceeding of the 33rd international conference on Software engineering*, ICSE '11, pages 551–560, Waikiki, Honolulu, HI, USA, 2011. ACM. ACM ID: 1985868.

Evaluation of Techniques for the Instrumentation and Extension of Proprietary OpenGL Applications

Maik Mory, Mario Pukall, Veit Köppen, Gunter Saake
 Otto-von-Guericke University, P.O. Box 4120, D-39016 Magdeburg, Germany
 (mory|pukall|vkoeppen|saake)@ovgu.de

ABSTRACT

Instrumentation of interfaces is a popular design pattern in engineering. Academic and industrial projects are already using instrumented OpenGL clients for various purposes. We perceive instrumentation of proprietary OpenGL applications as a basic technology to open up interactive three-dimensional graphics as a potent interoperability platform for heterogeneous simulation software in engineering. Hence, we describe and compare four instrumentation techniques on the MS Windows platform: relink library, dynamic library replacement, virtual display driver, and binary interception. We qualitatively evaluate them for four capabilities: to instrument proprietary simulation software; to instrument a subset only of the OpenGL interface; to instrument multiple interfaces simultaneously; and to chain intermediaries. The relink library technique is powerful, except that it cannot be used with proprietary simulation software. Dynamic library replacement and virtual display drivers potentially support all features, although some features are difficult to implement. The binary interception technique inherently supports all capabilities. We conclude with directions for future research.

Keywords

I.3.4 [Computer Graphics]: Graphics Utilities — Software support, Virtual device interfaces; D.2.12 [Software Engineering]: Interoperability — Distributed objects; J.2 [Physical Sciences and Engineering]: Engineering, Digital Engineering; I.6.8 [Simulation and Modeling]: Visual Simulation

1. INTRODUCTION

The digital revolution has changed the domain of engineering. Product values are determined increasingly by the software they contain. Even more, almost every engineering process is supported by computer simulations today. An important issue in digital working environments is interoperability between heterogeneous software components. We focus on interoperability between interactive simulations. The naive way to couple components is to implement one distinct adapter for each coupling wanted by the stakeholders. If we consider a project's size in number of components or products used, in the long term the naive approach results in the development of $O(n^2)$ adapters. Thus, the naive approach does not scale. Therefore, many projects define an interoperability platform. The interoperability platform captures a common concept, which is shared between components. Couplings that are wanted by the stakeholders are implemented by one adapter to the interoperability plat-

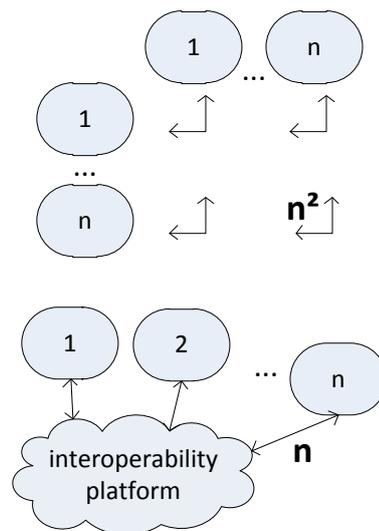


Figure 1: The curse of interoperability: Does a project that uses n applications implement $O(n^2)$ adapters; or is there a common concept, which serves as interoperability platform for $O(n)$ adapters?

form for each component. Because adapters can ideally be reused for different couplings, the interoperability platform approach results in the development of $O(n)$ adapters. Figure 1 illustrates the two strategies. The design of valuable interoperability platforms has become a critical concern in every bigger project.

The critical point when establishing an interoperability platform is to identify a common concept of candidate components. We want to improve interoperability of simulations in engineering. The question for an interoperability platform in industrial engineering software finds no answer since decades, e.g., [17, 26, 27]. Moreover, we do not want to establish yet another file based interoperability platform. We want to enable qualitative higher levels of interoperability through interactivity: components should exchange three-dimensional data during runtime, so that users are able to achieve information and gain knowledge in real time. Today, interactive visualization of simulation data is an omnipresent feature in simulation software. Often, simulation software uses OpenGL for interactive visualization of simulation data. We believe that the OpenGL industry standard serves very well as a common concept for interoperability of distributed virtual simulations [22].

As a first step towards the envisioned, OpenGL based interoperability platform we have to check, how to plug into the OpenGL pipeline. We focus on the pattern of instrumentation. Instrumentation by now primarily is known in performance analysis and debugging. But once the stream of function calls is tapped, it is only a matter of technical creativity which further applications can be implemented [6]. In the following sections, we describe a concrete application scenario and provide a distilled problem statement. Based upon component based system design, we work out an evaluation scheme with attestable properties. The properties are: capability to work with proprietary simulation software, to instrument interface subsets, to instrument multiple interfaces, and to chain instrumentation software. Then, we examine four instrumentation techniques for these properties. The techniques are relink library, dynamic library replacement, virtual display drivers, and binary interception. Finally, we conclude and discuss future directions of our research.

1.1 Application Scenario

Our scenario takes place in an engineering office environment. The project uses a heterogeneous setup: Microsoft Windows XP, Vista, and 7, which regarding our research question behave quite similar. The reference development platform for the operating systems is Microsoft Visual Studio.

Within the office environment, engineers use interactive simulation software (e.g., Matlab/Simulink¹) that renders three-dimensional data based upon results of simulations. In the engineering domain simulation software mostly is proprietary or it is developed as a side project with scarce resources. Thus, we consider the simulation software to be proprietary. The visualization component communicates with the OpenGL application programming interface. Note, the visualization component is not necessarily a dominant or permanent element of the user interface; we only require it to be available. So far, the application scenario resembles the state of the art as common in every engineering office.

The engineers in our scenario now bear a challenge, when they want to modify the rendering behavior of the proprietary simulation software. We represent the modification of the simulation software, as we introduce notional analysis software, which profiles access on databases. For real-time presentation of the profile data, OpenGL should be used as pragmatic interoperability platform (see Figure 2). This means, that the analysis software visualizes aggregate data and embeds its visualization into the simulation's visualization. Therefore, the analysis software has to instrument the simulation's database interface, what we do not discuss in detail. Furthermore, it has to instrument the simulation's OpenGL interface. More precisely, within the OpenGL interface the analysis software instruments the `SwapBuffers` function [19], which semantically indicates, that the simulation has done its rendering work and the new window content should appear on the screen. The instrumented variant of the function renders the analysis software's graphics, which are a stream of OpenGL commands, into the simulation's graphics; finally, it invokes the original `SwapBuffers` function to put the merged graphical content onto the screen.

¹The Mathworks Inc. provides the Matlab/Simulink simulation suite, which is very popular in the engineering domain. Detailed product descriptions are provided at the vendor's website (<http://www.mathworks.com/products>).

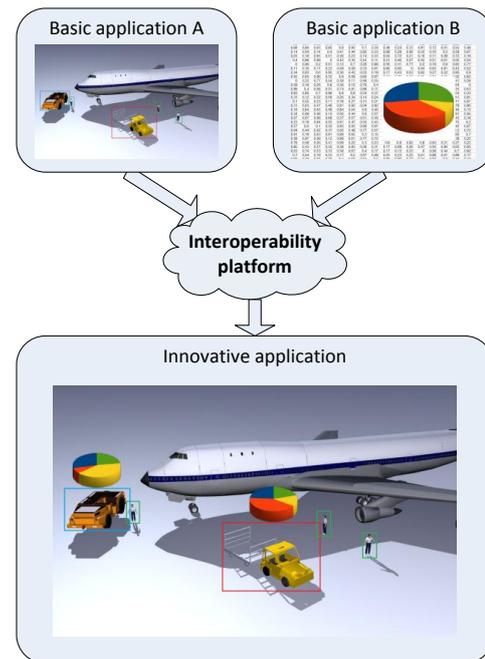


Figure 2: In our application scenario, we assume three-dimensional graphics as interoperability platform. This paper discusses instrumentation of proprietary OpenGL applications as basic technology.

1.2 Problem Statement

Based upon our application scenario with simulation software that should be instrumented, we ask the following questions:

- Which techniques are available for instrumentation of OpenGL?
- Which techniques are applicable to proprietary simulation software?
- Which technique is preferable, if a narrow subset of the OpenGL interface should be instrumented?
- Which technique is preferable, if multiple independent interfaces should be instrumented to implement interleaved behavior?
- Which technique is preferable, if proprietary intermediaries should be chained?

We do not discuss the inner functionality of the instrumentation software. There is already a lot of research about stylized rendering (e.g., [21]), distributed rendering (e.g., [3, 12, 14, 22]), and GPU debugging (e.g., [9, 24]) through instrumented OpenGL. In contrast to the literature that we are aware of we focus on the mechanics, how instrumentation software is attached to simulation software.

2. BACKGROUND

This section introduces terms and concepts, which are necessary to evaluate techniques for the instrumentation of proprietary OpenGL applications. Therefore, we declare a component and interface oriented notation together with an

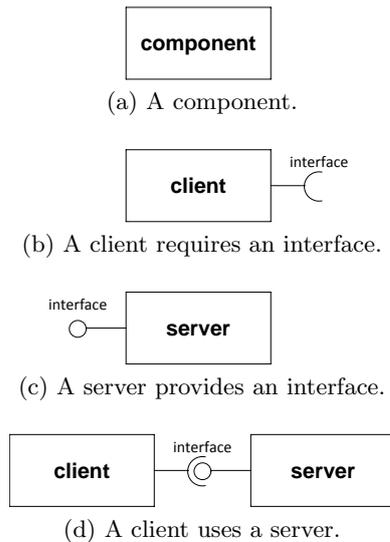


Figure 3: Definition of a simple component diagram notation based on UML 2

interoperability hierarchy. Building on that terminology, we discuss the instrumentation pattern and which properties are necessary, to cover our application scenario. We describe relevant implementation details of OpenGL on the MS Windows platform. Finally, we introduce four instrumentation techniques, which we evaluate in Section 3.

2.1 Modular Systems and Interoperability

Engineers describe system architectures in terms of components and interfaces. Depending on the engineer's domain and preferred method, what we call a component may be called an object, device, service, module, or other too. We focus on the domain of computer science. In our figures, we use a simplified dialect of UML 2's component diagram notation [11], which we present in Figure 3. A component is a definable software artifact. A server component provides an interface. A client component uses an interface. If a client's required interface and a server's provided interface are compatible, they can be connected. Then, the client uses the server.

Connections of components are differentiated between tight couplings and loose couplings. Tight coupling exploits dependencies and relations between the server and the client; the connected components are not supposed to be exchanged. A loose coupling minimizes dependencies and relations between client and server to a well-defined specification of the interface; loosely coupled components tend to be exchangeable. In real life, couplings are not clearly the one or the other. Rather, real couplings distribute in a continuum with ideal loose coupling on one end and with ideal tight coupling on the other end. The distinction is made, whether an instance is more the one or the other.

Loose couplings are the subject of interoperability. Interoperability is a field of active research. The most exhaustive, recent survey we know of was done by Manso et al. [17]. They declare seven levels of interoperability: technical, syntactic, semantic, pragmatic, dynamic, conceptual, and organizational. In our context it is sufficient to stick with a three level hierarchy of interoperability [16], which we briefly introduce as follows:



Figure 4: An instrumented connector has an intermediary component in the connection between client and server.

Syntactic interoperability is data exchange with a common set of symbols to which a formal grammar applies.

Semantic interoperability is information exchange with a shared, common vocabulary for interpretation of the syntactic terms.

Pragmatic interoperability is contextual exploitation of applications and services through shared knowledge.

An interface definition covers a subset of the interoperability hierarchy. Most interface definitions in computer science, especially application programming interfaces' documentations focus on syntactic and semantic interoperability. Software developers usually delegate technical interoperability to electrical engineers, who design computer chips and network links. The upper half of the interoperability hierarchy usually is in the responsibility of software project's stakeholders. The specification of OpenGL² defines syntax through function signatures together with a finite state machine and it defines semantics through human readable documentation for modules and functions.

2.2 Instrumented Interfaces

Instrumented connectors are a popular design pattern in engineering. For example in the domain of computer science, the Decorator, Proxy, and Composite design patterns [7] are object oriented interpretations of the instrumentation pattern. Given a connected client and server, a component commonly known as intermediary is inserted into the connector. The intermediary is the client's new server and the server's new client. Thus, an intermediary provides and requires the interface that connects the client and the server. Figure 4 depicts an instrumented connector. Instrumentation adds functionality to a given system. Thus, it can serve many purposes: Most instrumented connectors inspect the connector while the system is running; often, just like in our application scenario, connectors are instrumented to modify the behavior of the system [8].

Based upon the property that ideal loosely coupled components are interchangeable, Grechanik et al. [10] state, that an instrumented connector should be non-invasive and idempotent. They define an instrumented connector non-invasive, if it is undetectable in a connector's implementation. In today's complex world of software products with multi project source code bases, embedded scripting languages, and the "code is data" paradigm, Grechanik's definition is hard to decide for any real example. Therefore, we propose a pragmatic definition: An instrumented connector is non-invasive, if it works with proprietary client and server. Note that a free simulation model and script interpreted by proprietary infrastructure still builds up a proprietary software

²The Khronos Group provides the specification of OpenGL in a set of documents on their website (<http://www.opengl.org/documentation/specs/>).

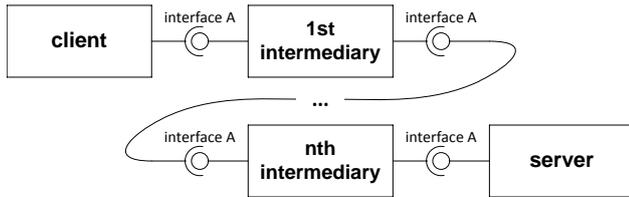


Figure 5: If intermediaries are idempotent, then they can be chained.

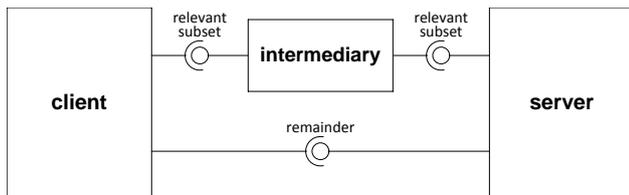


Figure 6: When a subset of the interface is instrumented, the remainder of the interface should be kept available to preserve interoperability between client and server.

product. If intermediaries can be chained because they have equal import and export interfaces and because they are non-invasive, then they are called idempotent [10]. Figure 5 shows chained intermediaries. We stick with Grechanik's definition of idempotency together with our adapted understanding of non-invasiveness. In our opinion, idempotency is the definite criterion whether a component matches the instrumentation pattern.

Interfaces can be composed of interfaces and conversely be part of other interfaces. OpenGL's formal interface definition indicates several decompositions. For example the notorious OpenGL extensions are self-contained subsets of the interface. In our application example in Section 1.1, we outline the instrumentation of a minimal interface subset, which consists of one function. Ideally, an instrumentation technique should be able to put an intermediary into a relevant subset of the interface while it keeps the remainder of the interface available to preserve interoperability between client and server. Figure 6 illustrates our requirement for interface subset instrumentation.

An intermediary not necessarily restricts to a singular, well defined interface. Sometimes, the intermediary should implement behavior that is interleaved between multiple interfaces. In our application scenario in Section 1.1, we want to interleave the behavior of two interfaces: the database application programming interface and the OpenGL application programming interface. If one intermediary instruments more than one connector, we call this multiple interface instrumentation. Figure 7 illustrates an intermediary, which implements cross-cutting functionality on multiple interfaces.

2.3 OpenGL on Microsoft Windows

OpenGL is part of MS Windows' software development environment. The OpenGL application programming interface is provided for all languages of Microsoft Visual Studio (e.g., C/C++, C#, Java). Because .NET and other script languages are built on top of the C/C++ toolchain, we constrain our discussion to the latter.

Developers and compilers access the syntactical interface

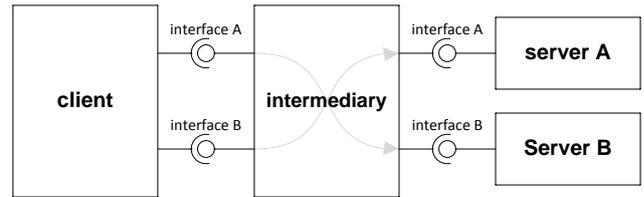


Figure 7: If the behavior of multiple connectors should be interleaved, they have to be instrumented with one intermediary.

definition of OpenGL through the `gl/gl.h` header file. The source code of the simulation software contains requests of OpenGL functions based upon the function definitions from the header file. At the time of this writing the OpenGL specification³ defines 2269 functions. The windowing system interface to the Microsoft Windows implementation of OpenGL (WGL⁴) adds 131 function definitions. Thus, an OpenGL application on Windows has access to a repository of 2400 functions. Microsoft Windows' OpenGL implementation is determined to be compatible with OpenGL version 1.1. Hence, the 2400 functions divide into three sets: 357 core functions, 1671 extension functions, and 372 alias functions. Core functions are directly accessible C/C++ routines. For extension functions the signature only is declared. A client has to request the procedure's entry point via the `wglGetProcAddress` function [20] before it can invoke the extension function. The server is not obligated to provide all extension functions; it might return the `NULL` value to indicate, that a particular extension function is not supported. Alias functions are identifiers that actually refer to one of the core or extension functions.

Windows establishes several abstraction layers between OpenGL client software and OpenGL server implementations (see Figure 8). During the compilation of the software, the compiler translates source code symbols to library symbols. In the linker step, the library symbols are resolved, so that the final binary executable image contains binary function code from `opengl32.lib` together with the simulation's binary code, which invokes the OpenGL functions. The `opengl32.lib` is a stub that redirects core function calls to the functions exported by the `opengl32.dll` dynamic library's symbol table during runtime. Extension functions are passed through the `wglGetProcAddress` function as it is stubbed by the static library likewise. While the linked functions of the static library are integral part of the simulation software, the dynamic library is searched, loaded and linked by the `LoadLibrary` function [18] during the initialization phase of the simulation's runtime. The `opengl32.dll` dispatches the application's function calls to a matching display driver. The concrete workings of this mechanism are an implementation detail of Windows' WGL facility. In our application scenario it is sufficient to assume that there is one screen that is driven by one graphics hardware with one display driver.

³We used the OpenGL specification as published on <http://www.opengl.org/registry/api/gl.spec> revision 12819 with the timestamp 2010-11-03 19:02:01.

⁴We used the WGL specification as published on <http://www.opengl.org/registry/api/wgl.spec> revision 10796 with the timestamp 2010-03-19 17:31:10 and <http://www.opengl.org/registry/api/wglext.spec> revision 12183 with the timestamp 2010-08-06 02:53:05.

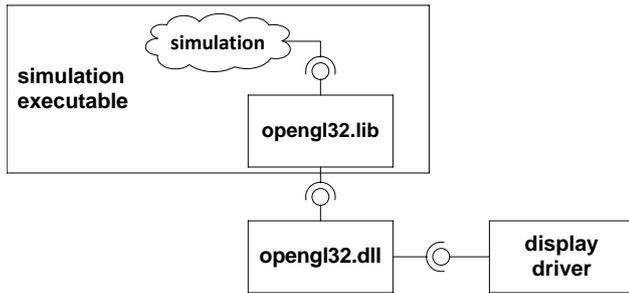


Figure 8: The OpenGL pipeline from a client, which requests visualization routines, to the server, which actually performs the rendering work, has several levels of indirection on Microsoft Windows.

2.4 Instrumentation Techniques

In this section, we describe four techniques that plug an intermediary into a simulation application: relink library, dynamic library replacement, virtual display drivers, and binary interception. We decrease redundancy in the descriptions as we anticipate an implementation detail that is common to each of the four instrumentation techniques. Extension functions are accessed through the `wglGetProcAddress` function. Therefore, it is sufficient to instrument extension functions by on-the-fly substitution of function addresses. One instruments the `wglGetProcAddress` function with an implementation, which returns intermediary’s extension functions to the client and stores server’s extension functions for later delegation. The `wglGetProcAddress` function is a core function. Thus, in the following it is sufficient to focus on instrumentation of core functions.

An intermediary that uses the **relink library** technique (e.g., [5]) is injected during compile time of the client software. In its simplest form, which gave the technique its name, a developer reconfigures the linker not to link the original `opengl32.lib` but to link the intermediary static library instead. The intermediary provides all symbols that the original library provided. With this simple approach, the intermediary static library cannot directly utilize the original static library, because otherwise there would be a name clash in the linker step. Today, one avoids the name clash issue by a modification of the relink library technique. A developer reconfigures the compiler not to use the original header files but to use intermediary’s header files. The intermediary’s header files define the OpenGL API for the client’s source code but link to distinct symbols for the intermediary’s static library instead. The implementation of the intermediary uses original header files, which link to the original static library. Throughout this paper, the shorthand term “relink library” refers to the latter, improved relink library technique.

An intermediary that uses the **dynamic library replacement** technique (e.g., [9, 15, 21, 22, 24], esp. [12]) is injected when the client software attempts to load the `opengl32.dll` dynamic library. The execution environment of the client process is configured, so that the call `LoadLibrary("opengl32")` does not load the original dynamic library but loads the intermediary dynamic library. For this, either the operating system’s dynamic library is replaced on the file system; or the intermediary’s dynamic library is in a place of `LoadLibrary`’s search sequence [18] prior to the original one. The intermediary library has to keep a reference to the location of

the original dynamic library. If the library is replaced on the file system, the intermediary dynamic library loads the original library with the qualified path to the original library’s backup. If the library has been put into `LoadLibrary`’s search sequence, the intermediary library loads the original library with the qualified path to the original dynamic library in the operating system installation. The intermediary then uses the symbols from the original dynamic library, which was loaded by qualified path, to invoke the original server.

A **virtual display driver** (e.g., [1, 2, 23], esp. [25]) is injected through the display driver framework of the operating system. The operating system maps displays to device drivers. The virtual display driver mimics a display. Client software that is executed on a virtual display is associated with the underlying virtual display driver. The virtual display driver invokes the original server through a proxy process, which is spawned on the display that is associated with the original display driver.

The technique of **binary interception** was designed by Hunt and Brubacher [13] with the intention to instrument and extend proprietary software. The intermediary manipulates the software’s binary image during runtime. For each function that should be instrumented, the intermediary installs a detour. The installation procedure for a function overwrites the first bytes of the server’s function with bytecode, which detours the execution path to the intermediary’s function. When the client invokes an intercepted server’s function, the overlaid detouring code is executed instead of the original code. In effect, the client invokes the intermediary’s function. The installation procedure produces a so-called trampoline, which keeps the original server’s function available. The trampoline contains a backup of the server function’s bytecode that was overwritten during installation of the detour and additional bytecode that repatriates the execution path to the unmodified remainder of the server function’s bytecode. In effect, invocations of the trampoline delegate calls to the server.

3. EVALUATION

In this section, we evaluate the four instrumentation techniques, which we introduced in Section 2.4, with the properties that we developed in Section 2.2. Because the instrumentation of extension functions is common to all of the four techniques, we anticipate its evaluation. We assume that the `wglGetProcAddress` core function is instrumented. If a subset of the extension functions should be instrumented, the intermediary substitutes the procedure addresses in the subset; in the remainder, it passes through the original addresses. The instrumentation of extension functions is not related to the instrumentation of interfaces other than OpenGL. Therefore, it has no effect for a technique’s ability to instrument multiple interfaces. If a chained intermediary uses an instrumented `wglGetProcAddress` function, it substitutes and uses the procedure addresses to instrumented extension functions, which are provided by the preceding intermediary. Thus, instrumentation of extension functions through procedure address substitution does not break idempotency. Hence, without loss of generality, we focus on the instrumentation of core functions, when we evaluate the four techniques in the following.

The **relink library** technique requires access to the client’s code base. Thus, it is not possible to instrument proprietary clients with this technique. For the instrumentation of an

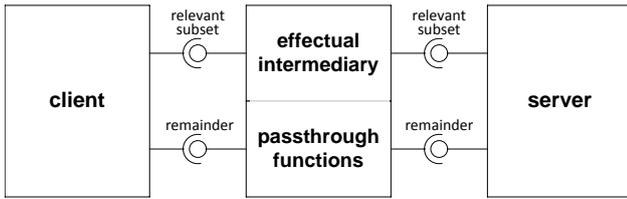


Figure 9: Dynamic library replacement and virtual display drivers do not inherently support instrumentation of subset interfaces. They need passthrough functions as workaround.

interface subset, the injected header file redefines the relevant symbols only. Functions that should not be instrumented are duplicated from the original OpenGL header file, which puts the original server’s core functions into effect. Thus, it is easy to instrument a subset. If multiple injected header files point their symbols to the same software artifact, they instrument multiple interfaces with one monolithic intermediary. Thus, the relink library technique allows for instrumentation of multiple interfaces. An intermediary’s implementation generally uses the original OpenGL header file to delegate calls to the original server. If one intermediary’s header file is injected into another intermediary’s implementation, then the latter one delegates its calls to the first one. Thus, the relink library technique is idempotent.

Dynamic library replacement does not require access to the client’s code base. Hence, it is suited for proprietary simulation software. The intermediary has to provide all functions of the OpenGL API to preserve interoperability with the client. For core functions that should not be instrumented, the intermediary has to export symbols for passthrough functions. Figure 9 illustrates how passthrough functions are used to pad the interface. Thus, instrumentation of an interface’s subset is possible. The interface provided by a dynamic library is well defined. Therefore, instrumentation of multiple interfaces with dynamic library replacement requires injecting distinct dynamic libraries for each of the multiple interfaces. These separate dynamic libraries, which are loaded into the process, frame a distributed system as depicted in Figure 10, which produces its own additional interoperability issues. Thus, we notice that instrumentation of multiple interfaces through dynamic library replacement is possible but elaborate and error prone. The chaining of proprietary intermediaries is limited by the way, how the library loader works. First, the intermediary only can be applied if the client does not reference the OpenGL dynamic library through a fully qualified path. Then, the intermediary references the original server through its fully qualified path. If we apply more than one intermediary, then one of them comes into effect first. This prioritized intermediary directly loads the original server through the fully qualified path, which effectively skips the other intermediaries. Figure 11 illustrates how one intermediary disables other intermediaries through a short circuit to the original server. Thus, dynamic library replacement is not idempotent.

Virtual display drivers do not require access to the client’s code base. Thus, virtual display drivers work with proprietary OpenGL clients and are non-invasive. If a virtual display driver instruments a subset of the interface, it still has to provide all other functions of the interface, because otherwise interoperability with the client would break. Therefore,

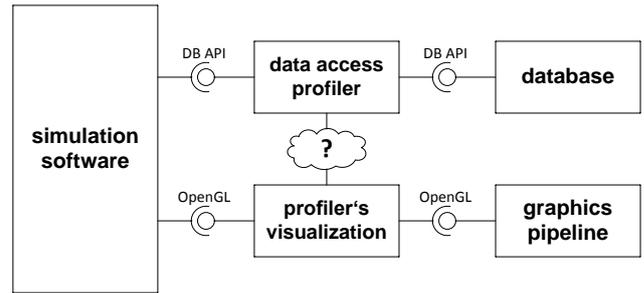


Figure 10: Dynamic library replacement and virtual display drivers do not inherently support instrumentation of multiple interfaces. They need distributed intermediaries as workaround, which produces additional interoperability issues.

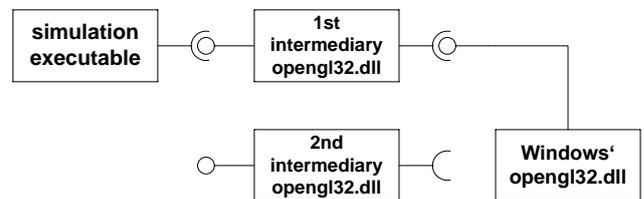


Figure 11: If dynamic library replacement is not done carefully, chained intermediaries break with a short circuit to the original dynamic library.

it has to implement passthrough functions for the remainder of the core interface as depicted in Figure 9. Thus, interface subset instrumentation is not inherently supported but possible. Whether virtual or not, a display driver provides display functionality. If multiple interfaces should be instrumented, each function that is not related to display drivers has to be handled by another intermediary than the virtual display driver. Comparable to multiple interface instrumentation with dynamic library replacement, this introduces a distributed system of intermediaries, which produces additional interoperability issues (see Figure 10). Thus, virtual display drivers are not suited for multiple interface instrumentation. An intermediary virtual display driver spawns a proxy client application on another server’s display. Whether the server is the original server or a chained intermediary virtual display driver makes no difference. Thus, virtual display drivers are idempotent.

The **binary interception** technique does not require access to the client’s code base. Therefore, it can be used with proprietary simulation software. The instrumentation of an interface’s subset is easy. One installs detours for every function that should be instrumented. The other functions, which should not be instrumented, remain untouched. A detour can be installed for every function that is provided by any server. One monolithic intermediary can install detours into an arbitrary interface inside the running process. Thus, the instrumentation of multiple interfaces is easy, too. In Section 2.4 we describe instrumentation through binary interception for the first intermediary. We show idempotency, as we assume, that there are already detours installed. The installation of yet another detour exactly overwrites the old detouring code with new detouring code. The new trampoline contains the detouring code of the previous intermediary

technique	proprietary client	interface subset	multiple interfaces	idempotency
Relink Library	--	++	++	++
Dynamic Library Replacement	++	+	-	-
Virtual Display Driver	++	+	-	++
Binary Interception	++	++	++	++

Table 1: The instrumentation techniques’ strengths and weaknesses in a shorthand comparison. The symbols are used as follows: (++) the technique inherently supports the feature; (+) the technique needs a workaround to support the feature; (-) the technique basically supports the feature with severe restrictions; (--) the feature is inherently not supported by the technique.

in its bytecode backup. In effect, a client’s call gets detoured to the last installed intermediary. An intermediary’s trampoline function invokes the detouring code of the intermediary that was installed previously. This pattern continues until the execution path arrives at the first installed intermediary. The first installed intermediary’s trampoline contains the original server’s function. Thus, binary interception is idempotent.

We recapitulate our results in Table 1. The relink library technique does not work with proprietary simulation software as OpenGL client. With dynamic library replacement it is hard to instrument multiple interfaces. Moreover, it is nearly impossible to achieve idempotency for proprietary intermediaries with dynamic library replacement. Virtual display drivers do not support instrumentation of interfaces that are not related to display functionality. Binary interception fulfills all declared requirements.

4. CONCLUSION AND OUTLOOK

In this paper, we define an application scenario, which we believe resembles the near future of pragmatically interoperable, distributed interactive simulations. Instrumentation of the OpenGL API is a basic technology in this application scenario. We identify requirements, which an instrumentation technique should fulfill to be useful in a proprietary software environment and to scale for various purposes. We describe four techniques for instrumentation of OpenGL: relink library, dynamic library replacement, virtual device drivers and binary interception. We argue that binary interception is the best available technology.

Today, binary interception is primarily used in malicious software and for game cheating. The experiments and products for instrumented OpenGL that we know of use the other three techniques that we outlined. The shortcomings of these three techniques, as documented in Section 3, restrict existing products to niches. We argue that binary interception is useful for the instrumentation of OpenGL. The next step is to provide an experiment, where binary interception is used for a complete computer graphics application with useful behavior in the intermediary. The following paragraphs outline our preliminary plans in doing so.

Based upon the research presented in this paper, we will implement a prototypical software using the Generative Programming Paradigm [4]. The OpenGL API’s formal specification⁵ will be input to a generative build process. Therefore,

⁵The Khronos Group provides a formal specification of OpenGL in their registry at <http://www.opengl.org/registry/>.

it will be easy to generate new versions of the software when new OpenGL versions are released. We already encapsulated Microsoft Research’s C-based Detours library⁶ into typesafe C++ classes. This keeps implementation efforts for additional detours low. The low implementation effort is accompanied by the fact, that debugging the intermediary is hassle-free. An intermediary’s runtime performance is dominated by its inner functionality. Our preliminary experience shows, that the cardinal bottleneck is bandwidth consumption when the OpenGL command and data stream is transmitted between processes.

On our way to pragmatical interoperability, as exemplified in the application scenario, the main research question is interoperability between OpenGL streams from different applications. In preliminary experiments we instrumented applications from the NeHe OpenGL tutorials⁷, the KUKA.Sim⁸ robotics simulation suite, and Bitmanagement’s BS Contact⁹ generic VR-platform. With these applications, which use traditional, geometry-based rendering techniques, we adjusted camera parameters and excavated background graphics. We are curious, how it will work out with shader-based rendering techniques. We will discuss interoperability of OpenGL command streams in our future work. Then, we will be able to show, whether OpenGL opens up interactive three-dimensional graphics as pragmatic interoperability platform.

Acknowledgements

This work is supported by the German Ministry of Education and Science (BMBF), within the ViERforES-II project No. 01IM10002B. Mario Pukall is funded by German Research Foundation (DFG), project number SA 465/31-2.

5. REFERENCES

- [1] Ricardo A. Baratto, Leonard N. Kim, and Jason Nieh. THINC: A Virtual Display Architecture for Thin-Client Computing. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles, SOSP '05*, pages 277–290, New York, NY, USA, 2005. ACM.
- [2] Emanuela Boutin-Boila. TechViz XL, March 2010.

⁶Detours’ project page is available at <http://research.microsoft.com/projects/detours/>.

⁷The NeHe tutorials are available at <http://nehe.gamedev.net>.

⁸KUKA.Sim Viewer is available at http://www.kuka-robotics.com/en/products/software/kuka_sim/.

⁹BS Contact is available at <http://www.bitmanagement.com/>.

- [3] Yuqun Chen, Han Chen, Douglas W. Clark, Zhiyan Liu, Grant Wallace, and Kai Li. Software Environments For Cluster-Based Display Systems. In *Proceedings of the 1st International Symposium on Cluster Computing and the Grid*, CCGRID '01, pages 202–210, Washington, DC, USA, 2001. IEEE Computer Society.
- [4] Krzysztof Czarnecki and Ulrich W. Eisenecker. *Generative Programming: Methods, Tools, and Applications*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000.
- [5] K. Doerr and F. Kuester. CGLX: A Scalable, High-Performance Visualization Framework for Networked Display Environments. *IEEE Transactions on Visualization and Computer Graphics*, 17(3):320–332, March 2011.
- [6] Craig Dunwoody. *The OpenGL Stream Codec – A Specification*. Silicon Graphics, 1996. <http://www.opengl.org/documentation/specs/gls/glsspec.txt> Accessed 2011-08-16.
- [7] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns*. Addison Wesley, Reading, MA, 1995.
- [8] Michael M. Gorlick and Rami R. Razouk. Using Weaves for Software Construction and Analysis. In *Proceedings of the 13th International Conference on Software Engineering*, ICSE '91, pages 23–34, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.
- [9] Graphic Remedy. gDEDebugger. Version 5.8, December 2010.
- [10] Mark Grechanik, Don Batory, and Dewayne E. Perry. Integrating and Reusing GUI-Driven Applications. In *Proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools*, ICSR-7, pages 1–16, London, UK, UK, 2002. Springer-Verlag.
- [11] Object Management Group. OMG Unified Modeling Language™ (OMG UML), Superstructure Version 2.1.2, 2007.
- [12] Greg Humphreys, Mike Houston, Ren Ng, Randall Frank, Sean Ahern, Peter D. Kirchner, and James T. Klosowski. Chromium: A Stream-Processing Framework for Interactive Rendering on Clusters. In *ACM SIGGRAPH ASIA 2008 courses*, SIGGRAPH Asia '08, pages 43:1–43:10, New York, NY, USA, 2008. ACM.
- [13] Galen Hunt and Doug Brubacher. Detours: Binary Interception of Win32 Functions. In *Proceedings of the 3rd conference on USENIX Windows NT Symposium - Volume 3*, WINSYM '99, page 14, Berkeley, CA, USA, 1999. USENIX Association.
- [14] Byungil Jeong, Luc Renambot, Ratko Jagodic, Rajvikram Singh, Julieta Aguilera, Andrew Johnson, and Jason Leigh. High-Performance Dynamic Graphics Streaming for Scalable Adaptive Graphics Environment. In *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, SC '06, New York, NY, USA, 2006. ACM.
- [15] Gi Jung and Soon Jung. A Streaming Engine for PC-Based 3D Network Games onto Heterogeneous Mobile Platforms. In Zhiheng Pan, Ruth Aylett, Holger Diener, Xiaogang Jin, Stefan Göbel, and Li Li, editors, *Technologies for E-Learning and Digital Entertainment*, volume 3942 of *Lecture Notes in Computer Science*, pages 797–800. Springer Berlin / Heidelberg, 2006.
- [16] Veit Köppen and Gunter Saake. Einsatz von Virtueller Realität im Prozessmanagement. *Industrie Management*, 2:49–53, 2010.
- [17] Miguel-Ángel Manso, Monica Wachowicz, and Miguel-Ángel Bernabé. Towards an Integrated Model of Interoperability for Spatial Data Infrastructures. *Transactions in GIS*, 13(1):43–67, 2009.
- [18] Microsoft Corporation. *Microsoft Developer Network Online Documentation – LoadLibrary Function (Windows)*, July 2011. <http://msdn.microsoft.com/en-us/library/ms684175.aspx> Accessed 2011-08-10.
- [19] Microsoft Corporation. *Microsoft Developer Network Online Documentation – SwapBuffers Function (Windows)*, March 2011. <http://msdn.microsoft.com/en-us/library/dd369060.aspx> Accessed 2011-08-19.
- [20] Microsoft Corporation. *Microsoft Developer Network Online Documentation – wglGetProcAddress Function (Windows)*, March 2011. <http://msdn.microsoft.com/en-us/library/dd374386.aspx> Accessed 2011-08-17.
- [21] Alex Mohr and Michael Gleicher. HijackGL: Reconstructing from Streams for Stylized Rendering. In *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '02, pages 13–20 and 154, New York, NY, USA, 2002. ACM.
- [22] John Stavrakakis, Masahiro Takatsuka, Zhen-Jock Lau, and Nick Lowe. Exposing Application Graphics to a Dynamic Heterogeneous Network. In *Proceedings of the 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, WSCG '2006, pages 71–78, 2006.
- [23] Daniel Stødle, John Markus Bjørndalen, and Otto J. Anshus. The 22 Megapixel Laptop. In *Proceedings of the 2007 Workshop on Emerging Displays Technologies: Images and Beyond: the Future of Displays and Interacton*, EDT '07, New York, NY, USA, 2007. ACM.
- [24] Magnus Strengert, Thomas Klein, and Thomas Ertl. A Hardware-Aware Debugger for the OpenGL Shading Language. In *Proceedings of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware*, pages 81–88, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
- [25] VirtualBox Service Desk. #475 – 3D acceleration support for VBox guests. <http://www.virtualbox.org/ticket/475> Accessed 2011-10-10.
- [26] Stephan Vornholt, Michael Stoye, Ingolf Geist, Veit Köppen, and Gunter Saake. Datenmodell zur flexiblen Verwaltung von Datenaustauschprozessen in der virtuellen Produktentwicklung. In Roland Kasper u.a., editor, *10. Magdeburger Maschinenbau-Tage 27.-29.09.2011*, Magdeburg, September 2011. Otto-von-Guericke-Universität Magdeburg.
- [27] Levent Yilmaz. Using Meta-Level Ontology Relations to Measure Conceptual Alignment and Interoperability of Simulation Models. In *Proceedings of the 39th Conference on Winter Simulation*, WSC '07, pages 1090–1099, Piscataway, NJ, USA, 2007. IEEE Press.

Hypermodelling

Introducing Multi-dimensional Concern Reverse Engineering

Tim Frey
Otto-von-Guericke-University
Magdeburg, Germany
tim.frey@tim-frey.com

Veit Köppen
Otto-von-Guericke-University
Magdeburg, Germany
veit.koeppen@ovgu.de

Gunter Saake
Otto-von-Guericke-University
Magdeburg, Germany
gunter.saake@ovgu.de

ABSTRACT

In data warehouse systems you search for all items that have a desired combination of features, called dimensions. Similar to features of an item, concerns of software are scattered throughout a software system. So, a developer might ask the following questions: Which modules are belonging to a concern? Which concerns are appearing in a package? In this paper, we introduce multi-dimensional concern reverse engineering supported by Data Warehouse technology. This approach enables to search for code fragments like it can be done for artifacts in the warehouse. We show that a transformation process from source code to the Data Warehouse is possible. Consequently, a developer can fasten up searches and perform source code analysis more easily.

Categories and Subject Descriptors

D.2.3 [Software Engineering]: Coding Tools and Techniques;
D.2.6 [Software Engineering]: Programming Environments;
D.2.7 [Software Engineering]: Distribution, Maintenance, and Enhancement

Keywords

Separation of Concerns; Data Warehouse; Hypermodelling

1. INTRODUCTION

One of the most important principles in software engineering is separation of concerns (SOC) [7]. This principle addresses the goal that modules have a primary and only responsibility. The term concern can be defined as a logical matter of interest in a software system [27]. Often programmers use or develop libraries, encoding functionality of hardware devices. Such functionality is a typical instance of a concern. Developers are confronted with the task to encapsulate functionality of hardware devices into modules or to compose new functionality through the use of libraries. Hence, we focus on the generic definition of concern as any matter of interest and see hardware functionality as a sample instance of this term.

In currently applied programming paradigms, it is not possible to separate all concerns into their own modules. Therefore, a module normally encodes multiple functionalities at the same time [21]. Different approaches enhance the capabilities of programming languages and enable a better separation [3,17]. However, a program is a complex system and concerns represent a multi-dimensional space [4]. Therefore, it is hard to understand how concerns are intertwined and related with each other; and still, a lot of modules weave the functionality of various concerns together. Consequently, a developer faces the challenge that modules represent multiple aspects of functionality at the same time. It is also occurring that functionality is encoded in multiple modules at the same time. The association of a source code

fragment with multiple concerns creates an access and navigation problem in software analysis; concerns cannot be used for navigation. Neither can software be viewed or visualized from the concern perspective.

Concern tools allow developers to associate concerns with code elements [25]. Other tools enable query operations on code [6,14,28]. It seems, there is a gap between query and concern detection tools. Hence, no queries for fragments belonging to one or another concern at the same time can be done. Such queries can be used to uncover modules that are belonging to different concerns. Additionally, query tools slow down with queries, addressing relations of program elements to each other. This results from the used primary hierarchy; normally a program is viewed from packages down to classes and their members. Query tools operate on top of this structure. This slows queries down, because element references are not directly connected.¹

We see the necessity to enable analysis of source code for internal hierarchies and relationships within the code and not only for the primary hierarchy. A multi-dimensional viewpoint towards software is needed. Thus, we need a technology that supports multi-dimensional viewpoints of program. Consequently, this kind of technology would enable to investigate concern relations in new ways. Furthermore, the former named limitation of the query time can be speed up. A multi-dimensional approach allows it to directly query for different relations besides the primary hierarchy.

We introduce the Hypermodelling approach to overcome addressed limitations. Our approach enables multi-dimensional concern reverse engineering. Hypermodelling is basically the idea to adopt mechanisms from the area of Data Warehousing (DW) in combination with SOC to create a multi-dimensional viewpoint towards software. We see Online Analytical Processing (OLAP) [16] as an efficient method to analyze multi-dimensional data.

More and more hardware devices are programmed in high level languages. In this paper, we focus on generic ways how concerns are encoded in the Java language. Previous attempts have shown that it is not a trivial task to build a holistic multi-dimensional schema for Java [11]. Thus, we see a first step to determine the application ability to use DW technology in the creation of a DW cube. We test the application of our cube with a sample application that we load into the cube. Since loading code into this cube is a high effort, we identify several areas that profit from it. This is done with respect to costs and benefits.

Our contribution is to describe the benefit of DW technology for code analysis. Furthermore, the Hypermodelling approach to use

¹For instance, in Eclipse references between classes are resolved by scanning the classpath and they are not connected statically.

a DW for code analysis is explained. We present a first cube and an exemplary report on it. This shows, DW technology makes it possible to query code from various viewpoints.

The paper is organized as follows: First, we describe different areas that could profit from the Hypermodelling approach. Afterwards, we briefly explain related research fields. Then, similarities between the different fields are shown and the Hypermodelling approach is presented. Following, we present the actual application to load source code into a DW. Reports demonstrate how measures are computed. Related research is compared and shown. Finally, we conclude and briefly describe future work paths.

2. MOTIVATION

Loading code into a DW comes with cost that should be paid only if convincing benefits can be obtained. Hence, we argue to apply DW technology to analyze source code in the following points.

2.1 Aggregation

Programmers spend most of their time exploring source code [24]. Thus, multiple attempts exist to improve the usage of Integrated Development Environment (IDE). This is done through support for concern analysis or enhancements for query operations [6,18,23,25,28]. A problem of the query tools is that when large code bases are queried, the performance drops down.² We develop Hypermodelling for Eclipse to allow developers to state multi-dimensional queries in the IDE for investigation of projects. The tool computes code slices based on element relations in the code. For instance, members of a slice are all classes that extend a specific class. The current implementation consists mainly of a query engine, responsible for parsing code and computing results. Thus, also the performance drops comparable to other tools. The queries have a multi-dimensional character through the inspiration from OLAP tools. One main application in DW is the usage of aggregates in queries. These allow us a fast computation and aggregated measures enable logical relations to be visualized through measures and indicators. For instance, the total number of employees in a certain district can be computed this way. Both dimensions, employee and district, can be aggregated against each other. Hence, it would be desirable to speed up the queries through the usage of DW technology to add the possibility of aggregate inclusions in queries. Such a query could be: all classes of a project that are annotated by a specific annotation and that are extending a certain class. Additionally, aggregates for various dimensional combinations can be (pre-) computed. The advantage of Hypermodelling in comparison to other tools would not only be the superior multi-dimensional query possibility, but also a performance advantage.

2.2 Various viewpoints

Additionally, various tools within an IDE are applied to investigate code from various perspectives. Future IDEs will use data from code and associated processes, leverage this data, and support synergy in data and functionality. The data of various tools will be used within other tools that identify a new meaning and therefore create a benefit. Furthermore, a great challenge is to use the immense amount of data and turn it into useful knowledge [29]. Process data are different kinds of facts, associated with code. They tie code elements or their combination with other data

together. Such facts can be; the association of code elements belonging to a certain task, or test outcomes and associated tested code fragments. Hence, an integrating technology, scaling for large amounts of data, is needed. The desired representation of various facts and their association with code must be possible in an extensible way and allow (re)composition to enable new usage scenarios.

2.3 Hierarchies

In commonly applied languages the object dimension is considered as the dominant mean to separate concerns. Further viewpoints have been proposed to advance software development [4,27]. Even though, the exploration of code is normally done by a primary hierarchy. This hierarchy is from project down to the package structure containing classes and their members. Other hierarchies like call or method hierarchies are supported, but the very main hierarchy is still used most [20]. New mechanisms to separate concerns are supported by IDE extensions, offering new viewpoints on the code [5]. Thus, a technology supporting various viewpoints towards code capable of representing multiple hierarchies is needed for advanced analysis.

2.4 Dynamic artifact extraction

Related to code visualizations is the representation of software through models. Models can be seen as various projections of a program and the related information that is used within the development process. Thus, software is a multi-dimensional space, visualized via various model planes that are used for projections. Therefore, we formulate the goal: create a holistic model that can be used to create models dynamically [1]. The dynamic creation of models out of a holistic model probably happens with transformations out of the central model. If a model represents a case like a class diagram it is just a subset of the code that is visualized. In the case of sequence diagram it is just data that is associated with various source code elements. Thus, a technology is needed to support the extraction of various facts out of the source model easily. Hence, again the need for a technology, allowing regarding software from various viewpoints excels.

2.5 Integration

Mining software archives deals with automated extraction, collection and abstraction of data generated in the development process [12]. Thereby, data are extracted out of various systems and aligned in a format for analysis [9,26]. Thus, to alter or switch to another perspective for the analysis means to alter the custom built extraction mechanism. A tool extracting facts at an aggregation level of interest would help to concentrate on the primary task to analyze the data and not to extract the data. Data of the development process are normally associated with the source code of an investigated program. Thus, a model of source code is needed that can be used to integrate data from various sources at one central point. This can be used to extract easily desired data from a central point.

2.6 Indicator Associations

When software operates a machine, multiple physical variables can be measured via sensors. For instance, heat, pressure, and energy consumption can be measured. Engineers face the challenge that indicators, as also the code structure, are needed to analyze the way of machine operates. Thus, a technology is needed, allowing the investigation of indicators as well as the code structure at the same time. A main application of a DW is

²For instance JQuery [28].

computations of indicators. Therefore, a first step towards analyzing code structure and indicators, is the realization of multi-dimensional model for software within a DW. This can then be used to associate indicators with source code elements.

3. HYPERMODELLING

The Hypermodelling idea, to combine OLAP and SOC, covers research areas that are normally not viewed together. We briefly introduce the concepts required for their combination and we explain Hypermodelling via an example.

3.1 Separation of Concerns

Programming language designers have developed numerous mechanisms for SOC at the source code level. In this paper we focus mainly on Metadata annotations [3,10], classes, methods and Aspects [8,17].

Annotations are elements that can be used as meta information for classes and their members like fields or methods. Normally, this information is used to enhance annotated elements with certain capabilities [3]. For example such annotations can influence how an object graph of an application is wired.

Aspects are program fragments that are used to enable programmers to code crosscutting concerns in their own modules [8,17]. Crosscutting concerns represent functionality of a program that is not clearly belonging to one module and would be scattered within various modules. Aspects enable grouping such kind of functionality together and define places where the functionality is applied.

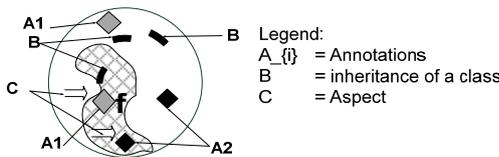


Figure 1. A Software System

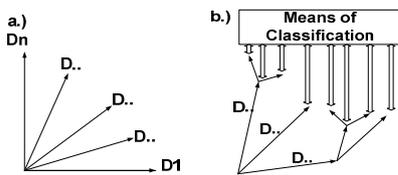


Figure 2. Fragment space

Figure 1 shows an abstract visualization of a programs source code. The circle symbolizes the code. The symbols represent occurring elements in the code. Figure 1 also shows a fragment of source code *f*. This fragment is affected by multiple concerns at the same time. It has different annotations, is affected by an aspect, and extends a certain class. All this concerns, affecting the element can be determined by parsing the code.

The generalized view of the software fragment can be seen in Figure 2a. A fragment belongs to different concerns at the same time; named as D1 to Dn. In Figure 2b it is shown that these concerns can have hierarchies, like inheritance or package structures. The large wide arrows and the arbor express that these concerns and the sorting of the corresponding source code fragments into hierarchies can be done automatically.

Examples for concern associations in Figure 2 are parent classes

or interfaces. A fragment is associated with various other classes that represent D1 to Dn. The parent classes are located within their own hierarchies: They have parent classes themselves or they are located within a package structure what is expressed as hierarchy. Likewise, annotations can be used for a fragment. These are also located within packages. A hierarchy can be used to visualize this relation of a code fragment with annotations and the corresponding hierarchy. Generally, every mean to apply SOC can be used to determine the association of a fragment with concerns. We kept the graphic generic to be not limited to a specific kind of concern associations. All concern associations are known and can be resolved automatically out of the language structure. This way, the various associations of a code fragment with concerns can be done.

3.2 Data Warehousing

DW and OLAP are used to analyze multi-dimensional data. The main data structure for DW is the data cube [16], which we depict in Figure 3 as a relational schema. This structure consists as one possibility out of a relational schema on top of which an OLAP-Cube is designed [16].³

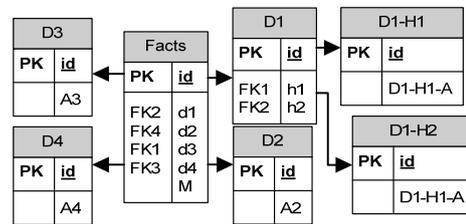


Figure 3. Snowflake schema for data Cube

We present a Snowflake schema of a DW in Figure 3. The fact table references multiple dimensional tables (D1-D4). A row in the fact table is associated with measures (M). Such measures are often called indicators. They are associated with combinations of dimensions. This means a row in the fact table is a connector of different points in dimensions and measures. Dimensions can have attributes (A1-A4) and also references to other tables in hierarchical levels (h1: D1-H1, h2: D1-H2).

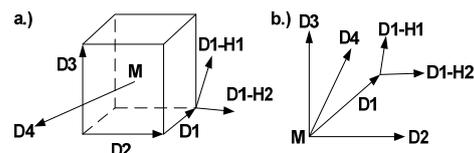


Figure 4. OLAP Cube

A data cube, on an abstract level, shown in Figure 4 visualizes a multi-dimensional structure. There, the association of the measure (M) with points in the different dimensions is visualized. A hierarchy of D1 is shown that can be used to structure members of a dimension in hierarchical levels. Finally, queries are executed in the data cube. Selections, filtering of dimensions as well as navigation between different dimensional hierarchies (called Roll-up, Drill-down, or Drill-across) are selected to determine corresponding measures. On main advantage of OLAP is that aggregations of measures are available and the navigation path is

³Since OLAP doesn't imply a relational schema, it is presented here anyways to ease comprehension, because the scope of this paper is to show similarities to software (re-)engineering.

easily and efficiently accessible. This enables us at the same time to generate reports on actual and aggregated data. Note; data mining techniques could also be applied within DW.

3.3 Hypermodelling

In Hypermodelling a source code fragment, for example a class or a method, can be associated with multiple concerns at the same time. All fragments that are associated with a specific concern are members of a slice belonging to this concern. Thus, concerns are quite similar to dimensions in the context of OLAP cubes. This leads to the technique to use OLAP similar methods to query for fragments, belonging to one or more concern. We use the dimensions of an OLAP cube as concerns. For us, the association of a class with concerns, like its parent classes, is the same like the associations of measures in OLAP with dimensions.

```

1: @Entity
2: @Deprecated
3: class Customer{
4:     @Deprecated
5:     Customer(){
6:         ...}
7: ...}
8: class CustomerDAO extends DaoSupport
9:
10:     @SuppressWarnings("deprecation")
11:     Customer createCustomer(){
12:         return new Customer();
13:     } ...}
    
```

Listing 1. Example for annotated source code

Table 1. Concerns of Listing 1 in a table

Element / Dimensions	extends DaoSupport	@Entity	@Deprecated	@SuppressWarnings
Customer		X	X	
Customer.Customer()			X	
CustomerDAO	X			
CustomerDAO.createCustomer()				X

We present an example in Listing 1 and Table 1. Listing 1 shows Java source code. A Customer class and a Data-Access-Object (DAO) are shown. The CustomerDAO class extends a helper class (DaoSupport) for table access. This is commonly done when frameworks are used. Table 1 shows rows that are representing source code fragments and columns representing concerns. The “X” indicates that a fragment belongs to a concern. For example the constructor of the class Customer is marked @Deprecated. Likewise the rest of the table-listing associations can be done.

In a nutshell, the main idea is to use the code fragments itself as fact table and allow queries on this “table”. Furthermore, also hierarchies exist in source code. A class, for instance, is member within a hierarchical package structure. This way, source code shows huge similarities to DW data structures.

A sample query for Table 1 could be; all fragments that belong to

@Entity and to @Deprecated. This leads to the result of the Customer class. Hence, queries can be used to determine code fragments, belonging to different concerns at the same time.

4. Data Warehouse-application evaluation

We load source code into a DW to evaluate the possibility of the application of actual OLAP technology. We use in our example the Microsoft Analysis Services Server Version 2008 R2. We lean the application partly on the previous shown sample and load annotations and inheritance. The loading is done via ETL (Extract-Transform and Load). For this we introduce an intermediate layer, the ETL-Relational source code schema. Afterwards, we explain the cube structure.⁴ Furthermore, we visualize exemplary queries and show a DW report.

4.1 The relational schema for ETL

As data source for the cube, we develop a relational schema for annotations and inheritance. The schema is inspired by the Eclipse internal Java model⁵. A reason to lean the model towards the Eclipse model is to be open for a portability of the Hypermodelling technique into the Eclipse IDE.

Nevertheless, the model of Eclipse is actually not build as a model for a relational database. Therefore, modifications have to be applied to create a similar model to a relational database. Thus, our relational model can just be seen as inspired by Eclipse and not as a one to one mapping. We perceive that some relations in the Eclipse model are based on the Java language specification logic. These differ compared to logical viewpoints of a programmer. For example, instances of annotations are not linked with the definition of the annotation type itself. This means that the occurrence of an annotation is actually not an instance of its type definition. This kind of “logical” gaps also makes a challenge for the transformation to a relational representation. However, we prefer that the relational model should represent reality in the programmers meaning and not the Java language specification. The schema, presented in Figure 5, shows the relational representation, whereby all fact tables are emphasized in grey. The fact tables are the source for the associations of the various dimensions. Through multiple fact tables it is possible to realize complex relations. Like a type (e.g., class) that has multiple members and also multiple types that are used for inheritance.

In Java primitive (e.g., integer and boolean) and complex (Classes, Enums, Annotations, Interfaces) types occur in source code. These are realized through AbstractType, ComplexType and TypeClassification table. The TypeClassification indicates the kind of type. The AbstractType is defined to have the possibility of a common base for complex and primitive types. As it can be seen, the ComplexType table, representing complex types, references the AbstractType and this way, indirectly, the TypeClassification.⁶ However, a complex type can also have additional properties in contrast to a primitive. Generally, a complex type is defined in a file that is, again, belonging to a package which is furthermore a member in a taxonomic package

⁴The shown graph is leaned on the visualization in [13].

⁵<http://www.eclipse.org/jdt>

⁶AbstractType enables to extend the model. Further associations between model elements can be done; e.g., method parameters that can be primitive or complex types.

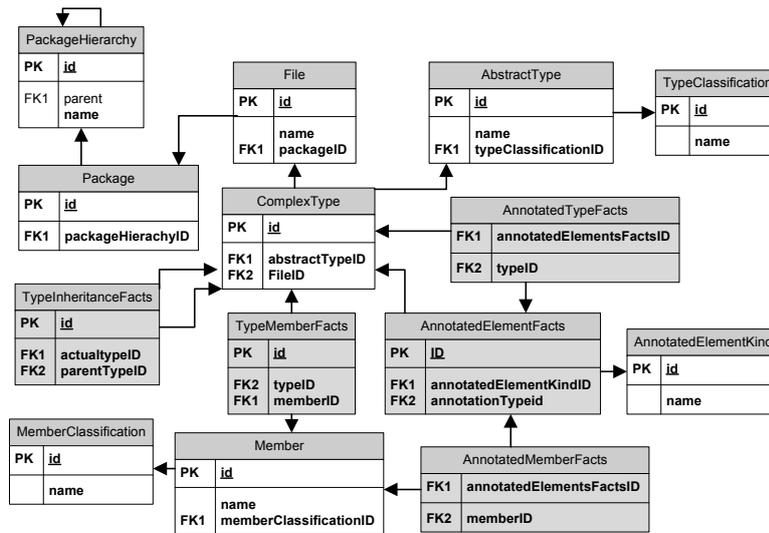


Figure 5. Relational schema

hierarchy. Often different package roots exist in a project that contains the same packages. Thus, a file is not a direct member in a package hierarchy and the bonding to the Package table is done it between.

As a matter of fact, a complex type can have multiple members. Such members are fields and methods. Like shown, this relation is realized though a fact table.⁷ A MemberClassification table is associated with the Member table to indicate the kind of the member. Hence, it is possible to extend the model for the different member kinds and mind their different properties. Implementation of interfaces and inheritance of classes are realized through the TypeInheritanceFacts table. The approach is taken to avoid a self reference of the ComplexType table, because a type can implement multiple interfaces.

AnnotatedElementFacts, AnnotatedTypeFacts and AnnotatedMemberFacts show, finally, the Annotated complex types and members. AnnotatedElementFacts is used to associate an annotation, represented through a complex type itself, with a row in the fact table. The kind of the annotated element is indicated through the AnnotatedElementKind. AnnotatedTypeFacts and AnnotatedMemberFacts associate a fact with a complex type or a complex type member.

4.2 The cube structure

In Figure 6, we present a visualization of a cube on top of the relational schema. The dimensions are the nodes and the hierarchies are shown by their connection.

The first dimension type represents elements, having an association to a complex type (named Type in the figure). They are located at the bottom, beneath the facts (Program nodes). They are affected elements by inheritance or annotations. Thus, they are “passive” elements in a relation. Since Members and Types can be annotated, both of them are directly connected with a fact table. In this way, we enable the possibility to sum up Member Annotations at Type dimension or to compute with the ones appearing at a Type.

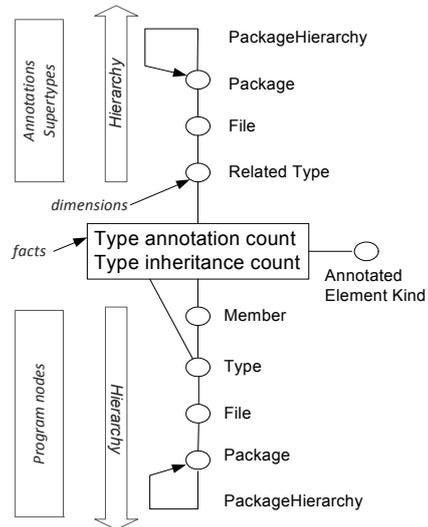


Figure 6. The Hypermodelling Cube

The second type of dimensions is the AnnotatedElementKind . We state this explicitly, because it is only used together with annotations and not with inheritance facts. The Member dimension is arranged parallel to the Type dimension. The reason is the hierarchic relation between members and types. Through that relation facts, associated with a member, can be aggregated at type level. Thus, the AnnotatedElementKind dimension is introduced to enable queries at type and also at member level. We visualize this vertical to the others to emphasize that it can be used to discriminate other dimensions.

The third type of dimension is the “related” dimension that is located at the top (Annotations, Supertypes). This represents “active” parts of a relation. Active means the annotation or the supertype itself.

Finally, two measures, consisting of occurrence counts, connect different dimension types. We show that the dimensions on top and bottom are ordered in a hierarchy. Therefore, queries can use hierarchical structures that occur in the code.

⁷The approach of a fact table in between was chosen to credit reality; often methods are moved between complex types.

Table 2. The source data of the report in Figure 9

package	All	*.stereotype (1)	*.transaction (2)	*.beans (3)	*.jmx (4)	*.web (5)	*.aspectj (6)	*.lang(excluded)	*.persistence (7)	*.xml (8)
org.springframework.petclinic (a)	5							2		3
org.springframework.petclinic.hibernate (b)	11	1	6	1				3		
org.springframework.petclinic.jdbc (c)	13	1	9	1	2					
org.springframework.petclinic.jpa (d)	11	1	6					3	1	
org.springframework.petclinic.web (e)	53	7		8		35		3		
org.springframework.petclinic.aspects (f)	13				6		7			

				Repository	Transactional	SuppressWarnings	PersistenceContext
EntityManagerClinic	Member	Field	em				1
		Method	getVets		1	1	
	Type			1	1		

Figure 7. Association measures at types and members

	org.springframework	java.lang
	org.springframework.samples.petclinic	
	Clinic	Object
EntityManagerClinic		1 1

Figure 8. Inheritance measures sample

4.3 The filled cube

The discussed relational schema and cube are filled with real Java data. As data, to support verification, we choose a sample application that is available publicly. The elected petclinic application is a demonstration of the capabilities of the spring framework that is applied widely in the industry. The main reason to choose this demo application is: The spring framework is widely known as a reference for good application design. Mainly, the application is a layered web application consisting of 31 Java files, containing application logic. It is making use of declarative transaction management, database access, and aspect oriented programming paradigms.⁹

In order to load the data, a parser for the Eclipse IDE is developed that inserts directly Java source code into the relational schema. Out of the filled schema the cube is processed.

4.3.1 Sample source code in the cube

In the following, we present sample queries containing results about a source code excerpt.

```

1: @Repository
2: @Transactional
3: public class EntityManagerClinic implements Clinic {
4:     @PersistenceContext
5:     private EntityManager em;
6:     @Transactional(readOnly = true)
7:     @SuppressWarnings("unchecked")
8:     public Collection<Vet> getVets() {

```

Listing 2. Excerpt source code of the demo project

⁹The petclinic application is described at <http://static.springsource.org/docs/petclinic.html> and can be downloaded at <http://hyperm modelling.com>

In Listing 2, we show an excerpt of a class of the org.springframework.sample.petclinic.jpa package. Corresponding query results are presented in Figure 7 and 8. In Figure 8 we present the inheritance measure. As it can be seen, the insertion also added the Object class as ancestor, since every class inherits in Java from this class. Additionally, the package hierarchy is shown to indicate that drilldowns over various hierarchy levels are possible. Generally, this enables queries in the style; all classes in a package, extending a class of another package (java.lang) and implementing an interface of another (org.springframework).

We show annotated elements in Figure 7. Likewise, the hierarchical structure of dimensions is shown on the left. The EntityManagerClinic has annotations at type and also at member level. Members can be divided into fields and methods. The count of one indicates that an annotation exists. Now, a query can determine the fragments that are fields and are marked as @PersistenceContext at the same time; the result is em. Clearly, dimensions like class name could be even used to discriminate further. Even a combination with the inheritance facts is possible. Divers concerns can be used to navigate through code.

In the following a report that uses the annotation measures is presented. We restrict ourselves to this example; however, further reports are possible.

4.3.2 A sample report

Table 2 shows the source data of our exemplary report in Figure 9. We present it to enable the verification of our report. We verify our loading technique of code by counting the occurrences of annotations in the sample loaded code that is publicly available. In Figure 9 a sample report visualizes the distribution of annotations that occur in the packages of the analyzed project. We use data of Table 2 as source. Annotations of the Java language specification like override or deprecated, have been excluded from the visualization of the report. This enables us to focus on annotations with a clear functional meaning. Annotations are grouped together to the packages where they are defined. For example, the previous shown Annotation Transactional is in the

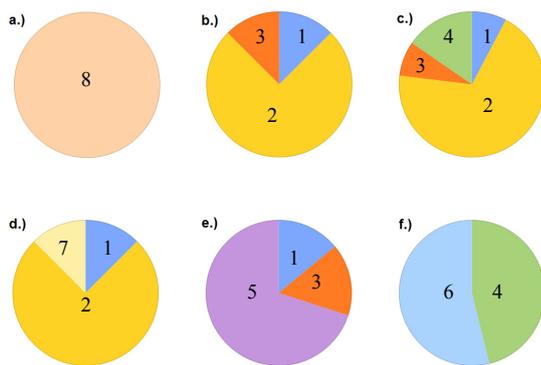


Figure 9. Sample report

*.transaction package.¹⁰ The Repository Annotation belongs to the *.stereotype package. However, the abstraction to the package level is done to prove that the hierarchies also work for annotations and to present a report on a generalized level. Further, the annotated elements are generalized to the package level.

4.3.3 Interpretation of the report

The report shows, two different dimensions and their hierarchies can be related to each other. So it is possible to see code not in one primary perspective anymore and to create reports from various perspectives. Also, the indicators can be computed for different perspectives, like for classes or packages. It is now possible to see code also from the viewpoint of annotations and their packages.

We also use the very main application of DW technology, aggregations. Aggregations show the total occurrences with the two dimensional hierarchies and enable drill downs over various dimensions.

The visualization shows that the occurrence of dimensional relations can be depicted, easily. This makes it possible to visualize various relations just with a few clicks. This is a huge advantage in comparison to the current state of the art in software visualization. There, the data to create such reports would have been collected through custom parsers and then a custom visualization would have been applied. Now, it is possible to create reports on the fly.

An advanced application for such reports is also to browse the code from various perspectives and interpret the results. The report shown in Figure 9 can be used to interpret the responsibility of various packages in the project. It excels that package *a* has just xml annotations and no other ones. The reason is that the package represents the domain model of the application, as it can be verified by inspection. Especially sticking out is package *e*, where a heavy use of annotations out of the web package is done. Thus, the conclusion can be done that the package is responsible for the web access of the application. Clearly, also the meaning of the aspect package *f*. can be derived from the consumed annotations. Package *b*, *c* and *d* make heavy use of transactions. This indicates their functionality has to be referred to the persistency logic of the application. In fact, these packages seem quite similar. Source inspection verifies this

¹⁰The wildcard '*' is used to indicate parent packages. Their name was stripped for readability issues and the package names are unique by the last name in this sample.

Legend

- g.) org.springframework.petclinic
 - h.) org.springframework.petclinic.hibernate
 - i.) org.springframework.petclinic.jdbc
 - j.) org.springframework.petclinic.jpa
 - k.) org.springframework.petclinic.web
 - l.) org.springframework.petclinic.aspects
-
- 1 ■ org.springframework.stereotype
 - 2 ■ org.springframework.transaction
 - 3 ■ org.springframework.beans
 - 4 ■ org.springframework.jmx
 - 5 ■ org.springframework.web
 - 6 ■ org.aspectj
 - 7 ■ javax.persistence
 - 8 ■ javax.xml

theory; these packages realize the access to a persistent data store with different data access technologies and are interchangeable within the application.

With such reports a developer can now do a query for classes and packages that contain annotations out of the persistence package *.persistence and also contain ones out of the transactional package. The result set is all classes in package *d*.. This is quite useful to get the required code fragments that are members of transactions and implemented with a specific persistence technology. Without Hypermodelling it would have been a longer and more complex search; through Hypermodelling concern oriented navigation is now possible.

All together, it seems straight forward to see what a package is used for within the application, just out of the annotations used in a package. Thus, there is the assumption that this is maybe the case in other applications as well. This needs to be investigated deeper and is out of the scope of this paper. However, this very first application, to load data into the cube shows the possibility to regard source code from one another viewpoint already created an interesting observation; annotations might be used to indicate the meaning of a package. Hence, browsing and measuring code from various perspectives seems very interesting for further investigation scenarios.

5. Related Work

The idea of Hypermodelling is, aside from the motivating, related technology, related to other approaches that try to integrate code and enable it for systematic inspection. Following we present related approaches and compare it to the idea to store code in a multi-dimensional model.

Storing source code in databases [14] describes code queries with a logic programming language. The source code is stored in a relational database and queries in the logic language are translated into SQL queries. This differs to the approach proposed in this paper. Hypermodelling uses DW technology to do the query and the relational model is just used to be the source for the cubes. Hence, Hypermodelling can improve the current approaches through the usage of aggregations at various levels.

It seems that in the area of repository mining [12] the idea of having a central access for the source code is gaining attention. The usage of Business Intelligence like means to support the

mining process is already mentioned. Since Business Intelligence is often used as a term for all the associated technologies, like OLAP or DW this can be seen related. Software Intelligence (SI) [15] describes this idea on an abstract level. In spite of the relation to SI, Hypermodelling is still unique. SI neither proposes multi-dimensional models nor takes the fact into account that concerns play a central role in software development. Also, the idea of using actual DW technology is not respected.

A source code search infrastructure name Sourcerer [2] based on a relational model with four tables enables searches of relations in code. The relations are realized through a relational table, connecting different rows of an entity table with each other. An entity represents a class method or a package. Hypermodelling uses in contrast various tables to represent the diverse relations that occur in source code. This has the advantage that the schema is quite to understand. Also, not so many self joins of a table are needed to calculate the relations. But, since the main application of Hypermodelling is to use multi-dimensional cubes and support aggregations it is also superior this way. Through the multi-dimensional model no self joins are needed and the relations can be determined faster.

In [22] a method to control a software development project with metrics is described. There, a Metrics Warehouse is mentioned. Since the measures calculated in the paper represent somehow metrics of source code this seems first to be related. But the metrics described are economic project figures, not representing code metrics. Therefore, the approach is supporting the business user and not the developer, what is one goal of Hypermodelling.

6. Conclusions and future Work

We presented various areas where the appliance of DW technology for source code analysis would be an advantage. We solved the problem in reverse engineering that concerns can't be used for navigation, visualization or as perspective through the Hypermodelling approach. Furthermore, we actually demonstrated the possibility to use DW technology, showing a first multi-dimensional model and a report. Now, source code can be navigated and also reported easily from various positions. Also, aggregations can be done. Nearby, it excelled that the meaning of the responsibility of packages can be indicated by annotations. The comparison to related work showed: the approach to use a multi-dimensional model is probably superior in determining relations in code and faster than normal SQL queries. It would be interesting to use the approach as code search engine.

Like described in the Section 2, various areas can benefit from the approach. We see the first step in investigating the portability of the DW technology directly into the IDE. This way, various query tools, as described in Section 2, can be accelerated. The model needs to be extended for other relations in code and associated with other facts, like process data and indicators. Also, the extraction of data and the application of mining with the DW are possible. We see the next step to extract information, already loaded in a schema, and use it for mining.

Nevertheless, only a small part of structures occurring in source code is currently used to demonstrate the actual application of the technology. For instance, we use annotations that are widespread in new Java programs, but older programs do not use them. We need more cubes to enable advanced possibilities in code investigation. These cubes should respect other facts as inheritance and annotations in their multi-dimensional model. This will represent a larger part of the Java language and enable

advanced queries. It will also be capable to inspect further applications. Furthermore, it can be investigated if queries can be used to create software models dynamically. Furthermore, a holistic model in the DW can be used by various tools in the IDE to sample their data for different viewpoints.

Currently, no historical versions of software are used within our model. Since historical data is a key application in classic DW, this can be an interesting field in the future how to load and organize historical software versions.

Moreover, we are confident that a DW can play an integrative role in software design. We believe it can be used to plan different concerns, like it is done in strategic enterprise management [19]. This comes out of the reason that the multi-dimensionality and responsibility of modules can be respected through our approach. The incursion is; concerns can be planned on an abstract level with measures. For instance, new functionality for a package is planned with concern combinations that are associated with a measure. Then, the concrete implementation is done in the classes of the package. Finally, a report compares the planned measure with the actual implementation level.

Another idea is to associate other indicators than counting with source code. An advanced trail can be energy analysis. Energy gain or loss can be associated as indicators with source code. Additionally, call hierarchies and execution probabilities can be also associated with elements in source code. Through the central analysis point, correlations of the various elements can maybe be determined. This can be used to expose energy loss spots in software, what would be especially interesting when software operates machines. It would be an advantage to know the exact regions of code that are most often executed and consume the most energy at same time. Exposing such spots can lead to build energy-optimized code.

Lastly, modern software systems contain millions lines of code. In the case we want to handle this amount of data a technology is needed that is designed to handle large scale data sets. However, all the code search engines share currently the same limitation: little structure information is used to explore the source code. Additionally, a dominant viewpoint towards the programs is always the perspective of project-packages-classes. The internal structures in the code itself are not respected. One advantage to use DW technology is that the multi-dimensionality of source code can be respected this way. We believe that the approach can be used to build better source code engines in the future that can be queried in a multi-dimensional way.

7. REFERENCES

- [1] C. Atkinson, D. Stoll. Orthographic Modelling Environment. FASE'08/ETAPS'08 Proceedings of the Theory and practice of software, 11th international conference on Fundamental approaches to software engineering. pages 93-96. Springer. 2008
- [2] K. S. Bajracharya, J. Oshser, C.V. Lopes. Sourcerer - An Infrastructure for Large-scale Collection and Analysis of Open-source Code. Third International Workshop on Academic Software Development Tools and Techniques (WASDeTT-3), Belgium, 2010
- [3] J. A. Bloch. Metadata Facility for the Java Programming Language. 2004.
- [4] W. Chung, W. Harrison, V. Kruskal et al.. Working with Implicit Concerns in the Concern Manipulation Environment, AOSD '05 Workshop on Linking Aspect Technology and Evolution (LATE), pages 1-5. ACM. 2005.
- [5] A. Colyer, A. Clement, M. Kersten. Aspect-oriented programming with ajdt. In Proceedings of AAOS 2003:

- Analysis of Aspect-Oriented Software, held in conjunction with ECOOP 2003. Springer. 2003.
- [6] B. de Alwis, G. C. Murphy, M. P. Robillard. A Comparative Study of Three Program Exploration Tools. pages 103-112 ACM. 2007
- [7] E. W. Dijkstra. Selected Writings on Computing: A Personal Perspective. On the role of scientific thought. Springer. 1982
- [8] R. E. Filman, T. Elrad, S. Clarke, M. Aksit Aspect-Oriented Software Development. Addison-Wesley Professional; 1st edition. 2004
- [9] M. Fischer, M. Pinzger, H. Gall. Populating a Release History Database from Version Control and Bug Tracking Systems. ICSM'03. pages 23-32. Merlin. 2003
- [10] M. Fowler, Domain-Specific Languages, Addison-Wesley, Netherlands. 2010
- [11] T. Frey. Vorschlag Hypermodellierung: Data Warehousing für Quelltext. 23rd GI Workshop on Foundations of Databases. CEUR-WS. pages 55-60. Austria. 2011
- [12] M. W. Godfrey, A. E. Hassan, J. D. Herbsleb et al.. Future of mining software archives: A roundtable. IEEE. 2009
- [13] M. Golfarelli, D. Maio, S. Rizzi. The Dimensional Fact Model: A Conceptual Model For Data Warehouses. In International Journal of Cooperative Information Systems, 7, pages 215-247. 1998.
- [14] E. Hajiyeve, M. Verbaere, O. de Moor. Codequest: scalable source code queries with datalog. ECOOP'06: proceedings. pages 2-27 Springer. 2006
- [15] A. E. Hassan, T. Xie. Software Intelligence: Future of Mining Software Engineering Data. In Proceedings of FSE/SDP Workshop on the Future of Software Engineering Research. Santa Fe. 2010
- [16] W. H. Inmon. Building the Data Warehouse. 4th ed., J. Wiley & Sons, New York. USA. 2005.
- [17] G. Kiczales, E. Hilsdale, et al.. An Overview of AspectJ. European Conference on Object-Oriented Programming (ECOOP). pages 327-353. Springer. 2001
- [18] I. Majid and M. P. Robillard. NaCIN - An Eclipse Plug-In for Program Navigation-based Concern Inference. In Proceedings of the Eclipse Technology Exchange at OOPSLA.ACM. 2005
- [19] M. C. Meier, W. Sinzig, P. Mertens. Enterprise Management with SAP SEM/Business Analytics. 2nd Edition, Springer. Berlin. 2005
- [20] G. C. Murphy, M. Kersten, L. Findlater. How Are Java Software Developers Using the Eclipse IDE?. IEEE. 2006
- [21] H. Ossher, P. Tarr. Multi-dimensional separation of concerns and the hyperspace approach. In Proceedings of the Symposium on Software Architectures and Component Technology. Kluwer. 2001.
- [22] C. R. Pandian. Software metrics: a guide to planning, analysis and application. Auerbach Publications. 2003
- [23] J. H. Pfeiffer, A. Sardos, J. R. Gurd. Complex code querying and navigation for aspectj. Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange. ACM. 2005
- [24] M. P. Robillard, W. Coelho, G. C. Murphy. How Effective Developers Investigate Source Code: An Exploratory Study. pages 889-903, IEEE. USA. 2004
- [25] M. P. Robillard, F. Weigand-Warr. ConcernMapper: simple view-based separation of scattered concerns. In Proceedings of the 2005 OOPSLA Workshop on Eclipse technology eXchange. ACM. 2005
- [26] J. Sliwerski, T. Zimmermann, A. Zeller. Don't Program on Fridays! How to Locate Fix-Inducing Changes. In Proceedings of the 7th Workshop Software Reengineering, 2005
- [27] S. M. Sutton Jr., P. Tarr. Aspect-oriented design needs concern modeling. AOSD workshop on Aspect-Oriented Design. 2002.
- [28] K. D. Volder. Jquery: A generic code browser with a declarative configuration language. In P. V. Hentenryck, editor, PADL, volume 3819 of Lecture Notes in Computer Science, pages 88-102. Springer, 2006
- [29] A. Zeller. The future of programming environments: Integration, synergy, and assistance. Future of Software Engineering, IEEE, USA, 2007

Management of Flexible Data Exchange Processes in Virtual Product Development

Michael Stoye
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
michaelstoye85@yahoo.de

Stephan Vornholt
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
vornholt@iti.cs.uni-
magdeburg.de

Ingolf Geist
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
ingolf.geist@iti.cs.uni-
magdeburg.de

ABSTRACT

To meet the strong requirements in product development many computer-aided systems are used today. Among these systems digital models have to be exchanged to avoid error-prone and time-consuming recreation of already defined product data. During product development computer-aided systems and file formats for data exchange among them can change because of updated versions of systems or new systems. In those changeable environments, for engineers it is hard to know which of the many possibilities to choose for data exchange between computer-aided systems. Another problem is the traceability of executed data exchange processes.

We propose a data model to support the management of flexible data exchange processes. Therefore, the data model describes and connects system- and transfer-specific data. To create an appropriate data model, detailed knowledge about possibilities for data exchange, computer-aided systems, exchange file formats and underlying processes are necessary. Consequently, we analysed frequently used data exchange file formats and an exemplary development process. In this paper the results of these analyses are presented. Furthermore, the developed data model and a prototypical implementation are described.

Categories and Subject Descriptors

H.2.1 [Database Management]: Logical Design — Schema and subschema; J.2 [Computer Applications]: Physical Sciences and Engineering — Engineering; J.6 [Computer-aided engineering]: Computer-aided design (CAD), computer-aided manufacturing (CAM)

Keywords

Virtual Product Development, Virtual Engineering, Data Exchange, Computer-Aided Systems, Workflow Management, Data Model, Product Data Management

1. INTRODUCTION

Companies, which are developing new products, are under a high pressure of the market. Products have to be developed and produced in a short time, with low costs and high quality. To meet these requirements the development is supported by using many different computer-aided systems (CAx systems). Engineers use some of these systems to digitally model products. Other CAx systems use the modelled

data for analyses, simulations, production planning or visualisation, for instance. A continuous usage of CAx systems in the product development is also called *virtual product development*.

Today more and more developed products are mechatronic systems which consist of components developed in different domains (e.g. mechanical, electric and software development). Traditionally separated, these domains have been evolved independently of each other. As a result, different methods and software systems exist for developing in the various domains. Consequently, several specialised CAx systems are used to develop a product. Among CAx systems of different domains, but also among systems inside a domain data has to be exchanged. Thus, for data exchange different interfaces between CAx systems exist. That leads to complex CAx system architectures inside a company or among several companies working together. Furthermore, CAx system architectures change over time because CAx systems can be replaced with a new version or a complete new CAx system. It is difficult to keep an overview of the architecture and the changing possibilities for data exchange among CAx systems. Another problem is the traceability of executed data exchange processes. For the described problems, a data model, which describes the required data (e.g. CAx systems, interfaces, process chains, executed process chains) and their relationships to establish a flexible management of data exchange processes was developed in [20]. We present the main results of that work in this paper.

The fundamentals for this paper are presented in section 2, starting with the reasons for data exchange among CAx systems. Hereafter, typical file formats for data exchange are compared and WFM systems as state of the art for process management are described. In section 3, we present an analysed example of a development process focussing on necessary data exchanges. After this, in section 4, a data model for the flexible management of data exchange processes is presented. A prototypical implementation based on the data model is described in section 5. Finally, we conclude the paper and present future work in section 6.

2. FUNDAMENTALS

In the following sections, the necessity of data exchange, typical file formats for data exchange and current approaches to support data exchange processes are presented.

2.1 Data Exchange

Data exchange between CAx systems is necessary to avoid

error-prone and time-consuming recreation of already defined product data. There are research works for data exchange among CAx systems that use *integrated product models* (e.g. [1, 6]). An integrated product model integrates partial models and can be used as a uniform database for several CAx systems. Because this approach is not supported by the CAx system manufacturers, it is currently not common to use integrated product models [19]. However, the growing complexity of products demands the integration of several domains and is in the focus of current research for this topic [7, 22].

This paper focuses on file based data exchange, the most common way to exchange information between CAx systems. One CAx system stores data into a file and another CAx system can read the data from this file. Data exchange often requires a transformation of data from one CAx system in a readable format for another system. File formats have a defined set of supported data that can be represented. Only data which is in the intersecting set of two CAx systems and the exchange file format can be exchanged between both systems (see Figure 1). For example, material data which can be defined in two different CAx systems, can only be transferred between both systems if the exchange file format supports it, too. Therefore, data exchange depends heavily on the right choice of the exchange file format according to a given use case.

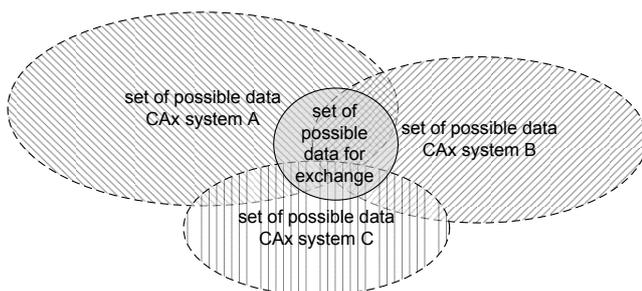


Figure 1: Intersecting sets of supported data that can be exchanged

For the realization of a file based interface there are two different approaches [22]:

- **System specific interfaces** define a specific exchange between two CAx systems. The advantage is that the interface can be optimized in order to support the transfer of many data between both systems. However, to support n CAx systems $n \cdot (n - 1)$ specific converters have to be implemented.
- **System neutral interfaces** define a common file format (also called neutral file format) for the exchange between systems. The disadvantage is that the possibilities to exchange data are limited to the supported set of data of the defined file format. The advantage is that the development of interfaces for n CAx systems requires only $2 \cdot n$ converter, which transform data in and from the defined file format.

It depends on a given use case which approach should be used for the realisation of an interface.

2.2 Neutral File Formats

Similar to interfaces, file formats can be separated in *native* and *neutral file formats*. Native file formats are CAx

system manufacturer specific and their specification is often not freely available. They are optimised for one CAx system or various CAx systems of one manufacturer and can store all in these CAx systems defined data. Usually, they are not useful for data exchange because only few CAx systems can read them. Thus, a transfer into another native or neutral file format is necessary.

Neutral file formats do not depend on one CAx system and their specification is often freely available. Some neutral file formats are normed. That means they are defined and maintained by a norming institution like the *International Organization for Standardization (ISO)*. We analysed typical standard formats to compare file exchange formats. More and detailed comparisons of neutral file formats are for example given in [2, 17, 21]:

- **Initial Graphics Exchange Specification (IGES)** also known as standard "ANSI Y14.26M" was first released in 1980 and is one of the first developed exchange formats for *computer-aided design (CAD)* systems [14]. Although the development ended in 1996, today IGES is used in many CAx systems especially for the exchange of geometry data.
- **Standard for the Exchange of Product Model Data (STEP)** is a working title for the international "ISO 10303" series of standards [8]. In contrast to many standard formats, STEP does not exclusively focus on geometry data. Many different kinds of product data that are generated in the life cycle of a product can be exchanged using STEP [16]. STEP defines a set of use case specific neutral file formats and can also be used as a construction kit to define new neutral file formats.
- **Jupiter Tessellation (JT)** is an industry standard, which is currently in development to an ISO standard [3]. The version "8.1b" is already available at ISO as a so called *public available specification* ISO/PAS 14306 [10]. JT files contain different representations of 3D models together (tessellated triangle surfaces and parametric models). Additionally, it is possible to define product manufacturing information (e.g. dimensioning, surface quality) and visualisation specific data (e.g. textures, lights, colours). Furthermore, JT has a strong compression algorithm.
- **Drawing Interchange Format (DXF)** is an industry standard developed by the CAx system manufacturer *Autodesk*. The DXF file format is not overloaded and is supported by many CAD systems as an export and import format.
- **Virtual Reality Modeling Language (VRML)** is normed as ISO standard 14772-1 [9]. VRML is a format for the description of 3D scenarios, their geometry, lights, animation and further visualisation related information. In the product development, it is mainly used for the exchange between CAD and *virtual reality* systems.

The paper shows a comparison of the basic properties of the described neutral file formats in Table 1. It describes which neutral file formats are normative standards and which are industry standards. In addition, the type of possible formats for files (ASCII, binary or XML) is shown. Furthermore, the supported content for an exchange is compared.

As conclusion of the comparison: To execute successfully

a data exchange, it is important to know which data can be transferred with a specific file format. The given comparison of the supported content is very compact and has to be refined according to a given use case. For example, if a 3D model with material data has to be transferred between two CAx systems then one has to use a file format that supports both: 3D model data and material data.

Table 1: Comparison of neutral file formats (✓ means can be transferred; - means cannot be transferred)

property	IGES	STEP	JT	DXF	VRML
norm (n) / industry standard (i)	n	n	i	i	n
ASCII (a) / binary (b) / XML (x)	a, b	a, x	b	a	a
2D models	✓	✓	-	✓	-
3D facet models	✓	✓	✓	-	✓
3D parametric models	✓	✓	✓	✓	-
non-geometry data	✓	✓	✓	✓	✓

2.3 State of the Art

WFM systems focus on sending correct information to the right people at the right time. The usage of WFM systems requires the definition of processes. Afterwards instances of those processes are executed and controlled. WFM systems mainly support structured, pre-planned and repetitive processes [13].

Compared to processes in production and logistics, the product design process cannot be pre-planned in many cases. The process often changes during development. Moreover, developing a product is dominated by creative and dynamic processes (e.g. iterations). Predefining the development process would constrain the creative freedom of developers and limit their ability to respond. Nevertheless, there are sub-processes in product development that can be pre-planned and automated, for example, release and change processes. Therefore, WFM functionality is often integrated in *product data management (PDM) systems* that are used in product development companies for data management in cooperation with the definition and management of processes [4].

WFM systems can be used for management of data exchange processes but since WFM systems are developed for process management in general they are not suitable to support the management of data exchange processes in detail. The data model presented in this paper could be used to extend a WFM system to meet the special requirements for data exchange processes.

3. ANALYSED DEVELOPMENT PROCESS

We analysed the procedure of a *multibody system (MBS) simulation* as an exemplary data exchange process. Engineers use MBS simulations to analyse the kinematic and dynamic behaviour of a product [18]. In the example process, several CAx systems are used in order to enable a MBS simulation. Data has to be exchanged among these CAx systems. The analysed process is not very complex but it is a typical process in product development environments and contains all problems that motivate the requirements of the traceability of data exchange processes. In Figure 2, a

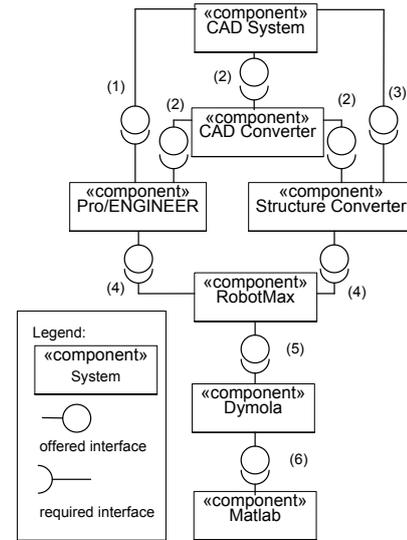


Figure 2: Component diagram showing the involved CAx systems of the analysed MBS simulation process and their interfaces

component diagram illustrates the CAx systems that are involved in the analysed process. In the picture the interfaces are numbered. They also represent process steps for data exchange between the CAx systems. Hereafter, we describe the process steps of the analysed MBS simulation:

As a basis for the MBS simulation a CAD model of a product is necessary. It can be created from any CAD system and saved in a format that can be imported in the CAD system *Pro/ENGINEER* (1). If this is not possible, a CAD converter has to be used to transform the CAD model into a readable format for *Pro/ENGINEER* (2). As an alternative possibility, the initial CAD model can be saved in the STEP format (3) or transformed into it (2) and be imported in a software system called *Structure Converter*. *Pro/ENGINEER* or the *Structure Converter* can create the required data for the *RobotMax* system (4). In this system the data from the CAD model is enriched with electronic components (see [11, 12] for more details). The result is a mechatronic MBS model that is used for a MBS simulation in *Dymola* (5). *Dymola* is a commercial system for different simulations. It is based on the object-oriented language *Modelica* [5, 15] that can be used for modelling physical systems, which consist of components of different domains. Another way to simulate the MBS system is to create an output file as *C source code* with *Dymola* for *Matlab* (6). *Matlab* is also a commercial system that can be used for simulations among other things.

When an instance of the described process has been executed, data is generated. In Figure 3, a UML class diagram is used to illustrate the possibly generated data and their relationships as well as their dependencies. The class *CAD-Data-Model for RobotMax* represents an abstract aggregation of the CAD data that is needed for *RobotMax* (one or more *VRML files* and an *XML file*). There are two possibilities to create them. One is based on a *STEP-CAD-Model* which might be converted based on another *CAD model*. A CAD model is, according to its format, represented by one or more *CAD files*. For example, for a CAD model in the

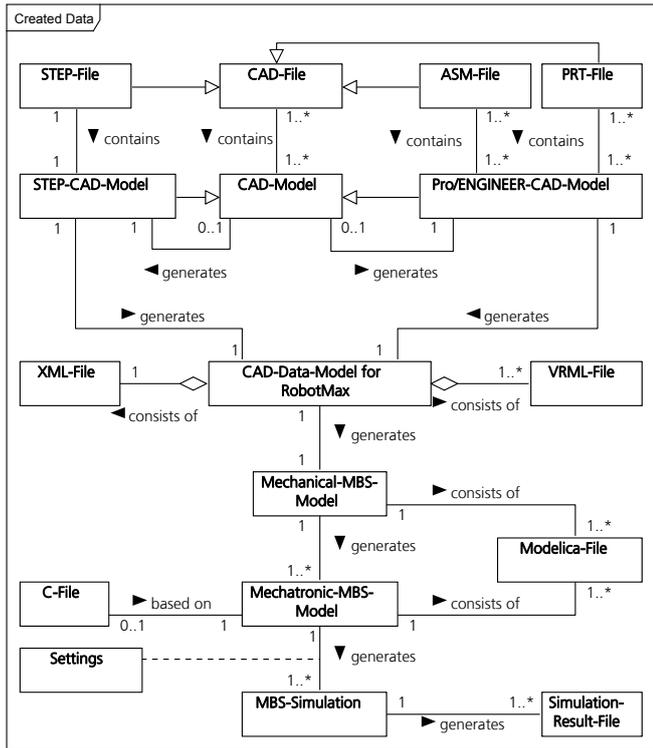


Figure 3: Class diagram with the data and their relationships that are created at the example process

STEP format exists one *STEP file*. The other possibility is a native *Pro/ENGINEER-CAD-Model* represented by one or more PRT and ASM files. Eventually, it can be the result of a conversion from another *CAD model*.

With a given *CAD-Data-Model RobotMax* creates at first a *Mechanical-MBS-Model*. It is represented by one or more *Modelica-Files*. In RobotMax it is possible to add electronic components, e.g. engines, so that the *Mechanical-MBS-Model* is extended to a *Mechatronic-MBS-Model*, which is also represented by one or more *Modelica-Files*. Based on one mechanical model different mechatronic models can be created. With a *Mechatronic-MBS-Model* different simulations can be executed because different settings in the simulation software can be used. A *MBS-Simulation* creates one or more *Simulation-Result-Files*.

It has to be mentioned that in general it is possible to use different settings, when a file is imported or exported in a CAx system. This aspect is not modelled in Figure 3 in order to reduce the complexity and improve the readability of the diagram. An analysis of import and export settings of data in CAx systems is given in [20]. To summarize, the characteristics of the analysed process are:

- different possibilities for the execution of the process
- different file formats are used as input
- data is generated with complex relationships and dependencies
- different settings can be used for importing and exporting files

Based on these characteristics, the requirements for supporting data exchange processes are:

- a description of different possibilities for execution and file formats
- the traceability of executed data exchange processes

The requirement for a description of different possibilities means that a data exchange process should be described in detail. This could be for example, information about the differences between alternate ways for data exchange as well as advantages and disadvantages of file formats in a specific use case. With the help of this information a user can choose the way for his given use case.

The traceability of data exchange processes enables the management of information about executed processes, for instance the chosen exchange steps and file formats, the used settings and also a reference to the files which have been transferred. Based on these requirements we created a data model that is presented in the following section.

4. DATA MODEL FOR FLEXIBLE DATA EXCHANGE PROCESSES

The development of the data model has the goal to enable the management of CAx system architectures and data exchange processes. It meets the demands that we outlined in the last section by the analysis of the exemplary data exchange process. The data model is shown as an UML class diagram in Figure 4. It is separated into two part data models to enhance the comprehensibility and maintainability:

- The first data model described in section 4.1 focuses on the management of CAx system architectures and is represented by the white coloured classes.
- The second data model described in section 4.2 focuses on the management of data exchange processes and is represented by light and dark grey coloured classes.

4.1 System Management

A typical CAx system architecture consists of several CAx systems from different manufacturers, as well as interfaces between them. The part of the data model that focuses on CAx system management allows the description of CAx systems and their versions. Every CAx system version can export and import zero or several file formats. The settings, which can be used by the import and export, can be defined with the class *Settings*. It consists of several separate defined settings. For some settings the user has to choose one of many options. These possible options can be defined with several instances of the class *Choice*. Furthermore, the data model supports the description of interfaces between CAx systems.

The difference between interfaces and file formats for export and import definition of CAx systems is that an interface encapsulates all necessary exchange processes for exchanging data between two CAx systems. For each exchange process different file formats can be used. In the analysed process for example, there is an interface between Pro/ENGINEER and RobotMax. To exchange data between both systems two different exchange processes with different file formats (XML and VRML) have to be executed.

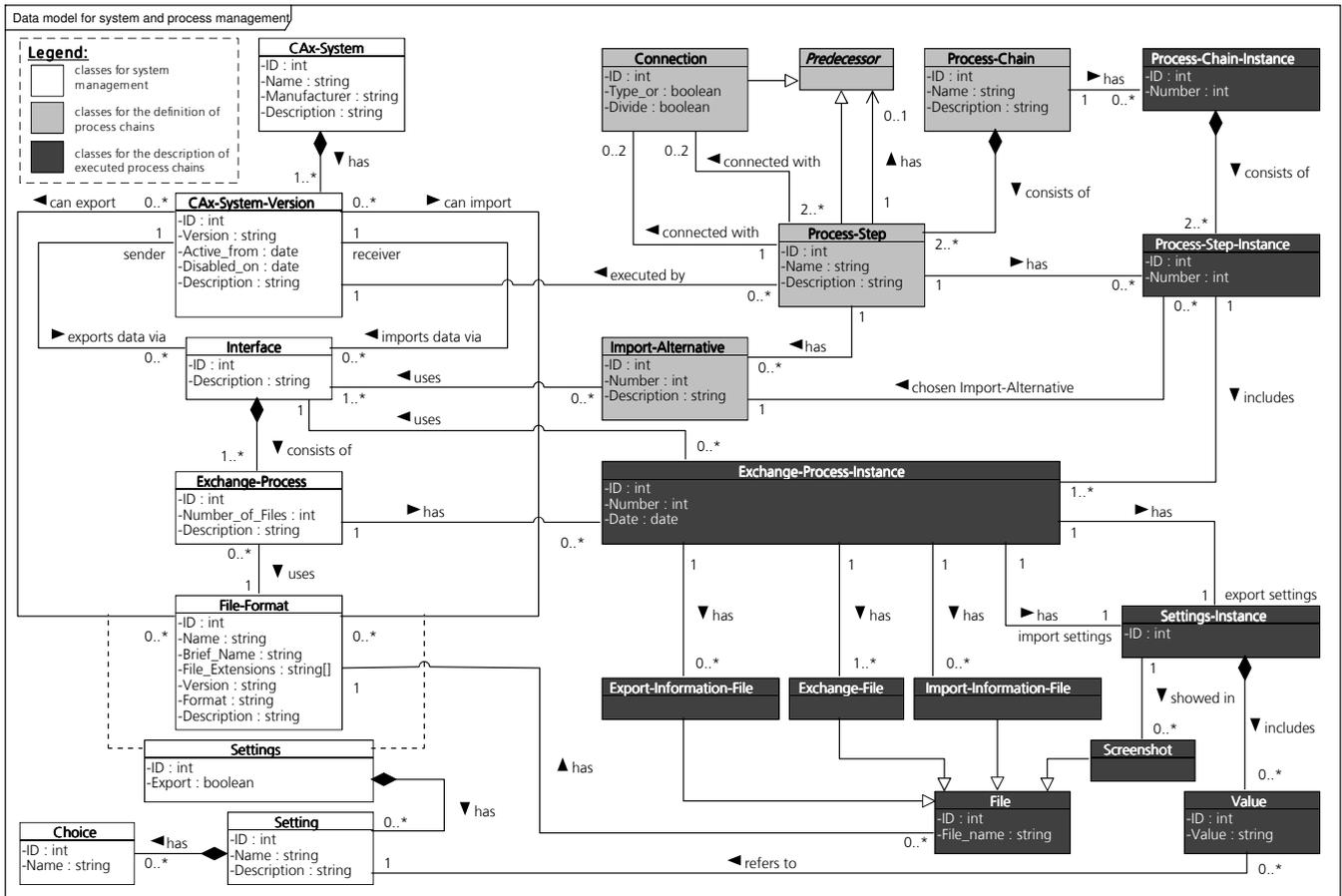


Figure 4: Class diagram showing the data model for system and process management

The defined data model allows the description of a CAX system architecture in a detailed way. It is possible to reconstruct old states of an architecture with the help of the date attributes (*Active_from*, *Disabled_on*). With the data that can be described by the data model, it is possible to create software that can give the following information to the users in an engineering company:

- the CAX systems in the company
- interfaces that can be used to exchange data between CAX systems
- differences or advantages/disadvantages if more than one interface can be used for a data exchange between two CAX systems
- import and export possibilities of CAX systems including settings
- file formats and which can be imported and exported by which CAX system

We use an example to illustrate and explain the data model. The example refers to the analysed process of section 3 of which the interface between the CAX system Pro/ENGINEER and RobotMax was chosen. Figure 5 shows an UML object diagram with instances of the presented data model. Two concrete versions of both CAX systems were

defined as well as the related interface, exchange processes and file formats. For an import in RobotMax an XML file and various VRML files are necessary. Pro/ENGINEER can export them by two separate exchange processes.

4.2 Process Management

Based on the proposed data model for system management a data model for process management was defined. It consists of two groups of classes:

- one group focuses on the definition of process chains (light grey classes)
- the other group focuses on the description of executed process chains (dark grey classes)

First, we describe the classes for the definition of process chains. A process chain defines the necessary and possible data exchange processes among several CAX systems. An example for a process chain is the analysed process in section 3. Several data exchange operations have to be executed to realize a MBS simulation. A process chain consists of several process steps, which are connected with each other. Each process step is executed by a defined CAX system version. An example for this is a MBS simulation that is executed by a concrete version of the CAX system Dymola. Each process step, except the first one, has one or more import alternatives. That means, one or different ways to import

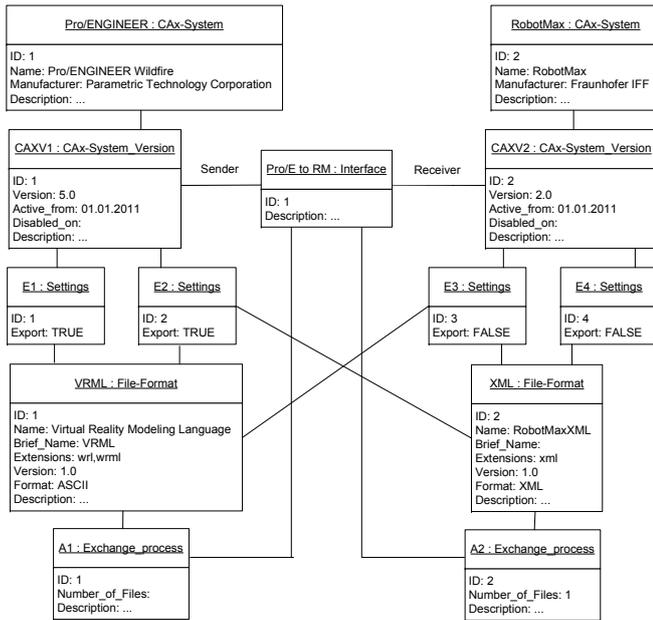


Figure 5: Object diagram for system management (example)

data that is necessary to execute the process step. For example, to generate a mechatronic MBS model in RobotMax one can import data from Pro/ENGINEER or the Structure Converter.

With this part of the data model it is possible to implement software that can be used to define and describe process chains. Every process chain consists of several process steps that are connected by one or different possibilities for data exchange. A user in an engineering company can obtain the following information by defined process chains:

- which possibilities exist to exchange data between two process steps
- what are the differences if there is more than one possibility for an exchange

The second group of classes of the process management data model can be used to describe data that is generated by an executed instance of a process chain. The process chain instance consists of several process step instances and exchange process instances. Every process step instance knows the selected import alternative because in a process chain instance only one of the various possibilities were selected. For each exchange process instance detailed information can be described like the used settings and generated files. This information about an executed process chain makes the data exchange processes traceable. Software that is implemented based on the data model can give a user the following information:

- how a process was executed and which of the possibilities for data exchange were chosen
- the used settings for import and export of files
- exchanged and generated files

5. IMPLEMENTATION

We evaluated the data model by implementing the model in a prototype. The prototype was realized as a web-based client-server application.

The database of the application was implemented as a relational *PostgreSQL* database. Therefore, a relational database schema was defined following the data model as a form of a conceptual schema (see Figure 6). The database schema includes tables to save the data for system and process management. The arrows represent foreign key relations.

The architecture of the implemented web-based application consists of three layers:

- **Presentation Layer:** The user interface is realized by the client side with a web browser. Therefore, several *Hypertext Markup Language (HTML)*, *Cascading Style Sheets (CSS)* and *JavaScript* files are provided by the web application.
- **Application Logic Layer:** The application logic is implemented in a *Java web application* that runs on an *Apache Tomcat* web server.
- **Database Layer:** A *PostgreSQL* database stores all persistent data of the web application and communicates with the web application over the *Java Database Connectivity (JDBC)* interface.

With the implemented prototype it is possible to define a CAx system architecture, process chains and describe executed process chains. Furthermore, the prototype gives access to the defined data.

6. CONCLUSIONS & PERSPECTIVES

Virtual product development includes complex CAx system architectures and between those systems complex data exchange processes. An analysis of typical exchange formats has shown that it is important to choose the right file formats for data exchange processes. Furthermore, based on an example development process we analysed the following requirements for the management of flexible data exchange processes:

- a description of different possibilities for execution and file formats
- the traceability of executed data exchange processes

To meet these requirements we presented a data model in this paper. To enhance the comprehensibility and maintainability it is separated into two data models. One data model describes data for a management of CAx system architectures, for example CAx systems, their versions, interfaces between the CAx systems and file formats that can be used for import and export.

The second data model describes data for the definition of process chains and executed process chains. The definition of process chains allows to describe the necessary and possible data exchange processes among several CAx systems. It is possible to describe different ways for data exchanges in a process chain including, for instance, what advantages and disadvantages for different file formats exist. Based on this information, a user can decide which alternative to choose for data exchange in a given use case. Furthermore, information about executed instances of defined process chains can be described, for example used setting, used alternatives for data exchange and generated files. With this information executed data exchange processes are traceable.

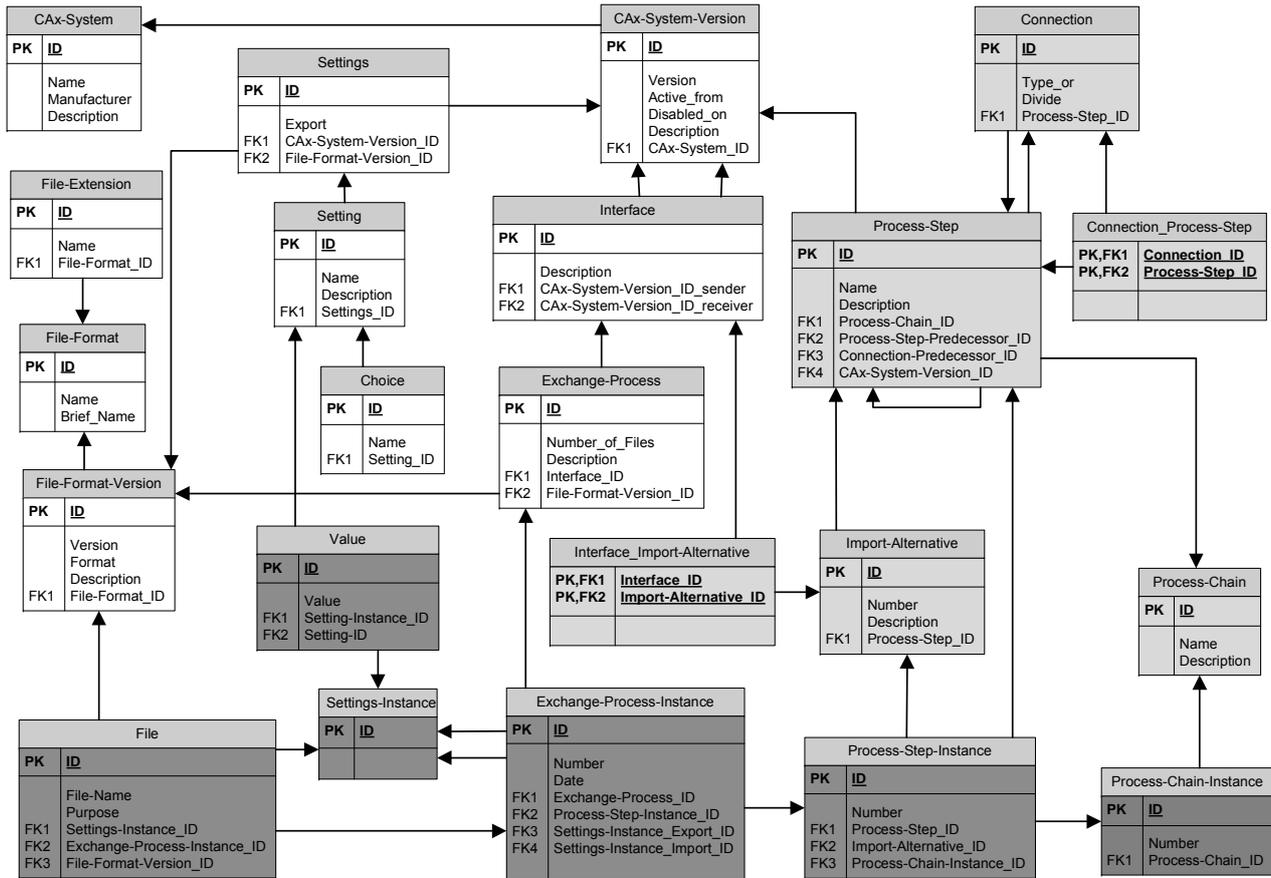


Figure 6: Database schema (white tables = system management data; grey tables = process management data)

To evaluate the data model we implemented a prototype. In the paper the database schema of the prototype was presented and the architecture was described. As a next step, an implementation of the data model as an extension of existing WFM systems or WFM components in PDM systems should be analysed. Furthermore, adjustments of the data model to other domains, and interchange formats of virtual product development are the subject of future research and development.

7. ACKNOWLEDGMENTS

The work described in this paper has been supported by the European Commission: European Regional Development Fund, project "COMO" C1-3201201 and C3-3201201.

8. REFERENCES

[1] G. Brunetti and B. Golob. A feature-based approach towards an integrated product model including conceptual design information. *Computer-Aided Design*, 32(14):pages 877 – 887, 2000.

[2] A. Dyla. *Modell einer durchgängig rechnerbasierten Produktentwicklung*. PhD thesis, Technical University Munich, 2002 (in German).

[3] M. Eigner, S. Handschuh, and F. Gerhardt. Concept to Enrichen Lightweight, Neutral Data Formats with

CAD-based Feature Technology. *Computer-Aided Design and Applications*, 7(1):pages 89–99, 2010.

[4] M. Eigner and R. Stelzer. *Product Lifecycle Management: Ein Leitfaden für Product Development und Life Cycle Management*. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2009 (in German).

[5] H. Elmqvist and S. E. Mattsson. An Introduction to the Physical Modeling Language Modelica. In W. Hahn and A. Lehmann, editors, *Proceedings of the 9th European Simulation Symposium ESS'97, 19-23th October, Passau, Germany*, pages 110–114. 1997.

[6] S. J. Fennes, Y. Choi, B. Gurumoorthy, G. M. Mocko, and R. D. Sriram. *Master Product Model for the Support of Tighter Integration of Spatial and Functional Design (NISTIR 7004)*. Technical report, U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, Gaithersburg, MD, 2003.

[7] M. Hirz. An approach of multi disciplinary collaboration in conceptual automotive development. *International Journal of Collaborative Enterprise*, 2(1):pages 39–56, 2011.

[8] ISO 10303-1: *Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles*, 1994.

[9] ISO 14772-1: *Information technology - Computer*

- graphics and image processing - The Virtual Reality Modeling Language - Part 1: Functional specification and UTF-8 encoding*, 1997.
- [10] ISO/PAS 14306: *JT File Format Specification harvesting for 3D Visualization of industrial data*, 2009.
- [11] T. Juhász. *Advanced Solutions in Object-Oriented Mechatronic Simulation*. PhD thesis, Budapest University of Technology and Economics, Budapest, 2008.
- [12] T. Juhász and U. Schmucker. Automatic Model Conversion to Modelica for Dymola-based Mechatronic Simulation. In B. Bachmann, editor, *Modelica 2008: Proceedings of the 6th International Modelica Conference; 3rd and 4th March, Bielefeld, Germany*, pages 719–726, 2008.
- [13] C.-H. Ma, Y.-T. Ko, and D.-B. Luh. A structure-based workflow planning method for new product development management. *International Journal of Management Science and Engineering Management*, 4(2):pages 83–103, 2008.
- [14] R. N. Nagel, W. W. Braithwaite, and P. R. Kennicott. *Initial Graphics Exchange Specification (IGES), Version 1.0, N.B.S. Report NBSIR 80-1978 (R)*, 1980.
- [15] M. Otter and H. Elmqvist. Modelica - Language, Libraries, Tools, and Conferences. *Simulation News Europe*, December:pages 3–8, 2000.
- [16] M. J. Pratt. Introduction to ISO 10303 - the STEP Standard for Product Data Exchange. *Journal of Computing and Information Science in Engineering*, 1(1):pages 102–103, 2001.
- [17] S. Schuhmann. *Methoden zur Optimierung des Austausches von Geometrie- und Parametrikdaten*. PhD thesis, Otto-von-Guericke University Magdeburg, 2001 (in German).
- [18] A. A. Shabana. *Dynamics of multibody systems*. Cambridge University Press, New York, NY, 2005.
- [19] A. Stekolschik. *Ein Beitrag zum ganzheitlichen Qualitätsmanagement von CAD-Modellen in der Produktentstehung*. PhD thesis, Ruhr-University Bochum, 2007 (in German).
- [20] M. Stoye. *Entwicklung eines Datenmodells zur Unterstützung des dateibasierten Datenaustauschs in der Produktentwicklung*, Diploma thesis, Otto-von-Guericke University Magdeburg, 2011 (in German).
- [21] VDI 2209: *3D product modelling - Technical and organizational requirements - Procedures, tools, and applications - Cost-effective practical use*. Beuth Verlag GmbH, Berlin, 2009.
- [22] S. Vornholt, I. Geist, and Y. Li. Categorisation of Data Management Solutions for Heterogeneous Data in Collaborative Virtual Engineering. In G. Saake and V. Köppen, editors, *Proceedings of the First International Workshop on Digital Engineering, IWDE '10, June 14, Magdeburg, Germany*, pages 9–16, 2010.



**VIRTUAL AND
AUGMENTED
REALITY**