Merging Model Driven and Ontology Driven System Development Approaches

Pervasive Computing Perspective

Ahmet Soylu, Patrick De Causmaecker
K.U. Leuven, Interdisciplinary Research on Technology Education and Communication (iTec),
Campus Kortrijk, Etienne Sabbelaan 53, 8500, Kortrijk, Belgium
Ahmet.Soylu@kuleuven-kortrijk.be, Patrick.DeCausmaecker@kuleuven-kortrijk.be

Abstract—**In this paper we present a view point on "intelligent" application development for pervasive computing environments. We first point out that today's traditional "intelligent" computing is built on strong and hard-coded logical assumptions and computational procedures which are pre-defined by developers, that is, what we call as simulated intelligence within the course of this paper. Such assumptions and procedures are based on enumerations of possible contexts of use which is pre-defined mappings between situations in contextual space to rational behaviors in behavior space. However pervasive computing applications extend the scope of application's context space and behavior space towards infinity which hardens development of "intelligent" systems having a certain degree of rationality. Therefore, approaches merging human intelligence and computing "intelligence" are required to be employed. We further advocate that pervasive computing era increases the complexity of application development because of the extended context space, hence software development approaches based on higher abstractions need to be employed where Model Driven approaches and Ontology Driven approaches are promising. We propose a basic methodology which merges Model Driven and Ontology Driven Development approaches. Resulting methodology employs use of formalized conceptual models to be employed both at run-time and development time in terms of reasoning and automatic code generation respectively. We finally point out that such application development paradigm based on pervasive computing perspective will enable users to program their own environment (i.e. smart spaces) in the future.**

Keywords-component; Model Driven Development, Ontology Driven Development, Software Development, Pervasive Computing, Simulated Intelligence, Environment Programming, Smart Spaces, Web of Things.

I. INTRODUCTION

Pervasive computing envisions a digital ecosystem which is seamlessly situated in real life. Such ecosystem is collection of various connected and autonomous devices and applications. Main purpose of such ecosystem is facilitating user life by organizing and utilizing available digital and physical resources, and responding properly and proactively in various settings of use. Although the emergence of pervasive computing goes to early 90s [1], we are still far away from the complete puzzle. It is reasonable to say that in the sense of hardware technologies we already achieved a lot by considering the advancements in the networking technologies, computing power, miniaturization, energy consumption, materials, sensors etc. [2]. However pervasive computing is not just about developing such small computing residents for the real life, variety of applications exploiting such extended hardware infrastructure are the other side of the coin. Application development within this new expanded computing setting requires more attention, since pervasive computing greatly changes the face of the traditional computing.

Pervasive computing settings are highly dynamic, mobile, heterogonous and connected [3]. It is populated with variety of seamlessly integrated devices, applications and different users. We do see available hardware resources through the physical environment simply as a farm of networked resources, and these networked resources are supposed to enable the real world setting to serve as an execution environment for the variety of applications. These computational entities are supposed to work collectively in order provide better service to the users in their vicinity without creating any distraction at the user site. Hence, seamless integration of computing into people's life is a must of the pervasive computing vision. If we don't want people to bother about the computing devices and the applications surrounding them, while they are making use of them, we need to make computing devices and applications to bother about people [3]. However utilizing all these physical resources and synchronizing various applications available through this extended infrastructure requires an "intelligence" behind and requires application development for such environments to be adjusted to cope with increasing complexity. An immediate requirement for such "intelligence" is context-awareness [3, 4] in order to enable applications to perceive the situation and react accordingly. Context-awareness enables pervasive computing environments to better serve users by applying available context information where context is any information characterizing the situation (e.g. characteristics, requirements, capabilities etc.) of an entity (e.g. applications, users, devices etc.) [3, 4, 5]. There is already a research line, in the pervasive computing literature, for context modeling. Although there are different approaches starting from simple key-value pairs, ontologies are promising for context modeling since they provide reasoning support, knowledge sharing and knowledge re-use [6, 7]. However in the pervasive computing domain, potential benefit of automatically generating application code from such models is greatly undermined. Such approach is possible since a typical context model includes most of the application logic, this is because application itself is also an entity in a context model.

This paper is based on research funded by the Industrial Research Fund (IOF) and conducted within the IOF Knowledge platform "Harnessing collective intelligence in order to make e-learning environments adaptive" (IOF KP/07/006).

978-1-4244-5023-7/09/$25.00 ©2009 IEEE

Furthermore ambiguity of context [8, 9, 10] is still a crucial problem where various artificial intelligence (AI) techniques are employed by several researchers. However human intelligence is still required to be in the loop, i.e. meditation [11, 12] , in order to prevent disastrous results of automatic actions taken by machines. Since the ultimate aim is to provide less user intervention, an appropriate balance need to be set.

Through this article we particularly elaborate on computer way of "intelligence" in relation with human intelligence and application development based on Model Driven Development (MDD) and Ontology Driven Development (ODD) approaches. We propose a basic methodology which merges MDD and ODD approaches. Resulting methodology employs use of formalized conceptual models to be employed both at run-time and development time in terms of contextual reasoning and automatic code generation respectively. We finally point out that such abstract application development paradigm based on pervasive computing perspective will enable users to be in the loop by means of mediation and to program their own environment (i.e. environment programming) in the future.

The rest of the paper is structured as follows: in section II we comment on computer "intelligence" in relation with the human intelligence, in section III we elaborate on importance of having human in the loop in the future's pervasive computing environments. In section IV, we elaborate on the reasons to shift traditional system development approaches to model driven and ontology driven approaches where these two approaches are clearly interrelated. In section V, we briefly introduce a vision, environment programming, based on abstract application development through MDD and ODD. Finally, we conclude the paper in section VI.

II. SIMULATED INTELLIGENCE

There has been always a debate about whether it is possible or not to build intelligent thinking machines which are as good as human, however AI is still in its infancy to exhibit the higher level of human cognitive abilities and thought process in computers [13].

Take the following example in order to create links between the human way of intelligence and the existing computing "intelligence" within the course of context-aware pervasive computing environments. Imagine four people playing a card game, most probably each player in this game will follow a different strategy. One might simply consider the cards available in his hand and can simply ignore the rest of the contextual information. One might both consider cards which he has presently, and the cards which have already been played by other players in order to predict remaining cards in his opponents' hands (i.e. say historical context). One might go further and can also consider the playing habits of the users (i.e. say player context) in order to adjust his strategy, like one might be tend to first play the biggest order cards in his hand in order to guarantee that at least he is not going to get any penalty. Variety of examples might be listed, however what is intelligence according to the such game, or simply what is the most intelligent way of playing such a game? Although, answer for such a question depends on the utility that each player seeks for in such a game (e.g. not necessarily to win but not to have least score either, just to have fun, absolutely win etc.), it is evident that the more contextual information considered by the user, the more likely he is going to win. However trying to consider all different context elements might not be sufficient, for instance, keeping played cards in mind requires player to have enough amount of short term memory, player should be able to estimate required probabilities roughly (e.g. normally in a qualitative manner), furthermore player should be able to realize required processing within the bound of acceptable time limits.

Figure 1. "Computers are incredibly fast, accurate, and stupid. Human beings are incredibly slow, inaccurate, and brilliant. Together they are powerful beyond imagination." - Albert Einstein

Considering the computer way of intelligence, it is possible to donate computational devices with variety of sensors, and it is also evident that in the sense of processing power and computational resources (e.g. memory, CPU), computers are far beyond the abilities of the humans. If so, why today's computing is still far behind the human level of intelligence (see Fig. 1 and Fig. 2.)? The answer lays behind the term "reasoning", computational systems are good at gathering and aggregating data and humans are good at recognizing contexts and determining (e.g. to reason) what is appropriate [10].

Figure 2. "All programmers are playwrights and all computers are lousy actors." – Anonymous

Today, intelligent computational systems are intelligent only to the extent of completeness of the real-world situations modeled by the developers. Their intelligence are strictly based on the strong logical assumptions (e.g. axioms) done by the developers, but what about exceptions and ambiguities? Humans are able to perceive, interpret and reason variety of contextual information (e.g. environment context, relational context, historical context etc.), besides human being is able to adjust his interpretation and reasoning under exceptional and ambiguous conditions. However applications are only able to do what they are told to do. Indeed, every application, whether we consider it context-aware or not, is designed for a specific and restricted context of use. Therefore these applications provide particular set of behaviors for a fixed context. Today's traditional "intelligent" computing is not more than hard coded enumeration of possible contexts of use. Hence, we prefer to call such intelligence as "*simulated intelligence*" (i.e. weak AI [14], where we prefer AI alone to refer to strong AI). This is because it is only a limited reflection of human intelligence which consists of a limited conceptual model and reasoning logic of developers (i.e. mental models) on a specific problem. In other words, it is not intelligence itself through imitating functional aspects (i.e. how such mental models and rules are created), but rather output/artifact of human intelligence. Accordingly, "intelligence" in early examples of pervasive computing applications are only limited with hard-coded bindings of extended context space to pre-defined behavior space, that is usually, automatic adaptive behaviors with a degree of rationality. However pervasive computing extends the scope and opens up the infinite context space (i.e. increasing complexity) since such computing is part of the daily life where hard-coded enumerations of use are hard to manage and not sufficient. Ambient environments should accommodate reasoning components where conceptual models and logical assumptions (i.e. rules) are easy to manage and external to applications. Although such approach is of use for manageability, and for application development, ambiguity of contextual information decrease the level of reliability and rationality of the reasoning results. The importance of such unreliability might have high or low depending on the situation. Although various AI techniques can be applied to reduce the ambiguity, such techniques do not provide hundred percent of success.

III. USER MEDIATION

It is reasonable to say that human intelligence is still the dominant intelligence, therefore when developing intelligent computational systems, particularly in the sense of ambient intelligence, human should be part of the loop mostly by means of mediation [11, 12]. Consider the following real life example. A recent airplane crash, which occurred on 25 Feb. 2009 at the Amsterdam's Schiphol international airport and lead to death of 9 people and injury of 84, is a tragic case which proves the necessity of such an approach.

Simply speaking, the airplane crash occurred because of the wrong sensory input which provides the distance of plane to the ground. This information provided by this sensor is used by the automatic pilot at the time of landing. In this case sensor indicated a lower distance than the actual one which caused autopilot to stop the power given to the engines of the airplane. Eventually, this caused airplane to be powerless while it was still on the air, and caused death of 9 people. It is said that there were two different sensors in the airplane providing the same information (one was providing true value), and although pilots realized the false input given by the first sensor, they couldn't make plane to use the correct sensory device because of either a defect or a restriction. More dramatically, another airplane from the same airline company mistakenly landed at a wrong airport near Tiblisi in Georgia. The original airport and the other one were located close to each other with similar alignments. As an explanation for the case, one said that because of the Schipol incident, nowadays pilots choose to land manually rather than using autopilot which is difficult for such

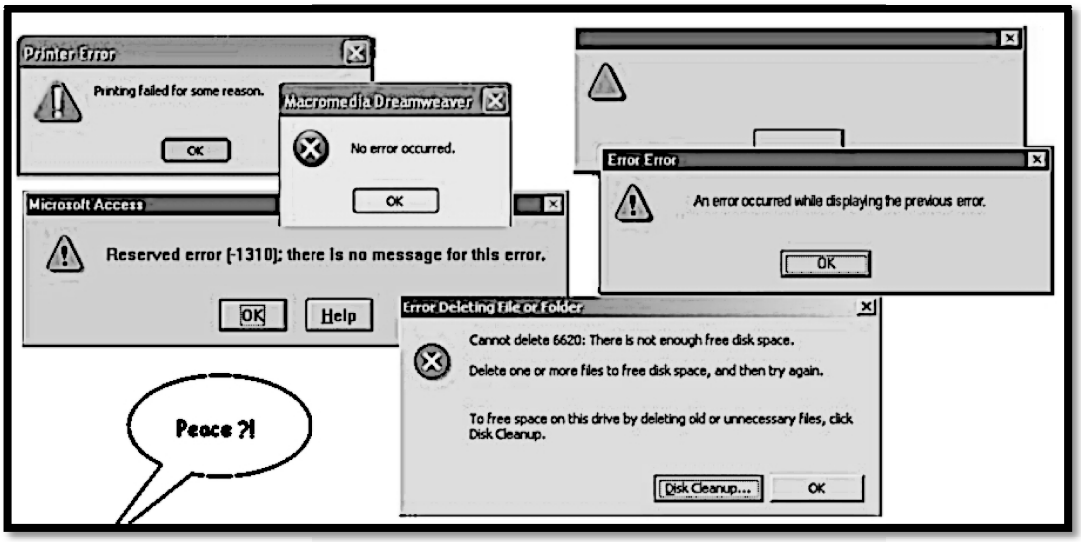

Figure 3. Poor human-computer communication

exceptional cases (i.e. two close airport which are similarly aligned), and particularly in bad weather conditions. If we reconsider cockpit of the airplane as a smart space, a reasoning component based on a context model could easily detect the

inconsistency arising from different sensory inputs, afterwards system could either try to reach more reliable contextual information (e.g. over control tower) or could inform pilots about the situation. Pilots could either decide on the reliability of autopilots decision (e.g. if system were able to explain the reasoning behind) or simply take over the control from autopilot for manual landing.

These incidents have something to tell, as previously noted intelligent computational systems only provide a "simulated intelligence" which requires intervention of the human user. However such systems should also be able to deliver the reasons of their decisions clearly to the user (see Fig. 3), then users will be tend to accept and use the reasoning of such systems. The matter is providing the right level of balance between automatic system decisions and user mediation. Apparently such optimization should be based on the priorities and significance of the situations in order to provide better user experience.

IV. MERGING APPROACHES

In order to be able to construct a bridge between humans and computers and to enable computers to decide on adaptive actions (i.e. automatic, semi-automatic, manual) through automated reasoning and/or mediation processes, computing systems need to maintain a formal model (i.e. context model) of the settings in which they are being used, and the complex relations between the various elements of these settings.

Ontologies [7, 15] might be an immediate solution since ontologies are based on ontological commitments, which means conceptualization represented by an ontology is shared by the related peers (e.g. users, applications etc.). They enable high level of interaction between the human users and "intelligent" applications, besides use of ontologies also enables systems to reason over the maintained formalized model and to explain the reasons of the decisions. Ontologies should take their part in every aspect of application development in order to cope increasing complexity, particularly at run time (e.g. for reasoning) and development time (e.g. specification, documentation, automatic code generation etc.). Ontologies are semantically richer, hence they can cover the functionalities of less expressive application and information artifacts such as UML, object oriented approaches, schemas etc.. Therefore a well defined ontology might be directly converted to application code itself, to database schemas, to abstract user interfaces etc., besides it moves reasoning logic apart from hard coded application which makes knowledge re-usable and management of reasoning logic more efficient . Although such efforts are already part of several ontology based researches [15], a real shift have not occurred yet and current application development research for pervasive computing is only limited with use of ontologies for context modeling and reasoning, e.g. [9, 16, 17]. This is because the scope of application development is still limited, hence available strategies are simple and sufficient. However a real shift from traditional computing to pervasive computing requires such ontological approaches to be extended and to be employed through the whole application development cycle.

Fortunately, MDD approaches [18, 19], which are based on increasing level of abstraction to cope with complexity, follows a similar approach with ontology-driven application development. Prominently, Model Driven Architecture (MDA) [20], which is initiated by OMG consortium, holds an important place for MDD. MDA initiative offers a conceptual framework for defining a set of standards in support of MDD [19]. However in ontology-driven approach, models are not only can be used to derive application code and other software artifacts automatically, besides they can be used for reasoning purposes which is of crucial. Indeed UML is in the core of model driven architecture, and in ontological engineering domain UML is considered to be a software engineering paradigm which can be used to model ontologies [7] where examples in pervasive computing domain are already available, e.g. [21]. That means, model-driven approach can also be considered as an ontology driven approach. However expressivity of UML is limited (i.e. light weight ontologies) where use of OCL (i.e. Object Constraint Language) to extend its expressivity is quite common [7], besides reasoning support is weak. Web Ontology Language, OWL, is a promising alternative since it allows to model heavyweight ontologies and it has a great support of reasoning tools, it has been already used for context modeling and reasoning in various research projects, e.g. [9, 16, 17]. Hence, we are lead to conclude that expertise and tools available in model driven approaches (particularly MDA) should be employed within Ontology-driven approaches where considerable amount of similarity between these approaches makes this really possible. Such merging approach will enable us to both get benefit of inference support of ontological approaches and the expertise of model driven approach for automated software development based on higher abstraction.

A possible merge of these two approaches might result in following roughly defined software development process (see Fig. 4), please note that definitions of the models adopted from MDA [22]:

(1) Define computing independent domain ontology (CIDO) and a generic ontology (GO, omitted in the figure for the sake of brevity),

(2) Convert part of CIDO to Platform Independent Application Model (PIAM),

(3) Convert PIAM to Platform Specific Application Models (PSAM) and Artifact Dependent Models (ADM, e.g. Database Schema), and define mapping technique and annotations,

(4) Convert part of the CIDO together with GO to Context Ontology (CO),

(5) Apply conversion of PSAM(s) and ADM(s) to application code and software artifacts.

In the given procedure, some portion of CIDO is converted into PIAM and some portion is converted into CO, this is because some portion of the CIDO might not necessarily need to map aspects of the application but rather to the reasoning logic of the application which is required to be used by the application externally. Existing work in the literature for transforming ontologies to software artifacts, including application code, shall be of use in such a merged approach. Hence we briefly refer to related literature, [22] shows how to convert RDF

schema and RuleML sources into Java classes, and [24] presents how to create a set of Java interfaces and classes from OWL ontology such that an instance of Java class represents an instance of a single class of the ontology with the most of its properties, class relationships and restriction-definitions maintained [24]. Similarly in [25], authors show how an OWL/RDF knowledge base can be integrated with conventional domain-centric data models (Enterprise Java Beans) and object-relational mapping tools (e.g. Hibernate). Considering relational databases, [26] presents a mapping technique from ontologies to relational databases in order to facilitate rapid operations (e.g. search, retrieval etc.) and to utilize benefits of relational management systems (e.g. transaction management, security etc.).

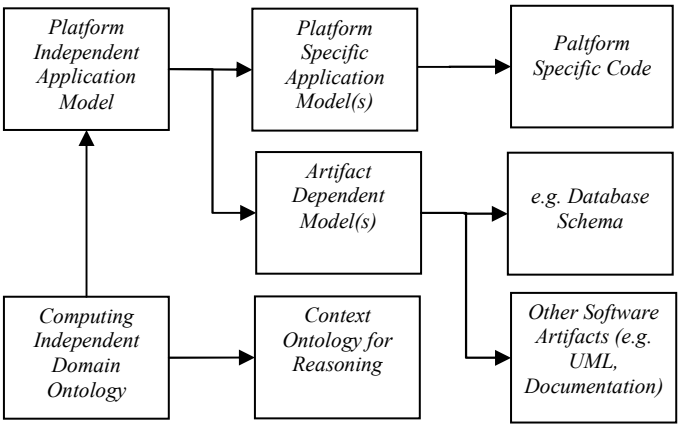

Figure 4. Toward merging MDD and ODD

Although some basic tools for automating transformations have been developed by aforementioned works, they don't seem to be sufficient, furthermore semantic differences (i.e. description logic vs. object oriented) between models to be transformed needs to be further investigated and should be define exhaustively. Success of proposed approach is mainly based on the completeness and quality (e.g. structure loss, data loss etc.) of transformations done, hence a complete evaluation is required. Indeed, most challenging job is to enlighten the conservative mindset of the software practitioners [22] where providing best practices and tools is crucial for such purpose.

V. ENVIRONMENT PROGRAMMING

We further advocate that human users should be able to program their own environment – *environment programming* – (more than ontology-driven programming) to some extend which seems to be possible by enabling applications and users sharing same conceptual world model. In other words, such approach allows users to program their own smart spaces through enabling higher level of human-computer interaction.

Such approach should be at the higher level of abstraction which will overcome possible technical barriers and turns physical environment into a huge programmable space where users can manage, synchronize, link etc. available functionalities and behaviors of applications and hardware entities in the environment according to available contextual information. Internet/intranet environment along with already available standards and protocols (e.g. HTTP, HTML etc.) might serve creation of such smart spaces by acting as [27] an (1) *application and communication space* (Web of Things [28]), that is, networking variety of appliances in an environment and employing web environment as an application space. Furthermore, as an (2) *information space*, that is, utilizing web as an ultimate information source by means of using variety of semantic web technologies available (e.g. RDF, embedded semantics etc.), we refer to our previous work further details of such an approach [27, 29].

A particular scenario might be as following. Tom has an "intelligent" home where various devices in his home such as TV, refrigerator etc. are connected to each other through the internet or intranet. These devices are able to identify different users in this home such as parents and children. Furthermore they are able to construct a model for each user which reflects user's habits etc. Considering TV related behavior, assume that there are different TV stations embedding their schedules in their web pages through one of the aforementioned technologies, RDFa, eRDF or Microformats, by using a common semantic (i.e. common vocabulary) for program descriptions and event descriptions (e.g. iCal). This intelligent TV can recommend TV programs for a particular user based on the user model by harvesting and querying embedded information in the web pages. We can further extend this scenario to cover pervasive aspects. Consider a refrigerator which is also connected to the web and which is able to detect food items which it contains. User instructs refrigerator to follow the availability of a particular food (probably through RFID). When a particular type of food is exhausted, it harvests web pages of different markets and according to the shopping habits and economical considerations of the family it recommends a list of markets. User reaches refrigerator through TV by coupling services given by TV and refrigerator, so he can view the available food items any time. Hence, refrigerator communicates with the TV for display purposes in this case. TV and refrigerator use RDF for communicating shared information.

A straight forward design of such a smart space consist of networked appliances which probably accommodate built-in micro web servers or gateways [30] together with semantic descriptions of its available physical and functional services [31]. Users shall be able to condition, link and synchronize services and information flow (e.g. listeners, triggers etc.) between appliances by making use of an abstract rule language. Web environment can be of use for such space as an information source by enabling agents located in these devices to harvest valuable web information [27]. A reasoning component together with a management infrastructure need to be accommodated by stronger computational devices, e.g. PC, moreover higher security practices need to be employed.

VI. CONCLUSIONS

Smart technologies should not make people dumb and adapt activities and environment to this dumbness [32]. Machines should support sensory limited human users with extended sensory information and should further reason on available information (where user processing is not sufficient) either in order to act automatically or let humans to interpret and reason based on available context information and

processing results. Adaptive behaviors (e.g. automatic actions) do not necessarily need to result in "must"s or "have-to"s but in many cases also in "should"s and "might"s (i.e. automatic adaptive mediation or guidance) to give users a degree of control with the possible directions and their reasoning behind. Ontologies are of great use for providing better human and machine experience/interaction and for software engineering in the new computing paradigm. [33] has noted that in the future, software will not be designed without using an ontological approach, especially when adequate tools are available. Ontologies will provide a neutral authoring mechanism where resulting ontologies will be automatically converted into different information and application artifacts. Thus, apart from greatly simplifying application development for pervasive computing systems, ontologies will also create bridges between human-human, computer-computer and crucially between human and computers which will enable us to interact, to program our environment and to connect human intelligence and computer "intelligence" tightly. View point represented through this article does not only target use of ontologies for software development, such approach is already concluded by several experts in the domain. However, more particularly, we advocate that pervasive computing opens up variety of challenges where use of ontologies in every phase of application development is of crucial while its complexity exceeds its possible benefits for traditional computing.

REFERENCES

[1] M. Weiser, "The Computer for the 21st Century", Scientific American, 1991, pp. 94-98.

[2] M. Bick, T. F. Kummer, "Ambient Intelligence and Ubiquitous Computing", In Handbook on Information Technologies for Education and Training, Ed. Bernurs, P., Blazewicz J., Schmidt, G., Shaw, M., Springer Verlag, 2008.

[3] A. Soylu, P. De Causmaecker, P. Desmet, "Context and Adaptivity in Context-Aware Pervasive Computing Environments", In Proc. Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, UFIRST'09 Workshop, Brisbane, Australia, 2009.

[4] A. K. Dey, G. D. Abowd, "Towards a better understanding of context and context-awareness", Technical Report GIT-GVU-99-22, Georgia Institute of Technology, College of Computing, 1999.

[5] L. Han, S. Jyri, J. Ma, K. Yu, "Research on Context-aware Mobile Computing", In Proc. Advanced Information Networking Applications – Workshops, pp 24-30, 2008.

[6] H. X. Wang, Q. D. Zhang, T. Gu, H. K. Pung, "Ontology based context modeling and reasoning using OWL", In Proc. PerCom 2004, Orlando, FL, USA, 2004.

[7] A. Gómez-Pérez, M. Fernández-López, O. Corcho, Ontological Engineering, Springer Verlag, 2003.

[8] K. Henricksen, J. Indulska, "Developing Context-Aware Pervasive Computing Applications: Models and Approach", Journal of Pervasive and Mobile Computing, 2(1), pp. 37-64, 2006.

[9] T. Strang, C. Linnhoff-popien, "A context modeling survey", In Proc. Advanced Context Modelling, Reasoning and Management, Ubicomp2004, Nottingham, UK, 2004.

[10] T. Erickson, "Some problems with the notion of context-aware computing", Communications of the ACM, 45(2), 2002, pp. 102-104.

[11] A. K. Dey, J. Mankoff, D. Gregory, "Distributed mediation of ambiguous context in aware environments", Abowd and Scott Carter Proceedings, UIST 2002, 2002.

[12] J. Mankoff, D. A. Gregory, S. E. Hudson, "OOPS: A toolkit supporting mediation techniques for resolving ambiguity in recognition-based interfaces", Computers and Graphics 24, 6, pp. 819-834, 2000.

[13] D. Nadeem, L. Sauermann, "From Philosophy and Mental Models to Semantic Desktop Research: Theoretical Overview", In Proc. I-Sematics'07, 2007.

[14] J. R. Searle, "Minds, Brains, and Programs", The Behavioral and Brain Sciences, Vol. 3, Cambridge University Press, 1980.

[15] F. Ruiz, J. R. Hilera, "Using Ontologies in Software Engineering and Technology", In Ontologies for Software Engineering and Software Technology, Ed. Calero, C., Ruiz, F., Piattini, M., Springer Verlag, 2006.

[16] H. X. Wang, Q. D. Zhang, T. Gu, H. K. Pung, "Ontology based context modeling and reasoning using OWL", In Proc. PerCom 2004, Orlando, FL, USA, 2004.

[17] H. Chen, T. Finin, A. Joshi, "An ontology for context-aware pervasive computing environments", Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review, 18(3), pp.197-207, 2004.

[18] B. Selic, "The Pragmatics of Model-Driven Development", IEEE Software, Special Issue on Model-Driven Development, pp. 19–25, 2003.

[19] D. C. Schmidt, "Model-Driven Engineering", IEEE Computer, 39(2), 2006.

[20] G. Booch, A Brown, S Iyengar, J. Rumbaugh and B. Selic, "An MDA Manifesto," In The MDA Journal: Model Driven Architecture Straight from the Masters, Chapter 11, Meghan-Kiffer Press, December 2004.

[21] K. Henricksen, J. Indulska, A. Rakotonirainy, "Modeling Context Information in Pervasive Computing Systems", Proceedings of First International Conference on Pervasive Computing, Pervasive 2002, pp. 79-117, 2002.

[22] Singh, Y., Sood, M., "Model Driven Architecture: A Perspective", In Proc. IEEE International Advance Computing Conference, 2009.

[23] A. Eberhart, "Automatic Generation of Java/SQL based Inference Engines from RDF Schema and RuleML", Proceedings of the First International Semantic Web Conference (ISWC 2002), Chia, Sardinia, Italy, pp. 102-116, 2002.

[24] A. Kalyanpur, D. Pastor, S. Battle, J. Padget."Automatic mapping of OWL ontologies into Java", Proceedings of the International Conference on Software Engineering & Knowledge Engineering (SEKE), 2004.

[25] I. N. Athanasiadis, F. Villa, A. E. Rizzoli, "Ontologies, JavaBeans and Relational Databases for enabling semantic programming", Proceedings of the 31th IEEE Annual International Computer Software and Applications Conference (COMPSAC), Vol 2, pp. 341-346, 2007.

[26] I. Astrova, N. Korda, A. Kalija, "Storing OWL Ontologies in SQL Relational Databases", International Journal of Electrical, Computer and Systems Engineering, Vol. 1, No. 4, 2007.

[27] A. Soylu, P. De Causmaecker, "Embedded Semantics Empowering Context-Aware Pervasive Computing Environments", In Proc. Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, USST09 Workshop, Brisbane, Australia, 2009.

[28] D. Raggett, "Towards the Web of Things", Talk given at UWE WDC, Bristol, www.w3c.org/Talks/0926-dsr-WDC/slides.pdf , September 2007.

[29] A. Soylu, S. Kuru, F. Wild, F. Mödrichter, "e-Learning and Microformats: A Learning Object Harvesting Model and a Sample Application", In Proc. Mupple'08 Workshop, pp. 57-65, 2008.

[30] T. Dillon, A. Talevski, V. Potdar , E. Chang, "Web of Things as a Framework for Ubiquitous Intelligence and Computing", In Proc. The 6th International Conference on Ubiquitous Intelligence and Computing, Brisbane, Australia, 2009.

[31] C. Pahl, "Semantic Model-Driven Architecting of Service-based Software Systems", Information and Software Technology, Vol. 49-8, 2007.

[32] J. Hundeboel, N. Helms, "Pervasive e-learning, In Situ Learning in Changing Contexts", In Proc. DREAM'06, 2006.

[33] D. M. Pisanelli, A. Gangemi, G. Steve, "Ontologies and Information Systems: the Marriage of the Centruy?", In Proc. LYEE Workshop, Paris, 2002.