Literatur

[AnidoPB02a] Manuel Lois Anido, Alexander Paar, and Nader Bagherzadeh. A novel method for improving the operation autonomy of SIMD processing elements. In *SBCCI '02: Proceedings of the 15th Symposium on Integrated Circuits and Systems Design*, volume 0, pages 49–54, Los Alamitos, CA, USA, 09 2002. IEEE Computer Society, IEEE Computer Society.

> **Abstract:** A novel method for improving the operation autonomy of the processing elements (PE) of SIMD-like machines is presented and discussed in this paper. The paper shows that it is feasible to avoid most branches and it is also possible to emulate nested if-then-else sentences on the processing elements by combining guarded instructions and pseudo branches. This prevents unnecessary intervention by the array control unit in many data-dependant computations, particularly those with short branch sections. The paper also shows that the simplicity of the method allows it to be implemented both in fine-grain and coarse-grain SIMD/ASIMD architectures because it does not require significant additional silicon area. Finally, it is shown that pseudo branches can be used to control the power saving of those processing elements that have instructions nullified.

[AnidoPB02b] Manuel Lois Anido, Alexander Paar, and Nader Bagherzadeh. Improving the operation autonomy of SIMD processing elements by using guarded instructions and pseudo branches. In Martyn Edwards, editor, *DSD '02: Proceedings of the Euromicro Symposium on Digital Systems Design*, volume 0, page 148, Washington, DC, USA, 09 2002. IEEE Computer Society, IEEE Computer Society.

> **Abstract:** This paper presents a novel method for improving the operation autonomy of the processing elements (PE) of SIMD-like machines. By combining guarded instructions and pseudo branches it is possible to achieve higher operation autonomy and higher instruction level parallelism than in previous SIMD/ASIMD architectures. The paper shows that it is

feasible to avoid most branches and it is also possible to emulate conditional execution on the processing elements, either by using guarded instructions or by using pseudo branches, thus avoiding unnecessary intervention by the array control unit in data-dependant computations. Pseudo branches are used when it is not possible to use guarded instructions. Additionally, they alsosupport the implementation of complex nested if-then-else constructs, improving the execution of irregular data-parallel applications. The paper also shows that the simplicity of the method allows it to be implemented both in fine-grain and coarse-grain SIMD/ASIMD architectures because it does not require significant additional silicon area. Finally, it is shown that pseudo branches can be used to control the power saving of those processing elements that have instructions nullified.

[CHILKBS] Alexander Paar. CHIL Knowledge Base Server. http://www.alexpaar.de/zhimantic/chilkbs/, 2008.

[FeldbuschPOI02] Fridtjof Feldbusch, Alexander Paar, Manuel Odendahl, and Ivan Ivanov. A Bluetooth remote control system. In Hartmut Schmeck, Theo Ungerer, and Lars C. Wolf, editors, *ARCS '02: Proceedings of the International Conference on Architecture of Computing Systems*, volume 2299/2002 of *Lecture Notes in Computer Science*, pages 241–255, Berlin/Heidelberg, Germany, 04 2002. Springer-Verlag.

Abstract: Emerging radio technologies like WLAN and Bluetooth enable electronic devices of any kind to communicate with one another. A simple and easy to implement application layer protocol called BTRC protocol was developed allowing devices to exchange data of any kind and format over different protocols like TCP/IP or Bluetooth. Based upon this protocol a universal remote control system was implemented. Software applications simulating cellular phones and personal digital assistants (PDA) were developed as remote control devices. BTRC server devices send their graphical XML based user interface to the remote control. This way the use of devices is simplified significantly.

[FeldbuschPOI03] Fridtjof Feldbusch, Alexander Paar, Manuel Odendahl, and Ivan Ivanov. The BTRC Bluetooth remote control system. *Personal Ubiquitous Computing*, 7(2):102–112, 07 2003.

> **Abstract:** Emerging radio technologies like WLAN and Bluetooth enable electronic devices of any kind to communicate with one another. A simple and easy to implement application layer protocol called BTRC protocol was developed allowing devices to exchange data of any kind and format over different protocols like TCP/IP or Bluetooth. Based upon this protocol a universal remote control system was implemented. Software applications simulating cellular phones and personal digital assistants (PDA) were developed as remote control devices. BTRC server devices send their graphical XML based user interface to the remote control. In this way, the use of household devices is simplified significantly.

[Paar03] Alexander Paar. Semantic software engineering tools. In *OOPSLA '03: Companion of the 18th Annual ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications*, pages 90–91, New York, NY, USA, 10 2003. ACM, ACM.

> **Abstract:** Recently, the paradigm of software engineering has shifted significantly to service orientation based on Web services. Web Services Description Language interface specifications provide sufficient information to physically access a service. However, these interface descriptions are semantically bleak. This work introduces a number of tools, which were developed to augment strict syntactic service descriptions with semantic information in order to elucidate the meaning of processed data and provided functionality. Semantic Web technologies such as DAML+OIL were supplemented with natural language support for usability improvements both at design- and at runtime.

[Paar07] Alexander Paar. Zhi# – programming language inherent support for ontologies. In Jean-Marie Favre, Dragan Gašević, Ralf Lämmel, and Andreas Winter, editors, *ateM '07: Proceedings of the 4th International Workshop on Software Language Engineering*, number 4/2007 in Mainzer Informatik-Berichte, pages

165–181, Mainz, Germany, 10 2007. Johannes Gutenberg Universität Mainz.

Abstract: XML Schema Definition (XSD) and the Web Ontology Language (OWL) have been widely used to define programming language independent data types and to conceptualize knowledge. However, writing software that operates on XML instance documents and on ontological knowledge bases still suffers from a lack of compile time support for XSD and OWL. In this paper, a novel compiler framework is presented that facilitates the cooperative usage of external type systems with C#. For the resulting programming language Zhi#, XSD and OWL compiler plug-ins were implemented in order to provide static type checking for constrained atomic value types and ontologies. XSD constraining facets and ontological inference rules could be integrated with host language features such as method overwriting. Zhi# programs are compiled to conventional C# and are interoperable with .NET assemblies.

[Paar10] Alexander Paar. Searching and using external types in an extensible software development environment. In *SUITE '10: Proceedings of the 2010 ICSE Workshop on Search-driven development: Users, Infrastructure, Tools and Evaluation*, pages 37–40, New York, NY, USA, 05 2010. ACM.

Abstract: Schema and ontology languages have proved to be useful for conceptualizing knowledge in a variety of applications. In many software projects, XML Schema Definition data types and ontological concept descriptions coexist with programming language class hierarchies. However, only programming language type definitions are fully integrated into today's software development environments. Support for external type systems is spotty. For programmers, it is particularly tedious to search type definitions in XML schema files and OWL ontologies, to browse external type hierarchies, to investigate external type members, and to analyze and comprehend the use of external type definitions in program code. In this work, it will be argued that improved search capabilities are required to ease the use of schema and ontology languages in software projects. Difficulties of searching type definitions

in software project workspaces will be indicated. An extensible compiler framework will be outlined that facilitates the use of schema and ontology languages in C# programs. An Eclipse-based integrated development environment will be described that makes XML data types and OWL concept descriptions first-class citizens of the source code editor. Finally, identical search and (just in time) program analysis features for programming language and external type definitions will be suggested.

[Paar12a] Alexander Paar. Position statement. https://sms.west.uni-koblenz.de/index.php/Main_Page, 05 2012.

[Paar12b] Alexander Paar. From program execution to automatic reasoning: Integrating ontologies into programming languages (keynote). In Alberto Simões, Ricardo Queirós, and Daniela Carneiro da Cruz, editors, *SLATE '12: Procededings of the 1st Symposium on Languages, Applications and Technologies*, volume 21 of *OpenAccess Series in Informatics (OASIcs)*, pages 5–5, Dagstuhl, Germany, 2012. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik.

Abstract: Since their standardizations by the W3C, the Extensible Markup Language (XML) and XML Schema Definition (XSD) have been widely adopted as a format to describe data and to define programming language agnostic data types and content models. Several other W3C standards such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL) are based on XML and XSD. At the same time, statically typed object-oriented programming languages such as Java and C# are most widely used for software development. This talk will delineate the conceptual bases of XML Schema Definition and the Web Ontology Language and how they differ from Java or C#. It will be shown how XSD facilitates the definition of data types based on value space constraints and how OWL ontologies are amenable to automatic reasoning. The superior modeling features of XSD and OWL will be elucidated based on exemplary comparisons with frame logic-based models. A significant shortcoming will become obvious: the deficient integration of XSD and OWL with the type systems of object-oriented programming languages. Eventual-

ly, the Zhi# approach will be presented that integrates XSD and OWL into the C# programming language. In Zhi#, value space-based data types and ontological concept descriptions are first-class citizens; compile time and runtime support is readily available for XSD and OWL. Thus, the execution of Zhi# programs is directly controlled by the artificial intelligence inherent in ontological models: Zhi# programs don't just execute, they reason.

[Paar12c] Alexander Paar. *Innovations in XML Applications and Metadata Management: Advancing Technologies*, chapter Foreword, pages xiii–xiv. IGI Global, 2012.

[PaarAB02] Alexander Paar, Manuel Lois Anido, and Nader Bagherzadeh. A novel predication scheme for a SIMD system-on-chip. In Burkhard Monien and Rainer Feldmann, editors, *Euro-Par '02: Proceedings of the 8th International Euro-Par Conference on Parallel Processing*, volume 0, pages 834–843, London, UK, 08 2002. Springer-Verlag.

Abstract: This paper presents a novel predication scheme that was applied to a SIMD system-on-chip. This approach was devised by improving and combining the unrestricted predication model and the guarded execution model. It is shown that significant execution autonomy is added to the SIMD processing elements and that the code size is reduced considerably. Finally, the implemented predication scheme is compared with predication schemes of general purpose processors, and it is shown that it enables more efficient if-conversion compilations than previous architectures.

[PaarDB03] Alexander Paar, Haitao Du, and Nader Bagherzadeh. A component oriented simulator for HW/SW co-designs. In Gerhard Fohler and Radu Marculescu, editors, *ESTImedia '03: Proceedings of the 1st Workshop on Embedded Systems for Real-Time Multimedia*, volume 0, pages 79–86, 10 2003.

Abstract: In order to extensively explore design space one has to specify a system on a very abstract level. Transforming a specification into a correct implementation is usually an error prone task. Moreover, one may want to specify a system on different abstraction levels. This work introduces a component orien-

ted simulator approach that comprises AsmL based executable specifications. Test cases are automatically generated from the specification. Concurrent verification techniques are used to run the specification in parallel with the implementation. Feasibility of such rapid specifying techniques is proved by simulating a 3D graphics engine for the UC Irvine MorphoSys SIMD system-on-chip.

[PaarDiplomarbeit] Alexander Paar. A novel predication scheme for a SIMD system-on-chip developed with a component oriented simulator. Diplomarbeit, Universität Karlsruhe (TH), Am Fasanengarten 5, 76128 Karlsruhe, Germany, 07 2002.

Abstract: This work introduces a novel predication scheme that was applied to the UC Irvine MorphoSys reconfigurable SIMD system-on-chip. This approach was devised by improving and combining the unrestricted predication model and the guarded execution model. It is shown that significant execution autonomy is added to the MorphoSys SIMD processing elements and that the code size is reduced considerably. The implemented predication scheme is compared with predication schemes of general purpose processors, and it is shown that it enables more efficient if-conversion compilations than previous architectures. These new capabilities to deal efficiently with parallel if-then-else clauses are evaluated by an implementation of a 3D graphics engine. The predication enhanced MorphoSys SIMD architecture is specified and simulated using a new component based simulator approach that incorporates mathematical rigorous abstract state machines to specify the behavior of both SIMD instruction sequences and the resulting predicated SIMD array and its processing elements. Finally, the specified system is implemented in VHDL to show its technological feasibility.

[PaarDissertation] Alexander Paar. *Zhi# – Programming Language Inherent Support for Ontologies.* PhD thesis, Universität Karlsruhe (TH), Am Fasanengarten 5, 76137 Karlsruhe, Germany, 07 2009.

[PaarG11] Alexander Paar and Stefan Gruner. Static typing with value space-based subtyping. In *SAICSIT '11: Proceedings of the South African Institute of Computer Scientists and Information*

Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment, SAICSIT '11, pages 177–186, New York, NY, USA, 10 2011. ACM.

Abstract: Numerous programming and schema languages contain the notion of value types. However, support for value space-based subtyping is spotty. This paper presents a formal type system for atomic value types as an extension of the simply typed lambda calculus with subtyping. In the λ_C-calculus, value types can be derived through the application of value space constraints. Type inference rules can be used to infer transient value space constraints that hold for limited scopes of a program. The type system of the λ_C-calculus is proved to be sound. The λ_C-calculus was fully implemented in Java and C#. The presented approach was successfully validated to subsume XML Schema Definition type construction and to facilitate the integration of XSD data types with the C# programming language.

[PaarR09] Alexander Paar and Jürgen Reuter. *Computers in the Human Interaction Loop*, chapter Ontological Modeling and Reasoning, pages 325–340. Human-Computer Interaction Series. Springer Verlag London, 04 2009.

Abstract: A major challenge in putting the CHIL vision into practice was to integrate contributions from project partners all over Europe and the United States. While standard Web technologies such as HTTP support core data exchange, perceptual component integration places higher demands on distributed intercommunication. Interfaces have to be defined for the type-safe exchange of structured data between components that are implemented in diverse computer languages.

[PaarRSSP06] Alexander Paar, Jürgen Reuter, John Soldatos, Kostas Stamatis, and Lazaros Polymenakos. A formally specified ontology management API as a registry for ubiquitous computing systems. In Ilias Maglogiannis, Kostas Karpouzis, and Max Bramer, editors, *AIAI '06: Proceedings of the 3rd IFIP Conference on Artificial Intelligence Applications and Innovations*, volume 204/2006, pages 137–146, Boston, MA, USA, 06 2006. IFIP

International Federation for Information Processing, Springer-Verlag.

Abstract: Recently, several standards have emerged for ontology markup languages that can be used to formalize all kinds of knowledge. However, there are no widely accepted standards yet that define APIs to manage ontological data. Processing ontological information still suffers from the heterogeneity imposed by the plethora of available ontology management systems. Moreover, ubiquitous computing environments usually comprise software components written in a variety of different programming languages, which makes it even more difficult to establish a common ontology management API with programming language agnostic semantics. We implemented an ontological Knowledge Base Server, which can expose the functionality of arbitrary off-the-shelf ontology management systems via a formally specified and well defined API. A case study was carried out in order to demonstrate the feasibility of our approach to use an ontological Knowledge Base Server as a registry for ubiquitous computing systems.

[PaarRSSP09] Alexander Paar, Jürgen Reuter, John Soldatos, Kostas Stamatis, and Lazaros Polymenakos. A formally specified ontology management API as a registry for ubiquitous computing systems. *Applied Intelligence*, 30(1):37–46, 02 2009.

Abstract: Recently, several standards have emerged for ontology markup languages that can be used to formalize all kinds of knowledge. However, there are no widely accepted standards yet that define APIs to manage ontological data. Processing ontological information still suffers from the heterogeneity imposed by the plethora of available ontology management systems. Moreover, ubiquitous computing environments usually comprise software components written in a variety of different programming languages, which makes it particularly difficult to establish a common ontology management API with programming language agnostic semantics. We implemented an ontological Knowledge Base Server, which can expose the functionality of arbitrary off-the-shelf ontology management systems via a formally specified and well defined API.

A case study was carried out in order to demonstrate the feasibility of our approach to use a formally specified ontology management API to implement a registry for ubiquitous computing systems.

[PaarRSch05] Alexander Paar, Jürgen Reuter, and Jaron Schaeffer. A pluggable architectural model and a formally specified programming language independent API for an ontological knowledge base server. In Thomas Meyer and Mehmet A. Orgun, editors, *AOW '05: Proceedings of the 2005 Australasian Ontology Workshop*, volume 58 of *CRPIT*, pages 83–91, Darlinghurst, Australia, 12 2005. Australian Computer Society, Inc., Australian Computer Society, Inc.

Abstract: Recently, ontology engineering has become ever more important when it comes to conceptualize knowledge. However, writing software applications that operate on ontological knowledge still suffers from a lack of connectivity provided by available ontology management systems. Interfaces of ontology management systems are either based on error prone programming language agnostic remoting protocols or they are restricted to one particular programming language. We implemented an ontological Knowledge Base Server, which can expose the functionality of arbitrary off-the-shelf ontology management systems via arbitrary remoting protocols. Based on XML Schema Definition, we defined a full-fledged API for processing OWL ontologies. Client access code can be generated automatically for virtually any object oriented programming language. Using Description Logics terminology, the Knowledge Base Server API was formally specified, such that it could be used to validate implementations based on three different adapted ontology management systems.

[PaarSGSch03] Alexander Paar, Gábor Szeder, Tom Gelhausen, and Marc Schanne. Grundlagen des Autonomen Rechnens. Interner Bericht 2003-14, Universität Karlsruhe (TH), Am Fasanengarten 5, 76128 Karlsruhe, Germany, 07 2003.

Abstract: Das vegetative Nervensystem (engl. autonomous nervous system) des Menschen kann das, wovon in der IT-Industrie noch geträumt wird. Abhängig von der aktuellen Umgebung und Tätigkeit

reguliert das vegetative Nervensystem mandatorische Körperfunktionen wie Herzfrequenz und Atmung. Reflexe, die dem Selbstschutz dienen, werden automatisch ausgelöst. Verletzungen heilen von selbst, ohne dass man seine normalen Tätigkeiten dafür unterbrechen müsste. Im Rahmen des Seminars „Autonomic Computing“ im Sommersemester 2003 am Institut für Programmstrukturen und Datenorganisation der Universität Karlsruhe wurden Grundlagen dieses Autonomen Rechnens besprochen. Als Basis für Selbstkonfiguration und Selbstoptimierung werden in „Kontextbewusstsein: Ein Überblick“ Techniken zur Erfassung des physischen und sozialen Kontexts einer Anwendung erläutert. Die dienstorientierte Architektur und konkrete Implementierungen wie z.B. UPnP, Jini oder Bluetooth werden in „Aktuelle Technologien zur Realisierung dienstorientierter Architekturen“ behandelt. Die Arbeit „Service-Orientierung und das Semantic Web“ beschreibt, wie Semantic Web Technologien zur Beschreibung von Web Services verwendet werden können mit dem Ziel der automatischen Dienstfindung. Danach wird der Begriff „Selbstbewusstsein“ in bezug auf Software anhand zweier komplementärer Forschungsprojekte definiert. Technologien zur Überwachung des Laufzeitverhaltens von Rechnersystemen mit dem Ziel der selbstständigen Optimierung sind Gegenstand der Arbeit „Selbst-Überwachung und Selbst-Optimierung“. Der Artikel „Selbst-Schutz“ fasst die Sicherheitsanforderungen zusammen, die an ein autonomes Computersystem gestellt werden müssen und die Techniken, um solche Anforderungen zu erfüllen. Ansätze aus dem Bereich wiederherstellungsorientiertes- und fehlertolerantes Rechnen werden in „Selbst-Heilung“, „ROC – Recovery Oriented Computing“ und „Recovery Oriented Computing: Modularisierung und Redundanz“ vorgestellt. Alle Ausarbeitungen und Präsentationen sind auch elektronisch auf der diesem Band beiliegenden CD oder unter www.autonomic-computing.org verfügbar.

[PaarStudienarbeit] Alexander Paar. Design and implementation of a Bluetooth based ubiquitous remote control system. Studienarbeit, Universität Karlsruhe (TH), Am Fasanengarten 5, 76128 Karls-

ruhe, Germany, 07 2002.

Abstract: In this paper we introduce the BTRC protocol. The Bluetooth Remote Control protocol introduces ubiquitous computing to our every-day life. We catch up here the ideas of the Network Working Group and their Request for Comments to provide a uniform access to any generic object in the web by means of a Uniform Resource Identifier (URI). In the near future of the ubiquitous computing revolution any device occurring in our every-day life will be supplied with the ability to be online – that means connected to the Internet. In that way the Internet will be considered to include objects accessed using an extendable number of protocols. The BTRC protocol is an ISO/OSI layer 7 (application layer) protocol running on any lower level protocol that is capable to convey device specific command sets to certain appliances. To demonstrate the might of BTRC we have realized this protocol in a PDA simulator application, which acts as a universal remote control. This system extends the web concept of uniform access to resources by applying the same access method, Uniform Resource Identifiers, to environment-specific resources, for example for access to the TV. Since these URIs may be embedded within common HTML pages, it is possible to access a real-life device just like a web object. This is of major consequence because the devices itself deliver XML based BTRC GUIs onto the universal remote control display. Further this paper describes how to access legacy (e.g. RC5 infrared controlled) devices. In summary, the BTRC protocol and its implementations institute a new epoch of ubiquitous computing by simplifying and standardizing the way human beings access appliance resources in their real-life environment as well as the way these devices collaborate with each other. The BTRC approach is really pervasive because the BTRC protocol is scalable down to pieces of equipment with even very restricted computing and networking capabilities.

[PaarT03] Alexander Paar and Walter F. Tichy. Semantic software engineering approaches for automatic service lookup and integration. In *AMS '03: Proceedings of the Autonomic Computing Workshop*

5th Workshop on Active Middleware Services, pages 103–110. IEEE Computer Society, 06 2003.

> **Abstract:** Using Web services today has two major drawbacks: firstly, a programmer has to guess the appropriate service operations by interpreting syntactic operation names provided by WSDL descriptions, and secondly, the decision which services to use is fixed at design time. Using ontological descriptions, we automate the lookup of services. The lookup occurs at runtime, supporting location aware selection and the choice of more relevant or alternate services. We use Semantic Web techniques such as DAML-OIL to avoid the exact match between services invocations and make both service lookup and invocation independent of the strict syntactic Web services description.

[PaarT05] Alexander Paar and Walter F. Tichy. Zhi#: Programming language inherent support for XML Schema Definition. In W.-T. Tsai and M.H. Hamza, editors, *SEA '05: Proceedings of the 9th IASTED International Conference on Software Engineering and Applications*, volume 0, pages 407–414, Anaheim, CA, USA, 11 2005. IASTED, ACTA Press.

> **Abstract:** Recently, XML Schema Definition (XSD) has become ever more important when it comes to define programming language independent content models and type systems. However, writing software that operates on XML instance documents still suffers from a lack of compile time support for XSD. Especially, obeying facet based constraints imposed on XSD simple data types is still error prone and laborious. This paper introduces the concept of XSD aware compilation. Zhi# provides static and dynamic type checking for XML simple data types, which can be used along with off-the-shelf implementations of the W3C XML DOM.

[PaarT06] Alexander Paar and Walter F. Tichy. Programming language inherent support for constrained XML Schema Definition data types and OWL DL. In *ASE '06: Proceedings of the 21st IEEE/ACM International Conference on Automated Software Engineering*, pages 281–284, Los Alamitos, CA, USA, 09 2006. IEEE Computer Society, IEEE Computer Society.

Abstract: Recently, the Web Ontology Language (OWL) and XML Schema Definition (XSD) have become ever more important when it comes to conceptualize knowledge and to define programming language independent type systems. However, writing software that operates on ontological data and on XML instance documents still suffers from a lack of compile time support for OWL and XSD. Especially, obeying lexical- and value space constraints that may be imposed on XSD simple data types and preserving the consistency of assertional ontological knowledge is still error prone and laborious. Validating XML instance documents and checking the consistency of ontological knowledge bases according to given XML Schema Definitions and ontological terminologies, respectively, requires significant amounts of code. This paper presents novel compile time- and code generation features, which were implemented as an extension of the C# programming language. Zhi# provides compile time and runtime support for constrained XML Schema Definition simple data types and it guarantees terminological validity for modifications of assertional ontological data.

[PaarV11] Alexander Paar and Denny Vrandečić. Zhi# – OWL aware compilation. In Grigoris Antoniou, Marko Grobelnik, Elena Paslaru Bontas Simperl, Bijan Parsia, Dimitris Plexousakis, Pieter De Leenheer, and Jeff Z. Pan, editors, *ESWC '11: Proceedings of the 8th Extended Semantic Web Conference*, volume 6644 of *Lecture Notes in Computer Science*, pages 315–329. Springer Verlag Berlin, Heidelberg, 06 2011.

Abstract: The usefulness of the Web Ontology Language to describe domains of discourse and to facilitate automatic reasoning services has been widely acknowledged. However, the programmability of ontological knowledge bases is severely impaired by the different conceptual bases of statically typed object-oriented programming languages such as Java and C# and ontology languages such as the Web Ontology Language (OWL). In this work, a novel programming language is presented that integrates OWL and XSD data types with C#. The Zhi# programming language is the first solution of its kind to make XSD data

types and OWL class descriptions first-class citizens of a widely-used programming language. The Zhi# programming language eases the development of Semantic Web applications and facilitates the use and reuse of knowledge in form of ontologies. The presented approach was successfully validated to reduce the number of possible runtime errors compared to the use of XML and OWL APIs.

[PaarWAK16] Stefan Wagner, Asim Abdulkhaleq, Kamer Kaya, and Alexander Paar. On the relationship of inconsistent software clones and faults: An empirical study. In *SANER '16: Proceedings of the 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering*, 2016.

Abstract: Code cloning - copying and reusing pieces of source code – is a common phenomenon in software development in practice. There have been several empirical studies on the effects of cloning, but there are contradictory results regarding the connection of cloning and faults. Objective: Our aim is to clarify the relationship between code clones and faults. In particular, we focus on inconsistent (or type-3) clones in this work. Method: We conducted a case study with TWT GmbH Science & Innovation where we detected the code clones in three Java systems, set them into relation to information from issue tracking and version control and interviewed three key developers. Results: Of the type-3 clones, 17 % contain faults. Developers modified most of the type-3 clones simultaneously and thereby fixed half of the faults in type-3 clones consistently. Type-2 clones with faults all evolved to fixed type-3 clones. Clone length is significantly correlated with faultiness. Conclusion: There are indications that the developers in two cases have been aware of clones. It might be a reason for the weak relationship between type-3 clones and faults. Hence, it seems important to keep developers aware of clones, potentially with new tool support. Future studies need to investigate if the rate of faults in type-3 clones justifies using them as cues in defect detection.

[PandisSPRCP05] Ippokratis Pandis, John Soldatos, Alexander Paar, Jürgen Reuter, Michael Carras, and Lazaros Polymenakos. An ontology-based framework for dynamic resource management in

ubiquitous computing environments. In *ICESS '05: Proceedings of the 2nd International Conference on Embedded Software and Systems*, pages 195–203, Los Alamitos, CA, USA, 12 2005. IEEE Computer Society, IEEE Computer Society.

> **Abstract:** Ubiquitous computing applications are supported by sophisticated middleware components enabling dynamic discovery, invocation and management of resources, as well as reasoning in cases of uncertainty. This paper advocates semantic Web technologies as primary vehicles to achieve dynamic management of resources in ubiquitous computing infrastructures and services. We introduce a framework for implementing ubiquitous computing services comprising a large number of sensors and perceptive interfaces, emphasizing the role of knowledge bases for dynamic registration and invocation of resources. We present the use of ontology-based mechanisms for controlling sensors and actuators. Moreover, we describe the implementation of a knowledge base server that can leverage different ontology management systems, while also exposing a host to different client access interfaces. The introduced framework has been exploited in implementing real prototype ubiquitous computing services, which we also outline in the paper.

[SolmsEPG11] Fritz Solms, Craig Edwards, Alexander Paar, and Stefan Gruner. A domain-specific language for URDAD based requirements elicitation. In *SAICSIT '11: Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment*, SAICSIT '11, pages 224–230, New York, NY, USA, 10 2011. ACM.

> **Abstract:** Use-Case Responsibility-Driven Analysis and Design (URDAD) is a service-oriented software analysis and design methodology. It is used by requirements engineers to develop technology-neutral, semi-formal platform-independent models (PIM) within the OMG's MDA. In the past, URDAD models were denoted in UML. However, that was tedious and error-prone. The resulting models were often of rather poor quality. In this paper we introduce and discuss a new Domain-Specific Language (DSL) for URDAD. Its meta model is consistent and satisfiable. We show

that URDAD DSL specifications are simpler and allow for more complete service contract specifications than their corresponding UML expressions. They also enable traceability and test case generation.

[ZhiSharpToolSuite] Alexander Paar. Zhi# tool suite. http://www.alexpaar.de/zhimantic/zhisharp/, 06 2010.

[ZhiSharpVideoDemos] Alexander Paar. Zhi# video demos. http://www.alexpaar.de/zhimantic/zhisharp/, 06 2008.