Record output: Abstract

NOTE: Your selected records (to a maximum of 500) will be kept until your session ends. However, to delete them after this task:

- Return to the Search results page and click Delete Selected Records, or
- Go to the Selected records page and click Remove All, or
- Click the End session link at the top of the page

---

1. **Visualization of software architecture graphs of Java systems: Managing propagated low level dependencies**

Schrettner, Lajos[1]; Fülöp, Lajos Jeno[1]; Ferenc, Rudolf[1]; Gyimóthy, Tibor[1]

Source: *Proceedings of the 8th International Conference on the Principles and Practice of Programming in Java, PPPJ 2010*, p 148-157, 2010, *Proceedings of the 8th International Conference on the Principles and Practice of Programming in Java, PPPJ 2010*; **ISBN-13:** 9781450302692; **DOI:** 10.1145/1852761.1852783; **Conference:** 8th Annual Conference on Principles and Practice of Programming in Java, PPPJ 2010, September 15, 2010 - September 17, 2010; **Sponsor:** ORACLE; Vienna University of Technology; computer languages; **Publisher:** Association for Computing Machinery

Author affiliation:
[1] Department of Software Engineering, University of Szeged, Hungary

Abstract: The availability of up-to-date documentation of the architecture is crucial for software maintenance tasks, but it is often missing or differs from the implemented architecture. An increasingly popular and feasible way to get a clear picture of the architecture is to reconstruct it from the source code. The result of the reconstruction procedure is a graph with special, architecture-specific properties. Nowadays software systems are typically very large, so the reconstructed architecture contains a lot of details and is really difficult to interpret. It is important therefore to have efficient methods that help in understanding and managing the architecture graph. The purpose of these methods is to try to present the information so that it is comprehensible to the users. Two important methods are selective subtree collapsion and lifting low level dependencies of the system into higher, visible levels. These enable an architect to investigate the dependencies of system components at higher levels, without the need to deal with an enormous quantity of low-level details. In this paper, first we overview the concepts related to lifting and present a conceptual framework that combines subtree collapsion with lifting to enable users to interactively explore and manipulate a software architecture graph. Then we define a set of algorithms that can be used to efficiently propagate dependency edges of a graph to higher levels. We also describe how the results can be integrated into SourceInventory, a software quality monitoring and visualization framework. © 2010 ACM. (15 refs.)

Main Heading: Software architecture

Controlled terms: Computer software maintenance - Computer software selection and evaluation - Java programming language - Visualization

Uncontrolled terms: Architecture reconstruction - architecture visualization - Conceptual frameworks - Efficient method - Java system - lifting - Low level - nested set model - Reconstruction procedure - Software Quality - Software systems - Software-maintenance tasks - Source codes - Subtrees - System components - Visualization framework

Classification Code: 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics

Database: Compendex

2. **Connecting research and practice: An experience report on research infusion with software architecture visualization and evaluation**

Lindvall, Mikael[1]; Stratton, William C.[2]; Sibol, Deane E.[2]; Ackermann, Christopher[1]; Reid, W.Mark[3]; Ganesan, Dharmalingam[1]; McComas, David[4]; Bartholomew, Maureen[4]; Godfrey, Sally[4] **Source:** *Innovations in Systems and Software Engineering*, v 8, n 4, p 255-277, December 2012; **ISSN:** 16145046, **E-ISSN:** 16145054; **DOI:** 10.1007/s11334-010-0135-y; **Publisher:** Springer London

Author affiliation:
[1] Fraunhofer Center for Experimental Software Engineering (FC-MD), College Park, MD, United States
[2] Space Department Information Systems Branch Ground Applications Group (SIG), Johns Hopkins University Applied Physics Laboratory (JHU/APL), Laurel, MD, United States
[3] Space Department Information Systems Branch Embedded Application Group (SIE), Johns Hopkins University Applied Physics Laboratory (JHU/APL), Laurel, MD, United States
[4] Software Engineering Division, NASA Goddard Space Flight Center, Greenbelt, MD, United States

Abstract: There are many technical challenges in ensuring high life-time quality of NASA's systems. Some of NASA's software-related challenges could potentially be addressed by the many powerful technologies that are being developed in software research laboratories. However, most such research technologies do not make the transition from the research lab to the software lab because research infusion and technology transfer is difficult. For example, there must be evidence that the technology works in the practitioner's particular domain, and there must be a potential for great improvements and enhanced competitive edge for the practitioner, for such infusion to take place. NASA IV&V's Research Infusion initiative strives to facilitate such infusion. In

2006, a research infusion project involving Johns Hopkins University Applied Physics Laboratory (JHU/APL) and the Fraunhofer Center for Experimental Software Engineering Maryland, was successfully completed infusing Fraunhofer's software architecture visualization and evaluation (SAVE) tool. The infusion project helped improve JHU/APL's software architecture and produced evidence that SAVE is applicable to software architecture problems in the aerospace domain, spawning a series of related research infusion projects. The project also led to the discovery of other needs that could not be addressed by current technologies and, therefore, spawned the research and development of a new technology that will be ready for infusion in the future. This paper describes the SAVE technology followed by a description of the infusion of SAVE at JHU/APL and the other projects that followed, as well as the newly started Dynamic SAVE research and development project. Lessons learned related to various aspects of research infusion conclude the paper. © 2010 Springer-Verlag London Limited. (12 refs.)

Main Heading: Research

Controlled terms: NASA - Research laboratories - Software architecture - Technology transfer - Visualization

Uncontrolled terms: Aerospace domain - Architecture visualization - Competitive edges - Current technology - Experience report - Experimental software engineering - Fraunhofer - Johns Hopkins University Applied Physics laboratories - Life-times - Maryland - Research and development - Research and development projects - Research technologies - Technical challenges

Classification Code: 655 Spacecraft - 656 Space Flight - 723.1 Computer Programming - 901.3 Engineering Research - 901.4 Impact of Technology on Society - 902.1 Engineering Graphics

Database: Compendex

3. **Communicating software architecture using a unified single-view visualization**

Panas, Thomas[1]; Epperly, Thomas[1]; Quinlan, Daniel[1]; Sbjrnsen, Andreas[1]; Vuduc, Richard[1] **Source:** *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, p 217-226, 2007, *Proceedings - 12th IEEE International Conference on Engineering Complex Computer Systems, ICECCS 2007*; **ISBN-10:** 0769528953, **ISBN-13:** 9780769528953; **DOI:** 10.1109/ICECCS.2007.20; **Article number:** 4276318; **Conference:** 12th IEEE International Conference on Engineering Complex Computer Systems, ICECCS 2007, July 11, 2007 - July 14, 2007; **Sponsor:** University of Auckland; Royal Society of New Zealand; Asian Office of Aerospace R and D; **Publisher:** Institute of Electrical and Electronics Engineers Inc.

Author affiliation:

[1] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

Abstract: Software is among the most complex human artifacts, and visualization is widely acknowledged as important to understanding software. In this paper, we consider the problem of understanding a software system's architecture through visualization. Whereas traditional visualizations use multiple stakeholder-specific views to present different kinds of taskspecific information, we propose an additional visualization technique that unifies the presentation of various kinds of architecture-level information, thereby allowing a variety of stakeholders to quickly see and communicate current development, quality, and costs of a software system. For future empirical evaluation of multi-aspect, single-view architectural visualizations, we have implemented our idea in an existing visualization tool, Vizz3D. Our implementation includes techniques, such as the use of a city metaphor, that reduce visual complexity in order to support single-view visualizations of large-scale programs. © 2007 IEEE. (36 refs.)

Main Heading: Software architecture

Controlled terms: Computer software - Computer software selection and evaluation - Computer systems - Computers - Visualization

Uncontrolled terms: complex computer systems - Empirical evaluations - In order - international conferences - Large scale programs - Software systems - Visual complexity - Visualization techniques - Visualization tools

Classification Code: 722 Computer Systems and Equipment - 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming - 902.1 Engineering Graphics

Database: Compendex

4. **Visualization and evolution of software architectures**

Khan, Taimur[1]; Barthel, Henning[2]; Ebert, Achim[1]; Liggesmeyer, Peter[2]
Source: *OpenAccess Series in Informatics*, v 27, p 25-42, 2012, *Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering, VLUDS 2011 - Proceedings of IRTG 1131 Workshop*; **ISSN:** 21906807; **ISBN-13:** 9783939897460; **DOI:** 10.4230/OASIcs.VLUDS.2011.25; **Conference:** International Research and Training Group 1131 Workshop on Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering, VLUDS 2011, June 10, 2011 - June 11, 2011; **Publisher:** Schloss Dagstuhl - Leibniz-Zentrum fur Informatik GmbH,

Author affiliation:
[1] Computer Graphics and HCI Group, University of Kaiserslautern, Germany
[2] Fraunhofer IESE, Kaiserslautern, Germany

Abstract: Software systems are an integral component of our everyday life as we find them in tools and embedded in equipment all around us. In order to ensure smooth, predictable, and accurate operation of these systems, it is crucial to produce and maintain systems that are highly reliable. A well-designed and well-maintained architecture goes a long way in achieving this goal. However, due to the intangible and often complex nature of software architecture, this task can be quite complicated. The field of software architecture visualization aims to ease this task by providing tools and techniques to examine the hierarchy, relationship, evolution, and quality of architecture components. In this paper, we present a discourse on the state of the art of software architecture visualization techniques. Further, we highlight the importance of developing solutions tailored to meet the needs and requirements of the stakeholders involved in the analysis process. (78 refs.)

Main Heading: Software architecture

Controlled terms: Computer software maintenance - Visualization

Uncontrolled terms: Architecture visualization - Developing solutions - Human perception - Integral components - Software comprehension - Software Evolution - State of the art - Tools and techniques

Classification Code: 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics

Database: Compendex

5. **Visual software architecture description based on design space**

Zhang, Qian[1] **Source:** *Proceedings - International Conference on Quality Software*, p 366-375, 2008, *Proceedings - 8th International Conference on Quality Software, QSIC 2008*; **ISSN:** 15506002; **ISBN-13:** 9780769533124; **DOI:** 10.1109/QSIC.2008.59; **Article number:** 4601566; **Conference:** 8th International Conference on Quality Software, QSIC 2008, August 12, 2008 - August 13, 2008; **Sponsor:** Oxford Brookes University; The University of Hong Kong; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] Department of Computer Science, National University of Defense Technology, Changsha, China

Abstract: Boxology is the essence of software architecture description. In comparison with text-based languages, well-defined visual notations model software architecture at a high level of abstraction. They are easy to understand and easy to use due to its simplicity, but less expressive as many architectural properties can not be adequately represented. A key question to be answered in the design of a visual notation for the description of software architectures is what properties should be visually represented. This paper applies the theory of design space in the development of a visual notation called ExSAVN for software architectural modeling. It is based on the design

space of software architectural elements to determine the properties of software architectures that are visually represented. It achieves balance between simplicity and expressiveness and supports incremental and iterative architectural design through a number of high level language facilities, which include the representation of undecided properties, hierarchical abstraction and type definition facilities. The paper also illustrated the style of ExSAVN by some examples of real systems. © 2008 IEEE. (21 refs.)

Main Heading: Architectural design

Controlled terms: Abstracting - Architecture - Computer software selection and evaluation - Design - High level languages - Large scale systems - Linguistics - Query languages - Software architecture - Software design

Uncontrolled terms: Architectural properties - Architecture description - Architecture description language - Design space - Design spaces - High level of abstraction - International conferences - Quality software - Real systems - Software architectural - Visual notation - Visual notations

Classification Code: 903.2 Information Dissemination - 903.1 Information Sources and Analysis - 723.1.1 Computer Programming Languages - 961 Systems Science - 723.1 Computer Programming - 408 Structural Design - 402 Buildings and Towers - 723 Computer Software, Data Handling and Applications

Database: Compendex

6. **Improved feedback for architectural performance prediction using software cartography visualizations**

Krogmann, Klaus[1]; Schweda, Christian M.[2]; Buckl, Sabine[2]; Kuperberg, Michael[1]; Martens, Anne[1]; Matthes, Florian[2] **Source:** *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 5581 LNCS, p 52-69, 2009, *Architectures for Adaptive Software Systems - 5th International Conference on the Quality of Software Architectures, QoSA 2009, Proceedings*; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3642023509, **ISBN-13:** 9783642023507; **DOI:** 10.1007/978-3-642-02351-4_4; **Conference:** 5th International Conference on the Quality of Software Architectures, QoSA 2009, June 24, 2009 - June 26, 2009; **Publisher:** Springer Verlag

Author affiliation:
1 Software Design and Quality Group, Universität Karlsruhe (TH), Germany
2 Software Engineering for Business Information Systems, Technische Universität München, Germany

Abstract: Software performance engineering provides techniques to analyze and predict the performance (e.g., response time or resource utilization) of software systems to avoid implementations with insufficient performance.

These techniques operate on models of software, often at an architectural level, to enable early, design-time predictions for evaluating design alternatives. Current software performance engineering approaches allow the prediction of performance at design time, but often provide cryptic results (e.g., lengths of queues). These prediction results can be hardly mapped back to the software architecture by humans, making it hard to derive the right design decisions. In this paper, we integrate software cartography (a map technique) with software performance engineering to overcome the limited interpretability of raw performance prediction results. Our approach is based on model transformations and a general software visualization approach. It provides an intuitive mapping of prediction results to the software architecture which simplifies design decisions. We successfully evaluated our approach in a quasi experiment involving 41 participants by comparing the correctness of performance-improving design decisions and participants' time effort using our novel approach to an existing software performance visualization. © 2009 Springer Berlin Heidelberg. (33 refs.)

Main Heading: Software architecture

Controlled terms: Architecture - Computer software selection and evaluation - Design - Mapping - Maps - Visualization

Uncontrolled terms: Architectural levels - Design alternatives - Design decisions - Design time - Interpretability - Model transformation - Performance prediction - Quasi-experiments - Resource utilizations - Response time - Software performance engineerings - Software performance visualization - Software systems - Software visualization

Classification Code: 902.1 Engineering Graphics - 723.5 Computer Applications - 723.1 Computer Programming - 912.2 Management - 723 Computer Software, Data Handling and Applications - 405.3 Surveying - 402 Buildings and Towers - 408 Structural Design

Database: Compendex

7. **SArF map: Visualizing software architecture from feature and layer viewpoints**

Kobayashi, Kenichi[1]; Kamimura, Manabu[1]; Yano, Keisuke[1]; Kato, Koki[1]; Matsuo, Akihiko[1] **Source:** *IEEE International Conference on Program Comprehension*, p 43-52, 2013, *2013 21st International Conference on Program Comprehension, ICPC 2013 - Proceedings*; **ISBN-13:** 9781467330923; **DOI:** 10.1109/ICPC.2013.6613832; **Article number:** 6613832; **Conference:** 2013 21st International Conference on Program Comprehension, ICPC 2013, May 20, 2013 - May 21, 2013; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Software Systems Laboratories, Fujitsu Laboratories Ltd., Kawasaki, Japan

Abstract: To facilitate understanding the architecture of a software system, we developed SArF Map technique that visualizes software architecture from feature and layer viewpoints using a city metaphor. SArF Map visualizes implicit software features using our previous study, SArF dependency-based software clustering algorithm. Since features are high-level abstraction units of software, a generated map can be directly used for high-level decision making such as reuse and also for communications between developers and non-developer stakeholders. In SArF Map, each feature is visualized as a city block, and classes in the feature are laid out as buildings reflecting their software layer. Relevance between features is represented as streets. Dependency links are visualized lucidly. Through open source and industrial case studies, we show that the architecture of the target systems can be easily overviewed and that the quality of their packaging designs can be quickly assessed. © 2013 IEEE. (25 refs.)

Main Heading: Open systems

Controlled terms: Clustering algorithms - Computer programming - Computer software reusability - Software architecture

Uncontrolled terms: city metaphor - Dependency graphs - Program comprehension - Software clustering - Software visualization

Classification Code: 721 Computer Circuits and Logic Elements - 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming

Database: Compendex

8. **Interactive visualization of quality variability at run-time**
Kuusijärvi, Jarkko **Source:** *VTT Publications*, n 746, p 1-111, 2010, *Interactive visualization of quality variability at run-time*; **ISSN:** 12350621, **E-ISSN:** 14550849; **ISBN-13:** 9789513874124; **Publisher:** Technical Research Center of Finland

Abstract: Smart environments are dynamic in nature, and the software running in these environments requires quality adaptations in order to function efficiently. The result of these adaptations, i.e., quality variability, must be verified in some way, and visualization can be used to aid this verification process. The research problem in this work was to find suitable visualization techniques to visualize quality variability and implement a visualization tool that encompasses these techniques and provides an interactive visualization of quality variability for the user. As a solution to the research problem, this work presents an interactive quality visualization tool. The requirements specification for the implemented tool was derived from the literature review and the intended usage context of the tool, i.e., smart environments. The literature review explores a set of applicable visualization techniques and compares existing visualization tools with regard to the features required to represent quality variability visually at run-time. The visualization techniques selected for the tool include interactive timelines,

charts and meters that enable analysis of the quality attributes and their variability in different time ranges or points in time. Some additional visualization techniques were also included such as treemaps and graphs to visualize the structure of the smart environment. The visualization techniques include open source visualization techniques and self-made techniques designed and implemented from the start to cover the specific requirements set for the tool. The main contribution of this work is the visualization tool that can be used to visualize different quality attributes and their variability. Moreover, the tool can easily be deployed in different environments due to its architecture and the selected implementation technologies that make the solution extensible and portable. The implemented visualization tool was evaluated in the context of a smart environment in which security was adapted at run-time. The case study demonstrated that the implemented tool can be used in the analysis of the variability of different quality attributes. The trend of a single quality attribute can be stud-ied for different time ranges or points in time according to need. The relation-ships between different quality attributes can also be studied with the help of appropriate visualization techniques. In addition, the visualization tool was successfully tested on mobile devices. (73 refs.)

Main Heading: Computer software selection and evaluation

Controlled terms: Equipment - Mobile devices - Quality control - Research - Ships - Visualization

Uncontrolled terms: Adaptive software - Quality attributes - Security - Smart environment - Software visualization

Classification Code: 913.3 Quality Assurance and Control - 902.1 Engineering Graphics - 901.3 Engineering Research - 901 Engineering Profession - 723 Computer Software, Data Handling and Applications - 716 Telecommunication; Radar, Radio and Television - 674 Small Craft and Other Marine Craft - 672 Naval Vessels - 671 Naval Architecture

Database: Compendex

9. **Extracting and facilitating architecture in service-oriented software systems**

Weinreich, Rainer[1]; Miesbauer, Cornelia[1]; Buchgeher, Georg[2]; Kriechbaum, Thomas[3] **Source:** *Proceedings of the 2012 Joint Working Conference on Software Architecture and 6th European Conference on Software Architecture, WICSA/ECSA 2012*, p 81-90, 2012, *Proceedings of the 2012 Joint Working Conference on Software Architecture and 6th European Conference on Software Architecture, WICSA/ECSA 2012*; **ISBN-13:** 9780769548272; **DOI:** 10.1109/WICSA-ECSA.212.16; **Article number:** 6337764; **Conference:** Joint 10th Working IEEE/IFIP Conference on Software Architecture, WICSA 2012 and 6th European Conference on Software, ECSA 2012, August 20, 2012 - August 24, 2012; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Johannes Kepler University Linz, Linz, Austria
[2] Software Competence Center Hagenberg, Hagenberg, Austria
[3] GRZ IT Center Linz GmbH, Linz, Austria

Abstract: In enterprises using service-oriented architecture (SOA) architectural information is used for various activities including analysis, design, governance, and quality assurance. Architectural information is created, stored and maintained in various locations like enterprise architecture management tools, design tools, text documents, and service registries/repositories. Capturing and maintaining this information manually is time-intensive, expensive and error-prone. To address this problem we present an approach for automatically extracting architectural information from an actual SOA implementation. The extracted information represents the currently implemented architecture and can be used as the basis for quality assurance tasks and, through synchronization, for keeping architectural information consistent in various other tools and locations. The presented approach has been developed for a SOA in the banking domain. Aside from presenting the main drivers for the approach and the approach itself, we report on experiences in applying the approach to different applications in this domain. © 2012 IEEE. (16 refs.)

Main Heading: Software architecture

Controlled terms: Information services - Quality assurance - Service oriented architecture (SOA)

Uncontrolled terms: Architectural views - Architecture visualization - Design tool - Enterprise architecture managements - Error prones - Service Oriented - Service registry - Software systems - Text document

Classification Code: 722.4 Digital Computers and Systems - 723.1 Computer Programming - 903.4 Information Services - 913.3 Quality Assurance and Control

Database: Compendex

10. **Software evolution: Analysis and visualization**

Gall, Harald C.[1]; Lanza, Michele[2] **Source:** *Proceedings - International Conference on Software Engineering*, v 2006, p 1055-1056, 2006, *Proceeding of the 28th International Conference on Software Engineering 2006, ICSE '06*; **ISSN:** 02705257; **ISBN-10:** 1595933751, **ISBN-13:** 9781595933751; **Conference:** 28th International Conference on Software Engineering 2006, ICSE '06, May 20, 2006 - May 28, 2006; **Sponsor:** ACM Special Interest Group on Software Engineering, SIGSOFT; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] Department of Informatics, University of Zurich

[2] Faculty of Informatics, University of Lugano, Switzerland

Abstract: Gaining higher level evolutionary information about large software systems is a key challenge in dealing with increasing complexity and decreasing software quality. Software repositories such as modifications, changes, or release information are rich sources for distinctive kinds of analyses: They reflect the reasons and effects of particular changes made to the software system over a certain period of time. If we can analyze these repositories in an effective way, we get a clearer picture of the status of the software. Software repositories can be analyzed to provide information about the problems concerning a particular feature or a set of features. Hidden dependencies of structurally unrelated but over time logically coupled files exhibit a high potential to illustrate software evolution and possible architectural deterioration. In this tutorial, we describe the investigation of software evolution by taking a step towards reflecting the analysis results against software quality attributes. Different kinds of analyses (from architecture to code) and their interpretation will be presented and discussed in relation to quality attributes. This will show our vision of where such evolution investigations can lead and how they can support development. For that, the tutorial will touch issues such as meta-models for evolution data, data analysis and history mining, software quality attributes, as well as visualization of analysis results. (14 refs.)

Main Heading: Software engineering

Controlled terms: Computational complexity - Data mining - Data reduction - Quality of service

Uncontrolled terms: History mining - Software evolution - Software quality

Classification Code: 716 Telecommunication; Radar, Radio and Television - 721.1 Computer Theory, Includes Formal Logic, Automata Theory, Switching Theory, Programming Theory - 723.1 Computer Programming - 723.2 Data Processing and Image Processing

Treatment: Theoretical (THR)

Database: Compendex

11. **SEA/I: INTEGRATED SOFTWARE DEVELOPMENT SUPPORT SYSTEM DESIGNED FOR IMPROVEMENT OF SOFTWARE QUALITY AND PRODUCTIVITY.**

Miyahara, Yuko[1]; Takahashi, Masaaki[1] **Source:** *NEC Research and Development*, n 87, p 63-75, Oct 1987; **ISSN:** 0547051X

Author affiliation:
[1] NEC, Jpn, NEC, Jpn

Abstract: NEC has developed an integrated software development support system, which consistently supports development and maintenance of business application software throughout its life cycle. This system is based on the SEA/I (Software Engineering Architecture One) and called the SEA/I system. It has been implemented as a system on NEC supplied operating system ACOS. Various kinds of methodologies and tools have been proposed to enhance software productivity. Among them, the SEA/I system has following features. First, its main ideas are 'standardization', 'componentization', and 're-use and co-use of components'. In addition, components are stored and 'unitarily managed' in SEA/I database. These contribute much to business application program development, because mass production of similar programs is required in such development. The SEA/I system contains many support tools, and by using such tools, software development processes such as design, manufacturing, test and maintenance can be visualized and automatized. Since its first release in 1984, it has been shipped to more than 1000 ACOS customers and applied to more than 330 projects. This figure shows that the SEA/I system is already being put into practical use. (7 refs.)

Main Heading: COMPUTER SOFTWARE

Controlled terms: COMPUTER AIDED ENGINEERING - COMPUTER PROGRAMMING - STANDARDIZATION

Uncontrolled terms: SEA/I (SOFTWARE ENGINEERING ARCHITECTURE ONE) - SOFTWARE AUTOMATION - SOFTWARE PRODUCTIVITY - SOFTWARE QUALITY

Classification Code: 723 Computer Software, Data Handling and Applications - 902 Engineering Graphics; Engineering Standards; Patents

Treatment: General review (GEN)

Database: Compendex

12. **An Architecture for Java-Based Real-Time Distributed Visualization**

Mahovsky, Jeffrey[1]; Benedicenti, Luigi[2] **Source:** *IEEE Transactions on Visualization and Computer Graphics*, v 9, n 4, p 570-579, October/December 2003; **ISSN:** 10772626; **DOI:** 10.1109/TVCG.2003.1260749; **Publisher:** Institute of Electrical and Electronics Engineers Computer Society

Author affiliation:
1 Department of Computer Science, University of Calgary, 2500 University Drive, N.W., Calgary, Alta. T2N 1N4, Canada
2 Faculty of Engineering, University of Regina, 3737 Wascana Parkway, Regina, Sask. S4S 0A2, Canada

Abstract: In this paper, we present a Java-based software architecture for real-time visualization that utilizes a cluster of conventional PCs to generate high-quality interactive graphics. Normally, a large multiprocessor computer

would be needed for interactive visualization tasks requiring more processing power than a single PC can provide. By using clusters of PCs, enormous cost savings can be realized, and proprietary "high-end" hardware is no longer necessary for these tasks. Our architecture minimizes the amount of synchronization needed between PCs, resulting in excellent scalability. It provides a modular framework that can accommodate a wide variety of rendering algorithms and data formats, provided that the rendering algorithms can generate pixels individually and the data is duplicated on each PC. Demonstration modules that implement ray tracing, fractal rendering, and volume rendering algorithms were developed to evaluate the architecture. Results are encouraging - using 15 PCs connected to a standard 100 Megabit's Ethernet network, the system can interactively render simple to moderately complex data sets at modest resolution. Excellent scalability is achieved; however, our tests were limited to a cluster of 15 PCs. Results also demonstrate that Java is a viable platform for real-time distributed visualization. (25 refs.)

Main Heading: Computer graphics

Controlled terms: Algorithms - Computer architecture - Computer hardware - Computer software - Java programming language - Multiprocessing systems - Personal computers - Real time systems - Synchronization

Uncontrolled terms: Distributed visualization - Pixels - Volume rendering

Classification Code: 722 Computer Systems and Equipment - 722.4 Digital Computers and Systems - 723.1.1 Computer Programming Languages - 723.5 Computer Applications - 731.1 Control Systems - 921 Mathematics

Treatment: Theoretical (THR)

Database: Compendex

13. **An extensible framework for improving a distributed software system's deployment architecture**

Malek, Sam[1]; Medvidovic, Nenad[2]; Mikic-Rakic, Marija[3] **Source:** *IEEE Transactions on Software Engineering*, v 38, n 1, p 73-100, 2012; **ISSN:** 00985589; **DOI:** 10.1109/TSE.2011.3; **Article number:** 5680912; **Publisher:** Institute of Electrical and Electronics Engineers Inc.

Author affiliation:
1 Department of Computer Science, George Mason University, Fairfax, VA 22030, United States
2 Computer Science Department, University of Southern California, Los Angeles, CA 90089, United States
3 Google, Inc., 604 Arizona Ave., Santa Monica, CA 90401, United States

Abstract: A distributed system's allocation of software components to

hardware nodes (i.e., deployment architecture) can have a significant impact on its quality of service (QoS). For a given system, there may be many deployment architectures that provide the same functionality, but with different levels of QoS. The parameters that influence the quality of a system's deployment architecture are often not known before the system's initial deployment and may change at runtime. This means that redeployment of the software system may be necessary to improve the system's QoS properties. This paper presents and evaluates a framework aimed at finding the most appropriate deployment architecture for a distributed software system with respect to multiple, possibly conflicting QoS dimensions. The framework supports formal modeling of the problem and provides a set of tailorable algorithms for improving a system's deployment. We have realized the framework on top of a visual deployment architecture modeling and analysis environment. The framework has been evaluated for precision and execution-time complexity on a large number of simulated distributed system scenarios, as well as in the context of two third-party families of distributed applications. © 2006 IEEE. (66 refs.)

Main Heading: Software architecture

Controlled terms: Computer software - Quality of service

Uncontrolled terms: Deployment architecture - Distributed applications - Distributed software system - Distributed systems - Extensible framework - Formal modeling - QoS properties - Runtimes - Self-adaptive software - Significant impacts - Software component - software deployment - Software systems

Classification Code: 723 Computer Software, Data Handling and Applications

Database: Compendex

14. **Scalable software architecture for high performance video codec's on parallel processing engines**
Rapaka, Krishnakanth; Mody, Mihir; Prasad, Keshava **Source:** *Proceedings of the International Symposium on Consumer Electronics, ISCE*, 2007, *2007 IEEE International Symposium on Consumer Electronics, ISCE*; **ISBN-10:** 1424411092, **ISBN-13:** 9781424411092; **DOI:** 10.1109/ISCE.2007.4382148; **Article number:** 4382148; **Conference:** 2007 IEEE International Symposium on Consumer Electronics, ISCE, June 20, 2007 - June 23, 2007; **Publisher:** Institute of Electrical and Electronics Engineers Inc.

Abstract: Video algorithm (e.g. H.264, MPEG2/4 etc) requires tremendous amount of computation power and data bandwidth. This complexity depends on encoding vs. decoding mode, video standard, resolution, frame-rate and visual quality constraints. Many video architecture solutions typically use multiple processing elements (e.g. multiple DSPs or MCU, DSP/MCU with dedicated accelerators or FPGA etc) to achieve the high computation requirements for video algorithms. These architectures provide new

challenges to video software's that are typically designed to run on a single processor. This paper presents software design for a video architecture using parallel processing elements. This paper explains following aspects in detail a) Software partitioning b) Algorithm specific optimizations c) Processor specific optimizations d) Efficient DMA/Cache usage e) Concurrent scheduling of all parallel processing elements. The given approach is explained with example of MPEG4 encoder on TMS320DM6446, which is Davinci™ family device from Texas Instruments Ltd. The given software architecture is scalable for various video standards (e.g. H.264, MPEG2/4 etc) as well as various parallel processing hardware solutions. The software achieves performance D1@30fsp on given device at less than 50% of DSP load. (6 refs.)

Main Heading: Software architecture

Controlled terms: Architectural design - Architecture - Chlorine compounds - Codes (symbols) - Concurrency control - Consumer electronics - Data handling - Decoding - Electronics industry - Hardware - Image coding - Optimization - Parallel algorithms - Software design - Standards - Technical presentations

Uncontrolled terms: Computation power - Computation requirements - Data bandwidths - Frame rates - H.264 - International symposium - MPEG4 - Multiple processing - Parallel processing - Parallel processing elements - Single processors - Software partitioning - Texas instruments - Video algorithms - Video architecture - Video codecs - Video coding - Video software - Video standard - Visual qualities

Classification Code: 921.5 Optimization Techniques - 902.2 Codes and Standards - 901.2 Education - 804.1 Organic Compounds - 741 Light, Optics and Optical Devices - 723.3 Database Systems - 723.2 Data Processing and Image Processing - 723.1 Computer Programming - 723 Computer Software, Data Handling and Applications - 715 Electronic Equipment, General Purpose and Industrial - 712 Electronic and Thermionic Materials - 605 Small Tools and Hardware - 402 Buildings and Towers

Database: Compendex

15. **Ontology-driven visualization of architectural design decisions**

De Boer, Remco C.[1]; Lago, Patricia[1]; Telea, Alexandru[2]; Van Vliet, Hans[1]

Source: *2009 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, WICSA/ECSA 2009*, p 51-60, 2009, *2009 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, WICSA/ECSA 2009*; **ISBN-13:** 9781424449859; **DOI:** 10.1109/WICSA.2009.5290791; **Article number:** 5290791; **Conference:** 2009 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, WICSA/ECSA 2009, September 14, 2009 - September 17, 2009; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Department of Computer Science, VU University Amsterdam, Netherlands
[2] Institute for Mathematics and Computer Science, University of Groningen, Netherlands

Abstract: There is a gradual increase of interest to use ontologies to capture architectural knowledge, in particular architectural design decisions. While ontologies seem a viable approach to codification, the application of such codified knowledge to everyday practice may be non-trivial. In particular, browsing and searching an architectural knowledge repository for effective reuse can be cumbersome. In this paper, we present how ontology-driven visualization of architectural design decisions can be used to assist software product audits, in which independent auditors perform an assessment of a product's quality. Our visualization combines the simplicity of tabular information representation with the power of on-the-fly ontological inference of decision attributes typically used by auditors. In this way, we are able to support the auditors in effectively reusing their know-how, and to actively assist the core aspects of their decision making process, namely trade-off analysis, impact analysis, and if-then scenarios. We demonstrate our visualization with examples from a real-world application. © 2009 IEEE. (18 refs.)

Main Heading: Software architecture

Controlled terms: Architectural design - Computer software - Decision making - Ontology - Structural design - Technology transfer - Visualization

Uncontrolled terms: Architectural knowledge - Decision attribute - Decision making process - Impact analysis - Information representation - Know-how - Non-trivial - On-the-fly - Real-world application - Software products - Trade-off analysis

Classification Code: 911.2 Industrial Economics - 903 Information Science - 902.1 Engineering Graphics - 901.4 Impact of Technology on Society - 912.2 Management - 723.5 Computer Applications - 723 Computer Software, Data Handling and Applications - 408.1 Structural Design, General - 402 Buildings and Towers - 723.1 Computer Programming

Database: Compendex

16. **PadSpace: A software architecture for the ad hoc federation of distributed visual components and web resources**

Lkhamsuren, D.[1]; Tanaka, Y.[1] **Source:** *CyberC 2009 - International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, p 113-120, 2009, *CyberC 2009 - International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*; **ISBN-13:** 9781424452187; **DOI:** 10.1109/CYBERC.2009.5342180; **Article number:** 5342180; **Conference:** 2009 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC '09, October 10,

2009 - October 11, 2009; **Sponsor:** IEEE Computer Society; IEEE CS Task Force on Networked Mobile Systems; IEEE Technical Committee on Simulation; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Meme Media Laboratory, Hokkaido University, North 13 West 8, Kitaku, Sapporo, Japan

Abstract: In this paper we will propose an extension of a typical Linda-like coordination model (tuplespace) to provide mechanisms for the cooperation of Web applications, Web services and end users' local resources. This new model assumes that all these different types of resources are wrapped into visual components called pads before their cooperation. Pads are defined based on the meme media architecture. This new Linda-like coordination model is called a PadSpace. A PadSpace uses an XML-tuplespace as its core coordination model. The XML-tuplespace is a distributed coordination model that extends the Linda coordination model with the ability to carry XML documents in tuple fields. We call each tuple an XML-tuple. A PadSpace uses the 2D meme media system IntelligentPad as its platform. Linda represents all the available services as tuples and stores them in its tuplespace. It also represents each service request as a tuple, and matches this requesting tuple with a compatible registered tuple representing a service. Similarly, a PadSpace represents Web applications, Web services, and end users' local resources as XML-tuples, and stores these XML-tuples in its core XML-tuplespace. In IntelligentPad, we represent every compound object as a pad. Therefore, we want to represent a service request also as a pad even before it is matched with a specific service, so that we can define a composite application pad using this service request pad without instantiating it to one candidate of the requested service. This service request pad, when matched with some service, works as a proxy of this service. A PadSpace represents such a service request pad as a requesting XML-tuple, and matches this XML-tuple with an appropriate service XML-tuple stored in its core XML-tuplespace. It establishes a connection between this service request pad and the matched service, and makes the service request pad to work as the proxy pad of this service. Using these mechanisms, a PadSpace provides an end user tool for the service composition of Web resources and local resources. This tool enables end users to register Web resources and local application resources into the core XML-tuplespace and to use those shared resources in combination with their own local resources by defining service compositions in the IntelligentPad architecture without writing any program codes. © 2009 IEEE. (15 refs.)

Main Heading: XML

Controlled terms: Computer science - Markup languages - Quality of service - Software architecture - Visual communication - Web services

Uncontrolled terms: Coordination model - Distributed Computing - Distributed coordination - End users - IntelligentPad - Linda coordination model - Media systems - Meme media architecture - New model - Program code - Service compositions - Service requests - Shared resources - Visual

components - WEB application - Web resources

Classification Code: 903.4 Information Services - 723.5 Computer Applications - 723.1 Computer Programming - 723 Computer Software, Data Handling and Applications - 722 Computer Systems and Equipment - 721 Computer Circuits and Logic Elements - 718 Telephone Systems and Related Technologies; Line Communications - 717.1 Optical Communication Systems - 717 Optical Communication - 716.4 Television Systems and Equipment - 716 Telecommunication; Radar, Radio and Television

Database: Compendex

17. **A demo on using visualization to aid run-time verification of dynamic service systems**

Kuusijärvi, Jarkko[1] **Source:** *ICSTW 2010 - 3rd International Conference on Software Testing, Verification, and Validation Workshops*, p 319-324, 2010, *ICSTW 2010 - 3rd International Conference on Software Testing, Verification, and Validation Workshops*; **ISBN-13:** 9780769540504; **DOI:** 10.1109/ICSTW.2010.40; **Article number:** 5463665; **Conference:** 3rd International Conference on Software Testing, Verification, and Validation Workshops, ICSTW 2010, April 6, 2010 - April 10, 2010; **Publisher:** IEEE Computer Society

Author affiliation:
[1] VTT Technical Research Centre of Finland, Kaitoväylä 1, 90571 Oulu, Finland

Abstract: Future software systems will be dynamic service oriented systems. Service-Oriented Architecture (SOA) provides an extensible and dynamic architecture to be used, for example, in smart environments. In such an environment, software has to adapt its behaviour dynamically. Thus, there is a need for Verifying and Validating (V&V) the adaptations at run-time. This paper contributes to that by introducing a novel visualization tool to be used with traditional V&V techniques to aid the software analysts in the verification process of dynamic software systems. When Quality of Service (QoS) of dynamic software systems varies due to the changing environment the Interactive Quality Visualization (IQVis) tool detects these changes and provides analysts an easier way of understanding the changed behaviour of the system. © 2010 IEEE. (23 refs.)

Main Heading: Computer software selection and evaluation

Controlled terms: Adaptive systems - Embedded systems - Information services - Quality of service - Software testing - Verification - Visualization

Uncontrolled terms: Changing environment - Dynamic architecture - Dynamic services - Dynamic softwares - Quality attributes - Run-time verification - Runtimes - Self-adaptive system - Smart environment - Software systems - Verification process - Visualization tools

Classification Code: 912.2 Management - 723.1 Computer Programming - 723.4 Artificial Intelligence - 723.5 Computer Applications - 731.1 Control Systems - 902.1 Engineering Graphics - 903.4 Information Services - 723 Computer Software, Data Handling and Applications - 721.2 Logic Elements - 721.1 Computer Theory, Includes Formal Logic, Automata Theory, Switching Theory, Programming Theory - 718 Telephone Systems and Related Technologies; Line Communications - 717 Optical Communication - 716 Telecommunication; Radar, Radio and Television - 722 Computer Systems and Equipment

Database: Compendex

18. **Architecture for monitoring, visualization, and control of gigabit networks**

Parulkar, Guru[1]; Schmidt, Douglas[1]; Kraemer, Eileen[1]; Turner, Jonathan[1]; Kantawala, Anshul[1] **Source:** *IEEE Network*, v 11, n 5, p 34-43, Sep-Oct 1997; **ISSN:** 08908044; **DOI:** 10.1109/65.620520; **Publisher:** IEEE

Author affiliation:
[1] Washington Univ, St. Louis, United States

Abstract: In this article we outline a design of a highly scalable network monitoring, visualization, and control (NMVC) system with advanced algorithmic and human-in-the-loop capability. This capability allows network administrators to calibrate and fine-tune network and application parameters in real time according to observed traffic patterns. The goal of the NMVC system is to ensure adequate quality of service to network users, while maintaining high network resource utilization. The main components of our system are: a network probe and an endsystem probe which can probe gigabit/s links, software network management agents that provide extensible multi-attribute event filtering for highly scalable data/event collection, efficient online event ordering algorithms that can help synthesize and display a consistent view of network health, status, and performance and a View Choreographer that allows management applications and administrators to specify the mapping of network events to higher-level events and to visualization objects and updates. (30 refs.)

Main Heading: Computer networks

Controlled terms: Algorithms - Computer software - Congestion control (communication) - Monitoring - Online systems - Real time systems - Telecommunication links - Telecommunication services - Telecommunication traffic - Visualization

Uncontrolled terms: Endsystem probe - Network probe - Software network management agents - View choreographer

Classification Code: 716.1 Information Theory and Signal Processing - 722.4 Digital Computers and Systems - 723 Computer Software, Data

Handling and Applications

Treatment: Theoretical (THR)

Database: Compendex

19. **Modeling approach for dynamic software architecture**

Yu, Zhenhua[1]; Cai, Yuanli[1]; Xu, Haiping[2] **Source:** *Hsi-An Chiao Tung Ta Hsueh/Journal of Xi'an Jiaotong University*, v 41, n 2, p 167-171, February 2007; **Language:** Chinese; **ISSN:** 0253987X; **Publisher:** Xi'an Jiaotong University

Author affiliation:
[1] School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[2] Department of Computer and Information Science, University of Massachusetts Dartmouth, North Dartmouth 02747, United States

Abstract: Aiming at the defects of existing methods for modeling of dynamic software architecture, a novel software architecture abstract model (SAAM) is presented, in which two complementary formalisms, namely object-oriented Petri nets (OPN) and π-calculus, are adopted as formal theory bases. The OPN are employed to visualize initial architecture as well as system behaviors; while π-calculus is used to describe software architecture evolutions. These two formal methods are integrated in the SAAM, and the SAAM can be analyzed and verified by the corresponding supporting tools. Furthermore, the evolving strategy of components and the consistency among components can also be analyzed using π-calculus so as to detect the design errors in early software design stage and significantly improve the quality of software. A classical gas station example is used to show that SAAM is intuitional and effectively describes dynamic software architecture. (10 refs.)

Main Heading: Software design

Controlled terms: Computer simulation - Formal methods - Object oriented programming - Petri nets - Software architecture

Uncontrolled terms: Modeling - Object oriented Petri net (OPN) - Software architecture abstract model (SAAM)

Classification Code: 723.5 Computer Applications - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory

Treatment: Experimental (EXP)

Database: Compendex

20. **Software product line architecture modeling and component composition implementation with extension of aspectual mechanism**

Shen, Li-Wei[1]; Peng, Xin[1]; Zhao, Wen-Yun[1] **Source:** *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, v 37, n SUPPL., p 140-145, April 2009; **Language:** Chinese; **ISSN:** 03722112; **Publisher:** Chinese Institute of Electronics

Author affiliation:
[1] School of Computer Science, Fudan University, Shanghai 200433, China

Abstract: Software product line (SPL) can increase the efficiency and quality of software development. Software architecture (SA), as the blueprint of SPL, defines the inter-relationships between components and guides the component composition implementation. However, the existing interface connection architecture is limited to describe the direct interactions between components. It cannot support the more complex interaction situations which emerge with the SPL variability. In this paper, we propose a method of software product line architecture modeling and component composition implementation with extension of aspectual mechanism. The core is an architecture description language (ADL) which extends xADL2.0 and combines with aspect-oriented techniques. The ADL supports the design and customization for SPL architecture based on variability, and instructs the component composition process for applications. Furthermore, we have developed a prototype tool FdSPLC which provides the visual modeling of architecture as well as the automatic application derivation. (11 refs.)

Main Heading: Software architecture

Controlled terms: Architectural design - Large scale systems

Uncontrolled terms: Architecture description languages - Aspect-oriented - Automatic applications - Complex interactions - Component composition - Component interaction style - Direct interactions - Inter-relationships - Prototype tools - Quality of softwares - Software product line architectures - Software product line development - Visual modeling

Classification Code: 912.3 Operations Research - 731.1 Control Systems - 723.5 Computer Applications - 961 Systems Science - 723.1 Computer Programming - 408.1 Structural Design, General - 402 Buildings and Towers - 461.1 Biomedical Engineering

Database: Compendex

21. **On quick comprehension and assessment of software**

Bartoszuk, Cezary[1]; Dabrowski, Robert[1]; Stencel, Krzysztof[1]; Timoszuk, Grzegorz[1] **Source:** *ACM International Conference Proceeding Series*, v 767, p 161-168, 2013, *Computer Systems and Technologies: 14th International Conference, CompSysTech 2013 - Proceedings*; **ISBN-13:** 9781450320214; **DOI:** 10.1145/2516775.2516806; **Conference:** 14th International Conference on Computer Systems and Technologies, CompSysTech 2013, June 28, 2013 - June 29, 2013; **Sponsor:** Technical University of Varna, Bulgaria (TECHUVB);

Federation of the Scientific Eng. Unions - Bulgaria (FOSEUB); Bulgarian Ministry of Education, Youth and Science (MEYS); CASTUVTB; University of Ruse, Bulgaria (UORB); **Publisher:** Association for Computing Machinery

Author affiliation:
[1] Institute of Informatics, University of Warsaw, ul. Banacha 2, 02-097 Warsaw, Poland

Abstract: By an architecture of a software system we mean the fundamental organization of the system embodied in its components, their relationships to one another and to the system's environment. It also encompasses principles governing the system's design and evolution. Architectures of complex systems are obviously complex as well. The goal of our research is to harness this complexity. In this paper we focus on providing software architects with ability to quickly comprehend the complexity and assess the quality of software. The essential tools we use are: (1) a graph-based repository for collecting information on software artefacts, accompanied by (2) tools to perform software intelligence tasks, like analyzing dependencies among those artefacts, calculating their importance, and quality. On top of those tools we implement visualization methods that render the relative importance using size and the quality using colours. By means of such methods a software architect can at glance comprehend and assess the software, He/she can (1) find the starting points to dig into a complex system; (2) judge the cohesion and coupling of system components; and (3) assess the overall quality. We demonstrate this method using selected open-source projects of various sizes and qualities. © 2013 ACM. (26 refs.)

Main Heading: Tools

Controlled terms: Architecture - Computer software - Image quality - Large scale systems - Software architecture

Uncontrolled terms: Cohesion and couplings - graph - intelligence - metrics - Open source projects - Quality of softwares - Software intelligences - Visualization method

Classification Code: 402 Buildings and Towers - 603 Machine Tools - 605 Small Tools and Hardware - 723 Computer Software, Data Handling and Applications - 741 Light, Optics and Optical Devices - 961 Systems Science

Database: Compendex

22. **Tool support for continuous quality control**

Deissenboeck, Florian[1]; Juergens, Elmar[1]; Hummel, Benjamin[1]; Wagner, Stefan[1]; Parareda, Benedikt Mas[1]; Pizka, Markus[2] **Source:** *IEEE Software*, v 25, n 5, p 60-67, 2008; **ISSN:** 07407459; **DOI:** 10.1109/MS.2008.129; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] Software Systems Engineering Group, Technische Universität München, München, Germany
[2] Itestra GmbH, Technische Universität München, München, Germany

Abstract: Over time, software systems suffer gradual quality decay and therefore costs can rise if organizations fail to take proactive countermeasures. Quality control is the first step to avoiding this cost trap. Continuous quality assessments help users identify quality problems early, when their removal is still inexpensive; they also aid decision making by providing an integrated view of a software system's current status. As a side effect, continuous and timely feedback helps developers and maintenance personnel improve their skills and thereby decreases the likelihood of future quality defects. To make regular quality control feasible, it must be highly automated, and assessment results must be presented in an aggregated manner to avoid overwhelming users with data. This article offers an overview of tools that aim to address these issues. The authors also discuss their own flexible, open-source toolkit, which supports the creation of dashboards for quality control. © 2008 IEEE. (6 refs.)

Main Heading: Quality control

Controlled terms: Computer networks - Computer software - Computer software maintenance - Customer satisfaction - Decision making - Industrial engineering - Problem solving - Quality assurance - Quality function deployment - Total quality management

Uncontrolled terms: Computer architecture - Data visualization - Monitoring - Program processors - Project control and modeling - Project management - Sensors - Software - Software engineering - Software quality

Classification Code: 921 Mathematics - 913.3 Quality Assurance and Control - 912.2 Management - 912.1 Industrial Engineering - 912 Industrial Engineering and Management - 922.2 Mathematical Statistics - 723.4 Artificial Intelligence - 722 Computer Systems and Equipment - 718 Telephone Systems and Related Technologies; Line Communications - 717 Optical Communication - 716 Telecommunication; Radar, Radio and Television - 723 Computer Software, Data Handling and Applications

Database: Compendex

23. **Towards quantitative evaluation of UML based software architecture**

Li, Jinhua[1]; Guo, Zhenbo[1]; Zhao, Yun[1]; Zhang, Zhenhua[1]; Pang, Ruijuan[1]
Source: *Proceedings - SNPD 2007: Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, v 1, p 663-669, 2007, *Proceedings - SNPD 2007: Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*;

ISBN-10: 0769529097, **ISBN-13:** 9780769529097; **DOI:** 10.1109/SNPD.2007.551; **Article number:** 4287589; **Conference:** SNPD 2007: 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, July 30, 2007 - August 1, 2007; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] College of Information Engineering, Qingdao University

Abstract: The architecture of a software system is a critical artifact in the software lifecycle and should be evaluated as early as possible. Recent efforts to software architecture evaluation are concentrated on scenario-based methods which are qualitative, subjective and need not any special architecture description languages. This paper investigates an approach to metrics based quantitative evaluation of UML software architecture. UML is a visual modeling language with well-formed hierarchical syntax and semantics, and is uniformly applied in various development stages. With supplementation UML has been adapted to describing software architecture. By utilization of these features three types of metrics for UML diagrams are proposed They measure the amount of information, visual effect and connectivity degree in different UML diagrams. The application of these metrics in quantitative evaluating qualities at the architecture-level such as system scale, complexity and structural characteristics is discussed. © 2007 IEEE. (17 refs.)

Main Heading: Unified Modeling Language

Controlled terms: Computational complexity - Feature extraction - Hierarchical systems - Software architecture - Syntactics

Uncontrolled terms: Architecture description languages - Hierarchical syntax - Software lifecycle - System scales

Classification Code: 721.1 Computer Theory, Includes Formal Logic, Automata Theory, Switching Theory, Programming Theory - 723.1.1 Computer Programming Languages - 723.5 Computer Applications - 903.2 Information Dissemination - 961 Systems Science

Treatment: Theoretical (THR)

Database: Compendex

24. **An architecture of geospatial sensor network for water quality monitoring**

He, Jin-Xin[1, 2]; Li, Jonathan[2]; Yan, Hao-Wen[2, 3] **Source:** *Jilin Daxue Xuebao (Diqiu Kexue Ban)/Journal of Jilin University (Earth Science Edition)*, v 41, n 4, p 1262-1266, July 2011; **Language:** Chinese; **ISSN:** 16715888; **Publisher:** Jilin University Press

Author affiliation:

1 College of Earth Sciences, Jilin University, Changchun 130061, China
2 Faculty of Environment, University of Waterloo, Waterloo N2L3G1, Canada
3 School of Mathematics, Physics and Software Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China

Abstract: A geospatial sensor network combines wireless sensor network with GIS, GPS and satellite remote sensing efficiently, but at present, it has no uniform architecture. So an architecture of geospatial sensor network oriented to water quality monitoring was proposed, and the design and implementation of data management subsystem is the main content. The software design of data management subsystem was based on service oriented architecture (SOA), since the monitored data is distributed, multiple sourced, and heterogeneous. Rich internet application (RIA) was the main developing mode in the software, so it has auto update, loading balance, and some other advantages. Compared with the traditional water quality monitoring approaches by the application in Ontario, Canada, this architecture can monitor more data, and visualize them more directly. (20 refs.)

Main Heading: Network architecture

Controlled terms: Information management - Information services - Internet - Remote sensing - Service oriented architecture (SOA) - Software design - Telecommunication networks - Water content - Water quality - Wireless networks

Uncontrolled terms: Auto-update - Geo-spatial - Geospatial sensor network - Network oriented - Ontario , Canada - Rich Internet Applications - Satellite remote sensing - Service Oriented - Water quality monitoring

Classification Code: 903.4 Information Services - 903.2 Information Dissemination - 731.1 Control Systems - 723 Computer Software, Data Handling and Applications - 722.4 Digital Computers and Systems - 722 Computer Systems and Equipment - 718 Telephone Systems and Related Technologies; Line Communications - 717 Optical Communication - 716 Telecommunication; Radar, Radio and Television - 453.2 Water Pollution Control - 444 Water Resources

Database: Compendex

25. **Visual Semantic Client a visualization tool for semantic content**

Wahl, Harald[1]; Mense, Alexander[1] **Source:** *ICETC 2010 - 2010 2nd International Conference on Education Technology and Computer*, v 2, p V219-V222, 2010, *ICETC 2010 - 2010 2nd International Conference on Education Technology and Computer*; **ISBN-13:** 9781424463688; **DOI:** 10.1109/ICETC.2010.5529443; **Article number:** 5529443; **Conference:** 2010 2nd International Conference on Education Technology and Computer, ICETC 2010, June 22, 2010 - June 24, 2010; **Sponsor:** Int. Assoc. Comput. Sci. Inf. Technol. (IACSIT); **Publisher:** IEEE Computer Society

Author affiliation:
[1] Information Engineering and Security, University of Applied Sciences Technikum Wien, Vienna, Austria

Abstract: The University of Applied Sciences Technikum Wien is a fastgrowing education organization that actually offers a set of 12 bachelor and 14 master degree programs. Coordination of lectures and therefore quality management has become more and more difficult. Knowledge management in terms of lecture contents and professional skills of lecturers seems to be an unsolvable task. As a matter of f act, nobody is able to overlook all teaching details of the whole university. Although information is available in several databases and documents even getting an overview of all detail content of a single degree program turns out to be impossible. To overcome this problem the Technikum Wien started a project to extract selected information from documents and store it in a Semantic Wiki by automatically setting up entities and their relations. To improve usage of semantic content a software tool to browse the information categories and their relations has being developed. The "Visual Semantic Client" visualizes entities with their attributes and allows following or searching their relations. The paper shows the concepts behind, the system architecture and the current state of development. © 2010 IEEE. (5 refs.)

Main Heading: Semantic Web

Controlled terms: Computer architecture - Knowledge management - Quality management - Semantics - Visualization

Uncontrolled terms: Applied science - Automatically setting - Degree program - Education organizations - Java - Knowledge Visualization - Master degree - Professional skills - RDF - Semantic content - Semantic wiki - Software tool - System architectures - Visual semantics - Visualization tools

Classification Code: 722 Computer Systems and Equipment - 723 Computer Software, Data Handling and Applications - 723.5 Computer Applications - 902.1 Engineering Graphics - 903 Information Science - 913.3 Quality Assurance and Control

Database: Compendex

26. **Hard- and software-configurable system for preoperative planning and intraoperative navigation of minimally invasive interventions**

Von Jan, Ute[1]; Sandkühler, D.[2]; Maas, S.[2]; Overhoff, H.M.[2] **Source:** *IFMBE Proceedings*, v 22 IFMBE, p 1769-1772, 2008, *4th European Conference of the International Federation for Medical and Biological Engineering - ECIFMBE 2008*; **ISSN:** 16800737; **ISBN-13:** 9783540892076; **DOI:** 10.1007/978-3-540-89208-3_422; **Conference:** 4th European Conference of the International Federation for Medical and Biological Engineering, ECIFMBE 2008, November 23, 2008 - November 27, 2008; **Publisher:** Springer Verlag

Author affiliation:
1 Peter L. Reichertz Institute for Medical Informatics, Carl-Neuberg-Str. 1, Hannover 30625, Germany
2 Medical Engineering Laboratory, University of Applied Sciences Gelsenkirchen, Gelsenkirchen, Germany

Abstract: At present, various commercial systems are available for preoperative planning and intraoperative navigation. Such systems are optimized for specialized hardware and selected intervention workflows, which complicates their adaption to other hardware and workflows. On the other hand, systems in use for research and development activities often lack essential features that are required for clinical applications. The effort for adding the necessary features to such systems is too high. Therefore, we developed an open, configurable software architecture for multimodal image based preoperative planning and intraoperative navigation under total quality management. The software architecture consists of several modules for preoperative visualization and operation planning as well as intraoperative navigation. Due to the open-interface architecture, image volumes from different modalities and manufacturers can be processed. The navigation module offers the possibility to integrate different localizer systems. In addition to a qualitative visual observation of the operative progress, the system gives quantitative hints for correctly moving the instruments to reach the desired position. So far, clinical applications for an ultrasound based approach in shoulder arthroplasty and for 3-D needle navigation have been implemented. Currently, four localizer systems (two optical and two electromagnetic) are included in the software. With the exception of the methods for surgical guidance, all components of the software architecture are standardized and developed under quality assurance. Thus, they can be used to build new clinical applications without any further software component tests. The software is currently being evaluated in clinical environments. The developed architecture allows for a reduced development time and costs for new clinically applications. Future work is addressed to the implementation of methods for standardized surgical guidance. © 2009 Springer Berlin Heidelberg. (4 refs.)

Main Heading: Software architecture

Controlled terms: Computer software selection and evaluation - Navigation - Planning - Quality assurance - Quality control - Surgery - Total quality management - Visualization

Uncontrolled terms: Clinical application - Clinical environments - Commercial systems - Configurable - Configurable systems - Desired position - Development time - Image volume - Interface architecture - Intraoperative navigation - Minimal invasive - Minimally invasive - Multi-modal image - Operation planning - Pre-operative planning - Research and development - Shoulder arthroplasty - Software component - Specialized hardware - Surgical guidance - Surgical planning - Visual observations - Work-flows

Classification Code: 403 Urban and Regional Planning and Development - 461.6 Medicine and Pharmacology - 716.3 Radio Systems and Equipment - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics - 913.3 Quality Assurance and Control

Database: Compendex

27. **Hard- and software-configurable system for preoperative planning and intraoperative navigation of minimally invasive interventions**

Von Jan, Ute[1]; Sandkühler, D.[2]; Maas, S.[2]; Overhoff, H.M.[2] **Source:** *IFMBE Proceedings*, v 22, p 1769-1772, 2008, *4th European Conference of the International Federation for Medical and Biological Engineering - ECIFMBE 2008*; **ISSN:** 16800737; **ISBN-13:** 9783540892076; **DOI:** 10.1007/978-3-540-89208-3_422; **Conference:** 4th European Conference of the International Federation for Medical and Biological Engineering, ECIFMBE 2008, November 23, 2008 - November 27, 2008; **Publisher:** Springer Verlag

Author affiliation:
1 Peter L. Reichertz Institute for Medical Informatics, Carl-Neuberg-Str. 1, Hannover 30625, Germany
2 Medical Engineering Laboratory, University of Applied Sciences Gelsenkirchen, Gelsenkirchen, Germany

Abstract: At present, various commercial systems are available for preoperative planning and intraoperative navigation. Such systems are optimized for specialized hardware and selected intervention workflows, which complicates their adaption to other hardware and workflows. On the other hand, systems in use for research and development activities often lack essential features that are required for clinical applications. The effort for adding the necessary features to such systems is too high. Therefore, we developed an open, configurable software architecture for multimodal image based preoperative planning and intraoperative navigation under total quality management. The software architecture consists of several modules for preoperative visualization and operation planning as well as intraoperative navigation. Due to the open-interface architecture, image volumes from different modalities and manufacturers can be processed. The navigation module offers the possibility to integrate different localizer systems. In addition to a qualitative visual observation of the operative progress, the system gives quantitative hints for correctly moving the instruments to reach the desired position. So far, clinical applications for an ultrasound based approach in shoulder arthroplasty and for 3-D needle navigation have been implemented. Currently, four localizer systems (two optical and two electromagnetic) are included in the software. With the exception of the methods for surgical guidance, all components of the software architecture are standardized and developed under quality assurance. Thus, they can be used to build new clinical applications without any further software component tests. The software is currently being evaluated in clinical environments. The developed architecture allows for a reduced development time and costs for new clinically applications. Future work is addressed to the implementation of

methods for standardized surgical guidance. © 2009 Springer Berlin Heidelberg. (4 refs.)

Main Heading: Software architecture

Controlled terms: Computer software selection and evaluation - Navigation - Planning - Quality assurance - Quality control - Surgery - Total quality management - Visualization

Uncontrolled terms: Clinical application - Clinical environments - Commercial systems - Configurable - Configurable systems - Desired position - Development time - Image volume - Interface architecture - Intraoperative navigation - Minimal invasive - Minimally invasive - Multi-modal image - Operation planning - Pre-operative planning - Research and development - Shoulder arthroplasty - Software component - Specialized hardware - Surgical guidance - Surgical planning - Visual observations - Work-flows

Classification Code: 403 Urban and Regional Planning and Development - 461.6 Medicine and Pharmacology - 716.3 Radio Systems and Equipment - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics - 913.3 Quality Assurance and Control

Database: Compendex

28. **Preventing erosion of architectural tactics through their strategic implementation, preservation, and visualization**

Mirakhorli, Mehdi[1] **Source:** *2013 28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013 - Proceedings*, p 762-765, 2013, *2013 28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013 - Proceedings*; **ISBN-13:** 9781479902156; **DOI:** 10.1109/ASE.2013.6693152; **Article number:** 6693152; **Conference:** 2013 28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013, November 11, 2013 - November 15, 2013; **Sponsor:** IEEE Computer Society; Association for Computing Machinery, Special Interest Group on Software Engineering (ACM SIGSOFT); IEEE Technical Council on Software Engineering (TCSE); ACM SIGART; NASA; **Publisher:** IEEE Computer Society

Author affiliation:
[1] DePaul University, School of Computing, Chicago, IL 60604, United States

Abstract: Nowadays, a successful software production is increasingly dependent on how the final deployed system addresses customers' and users' quality concerns such as security, reliability, availability, interoperability, performance and many other types of such requirements. In order to satisfy such quality concerns, software architects are accountable for devising and comparing various alternate solutions, assessing the trade-offs, and finally adopting strategic design decisions which optimize the degree to which each

of the quality concerns is satisfied. Although designing and implementing a good architecture is necessary, it is not usually enough. Even a good architecture can deteriorate in subsequent releases and then fail to address those concerns for which it was initially designed. In this work, we present a novel traceability approach for automating the construction of traceabilty links for architectural tactics and utilizing those links to implement a change impact analysis infrastructure to mitigate the problem of architecture degradation. Our approach utilizes machine learning methods to detect tactic-related classes. The detected tactic-related classes are then mapped to a Tactic Traceability Pattern. We train our trace algorithm using code extracted from fifty performance-centric and safety-critical open source software systems and then evaluate it against a real case study. © 2013 IEEE. (25 refs.)

Main Heading: Learning systems

Controlled terms: Architecture - Interoperability - Software architecture - Software reliability

Uncontrolled terms: Change impact analysis - Machine learning methods - Open source software systems - Software architects - Software production - tactics - traceability - traceability patterns

Classification Code: 402 Buildings and Towers - 716 Telecommunication; Radar, Radio and Television - 717 Optical Communication - 718 Telephone Systems and Related Technologies; Line Communications - 723 Computer Software, Data Handling and Applications

Database: Compendex

29. **A tool to visualize architectural design decisions**

Lee, Larix[1]; Kruchten, Philippe[1] **Source:** *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 5281 LNCS, p 43-54, 2008, *Quality of Software Architectures: Models and Architectures - 4th International Conference on the Quality of Software Architectures, QoSA 2008, Proceedings*; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3540878785, **ISBN-13:** 9783540878780; **DOI:** 10.1007/978-3-540-87879-7-3; **Conference:** 4th International Conference on the Quality of Software Architectures, QoSA 2008, October 14, 2008 - October 17, 2008; **Publisher:** Springer Verlag

Author affiliation:
[1] University of British Columbia

Abstract: The software architecture community is shifting its attention to architectural design decisions as a key element of architectural knowledge. Although there has been much work dealing with the representation of design decisions as formal structures within architecture, there still remains a need to investigate the exploratory nature of the design decisions themselves. We present in this paper a tool that should help improve the quality of software

architecture by enabling design decision exploration and analysis through decision visualization. Unlike many other design decision tools which acquire, list, and perform queries on decisions, our tool provides visualization components to help with decision exploration and analysis. Our tool has four main aspects: 1) the decision and relationship lists; 2) decision structure visualization view; 3) decision chronology view; and 4) decision impact view. Together, these four aspects provide an effective and powerful means for decision exploration and analysis. © 2008 Springer Berlin Heidelberg. (24 refs.)

Main Heading: Software architecture

Controlled terms: Architectural design - Computer software selection and evaluation - Visualization

Uncontrolled terms: Architectural knowledge - Decision impacts - Design decisions - Design-decision tools - Key elements - Quality of softwares

Classification Code: 912.2 Management - 902.1 Engineering Graphics - 723.5 Computer Applications - 723.1 Computer Programming - 723 Computer Software, Data Handling and Applications - 408.1 Structural Design, General - 402 Buildings and Towers

Database: Compendex

30. **A visual analysis and design tool for planning software reengineerings**

Beck, Martin[1]; Trümper, Jonas[1]; Döllner, Jürgen[1] **Source:** *Proceedings of VISSOFT 2011 - 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, 2011, *Proceedings of VISSOFT 2011 - 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis*; **ISBN-13:** 9781457708237; **DOI:** 10.1109/VISSOF.2011.6069458; **Article number:** 6069458; **Conference:** 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis, VISSOFT 2011, September 29, 2011 - September 30, 2011; **Sponsor:** IEEE Computer Society; IEEE Comput. Soc. Tech. Counc. Softw. Eng. (TCSE); **Publisher:** IEEE Computer Society

Author affiliation:
[1] Hasso-Plattner-Institute, University of Potsdam, Germany

Abstract: Reengineering complex software systems represents a non-trivial process. As a fundamental technique in software engineering, reengineering includes (a) reverse engineering the as-is system design, (b) identifying a set of transformations to the design, and (c) applying these transformations. While methods a) and c) are widely supported by existing tools, identifying possible transformations to improve architectural quality is not well supported and, therefore, becomes increasingly complex in aged and large software systems. In this paper we present a novel visual analysis and design tool to support software architects during reengineering tasks in identifying a given

software's design and in visually planning quality-improving changes to its design. The tool eases estimating effort and change impact of a planned reengineering. A prototype implementation shows the proposed technique's feasibility. Three case studies conducted on industrial software systems demonstrate usage and scalability of our approach. © 2011 IEEE. (37 refs.)

Main Heading: Software design

Controlled terms: C (programming language) - Computer software - Design - Quality control - Reengineering - Reverse engineering - Software architecture - Systems analysis

Uncontrolled terms: Architectural quality - Complex software systems - Industrial software - Large software systems - Non-trivial - Prototype implementations - Software architects - Visual analysis

Classification Code: 408 Structural Design - 723 Computer Software, Data Handling and Applications - 913.3 Quality Assurance and Control - 961 Systems Science

Database: Compendex

31. **Architecture of a high-performance surgical guidance system based on C-arm cone-beam CT: Software platform for technical integration and clinical translation**

Uneri, Ali[1]; Schafer, Sebastian[2]; Mirota, Daniel[1]; Nithiananthan, Sajendra[2]; Otake, Yoshito[1]; Reaungamornrat, Sureerat[1]; Yoo, Jongheun[1]; Stayman, J. Webster[2]; Reh, Douglas[3]; Gallia, Gary L.[4]; Khanna, A. Jay[5]; Hager, Gregory[1]; Taylor, Russell H.[1]; Kleinszig, Gerhard[6]; Siewerdsen, Jeffrey H.[1, 2] **Source:** *Progress in Biomedical Optics and Imaging - Proceedings of SPIE*, v 7964, 2011, *Medical Imaging 2011: Visualization, Image-Guided Procedures, and Modeling*; **ISSN:** 16057422; **ISBN-13:** 9780819485069; **DOI:** 10.1117/12.878191; **Article number:** 796422; **Conference:** Medical Imaging 2011: Visualization, Image-Guided Procedures, and Modeling, February 13, 2011 - February 15, 2011; **Sponsor:** The Society of Photo-Optical Instrumentation Engineers (SPIE); Dynasil Corporation/RMD Research; American Association of Physicists in Medicine (AAPM); DQE Instruments, Inc.; Ocean Thin Films, Inc.; **Publisher:** SPIE

Author affiliation:

1 Department of Computer Science, Johns Hopkins University, Baltimore, MD, United States

2 Department of Biomedical Engineering, Johns Hopkins University, Baltimore, MD, United States

3 Department of Otolaryngology, Johns Hopkins Medical Institute, Baltimore, MD, United States

4 Department of Neurosurgery, Johns Hopkins Medical Institute, Baltimore, MD, United States

5 Department of Orthopaedic Surgery, Johns Hopkins Medical Institute, Baltimore, MD, United States
6 Siemens Healthcare, Erlangen, Germany

Abstract: Intraoperative imaging modalities are becoming more prevalent in recent years, and the need for integration of these modalities with surgical guidance is rising, creating new possibilities as well as challenges. In the context of such emerging technologies and new clinical applications, a software architecture for cone-beam CT (CBCT) guided surgery has been developed with emphasis on binding open-source surgical navigation libraries and integrating intraoperative CBCT with novel, application-specific registration and guidance technologies. The architecture design is focused on accelerating translation of task-specific technical development in a wide range of applications, including orthopaedic, head-and-neck, and thoracic surgeries. The surgical guidance system is interfaced with a prototype mobile C-arm for high-quality CBCT and through a modular software architecture, integration of different tools and devices consistent with surgical workflow in each of these applications is realized. Specific modules are developed according to the surgical task, such as: 3D-3D rigid or deformable registration of preoperative images, surgical planning data, and up-to-date CBCT images; 3D-2D registration of planning and image data in real-time fluoroscopy and/or digitally reconstructed radiographs (DRRs); compatibility with infrared, electromagnetic, and video-based trackers used individually or in hybrid arrangements; augmented overlay of image and planning data in endoscopic or in-room video; real-time "virtual fluoroscopy" computed from GPU-accelerated DRRs; and multi-modality image display. The platform aims to minimize offline data processing by exposing quantitative tools that analyze and communicate factors of geometric precision. The system was translated to preclinical phantom and cadaver studies for assessment of fiducial (FRE) and target registration error (TRE) showing sub-mm accuracy in targeting and video overlay within intraoperative CBCT. The work culminates in the development of a CBCT guidance system (reported here for the first time) that leverages the technical developments in Carm CBCT and associated technologies for realizing a high-performance system for translation to clinical studies. © 2011 SPIE. (12 refs.)

Main Heading: Computerized tomography

Controlled terms: Augmented reality - C (programming language) - Data handling - Fluorescent screens - Image registration - Integration - Medical imaging - Navigation - Radiography - Remote control - Robotic arms - Software architecture - Surgical equipment - Three dimensional - Transplantation (surgical) - Virtual reality - Visualization

Uncontrolled terms: Cone-beam CT - Digitally reconstructed radiographs - Image guided surgery - Intra-operative - Surgical navigation

Classification Code: 921.2 Calculus - 902.1 Engineering Graphics - 801 Chemistry - 746 Imaging Techniques - 741.3 Optical Devices and Systems - 731 Automatic Control Principles and Applications - 723 Computer Software,

Data Handling and Applications - 716.3 Radio Systems and Equipment - 531 Metallurgy and Metallography - 462.4 Prosthetics - 462.1 Biomedical Equipment, General

Database: Compendex

32. **I-om: Intelligent optimization for computer graphics and visualization**

Moreira, Pedro Miguel[1]; Reis, Luís Paulo[2]; De Sousa, António Augusto[3]

Source: *Proceedings of the 5th Iberian Conference on Information Systems and Technologies, CISTI 2010*, 2010, *Proceedings of the 5th Iberian Conference on Information Systems and Technologies, CISTI 2010*; **ISBN-13:** 9789899624733; **Article number:** 5556644; **Conference:** 5th Iberian Conference on Information Systems and Technologies, CISTI 2010, June 16, 2010 - June 19, 2010; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Viana do Castelo, Viana do Castelo, Portugal
[2] DEI-FEUP, LIACC, Universidade do Porto, Porto, Portugal
[3] DEI-FEUP, INESC Porto, Universidade do Porto, Porto, Portugal

Abstract: There are several problems in the computer graphics and visualization domains which require optimization tasks to be performed in order to improve the quality of the overall process. In this context, we propose and describe an innovative optimization methodology and a supporting software framework: i-om. The design goals of the proposed framework were twofold. The first comprises the decoupling, as much as possible, of the optimization process from the application specific processing tasks. In order to attain this goal, we opted to make use of intelligent techniques (i.e. metaheuristics). The second goal is to allow remote operation, and consequently great portability and interoperability, between the optimization tools and the visualization application. To fulfill the latter requirement the proposed framework was designed with the ability to communicate with external application using a specifically developed high level message protocol. The optimization framework was implemented and the paper presents illustrative results demonstrating the usefulness and effectiveness of the proposed approach. (42 refs.)

Main Heading: Optimization

Controlled terms: Computer architecture - Computer graphics - Heuristic algorithms - Information systems - Visualization

Uncontrolled terms: Application specific - Design goal - Distributed architecture - Intelligent optimization - Intelligent techniques - Message protocol - Meta heuristics - Optimization framework - Optimization methodology - Optimization process - Optimization task - Optimization tools - Remote operation - Software frameworks - Visualization application

Classification Code: 722 Computer Systems and Equipment - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics - 903.2 Information Dissemination - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory - 921.5 Optimization Techniques

Database: Compendex

33. **Applying source code analysis techniques: A case study for a large mission-critical software system**

Haralambiev, Haralambi[1]; Boychev, Stanimir[1]; Lilov, Delyan[1]; Kraichev, Kraicho[1] **Source:** *EUROCON 2011 - International Conference on Computer as a Tool - Joint with Conftele 2011*, 2011, *EUROCON 2011 - International Conference on Computer as a Tool - Joint with Conftele 2011*; **ISBN-13:** 9781424474868; **DOI:** 10.1109/EUROCON.2011.5929241; **Article number:** 5929241; **Conference:** International Conference on Computer as a Tool, EUROCON 2011 - Joint with Conftele 2011, April 27, 2011 - April 29, 2011; **Sponsor:** Autoridade Nacional de Comunicacoes (ANACOM); LOGISER; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Applied Research and Development Center, MuSala Soft, Sofia, Bulgaria

Abstract: Source code analysis has been and still is extensively researched topic with various applications to the modern software industry. In this paper we share our experience in applying various source code analysis techniques for assessing the quality of and detecting potential defects in a large mission-critical software system. The case study is about the maintenance of a software system of a Bulgarian government agency. The system has been developed by a third-Party software vendor over a period of four years. The development produced over 4 million LOC using more than 20 technologies. MuSala Soft won a tender for maintaining this system in 2008. Although the system was operational, there were various issues that were known to its users. So, a decision was made to assess the system's quality with various source code analysis tools. The expectation was that the findings will reveal some of the problems' cause, allowing us to correct the issues and thus improve the quality and focus on functional enhancements. MuSala Soft had already established a special unit Applied Research and Development Center dealing with research and advancements in the area of software system analysis. Thus, a natural next step was for this unit to use the know-how and in-house developed tools to do the assessment. The team used various techniques that had been subject to intense research, more precisely: software metrics, code clone detection, defect and code smells detection through flow-sensitive and points-to analysis, software visualization and graph drawing. In addition to the open-source and free commercial tools, the team used internally developed ones that complement or improve what was available. The internally developed Smart Source Analyzer platform that was used is focused on several analysis areas: source code modeling, allowing easy navigation through the code elements and relations for different programming languages; quality audit through software

metrics by aggregating various metrics into a more meaningful quality characteristic (e.g. "maintainability"); source code pattern recognition to detect various security issues and "code smells". The produced results presented information about both the structure of the system and its quality. As the analysis was executed in the beginning of the maintenance tenure, it was vital for the team members to quickly grasp the architecture and the business logic. On the other hand, it was important to review the detected quality problems as this guided the team to quick solutions for the existing issues and also highlighted areas that would impede future improvements. The tool IPlasma and its System Complexity View (Fig. 1) revealed where the business logic is concentrated, which are the most important and which are the most complex elements of the system. The analysis with our internal metrics framework (Fig. 2) pointed out places that need refactoring because the code is hard to modify on request or testing is practically impossible. The code clone detection tools showed places where copy and paste programming has been applied. PMD, Find Bugs and Klockwork Solo tools were used to detect various "code smells" (Fig. 3). There were a number of occurrences that were indeed bugs in the system. Although these results were productive for the successful execution of the project, there were some challenges that should be addressed in the future through more extensive research. The two aspects we consider the most important are uSability and integration. As most of the tools require very deep understanding of the underlying analysis, the whole process requires tight cooperation between the analysis team and the maintenance team. For example, most of the metrics tools available provide specific values for a given metric without any indication what the value means and what is the threshold. Our internal metrics framework aggregates the metrics into meaningful quality characteristics, which solves the issue Partially. However, the user still often wonders about the justification behind the meaning of the given quality characteristic. There is a need for an explanation system one, which could point out the source code elements and explain why they are considered good or bad. The integration aspect is considered important because such analysis should be performed continuously. In our experience, the analysis is usually performed subsequent to an important event in this case: beginning of maintenance tenure. Some quality assurance practices should be developed and then adopted by the development teams so that the implementation quality is checked continuously. This should cover various activities and instruments, such as the integrated development environment, the code review process, automated builds, etc. In conclusion, we think that implementation quality audit and management is a vital activity that should be integrated into the software development process and the tools that support it should be uSable by the development team members without much knowledge of the underlying analysis. In this paper we presented a case study that showed the benefits of such a process.

Main Heading: Quality control

Controlled terms: Cloning - Codes (symbols) - Computer software - Computer software maintenance - Defects - Drawing (graphics) - Equipment - Integration - Maintainability - Object oriented programming - Odors - Pattern recognition - Problem oriented languages - Program

debugging - Quality assurance - Research - Software design - Systems analysis - Technology transfer - Visualization - Web services

Uncontrolled terms: Applied research - Business logic - Code clone detection - Code review - Code smell - Commercial tools - Copy-and-paste programming - Development teams - Explanation systems - Functional enhancements - Government agencies - Graph drawing - implementation quality - Integrated development environment - Know-how - Mission critical softwares - Open-source - Points-to analysis - Potential defects - Quality assurance practices - Quality characteristic - Quality problems - Refactorings - Security issues - Software development process - Software industry - software metrics - Software system analysis - Software systems - Software vendors - Software visualization - Source code analysis - Source codes - Specific values - System complexity - System's quality - Team members - Whole process

Classification Code: 951 Materials Science - 921.2 Calculus - 913.5 Maintenance - 913.3 Quality Assurance and Control - 902.1 Engineering Graphics - 961 Systems Science - 901 Engineering Profession - 716 Telecommunication; Radar, Radio and Television - 461.8.1 Genetic Engineering - 451.1 Air Pollution Sources - 423 Non Mechanical Properties and Tests of Building Materials - 723 Computer Software, Data Handling and Applications

Database: Compendex

34. **Envision: Visualizing water quality from geographically distributed wells**

Bradshaw, Jonathan L.[1]; Breen, David E.[2]; Gurian, Patrick L.[3] **Source:** *Proceedings of the 11th IASTED International Conference on Computer Graphics and Imaging, CGIM 2010*, p 231-238, 2010, *Proceedings of the 11th IASTED International Conference on Computer Graphics and Imaging, CGIM 2010*; **ISBN-13:** 9780889868243; **Conference:** 11th IASTED International Conference on Computer Graphics and Imaging, CGIM 2010, February 17, 2010 - February 19, 2010; **Publisher:** Acta Press

Author affiliation:
1 Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ, United States
2 Department of Civil, Architectural, and Environmental Engineering, Drexel University, Philadelphia, PA, United States
3 Department of Computer Science, Drexel University, Philadelphia, PA, United States

Abstract: Understanding trends in ground water contaminant concentrations is complicated by the fact that ground water quality presents complex three-dimensional trends over multiple spatial scales. To address this issue we have developed a method to jointly display three-dimensional ground water quality and topographic data. This approach allows the user to: 1) identify local areas

of elevated contaminant concentrations, 2) identify particular topographic features (river valleys, plateaus, etc) associated with elevated concentrations, 3) assess differences among aquifers and well use types, and 4) identify associations among different constituents. The approach used here links geospatially referenced concentration data with elevation data contained in Digital Terrain Elevation Data (DTED) files within an easy- to-use MATLAB-based visualization system. The method was applied to visualize information on nitrate and arsenic occurrence in a national ground water quality database. The method shows that high arsenic is associated with the transition from plains to piedmont in New Jersey. Nitrate in Iowa is shown to be associated with shallow wells in the southeastern portion of the state. The approach developed here is compatible with any Microsoft Excel Spreadsheet database which follows specific format conventions. (11 refs.)

Main Heading: River pollution

Controlled terms: Aquifers - Arsenic - Computer software - Concentration (process) - Data visualization - Geographic information systems - Groundwater pollution - Groundwater resources - Hydrogeology - Landforms - Spreadsheets - Three dimensional - Visualization - Water quality - Wells

Uncontrolled terms: Concentration data - Contaminant concentrations - Digital terrain elevation datum - Elevated concentrations - Elevation data - GIS - Groundwater quality - Microsoft excel - New Jersey - River valley - Shallow wells - Spatial scale - Topographic data - Topographic features - Visualization software. - Visualization system

Classification Code: 723.1 Computer Programming - 723.2 Data Processing and Image Processing - 723.3 Database Systems - 723.5 Computer Applications - 802.3 Chemical Operations - 804 Chemical Products Generally - 902.1 Engineering Graphics - 903.3 Information Retrieval and Use - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory - 723 Computer Software, Data Handling and Applications - 444.2 Groundwater - 445.2 Water Analysis - 446.1 Water Supply Systems - 453 Water Pollution - 453.1 Water Pollution Sources - 453.2 Water Pollution Control - 481.1 Geology - 512.1 Petroleum Deposits - 512.2 Natural Gas Deposits

Database: Compendex

35. **A software infrastructure for information quality assessment**

Ericsson, Morgan[1]; Wingkvist, Anna[1]; Löwe, Welf[1] **Source:** *ICIQ 2011 - Proceedings of the 16th International Conference on Information Quality*, p 400-414, 2011, *ICIQ 2011 - Proceedings of the 16th International Conference on Information Quality*; **Conference:** 16th International Conference on Information Quality, ICIQ 2011, November 18, 2011 - November 20, 2011; **Sponsor:** MIT Information QualityProgram; University of SouthAustralia; **Publisher:** Massachusetts Institute of Technology

Author affiliation:
1 Linnaeus University, School of Computer Science, Physics and Mathematics Växjö, Sweden

Abstract: Information quality assessment of technical documentation is nowadays an integral part of quality management of products and services. These are usually assessed using questionnaires, checklists, and reviews and consequently work that is cumbersome, costly and prone to errors. Acknowledging the fact that only humans can assess certain quality aspects, we suggest complementing these with automatic quality assessment using a software infrastructure that (i) reads information from documentations, (ii) performs analyses on this information, and (iii) visualizes the results to help stakeholders understand quality issues. We introduce the software infrastructure's architecture and implementation, its adaptation to different formats of documentations and types of analyses, along with a number of real world cases exemplifying feasibility and benefit of our approach. Altogether, our approach contributes to more efficient and automatic information quality assessments. (21 refs.)

Main Heading: Information analysis

Controlled terms: Quality management - Surveys

Uncontrolled terms: Automatic information - Information quality assessment - Integral part - Products and services - Quality aspects - Quality assessment - Software infrastructure - Software-based - Technical documentations

Classification Code: 903.1 Information Sources and Analysis - 913.3 Quality Assurance and Control

Database: Compendex

36. **Research and development of algorithm visualization teaching software on mechanical optimization design**

Li, Li[1]; Wu, Zhao Yun[2]; Wu, Li Hui[2]; Liu, Nan Bo[2] **Source:** *Applied Mechanics and Materials*, v 347-350, p 3403-3406, 2013, *Instruments, Measurement, Electronics and Information Engineering*; **ISSN:** 16609336, **E-ISSN:** 16627482; **DOI:** 10.4028/www.scientific.net/AMM.347-350.3403; **Conference:** 2013 International Conference on Precision Mechanical Instruments and Measurement Technology, ICPMIMT 2013, May 25, 2013 - May 26, 2013; **Publisher:** Trans Tech Publications Ltd

Author affiliation:
1 Automobile Engineering Department, Henan Vocational and Technical College, Zhengzhou, China
2 School of Mechanical and Electrical Engineering, Henan University of Technology, Zhengzhou, China

Abstract: In view of the teaching problems and difficulties in the course of mechanical optimization design, the algorithm visualization teaching software on mechanical optimization design was proposed. The functional requirements of the software were analyzed. The architecture and functional modules of the software were presented. The key technologies of the software were put forward, including the optimization algorithm library design, the method of visualization implementation, the method of preserving results and the animation reappearance technology. Based on these, the algorithm visualization teaching software was developed. The development environment and implementation methods of the software were proposed. It is proved that the software can play an important role in improving the teaching effect and quality. © 2013 Trans Tech Publications Ltd, Switzerland. (5 refs.)

Main Heading: Curricula

Controlled terms: Algorithms - Animation - Design - Optimization

Uncontrolled terms: Algorithm visualization - Development environment - Functional modules - Functional requirement - Mechanical optimization - Optimization algorithms - Research and development - Teaching reforms

Classification Code: 408 Structural Design - 723 Computer Software, Data Handling and Applications - 723.5 Computer Applications - 901.2 Education - 921 Mathematics - 921.5 Optimization Techniques

Database: Compendex

37. **VIZ - A graphical open-source architecture for use in structural bioinformatics**

Czekster, Ricardo M.[1]; De Souza, Osmar Norberto[1] **Source:** *Lecture Notes in Bioinformatics (Subseries of Lecture Notes in Computer Science)*, v 3594, p 226-229, 2005, *Advances in Bioinformatics and Computational Biology: Brazilian Symposium on Bioinformatics, BSB 2005. Proceedings*; **ISSN:** 03029743; **Conference:** Brazilian Symposium on Bioinformatics, BSB 2005: Advances in Bioinformatics and Computational Biology, July 27, 2005 - July 29, 2005; **Sponsor:** Brazilian Computer Society; Universidade Vale do Rio dos Sinos; The Rio Grande do Sul State Research Agency (FAPERGS); Hewlett-Packard; **Publisher:** Springer Verlag

Author affiliation:
[1] Laboratorio de Bioinformatica, Modelagem e Simulacao de Biossistemas, PPGCC - FACIN, PUCRS, 6681. Predio 16 - Sala 106, 90619-900 Porto Alegre, RS, Brazil

Abstract: Protein structure visualization is crucial for understanding its function inside the cell. Each year, laboratories around the world deposit protein structures on a central database for further analysis and research. The result is a large amount of structures being deposited (approximately 31,000 in may 2005). Visualization is a very powerful tool to help in the analysis, aiding

data understanding and interpretation. The present work suggests an architecture to help the rapid construction of visual biomolecular software, specifically designed to be simple, modular and scalable. The architecture, called VIZ, employs high quality open-source libraries offering simple data structures and customizable options. The architecture can be used to start a new visual software project to visualize and represent individual protein structures, as well as multiple conformations from molecular dynamics simulation trajectories. © Springer-Verlag Berlin Heidelberg 2005. (11 refs.)

Main Heading: Proteins

Controlled terms: Computer simulation - Computer software - Database systems - Molecular dynamics - Open systems - Visualization

Uncontrolled terms: Open-source libraries - Protein structure visualization - Structural bioinformatics - Visual biomolecular software

Classification Code: 722 Computer Systems and Equipment - 723.3 Database Systems - 723.5 Computer Applications - 801.4 Physical Chemistry - 804.1 Organic Compounds

Treatment: Theoretical (THR)

Database: Compendex

38. **Modeling architectural dependencies to support software release planning**

Nord, Robert L.[1]; Ozkaya, Ipek[1]; Brown, Nanette[1]; Sangwan, Raghvinder S.[2]
Source: *Invest on Visualization - Proceedings of the 13th International DSM Conference*, p 159-171, 2011, *Invest on Visualization - Proceedings of the 13th International DSM Conference*; **ISBN-10:** 3446430377, **ISBN-13:** 9783446430372; **Conference:** 13th International Dependency and Structure Modelling Conference, DSM'11, September 14, 2011 - September 15, 2011; **Publisher:** Carl Hanser Verlag GmbH and Co. KG

Author affiliation:
1 Software Engineering Institute, Carnegie Mellon University, 4500 Fifth Avenue, Pittsburgh, PA, United States
2 Pennsylvania State University, Malvern, PA, United States

Abstract: Organizations building products or systems that rely on software continue to demand increasingly rapid innovation and development processes that enable them to adjust products and systems to emerging needs. Release planning is a key activity in managing these processes. An essential aspect of release planning is balancing the development of new capabilities to address user needs against investment in system infrastructure and architecture to enable flexibility and maintainability. Providing quantifiable insight and visibility into both the delivered capabilities as well as the emerging quality of the software architecture is essential to product success. In this paper, we

describe our use of dependency structure and domain mapping matrices to model architectural dependencies. These dependencies provide insight into the value of the capabilities being delivered over total effort to better guide the process of release planning. (17 refs.)

Main Heading: Software architecture

Controlled terms: Economics - Maintainability - Visualization

Uncontrolled terms: Dependency analysis - Design structure matrix (DSM) - Domain mapping - Release planning - Software economics

Classification Code: 723.1 Computer Programming - 902.1 Engineering Graphics - 913.5 Maintenance - 971 Social Sciences

Database: Compendex

39. **Toward a multicore architecture for real-time ray-tracing**

Govindaraju, Venkatraman[1]; Djeu, Peter[2]; Sankaralingam, Karthikeyan[1]; Vernon, Mary[1, 3]; Mark, William R.[2] **Source:** *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, 2008 PROCEEDINGS, p 176-187, 2008, *2008 Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture, MICRO-41*; **ISSN:** 10724451; **ISBN-13:** 9781424428366; **DOI:** 10.1109/MICRO.2008.4771789; **Article number:** 4771789; **Conference:** 2008 - 41st Annual IEEE/ACM International Symposium on Microarchitecture, MICRO-41, November 8, 2008 - November 12, 2008; **Sponsor:** IEEE - Computer Society; IEEE-CS TC uARCH; ACM; NSF; Intel; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
1 Department of Computer Sciences, University of Wisconsin-Madison
2 Department of Computer Sciences, University of Texas at Austin
3 Vertical Research Group

Abstract: Significant improvement to visual quality for real-time 3D graphics requires modeling of complex illumination effects like soft-shadows, reflections, and diffuse lighting interactions. The conventional Z-buffer algorithm driven GPU model does not provide sufficient support for this improvement. This paper targets the entire graphics system stack and demonstrates algorithms, a software architecture, and a hardware architecture for real-time rendering with a paradigm shift to ray-tracing. The three unique features of our system called Copernicus are support for dynamic scenes, high image quality, and execution on programmable multicore architectures. The focus of this paper is the synergy and interaction between applications, architecture, and evaluation. First, we describe the ray-tracing algorithms which are designed to use redundancy and partitioning to achieve locality. Second, we describe the architecture which uses ISA specialization, multithreading to hide memory delays and supports only local coherence. Finally, we develop an analytical performance model for our 128-core system,

using measurements from simulation and a scaled-down prototype system. More generally, this paper addresses an important issue of mechanisms and evaluation for challenging workloads for future processors. Our results show that a single 8-core tile (each core 4-way multithreaded) can be almost 100% utilized and sustain 10 million rays/second. Sixteen such tiles, which can fit on a 240mm^2 chip in 22nm technology, make up the system and with our anticipated improvements in algorithms, can sustain real-time rendering. The mechanisms and the architecture can potentially support other domains like irregular scientific computations and physics computations. © 2008 IEEE. (43 refs.)

Main Heading: Software architecture

Controlled terms: Algorithms - Architecture - Building materials - Communication channels (information theory) - Image quality - Multitasking - Three dimensional

Uncontrolled terms: Analytical performance model - Core systems - Dynamic scenes - Graphics systems - Hardware architecture - High image quality - Illumination effect - Multi-threading - Multicore architectures - Multithreaded - Paradigm shifts - Prototype system - Ray-tracing algorithm - Real-time 3D graphics - Real-time rendering - Scientific computation - Unique features - Visual qualities - Z-buffer

Classification Code: 921 Mathematics - 722.4 Digital Computers and Systems - 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming - 723.5 Computer Applications - 741 Light, Optics and Optical Devices - 902.1 Engineering Graphics - 716.1 Information Theory and Signal Processing - 414 Masonry Materials - 413 Insulating Materials - 412 Concrete - 411 Bituminous Materials - 402 Buildings and Towers - 415 Metals, Plastics, Wood and Other Structural Materials

Database: Compendex

40. **3D visualization service for GIS based pipe network system**

Xu, Liutong[1]; Shi, Min[1]; Lin, Suping[1] **Source:** *Journal of Harbin Institute of Technology (New Series)*, v 15, n SUPPL., p 62-66, April 2008; **ISSN:** 10059113; **Publisher:** Harbin Institute of Technology

Author affiliation:
[1] Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: In GIS based pipe network system, the spatial relationships among pipelines are so complicated that two-dimensional (2D) visualization cannot meet the demands of reflecting real spatial relationship between pipelines clearly. However, three-dimensional (3D) visualization has such capabilities. In this paper, we propose a 3D visualization service for GIS based pipe network

system, comprised of three sub-services, 3D pipe unit library (PUL) service, 3D modeling (3DM) service and error rectification (ER) service. PUL service focuses on providing 3D pipe units for modeling, 3DM service creates and maintains the 3D scene graph, and ER service concentrates on adjusting pipe spatial attributes. Among them, the PUL service is highlighted with its detailed hierarchical class design. In PUL service, the pipe unit models are organized by inheritance hierarchy, which includes four basic categories (segment, valve, accessory, and connector) and their sub-categories. In the end, to illustrate the usage of the service, we share our experience in implementing the 3D visualization service to an existing GIS based 2D pipe network system and the results reach up to our design goals. (7 refs.)

Main Heading: Geographic information systems

Controlled terms: Computer simulation - Java programming language - Pipelines - Quality of service - Software architecture - Three dimensional - Visualization

Uncontrolled terms: 3D visualization - Java 3D - Pipe network - Service - Three dimensional Visualization service

Classification Code: 619.1 Pipe, Piping and Pipelines - 723.1.1 Computer Programming Languages - 723.5 Computer Applications - 903.3 Information Retrieval and Use

Treatment: Applications (APP); Experimental (EXP)

Database: Compendex

41. **A survey about the intent to use visual defect annotations for software models**

Rech, Jörg[1]; Spriestersbach, Axel[2] **Source:** *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 5095 LNCS, p 406-418, 2008, *Model Driven Architecture - Foundations and Applications - 4th European Conference, ECMDA-FA 2008, Proceedings*; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3540690956, **ISBN-13:** 9783540690955; **DOI:** 10.1007/978-3-540-69100-6-29; **Conference:** 4th European Conference on Model Driven Architecture - Foundations and Applications, ECMDA-FA 2008, June 9, 2008 - June 13, 2008; **Sponsor:** Eur. Proj. MODELPLEX (MODELling solut. comPLEX softw. syst.); Int. Bus. Mach. Corp.; Netfective Technol. SA; Objecteering Softw. SA; Testing Technologies IST GmbH; **Publisher:** Springer Verlag

Author affiliation:
[1] Fraunhofer IESE, Fraunhofer Platz 1, Kaiserslautern 67663, Germany
[2] SAP Research, Vincenz-Prienitz-Str. 1, Karlsruhe 76131, Germany

Abstract: Today, many practitioners have consolidated their experience with

software models in collections of design flaws, smells, antipatterns, or guidelines that have a negative impact on quality aspects (such as maintainability). Besides these quality defects, many compilability errors or conformance warnings might occur in a software design. Programming IDEs typically present problems regarding compilability in or near the code (e.g., icons at the line or underlining in the code). Modeling IDEs in MDSD follow a visual paradigm and need a similar mechanism for presenting problems in a clear, consistent, and familiar way. In this paper, we present different visualization concepts for visualizing quality defects and other problems in software models. These concepts use different dimensions such as color, size, or icons to present this information to the user. We used a survey to explore the opinions held by practitioners showing that 89.9% want to be informed about potential defects and prefer icon-, view- and underscore-based concepts to other types of concepts. © 2008 Springer-Verlag Berlin Heidelberg. (15 refs.)

Main Heading: Computer software selection and evaluation

Controlled terms: Defects - Maintainability - Software design - Surveys

Uncontrolled terms: Intelligent Assistance - MDSD - Quality Defects - Software Diagnostics - Software Models - Visual Annotations

Classification Code: 951 Materials Science - 933.1 Crystalline Solids - 913.5 Maintenance - 912.2 Management - 903.1 Information Sources and Analysis - 723.5 Computer Applications - 723.1 Computer Programming - 723 Computer Software, Data Handling and Applications - 531.2 Metallography - 423 Non Mechanical Properties and Tests of Building Materials - 405.3 Surveying

Database: Compendex

42. **Visualizing multiple program executions to assist behavior verification**

Zhao, Chunying[1]; Kang, Zhang[1]; Hao, Jie[1]; Wong, W. Eric[1] **Source:** *SSIRI 2009 - 3rd IEEE International Conference on Secure Software Integration Reliability Improvement*, p 113-122, 2009, *SSIRI 2009 - 3rd IEEE International Conference on Secure Software Integration Reliability Improvement*; **ISBN-13:** 9780769537580; **DOI:** 10.1109/SSIRI.2009.26; **Article number:** 5325386; **Conference:** 3rd IEEE International Conference on Secure Software Integration Reliability Improvement, SSIRI 2009, July 8, 2009 - July 10, 2009; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Department of Computer Science, University of Texas at Dallas, Richardson, TX, United States

Abstract: Visualization techniques have been widely used in representing software artifacts. They play a central role in conveying program information to software developers. While numerous tools have been developed to visualize

information such as static software architectures, dynamic program behaviors, and debugging processes, little attention has been paid to visualizing correlations and variations among program representations. This paper investigates the visualization of cross-references across multiple program executions based upon different testing inputs so that meaningful and viewable properties can be presented to the viewpoint from different perspectives. Visualizing such a comparison can help feature location and program behavior verification. It also helps programmers better understand and test their software which can have a significant impact on improving its reliability. © 2009 IEEE. (39 refs.)

Main Heading: Program debugging

Controlled terms: Computer software selection and evaluation - Quality assurance - Software architecture - Software reliability - Visualization

Uncontrolled terms: Debugging process - Dynamic behaviors - Dynamic programs - Feature location - Multiple program - Program behavior - Significant impacts - Software artifacts - Software developer - Software visualization - Visualization technique

Classification Code: 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming - 723.5 Computer Applications - 902.1 Engineering Graphics - 912.2 Management - 913.3 Quality Assurance and Control

Database: Compendex

43. **Initial design of the 'plug-n-analyze' framework for architecture tradeoff analysis**

In, Hoh[1]; Flores-Mendoza, Ana Erendira[1] **Source:** *Proceedings - IEEE Computer Society's International Computer Software and Applications Conference*, p 318-319, 1999; **ISSN:** 07303157; **Conference:** Proceedings of the 1999 23rd Annual International Computer Software and Applications Conference (COMPSAC '99), October 27, 1999 - October 29, 1999; **Sponsor:** IEEE Computer Society; **Publisher:** IEEE

Author affiliation:
[1] Texas A&M Univ, College Station, United States

Abstract: This paper presents an initial design of the 'Plug-n-Analyze' framework for the tradeoff analysis in determining architecture alternatives that have different strength and weakness on quality attributes. The framework provides: i) architecture alternative negotiation aids among multiple stakeholders, ii) visualization aids for understanding complicated interactions and tradeoff dependencies among quality attributes, and iii) a general environment by integrating independent analysis methods by simply 'plugging' and generating architecture options for fully 'analyzing'. (6 refs.)

Main Heading: Computer software

Controlled terms: Computer aided analysis - Computer architecture - Performance - Software engineering - Visualization

Uncontrolled terms: Quality attributes - Tradeoff analysis

Classification Code: 723.1 Computer Programming - 723.5 Computer Applications

Treatment: Applications (APP); Theoretical (THR)

Database: Compendex

44. **Visualization for situational awareness**

Feibush, Eliot[1, 2]; Gagvani, Nikhil[2]; Williams, Daniel[3] **Source:** *IEEE Computer Graphics and Applications*, v 20, n 5, p 38-45, September/October 2000; **ISSN:** 02721716; **DOI:** 10.1109/38.865878; **Publisher:** Institute of Electrical and Electronics Engineers Computer Society

Author affiliation:
1 Sarnoff Corporation, Princeton, NJ, United States
2 Sarnoff Corp., 201 Washington Road, Princeton, NJ 08540-6449, United States
3 Systems and Scientific Software, 263 Forrest Ave., Elkins Park, PA 19027, United States

Abstract: Visualization techniques in a situational awareness system aid rapid comprehension of a complex battlespace. Hardware scalability lets remote users share situational awareness with the command center. Some of the techniques used for effectively modeling and navigating geospatial and tactical data for situational awareness are presented. The visualization technologies discussed could be applied to commercial air traffic control, emergency management response, and geographic information systems as well. (14 refs.)

Main Heading: Computer simulation

Controlled terms: Air traffic control - Computational complexity - Computational geometry - Computer architecture - Data structures - Graphical user interfaces - Image analysis - Image quality - Visualization

Uncontrolled terms: Common object request broker architecture (CORBA) - Image Analysis - Situational awareness systems

Classification Code: 431.5 Air Navigation and Traffic Control - 721.1 Computer Theory, Includes Formal Logic, Automata Theory, Switching Theory, Programming Theory - 722.2 Computer Peripheral Equipment - 723.2 Data Processing and Image Processing - 723.5 Computer Applications - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory

Treatment: Applications (APP); Theoretical (THR)

Database: Compendex

45. **An architecture-based software reliability modeling tool and its support for teaching**

Wang, Wen-Li[1]; Scannell, Dan[1] **Source:** *Proceedings - Frontiers in Education Conference, FIE*, v 2005, p T4C-15-T4C-20, 2005, *Proceedings - Frontiers in Education, 35th Annual Conference: Pedagogies and Technologies for the Emerging Global Economy, FIE'05*; **ISSN:** 15394565; **ISBN-10:** 0780390776, **ISBN-13:** 9780780390775; **Article number:** 1611965; **Conference:** Frontiers in Education - 35th Annual Conference 2005, FIE' 05, October 19, 2005 - October 22, 2005; **Publisher:** Institute of Electrical and Electronics Engineers Inc.

Author affiliation:
[1] School of Engineering and Engineering Technology, Penn State Erie, Behrend College, 5091 Station Road, Erie, PA 16563

Abstract: This paper presents an architecture-based software reliability modeling tool for pedagogy and demonstrates its support for conveying learning materials to students. Software reliability is an important quality attribute. Improving this attribute early in the software life cycle is highly desirable, because it greatly reduces testing and maintenance effort later on. The architecture-based approach is for such a purpose. This tool enables students to conduct relative analyses on different architectural designs and to compute a more accurate measure once detailed information is available. It equips a GUI for architectureto-state modeling, taking into account four architectural styles. Students can incorporate additional architectural styles into this framework, and exercise different design alternatives. The GUI shows a graphical representation of software architecture, and helps students visualize the matrix construction for design changes. This tool has shortened students' learning curve, helped them understand the impact of different designs on reliability estimates, and increased their interests in other quality attributes. © 2005 IEEE. (13 refs.)

Main Heading: Software architecture

Controlled terms: Computer aided design - Learning systems - Mathematical models - Parameter estimation - Software reliability - Teaching

Uncontrolled terms: Architectural styles - Architectureto state modeling - Markov models - Reliability modeling

Classification Code: 723.1 Computer Programming - 723.4 Artificial Intelligence - 723.5 Computer Applications - 731.1 Control Systems - 901.2 Education - 921 Mathematics

Treatment: General review (GEN)

Database: Compendex

46. **Distributable features view: Visualizing the structural characteristics of distributed software systems**

Cosma, Dan C.[1]; Marinescu, Radu[1] **Source:** *VISSOFT 2007 - Proceedings of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, p 55-62, 2007, *VISSOFT 2007 - Proceedings of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*; **ISBN-10:** 1424406005, **ISBN-13:** 9781424406005; **DOI:** 10.1109/VISSOF.2007.4290700; **Article number:** 4290700; **Conference:** VISSOFT 2007 - 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis, June 25, 2007 - June 26, 2007; **Sponsor:** IEEE Computer Society; IEEE Comput. Soc. Tech. Council on Software Engineering (TCSE); IEEE Comput. Soc. Visualization and Graphics Tech. Comm. (VGTC); **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] LOOSE Research Group, Politehnica University of Timisoara, Romania

Abstract: The software industry is increasingly confronted with the issues of understanding and maintaining a special type of software systems, namely distributed systems. Although these systems are usually implemented in an object-oriented fashion, they raise very specific, and technology-dependent, understandability and quality assessment challenges. This paper presents a visual approach for comprehending the design of distributed software systems, by using technology awareness to isolate the functionally-distinct units within the code, so that the blueprint of the system's distributed behavior can be easily extracted. The approach provides means for observing the system's distributed architecture, visualizing the structure of the functional entities, and understanding their collaboration within the system, while focusing the analysis to the most substantial code fragments that deserve being taken into consideration. (20 refs.)

Main Heading: Software design

Controlled terms: Codes (standards) - Codes (symbols) - Computer software

Uncontrolled terms: (OTDR) technology - Code fragments - Distributed architectures - Distributed software systems - Distributed Systems - International (CO) - Object oriented (OO) - quality assessments - Software industry - Software systems - Structural characteristics - Understandability

Classification Code: 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming - 723.2 Data Processing and Image Processing - 902.2 Codes and Standards

Database: Compendex

47. **GLuRay: Ray tracing in scientific visualization applications using OpenGL interception**

Brownlee, Carson[1, 2]; Fogal, Thomas[2]; Hansen, Charles D.[1, 2] **Source:** *Eurographics Workshop on Parallel Graphics and Visualization*, p 41-50, 2012, *EG PGV 2012 - 12th Eurographics Symposium on Parallel Graphics and Visualization*; **ISSN:** 1727348X; **ISBN-13:** 9783905674354; **DOI:** 10.2312/EGPGV/EGPGV12/041-050; **Conference:** 12th Eurographics Symposium on Parallel Graphics and Visualization, EG PGV 2012, May 13, 2012 - May 14, 2012; **Sponsor:** EUROGRAPHICS Association; CRS4; Data Intensive Visualization and Analysis (DIVA); Universitas Studiorum Caralitana; **Publisher:** Eurographics Association

Author affiliation:
[1] School of Computing, University of Utah, United States
[2] SCI Institute, University of Utah, United States

Abstract: Ray tracing in scientific visualization allows for substantial gains in performance and rendering quality with large scale polygonal datasets compared to brute-force rasterization, however implementing new rendering architectures into existing tools is often costly and time consuming. This paper presents a library, GLuRay, which intercepts OpenGL calls from many common visualization applications and renders them with the CPU ray tracer Manta without modification to the underlying visualization tool. Rendering polygonal models such as isosurfaces can be done identically to an OpenGL implementation using provided material and camera properties or superior rendering can be achieved using enhanced settings such as dielectric materials or pinhole cameras with depth of field effects. Comparative benchmarks were conducted on the Texas Advanced Computing Center's Longhorn cluster using the popular visualization packages ParaView, VisIt, Ensight, and VAPOR. Through the parallel rendering package ParaView, scaling up to 64 nodes is demonstrated. With our tests we show that using OpenGL interception to accelerate and enhance visualization programs provides a viable enhancement to existing tools with little overhead and no code modification while allowing for the creation of publication quality renderings using advanced effects and greatly improved large-scale software rendering performance within tools that scientists are currently using. © The Eurographics Association 2012. (26 refs.)

Main Heading: Computer graphics

Controlled terms: Application programming interfaces (API) - Cluster computing - Dielectric materials - Flow visualization - Ray tracing - Software testing - Vapors

Uncontrolled terms: Code modifications - Computing center - Data sets - Depth of field - Iso surface - Parallel rendering - Petascale - Polygonal

models - Quality rendering - Rasterization - Rendering quality - Scaling-up - Software rendering - Space visualization - Visualization application - Visualization packages - Visualization tools

Classification Code: 631.1 Fluid Flow, General - 708.1 Dielectric Materials - 722.4 Digital Computers and Systems - 723 Computer Software, Data Handling and Applications - 723.5 Computer Applications - 804 Chemical Products Generally

Database: Compendex

48. **PadLogSpace: A prolog-based new software architecture for the service federation of web resources**

Lkhamsuren, D.[1]; Tanaka, Y.[1] **Source:** *IET Conference Publications*, v 2010, n 568 CP, p 390-395, 2010, *IET International Conference on Frontier Computing. Theory, Technologies and Applications*; **ISBN-13:** 9781849192088; **DOI:** 10.1049/cp.2010.0593; **Conference:** IET International Conference on Frontier Computing. Theory, Technologies and Applications, August 4, 2010 - August 6, 2010; **Publisher:** Institution of Engineering and Technology

Author affiliation:
[1] Hokkaido University, Japan

Abstract: Service composition has received much interest from users on its development tools. Users may often need to use not only a single service, but also several services, and in some cases, use those services in combination with their own local applications. Users need a simple and interactive tool for service composition development to realize their needs without writing any program codes, and to reduce realization time. In our previous work, we proposed the software architecture for ad hoc service federation of Web and local resources. This architecture enables users to directly manipulate visual components, and to create new composite components for the creation of services that use Web applications, Web services, and local applications. In this architecture, every service composition is done by users' manipulation. In this paper, we will propose a new software architecture, called a PadLogSpace, for extending the service registration function and the service matching function of our previously proposed PadSpace to provide a semantic rule-definition mechanism for the service composition development. (7 refs.)

Main Heading: Software architecture

Controlled terms: Computation theory - PROLOG (programming language) - Quality of service - Web services

Uncontrolled terms: A-prolog - Composite components - Development tools - Interactive tool - Program code - Semantic rules - Service compositions - Service matching - Service registration - Visual components - WEB application - Web resources

Classification Code: 716 Telecommunication; Radar, Radio and Television -

717 Optical Communication - 718 Telephone Systems and Related Technologies; Line Communications - 723 Computer Software, Data Handling and Applications - 921 Mathematics

Database: Compendex

49. **A model-driven traceability framework for software product lines**

Anquetil, Nicolas[1]; Kulesza, Uirá[2]; Mitschke, Ralf[3]; Moreira, Ana[2]; Royer, Jean-Claude[1]; Rummler, Andreas[4]; Sousa, André[2] **Source:** *Software and Systems Modeling*, v 9, n 4, p 427-451, 2010; **ISSN:** 16191366, **E-ISSN:** 16191374; **DOI:** 10.1007/s10270-009-0120-9; **Publisher:** Springer Verlag

Author affiliation:
[1] ASCOLA, EMN-INRIA, Nantes, France
[2] CITI/DI/FCT, Universidade Nova de Lisboa, Caparica, Portugal
[3] TU Darmstadt, Darmstadt, Germany
[4] SAP Research, Dresden, Germany

Abstract: Software product line (SPL) engineering is a recent approach to software development where a set of software products are derived for a well defined target application domain, from a common set of core assets using analogous means of production (for instance, through Model Driven Engineering). Therefore, such family of products are built from reuse, instead of developed individually from scratch. SPL promise to lower the costs of development, increase the quality of software, give clients more flexibility and reduce time to market. These benefits come with a set of new problems and turn some older problems possibly more complex. One of these problems is traceability management. In the European AMPLE project we are creating a common traceability framework across the various activities of the SPL development. We identified four orthogonal traceability dimensions in SPL development, one of which is an extension of what is often considered as "traceability of variability". This constitutes one of the two contributions of this paper. The second contribution is the specification of a metamodel for a repository of traceability links in the context of SPL and the implementation of a respective traceability framework. This framework enables fundamental traceability management operations, such as trace import and export, modification, query and visualization. The power of our framework is highlighted with an example scenario. © 2009 Springer-Verlag. (56 refs.)

Main Heading: Software design

Controlled terms: Concurrent engineering - Network architecture - Visualization

Uncontrolled terms: Core asset - Management operation - Meta model - Model-driven - Model-driven Engineering - Quality of softwares - Software development - Software Product Line - Software product lines - Software products - Target application - Time to market - Traceability - Traceability links

Classification Code: 722 Computer Systems and Equipment - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics

Database: Compendex

50. **Autostereoscopic display of large-scale scientific visualization**

Peterka, Tom[1]; Ross, Robert[1]; Yu, Hongfeng[2]; Ma, Kwan-Liu[3]; Kooima, Robert[4]; Girado, Javier[5] **Source:** *Proceedings of SPIE - The International Society for Optical Engineering*, v 7237, 2009, *Proceedings of SPIE-IS and T Electronic Imaging - Stereoscopic Displays and Applications XX*; **ISSN:** 0277786X; **DOI:** 10.1117/12.805422; **Article number:** 723706; **Conference:** Stereoscopic Displays and Applications XX, January 19, 2009 - January 21, 2009; **Publisher:** SPIE

Author affiliation:
1 Argonne National Laboratory, 9700 South Cass Ave., Argonne, IL, United States
2 Sandia National Laboratories, California, PO Box 969, Livermore, CA, United States
3 University of California at Davis, One Shields Ave., Davis, CA, United States
4 University of Illinois at Chicago, 851 S. Morgan St., Chicago, IL, United States
5 Qualcomm Inc., 5775 Morehouse Dr., San Diego, CA, United States

Abstract: Modern computational science poses two challenges for scientific visualization: managing the size of resulting datasets and extracting maximum knowledge from them. While our team attacks the first problem by implementing parallel visualization algorithms on supercomputing architectures at vast scale, we are experimenting with autostereoscopic display technology to aid scientists in the second challenge. We are building a visualization framework connecting parallel visualization algorithms running on one of the world's most powerful supercomputers with high-quality autostereo display systems. This paper is a case study of the development of an end-to-end solution that couples scalable volume rendering on thousands of supercomputer cores to the scientists' interaction with autostereo volume rendering at their desktops and larger display spaces. We discuss modifications to our volume rendering algorithm to produce perspective stereo images, their transport from supercomputer to display system, and the scientists' 3D interactions. A lightweight display client software architecture supports a variety of monoscopic and autostereoscopic display technologies through a flexible configuration framework. This case study provides a foundation that future research can build upon in order to examine how autostereo immersion in scientific data can improve understanding and perhaps enable new discoveries. © 2009 SPIE-IS&T. (38 refs.)

Main Heading: Visualization

Controlled terms: Computers - Display devices - Parallel algorithms - Parallel architectures - Software architecture - Supercomputers - Three dimensional - Volume rendering

Uncontrolled terms: 3D interactions - Auto-stereoscopic displays - Autostereoscopic 3D displays - Computational science - Data-sets - Display systems - End-to-end solutions - High qualities - Large scale visualization - Monoscopic - Parallel visualizations - Scientific datum - Scientific visualizations - Stereo images - Visualization frameworks - Volume rendering algorithms

Classification Code: 921 Mathematics - 902.1 Engineering Graphics - 723.5 Computer Applications - 723.2 Data Processing and Image Processing - 723.1 Computer Programming - 723 Computer Software, Data Handling and Applications - 722.4 Digital Computers and Systems - 722.2 Computer Peripheral Equipment - 722 Computer Systems and Equipment

Database: Compendex

51. **Optimization of routing architecture and performance for FPGA routing fabric**

Tan, Biley[1] **Source:** *Proceedings of the 4th Asia Symposium on Quality Electronic Design, ASQED 2012*, p 8-15, 2012, *Proceedings of the 4th Asia Symposium on Quality Electronic Design, ASQED 2012*; **ISBN-13:** 9781467326889; **DOI:** 10.1109/ACQED.2012.6320468; **Article number:** 6320468; **Conference:** 4th Asia Symposium on Quality Electronic Design, ASQED 2012, July 10, 2012 - July 11, 2012; **Sponsor:** International Society for Quality Electronic Design; Altera; Cadence; InnovoTek; MIMOS; **Publisher:** IEEE Computer Society

Author affiliation:
[1] IC Design Engineering, Altera Corporation (M) Sdn. Bhd., Bayan Lepas Technoplex, Plot 6, Medan Bayan Lepas, 11900 Penang, Malaysia

Abstract: Being a critical highway between FPGA core IPs, the routing fabric components play an integral part in overall core performance. Optimizing the components in the routing fabric region is not a trivial task when considering the multitude of variable physical parameters involved especially the profound association with software programming and constantly changing process parameters. Common optimization algorithm would be discussed, incorporating propose multi-dimensional graphical method. Case study illustrates the potential of graphical method in identifying optimal component sizing and tradeoffs, aiding design and decision making in delivering the most optimal designs in routing fabric region. Supported by comprehensive graphical performance visualization, basic routing element architecture would also be discussed, developing expressions to promote faster optimization by narrowing down the region of interest; improving resource utilization on optimization. © 2012 IEEE. (10 refs.)

Main Heading: Optimization

Controlled terms: Algorithms - Design - Field programmable gate arrays (FPGA) - Visualization

Uncontrolled terms: Component sizing - Core performance - FPGA core - Graphical methods - Graphical visualization - Integral part - Optimal design - Optimization algorithms - Performance visualization - Physical parameters - Process parameters - Region of interest - Resource utilizations - Routing architecture - Routing resources - Simplex algorithm - Software programming

Classification Code: 721.3 Computer Circuits - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics - 921 Mathematics - 921.5 Optimization Techniques

Database: Compendex

52. **Tracking concept drift of software projects using defect prediction quality**

Ekanayake, Jayalath[1]; Tappolet, Jonas[1]; Gall, Harald C.[2]; Bernstein, Abraham[1] **Source:** *Proceedings of the 2009 6th IEEE International Working Conference on Mining Software Repositories, MSR 2009*, p 51-60, 2009, *Proceedings of the 2009 6th IEEE International Working Conference on Mining Software Repositories, MSR 2009*; **ISBN-13:** 9781424434930; **DOI:** 10.1109/MSR.2009.5069480; **Article number:** 5069480; **Conference:** 2009 6th IEEE International Working Conference on Mining Software Repositories, MSR 2009, May 16, 2009 - May 17, 2009; **Publisher:** IEEE Computer Society

Author affiliation:
1 Dynamic and Distributed Systems Group, Department of Informatics, University of Zurich
2 Software Evolution and Architecture Lab., Department of Informatics, University of Zurich

Abstract: Defect prediction is an important task in the mining of software repositories, but the quality of predictions varies strongly within and across software projects. In this paper we investigate the reasons why the prediction quality is so fluctuating due to the altering nature of the bug (or defect) fixing process. Therefore, we adopt the notion of a concept drift, which denotes that the defect prediction model has become unsuitable as set of influencing features has changed - usually due to a change in the underlying bug generation process (i.e., the concept). We explore four open source projects (Eclipse, OpenOffice, Netbeans and Mozilla) and construct file-level and project-level features for each of them from their respective CVS and Bugzilla repositories. We then use this data to build defect prediction models and visualize the prediction quality along the time axis. These visualizations allow us to identify concept drifts and-as a consequence-phases of stability and instability expressed in the level of defect prediction quality. Further, we

identify those project features, which are influencing the defect prediction quality using both a tree induction-algorithm and a linear regression model. Our experiments uncover that software systems are subject to considerable concept drifts in their evolution history. Specifically, we observe that the change in number of authors editing a file and the number of defects fixed by them contribute to a project's concept drift and therefore influence the defect prediction quality. Our findings suggest that project managers using defect prediction models for decision making should be aware of the actual phase of stability or instability due to a potential concept drift. © 2009 IEEE. (18 refs.)

Main Heading: Computer software selection and evaluation

Controlled terms: Defects - Linear regression - Mathematical models - Phase stability - Project management - Software engineering - System stability

Uncontrolled terms: Concept drifts - Defect prediction - Defect prediction models - Evolution history - Generation process - Linear regression models - Mozilla - NetBeans - Open source projects - Prediction quality - Project managers - Quality of predictions - Software project - Software repositories - Software systems - Stability and instability - Time axis - Tree induction

Classification Code: 912.2 Management - 921 Mathematics - 922.2 Mathematical Statistics - 931.2 Physical Properties of Gases, Liquids and Solids - 933.1 Crystalline Solids - 951 Materials Science - 961 Systems Science - 801.4 Physical Chemistry - 423 Non Mechanical Properties and Tests of Building Materials - 531 Metallurgy and Metallography - 531.2 Metallography - 641 Heat and Mass Transfer; Thermodynamics - 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming - 731.4 System Stability

Database: Compendex

53. **A GPU-based architecture for real-time data assessment at synchrotron experiments**

Chilingaryan, Suren[1]; Kopmann, Andreas[1]; Mirone, Alessandro[2]; Dos Santos Rolo, Tomy[3] **Source:** *Conference Record - 2010 17th IEEE-NPSS Real Time Conference, RT10*, 2010, *Conference Record - 2010 17th IEEE-NPSS Real Time Conference, RT10*; **ISBN-13:** 9781424471096; **DOI:** 10.1109/RTC.2010.5750342; **Article number:** 5750342; **Conference:** 2010 17th IEEE-NPSS Real Time Conference, RT10, May 24, 2010 - May 28, 2010; **Publisher:** IEEE Computer Society

Author affiliation:
1 Institute of Data Processing and Electronics, Karlsruhe Institute of Technology, Karlsruhe, Germany
2 European Synchrotron Radiation Facility, Grenoble, France

3 Institute for Synchrotron Radiation, Karlsruhe Institute of Technology, Karlsruhe, Germany

Abstract: Current imaging experiments at synchrotron beam lines often lack a real-time data assessment. X-ray imaging cameras installed at synchrotron facilities like ANKA provide millions of pixels, each with a resolution of 12 bits or more, and take up to several thousand frames per second. A given experiment can produce data sets of multiple gigabytes in a few seconds. Up to now the data is stored in local memory, transferred to mass storage, and then processed and analyzed off-line. The data quality and thus the success of the experiment, can, therefore, only be judged with a substantial delay, which makes an immediate monitoring of the results impossible. To optimize the usage of the micro-tomography beam-line at ANKA we have ported the reconstruction software to modern graphic adapters which offer an enormous amount of calculation power. We were able to reduce the reconstruction time from multiple hours to just a few minutes with a sample dataset of 20 GB. Using the new reconstruction software it is possible to provide a near real-time visualization and significantly reduce the time needed for the first evaluation of the reconstructed sample. The main paradigm of our approach is 100% utilization of all system resources. The compute intensive parts are offloaded to the GPU. While the GPU is reconstructing one slice, the CPUs are used to prepare the next one. A special attention is devoted to minimize data transfers between the host and GPU memory and to execute I/O operations in parallel with the computations. It could be shown that for our application not the computational part but the data transfers are now limiting the speed of the reconstruction. Several changes in the architecture of the DAQ system are proposed to overcome this second bottleneck. The article will introduce the system architecture, describe the hardware platform in details, and analyze performance gains during the first half year of operation. © 2010 IEEE. (33 refs.)

Main Heading: Experiments

Controlled terms: Data transfer - Program processors - Synchrotrons - Visualization

Uncontrolled terms: Current imaging - DAQ system - Data quality - Data sets - Frames per seconds - Hardware platform - I/O operations - Local memories - Mass storage - Micro-tomography - Performance Gain - Real time visualization - Real-time data - Reconstruction software - Sample dataset - Substantial delays - Synchrotron beamlines - System architectures - System resources - Xray imaging

Classification Code: 723.1 Computer Programming - 723.2 Data Processing and Image Processing - 901.3 Engineering Research - 902.1 Engineering Graphics - 932.1.1 Particle Accelerators

Database: Compendex

54. **Mobile applications software testing methodology**

Kim, Haeng-Kon[1] **Source:** *Communications in Computer and Information Science*, v 342 CCIS, p 158-166, 2012, *Computer Applications for Web, Human Computer Interaction, Signal and Image Processing, and Pattern Recognition - Int. Conf., SIP, WSE, and ICHCI 2012, Held in Conjunction with GST 2012, Proceedings*; **ISSN:** 18650929; **ISBN-13:** 9783642352690; **DOI:** 10.1007/978-3-642-35270-6_22; **Conference:** 2012 Int. Conf. on Signal Processing, Image Processing and Pattern Recognition, SIP 2012, 2012 Int. Conf. on Web Science and Engineering, WSE 2012, and the 2012 Int. Conf. on Human Computer Interaction, ICHCI 2012, Held in Conjunction with GST 2012, November 28, 2012 - December 2, 2012; **Publisher:** Springer Verlag

Author affiliation:
[1] School of Information Technology, Catholic University of Daegu, Korea, Republic of

Abstract: Today's Mobile Applications deliver complex functionality on platforms that have limited resources for computing. Yet, unlike the PC-based environment, the Mobile environment comprises a number of devices with diverse hardware and software configurations and communication intricacies. This diversity in mobile computing environments presents unique challenges in mobile application development, quality assurance, and deployment, requiring unique testing strategies. Many enterprise applications that were deployed as desktop/web applications are now being ported to Mobile devices. In this paper, we have constructed the Mobile Applications Quality Assurance Tool(MAQAT) by integrating tools and prototype systems that we built for program analysis and testing for mobile applications software. MAQAS provides a architecture of program analysis and testing for mobile, and supports many program-analysis-based techniques, including automated mobile applications software inspection, software visualization, testing coverage analysis, performance evaluation, concurrent program debugging, software measurement, etc. The paper briefly describes the overall architecture of MAQAS, and introduces the implementation of its tools and components. © 2012 Springer-Verlag Berlin Heidelberg. (11 refs.)

Main Heading: Software testing

Controlled terms: Computer software selection and evaluation - Image processing - Inspection - Internet protocols - Mobile computing - Mobile devices - Pattern recognition - Quality assurance - Software engineering

Uncontrolled terms: Concurrent program - Coverage analysis - Enterprise applications - Hardware and software - Mobile application development - Mobile applications - Mobile computing environment - Mobile environments - PC-based - Performance evaluation - Program analysis - Prototype system - Quality assurance tools - Software Measurement - Software visualization - Testing methodology - Testing strategies

Classification Code: 716 Telecommunication; Radar, Radio and Television - 723 Computer Software, Data Handling and Applications - 741 Light, Optics

and Optical Devices - 913.3 Quality Assurance and Control - 913.3.1 Inspection

Database: Compendex

55. **Research on the Application of Visualization Computer Technology in the Design of Architectural Decoration**

Zhou, Jun[1]; Zhou, Jian[2]; Ma, Xiao Li[3] **Source:** *Applied Mechanics and Materials*, v 380-384, p 4179-4183, 2013, *Vehicle, Mechatronics and Information Technologies*; **ISSN:** 16609336, **E-ISSN:** 16627482; **ISBN-13:** 9783037858202; **DOI:** 10.4028/www.scientific.net/AMM.380-384.4179; **Conference:** 2013 International Conference on Vehicle and Mechanical Engineering and Information Technology, VMEIT 2013, August 17, 2013 - August 18, 2013; **Publisher:** Trans Tech Publications Ltd

Author affiliation:

1 Architecture and Art College, Shijiazhuang Tiedao University, Shijiazhuang, China

2 Digital Media Department, Hebei Software Institute, Baoding, China

3 Microsoft IT Department, Shijiazhuang Information Engineering Vocational College, Shijiazhuang, China

Abstract: With the advances of computer science and technology, the visualization computer technology profoundly affects and changes people's understanding of the traditional visual language and promotes the continuous progress of the building decoration design. Its function of repeatedly revised not only can improve the quality of design but also can shorten the design cycle. The characteristics of building decoration design and the advantages of application of visual computer technology were first analyzed; based on analyses, the article explores the reliabilities of visualization computer technology in architectural decoration design. Linear model provides a theoretical basis and practical support for the development of architectural decoration design in a certain extent. © (2013) Trans Tech Publications, Switzerland. (9 refs.)

Main Heading: Architectural design

Controlled terms: Computer applications - Flow visualization - Information technology - Visualization

Uncontrolled terms: Computer science and technologies - Computer technology - Design cycle - Empirical analysis - Linear modeling - Quality of design - Visual Computer

Classification Code: 402 Buildings and Towers - 631.1 Fluid Flow, General - 723.5 Computer Applications - 902.1 Engineering Graphics

Database: Compendex

56. **Designing test engine for computer-aided software testing tools**

Ma, Xue-Ying[1]; Sheng, Bin-Kui[1] **Source:** *WSEAS Transactions on Computers*, v 10, n 5, p 135-145, May 2011; **ISSN:** 11092750; **Publisher:** World Scientific and Engineering Academy and Society

Author affiliation:
[1] College of Information Management, Zhejiang University of Finance and Economics, Hangzhou, China

Abstract: With the rapid development of software scale and programming languages, it is impossible to test software manually. The case for automating the software testing process has been made repeatedly and convincingly by numerous testing professionals. Automated tests can promote the efficiency of software testing and then to increase software productivity, improve software quality, and reduce cost in almost all processes of software engineering. White-box testing is one of the most important software testing strategies that can detect error even when the software specification is vague or incomplete. This paper gives a detailed description of the design and implementation of a testing engine. The testing engine, which is the kernel of a developed structured software-testing tool for the Visual Basic and C/C++ language, mainly consists of three components: program analyzer, source code instrumentation tool and intermediate database. In the testing engine, a block division mechanism and a new block-based CFG model are introduced and some block-based test adequacy criteria are extended. The programs are divided into a sequence of blocks and then instrumented and compiled in the testing engine, and all the information related to the test is saved in the intermediate database. The testing engine, acting as an agency, associates the testing automation module with instrumented executable program rather than the source code, and therefore the testing tool can easily be developed to accommodate new requirements and different testing adequacy criteria. It is also convenient to build a testing environment for multi-languages by modifying the program analyzer only, due to the flexibility of the software architecture. (16 refs.)

Main Heading: Software testing

Controlled terms: Computer programming languages - Computer software selection and evaluation - Database systems - Engines - Instruments - Software architecture - Software design

Uncontrolled terms: Automated test - Block division - Executable programs - Intermediate database - Object-oriented software-testing - Program instrumentation - Rapid development - Software productivity - Software Quality - Software Specification - Software test - Source Code Instrumentation - Source codes - Test adequacy criteria - Testing automation - Testing environment - Testing strategies - Testing tools - Three component - VISUAL BASIC - White-box testing

Classification Code: 612 Engines - 723 Computer Software, Data Handling and Applications - 941 Acoustical and Optical Measuring Instruments - 942

Electric and Electronic Measuring Instruments - 943 Mechanical and Miscellaneous Measuring Instruments - 944 Moisture, Pressure and Temperature, and Radiation Measuring Instruments

Database: Compendex

57. **VIZ - A graphical open-source architecture for use in structural bioinformatics**

Czekster, Ricardo M.[1]; De Souza, Osmar Norberte[1] **Source:** *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 3594 LNBI, p 226-229, 2005; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3540280081, **ISBN-13:** 9783540280088; **Conference:** Brazilian Symposium on Bioinformatics - Advances in Bioinformatics and Computational Biology, July 27, 2005 - July 29, 2005; **Sponsor:** Bracilian Computer Society; Universidade Vale do Rio dos Sinos; Brazilian National Council for Research; Rio Grande do sul State Research Agency; Hewlett-Packard; **Publisher:** Springer Verlag

Author affiliation:
[1] Laboratorio de Bioinformatica, Modelagem e Simulacao de Biossistemas - LABIO, PPGCC - FACIN, PUCRS, 6681. Predio 16 - Sala 106, 90619-900 Porto Alegre, RS, Brazil

Abstract: Protein structure visualization is crucial for understanding its function inside the cell. Each year, laboratories around the world deposit protein structures on a central database for further analysis and research. The result is a large amount of structures being deposited (approximately 31,000 in may 2005). Visualization is a very powerful tool to help in the analysis, aiding data understanding and interpretation. The present work suggests an architecture to help the rapid construction of visual biomolecular software, specifically designed to be simple, modular and scalable. The architecture, called VIZ, employs high quality open-source libraries offering simple data structures and customizable options. The architecture can be used to start a new visual software project to visualize and represent individual protein structures, as well as multiple conformations from molecular dynamics simulation trajectories. © Springer-Verlag Berlin Heidelberg 2005. (11 refs.)

Main Heading: Proteins

Controlled terms: Biotechnology - Cell culture - Data reduction - Database systems - Molecular dynamics - Visualization

Uncontrolled terms: Molecular dynamics simulation trajectories - Protein structure visualization - Visual biomolecular software

Classification Code: 461.8 Biotechnology - 723.2 Data Processing and Image Processing - 723.3 Database Systems - 801.4 Physical Chemistry - 804.1 Organic Compounds

Treatment: Theoretical (THR)

Database: Compendex

58. **Visual configuration in automotive software product lines**

Botterweck, Goetz[1]; Thiel, Steffen[1]; Cawley, Ciarán[1]; Nestor, Daren[1]; Preuner, André[2] **Source:** *Proceedings - International Computer Software and Applications Conference*, p 1070-1075, 2008, *Proceedings - 32nd Annual IEEE International Computer Software and Applications Conference, COMPSAC 2008*; **ISSN:** 07303157; **ISBN-13:** 9780769532622; **DOI:** 10.1109/COMPSAC.2008.108; **Article number:** 4591724; **Conference:** 32nd Annual IEEE International Computer Software and Applications Conference, COMPSAC 2008, July 28, 2008 - August 1, 2008; **Publisher:** Inst. of Elec. and Elec. Eng. Computer Society

Author affiliation:
[1] Lero, University of Limerick, Limerick, Ireland
[2] BTU Cottbus, Institute of Computer Science, Cottbus, Germany

Abstract: Software Product Line engineering has emerged as a viable and important software development paradigm in the automotive industry. It allows companies to realise significant improvements in time-to-market, cost, productivity, and system quality. One major difficulty with software product line engineering is related to the fact that a product line of industrial size can easily incorporate thousands of variation points. This scale of variability can become extremely complex to manage resulting in a product configuration process that bears significant costs. This paper introduces a meta-model and research tool that employs visualisation and interaction techniques to improve product configuration in high-variability product lines. The meta-model and techniques utilised are illustrated using an automotive restraint system example. © 2008 IEEE. (15 refs.)

Main Heading: Computer software

Controlled terms: Automotive engineering - Automotive industry - Chlorine compounds - Computer applications - Computers - Network architecture - Product development - Production engineering - Research - Software design - Technology - Word processing

Uncontrolled terms: Automotive software - Industrial size - Interaction techniques - Meta modelling - Product configurations - Product lines - Software development - Software Product Line Engineering - System quality - Time-to-market - Visual configuration - Visualisation

Classification Code: 913.1 Production Engineering - 901.3 Engineering Research - 901 Engineering Profession - 804.1 Organic Compounds - 723.5 Computer Applications - 913.6 Product Development; Concurrent Engineering - 723.2 Data Processing and Image Processing - 723 Computer Software, Data Handling and Applications - 722 Computer Systems and

Equipment - 664 Automotive Engineering, General - 661 Automotive Engines and Related Equipment - 723.1 Computer Programming

Database: Compendex

59. **SWB: An analysis and visualization tool for COSMO SkyMed**

Lore, V.A.[1]; Milillo, G.[2]; Milillo, P.[3]; Valentino, A.[1]; Franceschetti, G.[4] **Source:** *Progress in Electromagnetics Research Symposium*, p 1623-1627, 2012, *PIERS 2012 Kuala Lumpur - Progress in Electromagnetics Research Symposium, Proceedings*; **ISSN:** 15599450; **ISBN-13:** 9781934142202; **Conference:** Progress in Electromagnetics Research Symposium, PIERS 2012 Kuala Lumpur, March 27, 2012 - March 30, 2012; **Sponsor:** TELEKOM Malaysia; TELEKOM Malaysia Research and Development; Motorola Solutions; Malaysia Convention and Exhibition Bureau (MyCeb); **Publisher:** Electromagnetics Academy

Author affiliation:

1 INNOVA Consorzio per L'informatica e la Telematica s.r.l., Italy
2 Agenzia Spaziale Italiana (ASI), Italy
3 Fisiche E Naturali Dipartimento Interateneo di Fisica, Facolta di Scienze Matematiche, Universita Degli Studi di Bari, Italy
4 Dipartimento di Ingegneria Elettronica e delle Telecomunicazioni, Universita di Napoli Federico II, Italy

Abstract: With the increase of the availability of high quality data for Earth Observation (EO), and SAR in particular, it is even more important to have advanced applications that can exploit this great amount of data.Also it is important to have software (SW) tools that allow scientists and engineers to process and analyze data, and to develop innovative applications."SAR WorkBench" (SWB) is a SW for the fast visualization/browsing, analysis and post processing of multi-mission EO data.It was developed in the framework of one of the Italian Space Agency (ASI) projects for COSMO SkyMed data exploitation and has, among others, some features that are very peculiar like the Quality Analysis ones.SWB has a modular architecture and it is extendible via plug-ins.It can be programmed in an high level programming language, close to the ones scientists are used to, and do not requires expensive SW licenses.In this paper, it is presented an overview of the SW and its architecture and some examples of data analysis performed with SWB. (7 refs.)

Main Heading: Quality control

Controlled terms: Condensed matter physics - Radiation

Uncontrolled terms: Advanced applications - Cosmo skymed - Earth observations - Fast visualization - High quality data - High-level programming language - Italian Space Agency - ITS architecture - Modular architectures - Post processing - Scientists and engineers - Visualization tools

Classification Code: 711 Electromagnetic Waves - 913.3 Quality Assurance and Control - 931 Classical Physics; Quantum Theory; Relativity - 933 Solid State Physics

Database: Compendex

60. **Quality of Software Architectures: Models and Architectures - 4th International Conference on the Quality of Software Architectures, QoSA 2008, Proceedings**
Source: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 5281 LNCS, 2008, *Quality of Software Architectures: Models and Architectures - 4th International Conference on the Quality of Software Architectures, QoSA 2008, Proceedings*; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3540878785, **ISBN-13:** 9783540878780; **Conference:** 4th International Conference on the Quality of Software Architectures, QoSA 2008, October 14, 2008 - October 17, 2008; **Publisher:** Springer Verlag

Abstract: The proceedings contain 14 papers. The topics discussed include: rethinking the use of models in software architecture; design reasoning improves software design quality; a tool to visualize architectural design decisions; designing the enterprise architecture function; quality prediction of service compositions through probabilistic model checking; model-driven performance analysis; architectural specification and static analyses of contractual application properties; integrating quality-attribute reasoning frameworks in the ArchE design assistant; middleware architecture evaluation for dependable self-managing systems; comprehensive architecture evaluation and management in large software-systems; and sharing the architectural knowledge of quantitative analysis.

Database: Compendex

61. **Code rocket: Improving detailed design support in mainstream software development**
Parkes, Steve[1]; Ramsay, Craig[1]; Spark, Alan[2] **Source:** *2011 International Conference on Computer and Management, CAMAN 2011*, 2011, *2011 International Conference on Computer and Management, CAMAN 2011*; **ISBN-13:** 9781424492831; **DOI:** 10.1109/CAMAN.2011.5778773; **Article number:** 5778773; **Conference:** 2011 International Conference on Computer and Management, CAMAN 2011, May 19, 2011 - May 21, 2011; **Sponsor:** IEEE Wuhan Section; Hunan University; Wuhan University; Engineering Information Institute; Chongqing VIP Information Co., Ltd; **Publisher:** IEEE Computer Society

Author affiliation:
1 School of Computing, University of Dundee, Dundee, United Kingdom
2 Research and Development, Rapid Quality Systems Ltd., Dundee, United Kingdom

Abstract: In mainstream software development there can often be a gap which exists between the stages of performing the architectural design of a software system and implementing the detailed algorithms and processes required in the program code. Code Rocket is a code visualization and documentation system which has been developed to fill this gap. Code Rocket provides automated design and documentation support for software developers during detailed stages of code construction. It integrates seamlessly with existing development tools to provide extensive documentation with little or no effort on behalf of the software engineer. Code and documentation remain fully synchronized even when changes are implemented in the code. This paper describes Code Rocket, the rationale for its development, and the key features and benefits it delivers to different stakeholders on a software project. ©2011 IEEE. (12 refs.)

Main Heading: Software design

Controlled terms: Architectural design - Computer software - Rockets - Visualization

Uncontrolled terms: Automated design - Code construction - Code visualization - Design tools and techniques - Detailed design - Development tools - Documentation systems - Key feature - Program code - Software developer - Software engineers - Software project - Software systems

Classification Code: 402 Buildings and Towers - 404.1 Military Engineering - 723 Computer Software, Data Handling and Applications - 902.1 Engineering Graphics

Database: Compendex

62. **Proceedings of the 5th European Conference on Software Architecture, ECSA 2011 - Companion Volume: Workshop on Traceability, Dependencies and Software Architecture, TDSA 2011, 1st International Workshop on Software Architecture Variability, SAVA 2011**
Source: *ACM International Conference Proceeding Series*, 2011, *Proc. of the 5th ECSA 2011 - Companion Volume: Workshop on Traceability, Dependencies and Software Architecture, TDSA 2011, 1st International Workshop on Software Architecture Variability, SAVA 2011*; **ISBN-13:** 9781450306188; **Conference:** 5th European Conference on Software Architecture, ECSA 2011 - Workshop on Traceability, Dependencies and Software Architecture, TDSA 2011, 1st International Workshop on Software Architecture Variability, SAVA 2011, September 13, 2011 - September 16, 2011; **Sponsor:** Universitat Duisburg-Essen; PALUNO - The Ruhr Institute for Software Technology; adesso - business. people. technology.; e.on; **Publisher:** Association for Computing Machinery

Abstract: The proceedings contain 10 papers. The topics discussed include: dependencies, traceability and consistency in software architecture: towards a view-based perspective; architectural design decision visualization for architecture design: preliminary results of a controlled experiment; concepts

and diagram elements for architectural knowledge management; rationale, decisions and alternatives traceability for architecture design; delta-oriented architectural variability using MontiCore; quality attributes and variability in AO-ADL software architectures; towards variability management in business document types using product line engineering; a model-driven approach for automating mobile applications testing; and analysis of a cross-domain reference architecture using change scenarios.

Database: Compendex

63. **Is it possible to decorate graphical software design and architecture models with qualitative information? - An experiment**

Bratthall, Lars[1]; Wohlin, Claes[2] **Source:** *IEEE Transactions on Software Engineering*, v 28, n 12, p 1181-1193, December 2002; **ISSN:** 00985589; **DOI:** 10.1109/TSE.2002.1158290; **Publisher:** Institute of Electrical and Electronics Engineers Inc.

Author affiliation:
[1] Corporate Research Department, ABB AS, Norway, Bergervn 12, N-1375 Billingstad, Norway
[2] Dept. of Software Eng./Comp. Sci., Blekinge Institute of Technology, PO Box 520, SE-372 25 Ronneby, Sweden

Abstract: Software systems evolve over time and it is often difficult to maintain them. One reason for this is that often it is hard to understand the previous release. Further, even if architecture and design models are available and up to date, they primarily represent the functional behavior of the system. To evaluate whether it is possible to also represent some nonfunctional aspects, an experiment has been conducted. The objective of the experiment is to evaluate the cognitive suitability of some visual representations that can be used to represent a control relation, software component size and component external and internal complexity. Ten different representations are evaluated in a controlled environment using 35 subjects. The results from the experiment show that representations with low cognitive accessibility weight can be found. In an example, these representations are used to illustrate some qualities in an SDL block diagram. It is concluded that the incorporation of these representations in architecture and design descriptions is both easy and probably worthwhile. The incorporation of the representations should enhance the understanding of previous releases and, hence, help software developers in evolving and maintaining complex software systems. (44 refs.)

Main Heading: Software engineering

Controlled terms: Computational complexity - Computer software maintenance - Mathematical models - Statistical methods

Uncontrolled terms: Software evolution

Classification Code: 721.1 Computer Theory, Includes Formal Logic,

Automata Theory, Switching Theory, Programming Theory - 723.1 Computer Programming - 921 Mathematics - 922.2 Mathematical Statistics

Treatment: Theoretical (THR); Experimental (EXP)

Database: Compendex

64. **Modelling airborne mission systems using the architecture analysis and design language**

Sioutis, C.[1]; Nguyen, T.[1] **Source:** *18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation: Interfacing Modelling and Simulation with Mathematical and Computational Sciences, Proceedings*, p 1650-1655, 2009, *18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation: Interfacing Modelling and Simulation with Mathematical and Computational Sciences, Proceedings*; **ISBN-13:** 9780975840078; **Conference:** 18th World IMACS Congress and International Congress on Modelling and Simulation: Interfacing Modelling and Simulation with Mathematical and Computational Sciences, MODSIM09, July 13, 2009 - July 17, 2009; **Sponsor:** CSIRO; Australian Mathematical Sciences Institute; Griffith University; eWater Cooperative Research Centre; Department of Sustainability and Environment; **Publisher:** Acta Press

Author affiliation:
[1] Airborne Mission Systems, Air Operations Division, Defence Science and Technology Organisation, SA, Australia

Abstract: One of the critical success factors for integration of a Defence system is a good architectural design. A right architecture can help ensure that a system will satisfy its key operational requirements as well as its quality attributes such as real-time performance, reliability, security, and maintainability. A bad architecture on the other hand is a recipe for disaster (Clements et. al. 2001). Emerging properties of systems such as scheduling, fault tolerance, and security can cause significant problems in integration of complex Defence systems. Life-cycles are becoming evolutionary as components of systems are upgraded to avoid obsolescence, but such upgrades must be done in the context of system impact (Allen et. al. 2002). Software intensive acquisition projects are historically considered the most risk prone in the Defence domain and often incur schedule delays, cost overruns, and reduced operational capability (Commonwealth of Australia 2004). Capability systems have a "life cycle" that begins with the identification of the need to address a current or prospective capability gap. This need is progressively translated into a working capability system that is operated and supported until it is ultimately withdrawn from service. Architecture modelling and analysis can potentially be incorporated to the Defence Capability Life Cycle (DCLC) (Commonwealth of Australia 2006) for assessing technical risks associated with proposed system architectures. This paper describes initial research conducted for modelling and analysing Airborne Mission Systems (AMS). The Architecture Analysis and Design Language (AADL) was chosen for this purpose because it is specifically geared for model-based development and

analysis of real-time embedded systems. The AADL is a textual and graphical language that models the architecture of systems as an assembly of software components mapped on an execution platform. The primary advantage for using the AADL is that it provides different ways of expressing a model being developed. Graphical AADL is useful for development and visualization, text AADL is useful editing and parameterisation, XML AADL is useful to conduct various types of analysis with different third party tools. This allows finding problems early and potentially saving costs throughout the system's entire life cycle. This paper introduces the AADL and then describes the two case studies employed to understand and apply it. The aim of the first case study was to gain an appreciation of how to develop an AADL model in general. It was decided to develop an AADL model that was related to the air domain but not be overly complex. This reasoning led to the modelling a model-helicopter in flight, which includes a pilot controlling the helicopter via a controller. The second study focused on investigating how to specifically model a subset of the mission system hardware and software of a Royal Australian Navy S-70B-2 Seahawk helicopter. This research revealed that an AADL model must be developed based on what is the specific question being investigated. Models intended to simply describe a system can afford to be more abstract in order to highlight higher level architectural elements. Conversely, models intended for specific analysis must have an adequate fidelity and be populated with any parameters needed to perform that analysis. (13 refs.)

Main Heading: Mathematical models

Controlled terms: Architectural design - Embedded systems - Fault tolerance - Helicopters - Life cycle - Maintainability - Obsolescence - Real time systems - Research - Visualization

Uncontrolled terms: Acquisition projects - Analysis - Architectural element - Architecture analysis - Australia - Cost overruns - Critical success factor - Defence systems - Graphical languages - Mission systems - Model based development - Modelling - Operational capabilities - Operational requirements - Quality attributes - Real time performance - Real-time embedded systems - Royal Australian Navy - Schedule delay - Software component - System architectures - System impact - Technical risks - Third-party tools

Classification Code: 921 Mathematics - 913.5 Maintenance - 913.1 Production Engineering - 902.1 Engineering Graphics - 901.3 Engineering Research - 901 Engineering Profession - 723 Computer Software, Data Handling and Applications - 722.4 Digital Computers and Systems - 722 Computer Systems and Equipment - 652.4 Helicopters - 402 Buildings and Towers

Database: Compendex

65. **A Software-based H.264/AVC HDTV real-time interactive codec architecture using parallel processing**

Sano, Takashi[1]; Ohnishi, Takayuki[1]; Iwasaki, Hiroe[1]; Kamikura, Kazuto[1];

Naganuma, Jiro[2] **Source:** *ICCE 2010 - 2010 Digest of Technical Papers International Conference on Consumer Electronics*, p 467-468, 2010, *ICCE 2010 - 2010 Digest of Technical Papers International Conference on Consumer Electronics*; **ISBN-13:** 9781424443161; **DOI:** 10.1109/ICCE.2010.5418798; **Article number:** 5418798; **Conference:** 2010 International Conference on Consumer Electronics, ICCE 2010, January 11, 2010 - January 13, 2010; **Publisher:** IEEE Computer Society

Author affiliation:
[1] NTT Cyber Space Laboratories, NTT Corporation, 1-1, Hikarinooka, Yokosuka, Kanagawa, 239-0847, Japan
[2] NTT Electronics Corporation, 1-1-32, Shin-Urashima, Kanagawa, Yokohama, Kanagawa, 221-0031, Japan

Abstract: This paper describes a software-based H.264/AVC HDTV real-time interactive CODEC architecture (named RISCA264-HD) using parallel processing. The RISCA264-HD consists of multiple high-speed encoder/decoder cores, an IP communication part, and an error recovery controller with FEC. It provides Full-HD quality (1920 x 1080 pixels, 29.97 frames per second) using parallel encoding, natural interactive conversation with low delay of less than 165 ms, and smooth visual communication free from macro block noises. This software with a home television and a home digital video camera achieves HDTV-quality bidirectional video communication via commercially IP broadband network. ©2010 IEEE. (6 refs.)

Main Heading: Computer software selection and evaluation

Controlled terms: Communication - Computer graphics - Consumer electronics - Digital television - High definition television - Image communication systems - Motion Picture Experts Group standards - Multimedia systems - Telecommunication systems - Television broadcasting - Television networks - Video cameras - Visual communication

Uncontrolled terms: Digital video cameras - Encoder/decoder - Error recovery - Frames per seconds - H.264/AVC - High-speed - IP communications - Low delay - Macro block - Parallel processing - Software-based - Video communications

Classification Code: 723.2 Data Processing and Image Processing - 723.5 Computer Applications - 741.1 Light/Optics - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory - 742.1 Photography - 912.2 Management - 913 Production Planning and Control; Manufacturing - 742.2 Photographic Equipment - 723 Computer Software, Data Handling and Applications - 722.3 Data Communication, Equipment and Techniques - 715 Electronic Equipment, General Purpose and Industrial - 716 Telecommunication; Radar, Radio and Television - 716.4 Television Systems and Equipment - 717 Optical Communication - 717.1 Optical Communication Systems - 718 Telephone Systems and Related Technologies; Line Communications

Database: Compendex

66. **7th Edition of the French-Speaking Conference on Software Architecture, CAL 2013 (7th Edition of the French-Speaking Conference on Software Architecture, CAL 2013)**
Source: *CAL 2013 - 7eme Edition de la Conference Francophone sur les Architectures Logicielles*, 2013, *CAL 2013 - 7eme Edition de la Conference Francophone sur les Architectures Logicielles*; **Language:** French; **Conference:** 7eme Conference Francophone sur les Architectures Logicielles, CAL 2013 - 7th French-Speaking Conference on Software Architecture, CAL 2013, May 30, 2013 - May 31, 2013; **Publisher:** Revue des Nouvelles Technologies de l'Information (RNTI)

Abstract: The proceedings contain 11 papers. The topics discussed include: IMOCA: a mode-based architecture of operation for control application in an uncertain environment; Kalimucho: software platform distributed by applications supervision; modeling interactive scientific visualization applications; UML extension by XML schema profiles: designing a profile for reference data management; formal verification of compound, transactional and dynamic web services; evaluating the quality of an information system as a whole; validation of systems using software architecture: the test-based approach; comparative study of the cloud computing monitoring tools; towards a diagnostic approach for context-sensitive M2M architectures; and space exploration of architectural configurations guided by the QoS of publish-subscribe systems on MANET.

Database: Compendex

67. **Visualization of large medical data sets using memory-optimized CPU and GPU algorithms**
Kiefer, Gundolf[1]; Lehmann, Heiko[1]; Weese, Jürgen[1] **Source:** *Progress in Biomedical Optics and Imaging - Proceedings of SPIE*, v 5744, n II, p 677-687, 2005, *Medical Imaging 2005 - Visualization, Image-Guided Procedures, and Display*; **ISSN:** 16057422; **DOI:** 10.1117/12.595025; **Article number:** 76; **Conference:** Medical Imaging 2005 - Visualization, Image-Guided Procedures, and Display, February 13, 2005 - February 15, 2005; **Sponsor:** SPIE; **Publisher:** SPIE

Author affiliation:
[1] Philips Research Laboratories, Aachen, Germany

Abstract: With the evolution of medical scanners towards higher spatial resolutions, the sizes of image data sets are increasing rapidly. To profit from the higher resolution in medical applications such as 3D-angiography for a more efficient and precise diagnosis, high-performance visualization is essential. However, to make sure that the performance of a volume rendering algorithm scales with the performance of future computer architectures, technology trends need to be considered. The design of such scalable volume rendering algorithms remains challenging. One of the major trends in the

development of computer architectures is the wider use of cache memory hierarchies to bridge the growing gap between the faster evolving processing power and the slower evolving memory access speed. In this paper we propose ways to exploit the standard PC's cache memories supporting the main processors (CPU's) and the graphics hardware (graphics processing unit, GPU), respectively, for computing Maximum Intensity Projections (MIPs). To this end, we describe a generic and flexible way to improve the cache efficiency of software ray casting algorithms and show by means of cache simulations, that it enables cache miss rates close to the theoretical optimum. For GPUbased rendering we propose a similar, brick-based technique to optimize the utilization of onboard caches and the transfer of data to the GPU on-board memory. All algorithms produce images of identical quality, which enables us to compare the performance of their implementations in a fair way without eventually trading quality for speed. Our comparison indicates that the proposed methods perform superior, in particular for large data sets. (34 refs.)

Main Heading: Algorithms

Controlled terms: Angiography - Computer architecture - Diagnosis - Medical imaging - Program processors - Visualization

Uncontrolled terms: 3D angiography - Graphics accelerators - Maximum intensity projection - Volume visualization

Classification Code: 461.1 Biomedical Engineering - 461.6 Medicine and Pharmacology - 722 Computer Systems and Equipment - 723 Computer Software, Data Handling and Applications - 723.1 Computer Programming

Treatment: Theoretical (THR)

Database: Compendex

68. **NRC-IRC computer controlled acoustic measurement and quality system**

Estabrooks, Timothy[1] **Source:** *Canadian Acoustics - Acoustique Canadienne*, v 37, n 3, p 46-47, September 2009; **ISSN:** 07116659; **Publisher:** Canadian Acoustical Association

Author affiliation:
[1] National Research Council, 1200 Montreal Road, Ottawa, ON Kl A 0R6, Canada

Abstract: The architecture, development, and functionality of acoustics software developed at the National Research Council Canada is described. The software incorporates methods to formalize data verification and data approval for quality control along with performing data storage functions. The software has been designed to allow the user to access various data results quickly and efficiently, while providing a quick visual reference to users. The use of the software helps the user to review any test result or the raw data from which

result has been derived. All calculated or measured values are stored in an Oracle database, while the raw unprocessed data are stored on a secure backed-up disk system. The system design includes a relational database with the capability of linking the contract, specimens, and test data together.

Main Heading: Computer software selection and evaluation

Controlled terms: Acoustics - Computer control systems - Quality control - Quality function deployment - Software architecture - Total quality management - Verification

Uncontrolled terms: Acoustic measurements - Data storage function - Data verification - Disk systems - National Research Council - Oracle database - Quality systems - Relational Database - System design - Test data - Test results - Visual reference

Classification Code: 913.3 Quality Assurance and Control - 912.2 Management - 751 Acoustics, Noise. Sound - 731.1 Control Systems - 922.2 Mathematical Statistics - 723.5 Computer Applications - 723 Computer Software, Data Handling and Applications - 721.2 Logic Elements - 721.1 Computer Theory, Includes Formal Logic, Automata Theory, Switching Theory, Programming Theory - 723.1 Computer Programming

Database: Compendex

69. **Lower cost spatially immersive visualization for human environments**

Parke, Frederic I.[1, 2, 3] **Source:** *Landscape and Urban Planning*, v 73, n 2-3, p 234-243, October 15, 2005, *Research on the Built and Virtual Environments*; **ISSN:** 01692046; **DOI:** 10.1016/j.landurbplan.2004.11.009; **Publisher:** Elsevier

Author affiliation:
1 Visualization Sciences, College of Architecture, Texas A and M University, 216A Langford, College Station, TX 77843-3137, United States
2 Visualization Sciences Faculty, Department of Architecture, Texas A and M University
3 Texas A and M College of Architecture

Abstract: Access to computer simulation and computer based visualization has dramatically impacted our ability to design complex human environments. Immersive visualization, with its ability to present high quality interactive three-dimensional representations of environments, is the next step in this evolution. This paper explores the development of lower cost modular immersive visualization systems as a way to extend and augment our ability to plan, design and evaluate human environments. Technology now available enables spatially immersive visualization systems created using off the shelf components including high performance, relatively inexpensive, commodity computers, inexpensive commodity projectors and open source software. Flexible modular configurations utilizing polyhedral display surfaces with

many identical modular components and networked visual computer clusters is one approach to such systems. Work is underway at the Texas A&M College of Architecture focused on developing and evaluating several prototypes of this new class of systems to determine their practicality and effectiveness. Underlying concepts, issues and trade-offs related to the design and development of these systems are presented. Initial applications using these systems in human environment planning, design and evaluation are discussed. (21 refs.)

Main Heading: Virtual reality

Controlled terms: Computer simulation - Cost effectiveness - Environmental engineering - Sustainable development - Visualization

Uncontrolled terms: Modular visualization systems - Parallel visual computing - Spatially immersive visualization - Virtual environments - Visualizing human environments

Classification Code: 454 Environmental Engineering - 723 Computer Software, Data Handling and Applications - 723.5 Computer Applications - 901.4 Impact of Technology on Society - 911.2 Industrial Economics

Treatment: Theoretical (THR)

Database: Compendex

70. **Architectures for Adaptive Software Systems - 5th International Conference on the Quality of Software Architectures, QoSA 2009, Proceedings**
Source: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 5581 LNCS, 2009, *Architectures for Adaptive Software Systems - 5th International Conference on the Quality of Software Architectures, QoSA 2009, Proceedings*; **ISSN:** 03029743, **E-ISSN:** 16113349; **ISBN-10:** 3642023509, **ISBN-13:** 9783642023507; **Conference:** 5th International Conference on the Quality of Software Architectures, QoSA 2009, June 24, 2009 - June 26, 2009; **Publisher:** Springer Verlag

Abstract: The proceedings contain 13 papers. The topics discussed include: a model-based framework to design and debug safe component-based autonomic systems; automated architecture consistency checking for model driven software development; improved feedback for architectural performance prediction using software cartography visualizations; predicting performance properties for open systems with KAMI; compositional prediction of timed behavior for process control architecture; timed simulation of extended AADL-based architecture specifications with timed abstract state machines; achieving agility through architecture visibility; toward a catalogue of architectural bad smells; on the consolidation of data-centers with performance constraints; and adaptive application composition in quantum chemistry.

Database: Compendex

71. **Best practice for achieving real-time virtual reality architectural visualization: Achieving virtual reality presentation performance efficiency**

Chung, Daniel Hii Jun[1] **Source:** *CAADRIA 2007 - The Association for Computer-Aided Architectural Design Research in Asia: Digitization and Globalization*, p 689-691, 2007, *CAADRIA 2007 - The Association for Computer-Aided Architectural Design Research in Asia: Digitization and Globalization*; **Conference:** 12th International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA 2007, April 19, 2007 - April 21, 2007; **Publisher:** The Association for Computer-Aided Architectural Design

Author affiliation:
[1] Department of Architecture, School of Design and Environment, National University of Singapore, Singapore

Abstract: Real-time rendering is the most fundamental aspect of the virtual reality technology, which is always, a balance between hardware advancement and efficient software algorithm upgrades on the one hand and the demands of the quality output on the other. As architects who utilize this technology, what we can do as users is to optimize the parameters we can control in the files we input to run the simulation. Thus the identification of the few variables we can control to prevent a lagging performance or ghosting which is no longer realtime. The frame rate above all is the measurement to determine whether a simulation is real-time or not. The research aim is to identify the biggest contributing factors among the few identified. The intention is to save time and effort by optimizing the selected factors in projects so that we do not have to optimize all of them. The workflow will also be a lot faster and projects can be done efficiently. The final part is to run satisfaction tests among users to evaluate those biggest contributing factors and the qualities acceptable by users. This is to eliminate the added processing power needed to run bigger files if they do not contribute much to improved quality. (13 refs.)

Main Heading: Quality control

Controlled terms: Architectural design - Optimization - Research - Virtual reality

Uncontrolled terms: Best practices - Contributing factor - Frame rate - Performance efficiency - Processing power - Real-time rendering - Software algorithms - Virtual reality technology

Classification Code: 402 Buildings and Towers - 723 Computer Software, Data Handling and Applications - 901.3 Engineering Research - 913.3 Quality Assurance and Control - 921.5 Optimization Techniques

Database: Compendex

72. **Plugging-in Visualization: Experiences Integrating a Visualization Tool with Eclipse**

Lintern, Rob[1]; Michaud, Jeff[1]; Storey, Margaret-Anne[1]; Wu, Xiaomin[1]
Source: *Proceedings of ACM Symposium on Software Visualization*, p 47-56, 2003, *Proceedings of the 2003 ACM Symposium on Software Visualization, SoftVis '03*; **ISBN-10:** 1581136420, **ISBN-13:** 9781581136425; **DOI:** 10.1145/774833.774840; **Conference:** Proceedings of the ACM 2003 Symposium on Software Visualization (SoftVis 2003), June 11, 2003 - June 13, 2003; **Sponsor:** ACM SIGCHI; **Publisher:** Association for Computing Machinery

Author affiliation:
[1] Dept. of Computer Science, University of Victoria, Victoria, BC, Canada

Abstract: The Eclipse platform presents an opportunity to openly collaborate and share visualization tools amongst the research community and with developers. In this paper, we present our own experiences of "plugging-in" our visualization tool, SHriMP Views, into this environment. The Eclipse platform's Java Development Tools (JDT) and CVS plug-ins provide us with invaluable information on software artifacts relieving us from the burden of creating this functionality from scratch. This allows us to focus our efforts on the quality of our visualizations and, as our tool is now part of a full-featured Java IDE, gives us greater opportunities to evaluate our visualizations. The integration process required us to re-think some of our tool's architecture, strengthening its ability to be plugged into other environments. We step through a real-life scenario, using our newly integrated tool to aid us in merging of two branches of source code. Finally we detail some of the issues we have encountered in this integration and provide recommendations for other developers of visualization tools considering integration with the Eclipse platform. (30 refs.)

Main Heading: Computer aided software engineering

Controlled terms: Codes (symbols) - Graphical user interfaces - Java programming language

Uncontrolled terms: Software visualization

Classification Code: 722.2 Computer Peripheral Equipment - 723.1 Computer Programming - 723.1.1 Computer Programming Languages - 723.2 Data Processing and Image Processing - 723.5 Computer Applications

Treatment: Theoretical (THR)

Database: Compendex

73. **The ALICE data quality monitoring**

Von Haller, B.[1]; Roukoutakis, F.[1, 3]; Chapeland, S.[1]; Altini, V.[2]; Carena, F.[1];

Carena, W.[1]; Chibante Barroso, V.[1]; Costa, F.[1]; Divià, R.[1]; Fuchs, U.[1]; Makhlyueva, I.[1, 4]; Schossmaier, K.[1]; Soós, C.[1]; Vande Vyvre, P.[1] **Source:** *Journal of Physics: Conference Series*, v 219, n 1 PART 2, 2010, *17th International Conference on Computing in High Energy and Nuclear Physics, CHEP09*; **ISSN:** 17426588, **E-ISSN:** 17426596; **DOI:** 10.1088/1742-6596/219/2/022023; **Article number:** 022023; **Conference:** 17th International Conference on Computing in High Energy and Nuclear Physics, CHEP 2009, March 21, 2009 - March 27, 2009; **Publisher:** Institute of Physics Publishing

Author affiliation:
[1] CERN, Physics Department, Geneva, Switzerland
[2] INFN, Dipartimento di Fisica dell'Università, Sezione INFN, Bari, Italy
[3] University of Athens, Physics Department, Athens, Greece
[4] ITEP, Institute for Theoretical and Experimental Physics, Moscow, Russia

Abstract: ALICE is one of the four experiments installed at the CERN Large Hadron Collider (LHC), especially designed for the study of heavy-ion collisions. The online Data Quality Monitoring (DQM) is an important part of the data acquisition (DAQ) software. It involves the online gathering, the analysis by user-defined algorithms and the visualization of monitored data. This paper presents the final design, as well as the latest and coming features, of the ALICE's specific DQM software called AMORE (Automatic MonitoRing Environment). It describes the challenges we faced during its implementation, including the performances issues, and how we tested and handled them, in particular by using a scalable and robust publish-subscribe architecture.We also review the on-going and increasing adoption of this tool amongst the ALICE collaboration and the measures taken to develop, in synergy with their respective teams, efficient monitoring modules for the sub-detectors. The related packaging and release procedure needed by such a distributed framework is also described. We finally overview the wide range of usages people make of this framework, and we review our own experience, before and during the LHC start-up, when monitoring the data quality on both the sub-detectors and the DAQ side in a real-world and challenging environment. (11 refs.)

Main Heading: Data visualization

Controlled terms: Computer software - Elementary particles - Nuclear physics - Visualization

Uncontrolled terms: Automatic monitoring - Data quality - Distributed framework - Efficient monitoring - Heavy-ion collisions - Large Hadron collider LHC - Online data - Publish-subscribe - Real-world - Start-ups

Classification Code: 932.2 Nuclear Physics - 931.3 Atomic and Molecular Physics - 921.4 Combinatorial Mathematics, Includes Graph Theory, Set Theory - 902.1 Engineering Graphics - 723.5 Computer Applications - 723.2 Data Processing and Image Processing - 723 Computer Software, Data Handling and Applications

Database: Compendex

74. **Reducing the effort of building object-oriented visualizations**

Orosco, Ricardo[1]; Moriyon, Roberto[1] **Source:** *Proceedings - IEEE Computer Society's International Computer Software and Applications Conference*, p 568-573, 1998; **ISSN:** 07303157; **ISBN-10:** 0818685859; **DOI:** 10.1109/CMPSAC.1998.716720; **Conference:** Proceedings of the 1998 IEEE 22nd Annual International Computer Software & Applications Conference, August 19, 1998 - August 21, 1998; **Sponsor:** IEEE; **Publisher:** IEEE Comp Soc

Author affiliation:
[1] Univ Autonoma de Madrid, Madrid, Spain

Abstract: The construction of visualization systems is a difficult task. The management of large quantities of data and the support of dynamic exploration processes are some of the main causes of this difficulty. However, current approaches are far away from the system-designer needs. Automatic construction systems, toolkits and dataflow-based systems do not considerate all quality aspects such as flexibility, extensibility, modularity, reusability and generality. In this work, the design of a software architecture for construction of visualization systems, called Telescope, is described. This architecture allows building of visualization applications in different domains, as exemplified by the several developed applications. Also, it allows the incorporation of new tasks, algorithms and data types, and the specification of presentations in a declarative way. The development of applications using Telescope is described, along with a system developed with it, WarVis, for visualizing information about military conflicts. (19 refs.)

Main Heading: Object oriented programming

Controlled terms: Algorithms - Computer architecture - Computer software - Visualization

Uncontrolled terms: Information visualization systems - Object oriented visualizations

Classification Code: 723.1 Computer Programming

Treatment: General review (GEN)

Database: Compendex

75. **Visualization of work flow to support lean construction**

Sacks, R.[1]; Treckmann, M.[2]; Rozenfeld, O.[3] **Source:** *Journal of Construction Engineering and Management*, v 135, n 12, p 1307-1315, 2009; **ISSN:** 07339364; **DOI:** 10.1061/(ASCE)CO.1943-7862.0000102; **Publisher:** American Society of Civil Engineers

Author affiliation:
[1] Technion-Israel Institute of Technology, Faculty of Civil and Environmental Engineering, Haifa 32000, Israel
[2] Ruhr-Univ. Bochum, Faculty of Civil and Environmental Engineering, Bochum, Germany
[3] Technion-Israel Institute of Technology Deputy Chief Engineer, Plaza Centers, Budapest, Hungary

Abstract: Implementation of advanced production management techniques, such as lean construction concepts like filtering of work packages to stabilize work flows, pull flow of teams and materials, and in-process quality control, demands effective and timely flows of information both to and from the workface. The key requirement-making the process state transparent to all participants-is more difficult to achieve in construction than in manufacturing, because work crews move continuously within a physical environment that is itself changing. Novel computer-aided visualization tools can fulfill the needs that simpler tools, such as Kanban cards, fulfill in manufacturing. Two prototypes with user interfaces designed to facilitate process flow have been devised and implemented within the context of building information modeling (BIM) software systems. They demonstrate aspects of the synergy between BIM and lean construction. Given the dynamic and dispersed physical environments and the fractured contracting arrangements typical of construction, BIM-based visualization interfaces are important tools for providing process transparency. © 2009 ASCE. (38 refs.)

Main Heading: Architectural design

Controlled terms: Computer aided design - Computer aided manufacturing - Imaging techniques - Industrial management - Project management - Quality function deployment - Surface chemistry - Total quality management - User interfaces - Visualization

Uncontrolled terms: Building information modeling - Construction management - In-process - Lean construction - Physical environments - Process flows - Process state - Production management - Software systems - Visualization tools - Work packages - Work-flows

Classification Code: 922.2 Mathematical Statistics - 913.4.2 Computer Aided Manufacturing - 913.3 Quality Assurance and Control - 912.2 Management - 902.1 Engineering Graphics - 801.4 Physical Chemistry - 746 Imaging Techniques - 741 Light, Optics and Optical Devices - 723.5 Computer Applications - 723.2 Data Processing and Image Processing - 722.2 Computer Peripheral Equipment - 408.1 Structural Design, General - 402 Buildings and Towers

Database: Compendex

76. **Towards architecture model based deployment for dynamic grid services**

Huang, Gang[1]; Wang, Meng[1]; Ma, Liya[1]; Lan, Ling[1]; Liu, Tiancheng[1]; Mei, Hong[1] **Source:** *Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business, CEC-East 2004*, p 14-21, 2004, *Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business, CEC-East 2004*; **ISBN-10:** 0769522068, **ISBN-13:** 9780769522067; **Conference:** Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business, CEC-East 2004, September 13, 2004 - September 15, 2004; **Sponsor:** IEEE Computer Society Technical Committee on Electronic Commerce; **Publisher:** IEEE Computer Society

Author affiliation:
[1] Sch. Electronics Eng. and Comp. Sci., Peking University, Beijing, 100871, China

Abstract: The deployment of grid services should make the services, including those to be deployed and those already deployed, operate with desired functionalities and qualities. The critical challenge in the deployment is that many technical and non-technical factors have to be taken into account, such as performance, reliability, utilization, operating cost, incomes, and so on. Since the factors change continuously, some deployed services may have to be re-deployed for guaranteeing their functionalities and qualities. This position paper presents an approach to the deployment and re-deployment of grid services based on software architecture models. In this approach, all services in a grid consist in a software architecture, which represents the services, their relationships and other factors in a global, understandable and easy-to-use way. To demonstrate the approach, a visual tool for deploying services onto a set of popular grid infrastructures, including J2EE application servers and BPEL engines, with the help of software architectures is developed. (28 refs.)

Main Heading: Computer science

Controlled terms: Automation - Computer architecture - Computer software - Costs - Electronic communities - Quality of service - Reliability

Uncontrolled terms: Grid services - Operating cost - Software architectures - Virtual organization (VO)

Classification Code: 723 Computer Software, Data Handling and Applications - 723.5 Computer Applications - 731 Automatic Control Principles and Applications - 911 Cost and Value Engineering; Industrial Economics - 922.2 Mathematical Statistics

Treatment: Theoretical (THR)

Database: Compendex