Factors affecting developers' use of MDSD in the Healthcare Domain: Evaluation from the MPOWER Project

Ståle Walderhaug[a,b], Marius Mikalsen[a], Ivan Benc[c], Greg Loniewski[d], Erlend Stav[a]

[a]SINTEF ICT, Norway
[b]Department of Computer Science, University of Tromsø, Norway
[c]Ericsson Nikola Tesla, Zagreb, Croatia
[d]Dimension-Informatica, Valencia, Spain

Abstract: Model-driven software development (MDSD) is steadily gaining popularity, and new and more advanced tools are being developed. However, the paradigm shift that has been expected has not yet come, despite reliable reports on both quality and productivity improvements using model-driven approaches. This paper investigates which factors are important for developers to use MDSD in their work. In January 2007, a total of 16 developers from one university, two SMEs, one research organization and one large multinational company were introduced to a MDSD toolchain for software service development. After using the toolchain for one year in development of middleware services and end-user applications, the developers evaluated the toolchain and their use of MDSD in general. The evaluation was done using proven research methodologies that were adapted to be able to evaluate MDSD. All 16 developers from the four European countries participated. The findings suggest that perceived usefulness and ease of use are the most important factors for using a MDSD development approach. No significant relationships between tool performances or subjective norms were found. However, the "traceability" feature of the development approach was found useful. The work was carried out as a part of the MPOWER EU-IST project, and the results will be used for improving the project toolchain and the evaluation processes.

Introduction

Software engineering projects often fail to deliver the promised results. Several studies show that projects expend more effort than is estimated and deliver less functionality than initially planned[1-3]. Also, software systems are getting more complex, incorporate more functionality, and users have a higher demand for performance and reliability. One approach to control system complexity apply abstraction and reuse of existing well-proven artifacts[4, 5]. This is accordance with the fundamental concepts of model-driven software development (MDSD), that separates business functionality from technological details to hide technical complexity, and applies model transformation and code generation to reuse code and software patterns [6, 7]. Many publications reports from successful adoption of

MDSD in IS organizations, but very few provide empirical results or an explanation of the success criteria. In [8], Staron tries to explain the requirements and factors for adopting MDSD in organizations based on a case study in two companies. The study shows that for a company to adopt MDSD, it should make it possible to estimate costs based on models, improve quality and understanding, improve communication between developers, and enable traceability throughout the software artefacts. Staron found that a key factor for determining the adoption is the availability of modelling tools. Tool support was also addressed by MacDonald et al in [9]. They describe the “perfect tool” where platform independence, access to rich libraries and possibility to perform sophisticated analysis (including traceability) are key features in addition to those described by Kleppe [10]. Other recent studies on MDSD adoption includes the book by Guttman and Perodi [11], where six real-life projects, all report positive experiences from using a model-driven software development approach. However, the results and benefits are based on interviews with the CEOs and CTOs in the companies, and do not provide reliable scientific evidence. Despite many reports of successful adoption of MDSD principles in companies around the world, the paradigm shift that MDSD was supposed to make [10, 12] has not happened. At the Future of Software Engineering conference in 2007, France and Rumpe presented the main challenges of MDE (MDSD included)[13]. They identified modelling language, separation of concern and model manipulation and management challenges. All identified challenges are focused on technological concepts and constructs.

The healthcare domain has and continues to be, undergoing a digitization process where all information should be stored and distributed electronically. The demand for healthcare services increases dramatically and soon there will be a shortage of healthcare personnel[14]. Healthcare systems are complex, and projects often fail because of the problem of transforming the knowledge of the user domain into knowledge in the domain of those that implements the solutions[13]. But, information technology is considered promising in order to enable care at home, facilitating active ageing at home, and leveraging the informal care given by family and friends. In turn, these are prerequisites to have sustainable healthcare services in the future. There are many factors creating the complexity. Many stakeholders and legacy systems are involved, and security and reliability requirements are strong. The development costs are compared against the acquisition of more advanced medical equipment, more effective (and expensive) medicine and more healthcare personnel on duty. Consequently, there is a need to reduce complexity to reduce development cost and improve system quality in healthcare information systems. If MDSD were to deliver upon its promise in the healthcare domain, this would be deemed very beneficial for all involved actors.

To investigate the use of MDSD in healthcare, the MPOWER project [15] has designed and evaluated a MDSD toolchain that supports development of home care services on a SOA platform. This paper presents the evaluation of MDSD done in the MPOWER project focusing on developers’ acceptance of the model-based approach to doing system development of healthcare middleware services. A MDSD toolchain was introduced in the project and the users of the toolchain were asked to fill out an evaluation questionnaire after one year of use. The results suggest that tool usefulness and ease of use are important for the acceptance of MDSD. Other factors such as traceability and code generation are found useful.

The remainder of this paper presents the conceptual model along with the hypotheses that will be tested, the research method and the results from the survey. Finally, a discussion with concluding remarks and suggested further research is given.

Conceptual model and hypotheses

MDSD should be regarded as a new technology supporting developers, and its use must be evaluated not only on the project outcome level (e.g. delivery on time and on budget), but also on the individual level to discover why or why not MDSD works. MDSD often represents innovations for the potential adopters, and the evaluation of its acceptance should use methodologies from the diffusion of innovation knowledge base. Perceived characteristics of innovating (PCI) is posited to have a significant influence on user acceptance [16]. The technology acceptance model (TAM) [17, 18] and the theory of planned behaviour (TPB) [19] are other models that attempt to explain the user perceptions and use intentions of a certain technology. The conceptual model for the study presented builds upon the factors of TAM, TAM2, PCI and TPB. This study reuses four of five components from the study by Dybå in [20]. In addition, a new component on *tool performance* is introduced. The *tool performance* component is included based on the findings by Staron[8], MacDonald[9], and specifically targets the core MDSD features business analysis, traceability and implementation automation[6, 7, 21].

Two dependent variables are measured: the current use of model-driven development techniques, and future use intentions. Figure 1 shows the proposed hypotheses about the factors affecting the use of MDSD techniques (+ sign indicates a positive association).

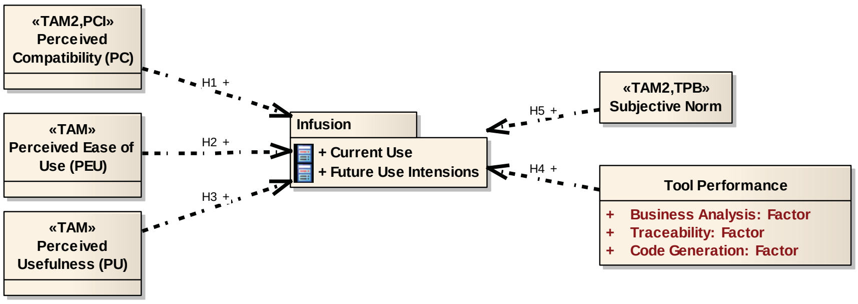

Fig. 1. The conceptual model for the investigation. Five factors are suggested to affect the usage of MDD

- The Perceived Compatibility : "the degree to which an innovation is perceived as being consistent with the existing values, needs and past experiences of potential adopters" [16]: *H1: The perceived compatibility is positively associated with current usage and future use intentions of MDD*

- The Perceived Usefulness: "the degree to which a person believes that using a particular system would enhance his or her job performance" [17]: *H2: The perceived usefulness is positively associated with the current usage and future use intentions of MDD*
- The Perceived Ease of Use (PEU): "the degree to which a person believes that using a particular system would be free of effort" [17]: *H3: The perceived ease of use is positively associated with current usage and future use intentions of MDD*
- Tool performance: the degree to which the tools support the development process and the tasks at hand: *H4: The perceived tool performance is positively associated with current usage and future use intentions of MDD, H4.1: Business Analysis, H4.2: Traceability and H4.3: Code Generation*
- Subjective Norm: In an extension of the TAM from 2000, Venktatesh and Davis added the subjective norm [18]: "*the person's perception that most people who are important to him think he should or should not perform the behaviour in question*" : *H5: The subjective norm is positively associated with current usage and future use intentions of MDD*

Research Method

Study context and subjects

The study was carried out in the MPOWER project that started in October 2006 and runs until April 2009. The project uses and agile model-driven development approach [22] for the design, development and evaluation of homecare middleware services. The main objective of the project is to build a middleware platform that enables rapid development of smart homecare services by reusing concepts services in a SOA framework [15]. The services are designed and developed using a set of MDSD tools and techniques described in the next section.

In January 2007, developers from one university, two SMEs, one research organization and one large multinational company were introduced to a MDSD toolchain for software service development. The subjects in the study are the developers of the middleware services and end user applications. They were using the MDSD tools from the initial design to final deployment and pilot testing. A total of 16 developers (n=16) from four European countries (Austria, Croatia, Cyprus and Spain) participated.

Table 1. Summary of study subjects' characteristics

Work duties	Highest completed education	Years experience with programming /systems engineering

81 % are developers 30% involved in arch 31% are doing research 6 % project management 6 % product support	31% Bachelor's degree 63 % Master's degree 6% Other education	From 1 -20 years Mean: 5,63

Model-Driven Tools introduced in the project

The developers in the project were introduced to a model driven methodology with tool support in January 2008. Fig 2 shows the UseCase Model, ActorModel and ServiceModel that were created using Enterprise Architect (EA)[23]. The IBM Profiles for Software Services [24] were imported into EA to specify the ServiceModel in terms of Service Oriented Architecture (SOA) [25]concepts. Using model transformation and code generation, WSDL models and code were created from the ServiceModel. From these, web services generated and web applications developed. The applications will be evaluated by healthcare personnel and patients during the spring of 2008.

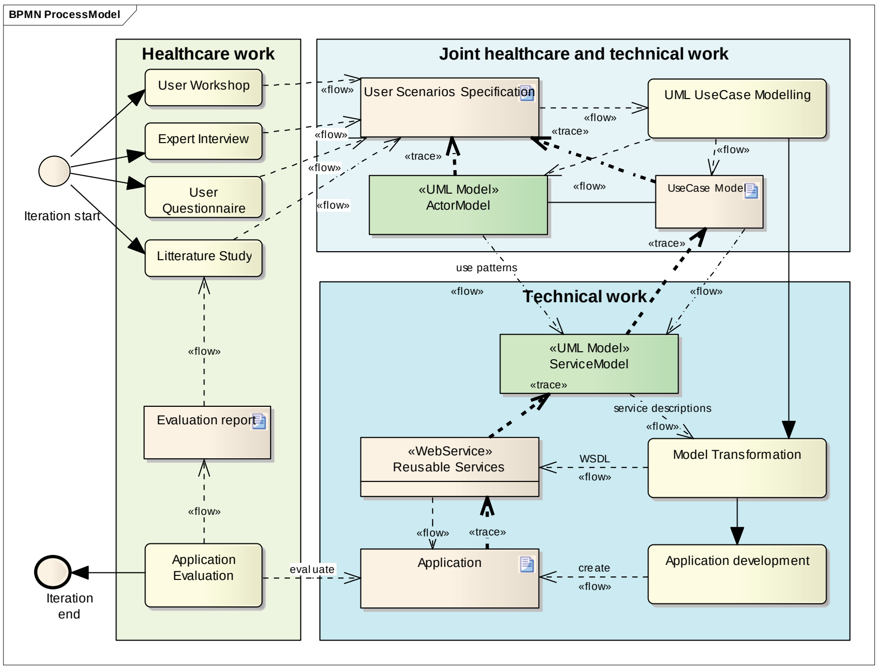

Fig. 2. The MPOWER Toolchain

A key methodological issue was the ability to trace the reusable services back to the context and user interactions for which it was designed. This was done using UML dependency links in EA and periodically exporting the model documentation to a web-server for easy navigation. The trace links are denoted as bold arrows in Fig. 2.

Data collection procedure

Limesurvey[1], an open source web survey tool were used for data collection. The requests for participation were sent by email, and all respondents filled in the questionnaire within 3 working days. 16 out of a total of17 developers participated.

The questionnaire had seven groups of questions, each group including a free-text field for comments: 1) candidate information (table 1), 2) current use, 3) future use intentions, 4) perceived usefulness, perceived ease of use, perceived compatibility, 5) subjective norm, 6) tool performance, and 7) MPOWER Toolchain experience. Question group 2 and 3 used a scale ranging from 0-4 (not used at all, used on an experimental basis, used on a regular basis by a few people/project, used on regular basis by most people/projects, used on regular basis by all people/projects)[26] , while group 4, 5 and 6 used a five-point Likert scale [27]. The results from the questionnaire were exported from LimeSurvey to be imported into SPSS 15[2] for statistical analysis.

Assessment of reliability and validity

The reliability of the independent and dependent factors in the conceptual model is presented in table 2. Factors in *italic* are subfactors of "Tool Performance (TP)".* Question *"given a choice, I would prefer <u>not</u> to use model-driven development in any future"* was misinterpreted by some respondents. Reliability was strongly improved when this question was removed from the factor. The MPOWER specific questions were not subject for factor analysis.

Table 2. The reliability of the independent and dependent factors measured.

Factor	#items	Cronbach's alpha
Perceived Usefulness (PU)	5	0,872
Perceived Ease of Use (PEU)	6	0,917
Perceived Compatibility (PC)	4	0,937
Subjective Norm (SN)	3	0,966
Tool Performance (TP)	13	0,878
- Business Analysis (BUS)	*3*	*0,573*
- Traceability(TR)	*3*	*0,944*
- Code Generation (GEN)	*4*	*0,836*
Current use (CU)	6	0,947
Future use intentions (FUI)	3 (4*)	0,970 (0,675*)

The content validity, as defined by Dybå[28], has to do with the appropriateness of the scale items in the domain under study. For this study, this is ensured through the

[1] Limesurvey website: http://www.limesurvey.org/
[2] Statistical Package for the Social Sciences (SPSS) website: http://www.spss.com/

reuse of validated scales from TAM, TPB and PCI – all with questions adapted to the MDSD domain. The criterion-related validity from Dybå is concerned with the degree to which the scales under study are relied to an independent measure of the relevant criterion. This was evaluated by computing the multiple correlations between the independent variables and the dependent variables. The results are shown in table 3.

For the dependent variables, only "Current Use (CU)" has a significant correlation with the independent variables "Perceived Usefulness (PU)" and "Perceived Ease of Use (PEU)". The results also show correlations between "Perceived Usefulness (PU)", "Perceived Ease of Use (PEU)" and "Perceived Compatibility (PC)".

Table 3. Intercorrelations between independent and dependent variables

Variable	PU	PEU	PC	SN	TP	*BUS*	*TR*	*Gen*
PU	1							
PEU	,783(**)	1						
PC	,557(*)	,776(**)	1					
TR	ns	ns	ns	ns	-	,599(*)	1	
GEN	ns	ns	ns	ns	-	,618(*)	ns	1
CU	,596(*)	,596(*)	ns	ns	ns	ns	ns	ns
FUI	ns	ns	ns	ns	ns	ns	ns	ns

** Correlation is significant at the 0.01 level (2-tailed), * Correlation is significant at the 0.05 level (2-tailed)., ns = not significant

Results

To further investigate to which degree the factors correlate, regression analysis were conducted on the significant correlations. The partial regression analysis, checking each of the independent variables (PU and PEU) with the dependent variable CU is shown in table 4. The results show that both PU and PEU can explain 30,9% of the variance in current use. Both models are significant on the 5% level.

Table 4. Partial regression analysis of CU with PU and PEU

Regression equation	Adjusted R Square	Beta	*t*-value	Significance
CU = PU	,309	0,596	2,775*	,015
CU = PEU	,309	0,596	2,774*	,015

Table 5 shows the results for the regression between the dependent variable CU and the two independent variables that had a significant correlation with CU, namely PU and PEU. The analysis shows that the complete model is significant on the 5% level and 30,5% of the variance of CU can be explained using the two independent variables PU and PEU. However, the coefficients are not statistically significant; hence an equation for expressing the value of CU from PU and PEU cannot be created.

Table 5. Regression analysis for CU = PU + PEU(*p<0,05)

Regression equation		Adjusted R Square	Beta	*t*-value	Significance
CU =	PU		,334	,967	,351
	+				
	PEU	,305*	,334	,965	,352

The three subfactors of Tool Performance (TP) were measured, namely Business analysis (BUS), Traceability (TR) and Code Generation (GEN). None of these correlated with CU or FUI. However, descriptive statistics in table 6 shows that traceability is a feature that the developers found useful. The variance in code generation was high.

Table 6. Descriptive Statistics of Tool Performance Subfactors

	Minimum	Maximum	Mean	Std. Deviation	Variance
Code Generation	1,25	4,25	3,0000	,83166	,692
Business analysis	2,67	4,00	3,5111	,48578	,236
Traceability in toolchain	3,00	5,00	3,7500	,67220	,452

A separate group of questions targeted the concrete experience from using the MPOWER MDSD toolchain and methodology proposed in the MPOWER project. The questions and the statistics are presented in table 7. 13 out of 16 respondents (n=13) developers had used the complete toolchain and qualified to answer this group of questions. The scale is 1-5 (Strongly disagree – Strongly agree).

Table 7. Descriptive statistics for MPOWER specific features

Question	Min	Max	Mean	St.d
Using Model-driven development in MPOWER is useful	3	4	3,77	,439
Using Model-driven development in MPOWER improves the collaboration between the partners	2	5	3,77	,832
Using Model-driven development in MPOWER requires much coordination	2	5	4,00	,913
Using Model-driven development in MPOWER decreases my performance as a developer	2	5	2,85	,899
UML is easy to understand	3	5	4,00	,707
UML provides all the functions/mechanisms I need	2	4	3,38	,768
The tools (i.e. Enterprise Architect and Netbeans) were easy to use / learn	1	5	3,46	1,198
The tools limited my performance as a software developer	2	5	2,92	,954
Using models just adds more work to the development process	2	4	2,85	,801

Models are important to the development process	3	4	3,77	,439
Models can be used in programming	3	4	3,62	,506
UML is useful for Service Oriented Architectures	3	5	3,92	,494
Traceability between model elements (components, features, actors) is important for the development process	3	5	3,85	,801
Traceability is for system documentation only	2	3	2,46	,519
Traceability improves my understanding of the system	3	5	3,69	,751

Comments from the respondents

To capture qualitative comments from the subjects, free-text fields were included on each group of questions. One developer made a comment in the group with questions on PU, PEU and PC:

- "Using Model-Driven development improves my job performance and productivity, only if everything works well with the transformation of models... Otherwise you can find yourself spending too much time trying to make things work (and doing the required changes manually). If this is the case then using Model-Driven development takes too much time from my normal duties. "

Three developers made comments on the MPOWER Toolchain:

- "By my opinion, the problem in MPOWER was the selected tool. EA is a powerful tool but 1) not fix very well with Netbeans and 2) unknowledging of the tool by the involved team, did that the project development was difficult, sometimes losing a lot of time instead of reducing (that's the idea of the use of the tool)."
- "Maybe we could select more stable tools (instead of Netbeans) or tools that work well together"
- "The idea of using EA + Netbeans is good, but I found a lot of problems in the use of these tools (more to integrate the work done in the first one to the second one), modifying the code by hand sometimes. Somehow, I thought that people hadn't tested the tools before deciding to use it."

Discussion

A survey was performed in a MPOWER project to investigate factors affecting developers' acceptance and utilization of MDSD. After using a MDSD toolchain for designing and developing healthcare middleware services, 16 developers from five different organizations in four European countries participated. The evaluation results show that perceived usefulness and perceived ease of use have an impact on the developers' current use of MDSD. This finding is confirmed by the statistics from the tool performance factor and the free-text comments made by some respondents. They find models useful in development, and MDSD is useful to improve collaboration, traceability and generate code. This is fully in accordance with the results from the study presented by Staron in 2006, where improving quality by increasing understanding, improving communication within development and improved traceability are identified as key requirements for MDSD. On the other hand, the results in table 8 indicate that MDSD requires much coordination in the development team. Coordination should be supported by tools, requiring flexible management systems, as identified as a key MDSD challenge for the future[13].

Furthermore, the analysis shows that none of the proposed factors an influence on the developers' future use intentions of MDSD. This is a neutral result that can be explained by a general positive attitude to MDSD in the development process (table 7, first item: mean 3,77), and mixed experience with tool stability and functionality (table 7, neutral mean values for items on developer's performance, and free-text comments).

Subjective norm was not found to affect the current or future use of MDSD. There are no results that can explain this. One explanation could be that the number of participants in the survey should be higher to ensure a normal distribution for performing statistical analysis. In most software engineering projects, getting sufficient evaluation results is a hard goal to achieve, and a combination of quantitative and qualitative approaches should be used as suggested by Miller in [29]. The free text comments provided in the survey made it possible to explain findings from the statistical analysis and thus increase the validity of the survey as a whole.

To summarize, two out of five proposed hypotheses are supported by the findings in the survey: perceived usefulness and ease of use have are positively affected with current use of MDSD. Traceability between artefacts is also found useful.

Clearly, the MDSD tools introduced in this study were not good enough. The developers did not agree on the effectiveness and usefulness of MDSD. A specific case is code generation functions that got a mean score on 3,00 (neither agree nor disagree), but with a range from 1,25 to 4,25 (table 6). The comments given explain this high variance with the fact that the generated code was not 100% correct and that a time-consuming process of manually changing the generated code was required.

As presented in the introduction, there are both positive and negative experience reports from adopting MDSD in industry. This fact emphasizes the need to investigate the factors that affect the developers' adoption and use of MDSD.

In healthcare systems development, with its specical characteristics, traceability will be a key concern. Bridging the gap between healthcare processes and IT support tools is essential to defend the investments in IT systems for healthcare. In accordance with trends in MDSD literature [9, 13], the use of domain specific languages / domain specific application development environments should be investigated. The MDSD concepts along with a domain specific language extension such as a UML profile for HomeCare could improve the developed systems' adherence to domain information standards through reuse of conceptual models and transformation templates, and at the same time improve quality and development cost [30]. The process of adopting MDSD should use experience from other domains and organizations[31, 32].

Future work

The study presented in this paper will be followed up by a new evaluation in the MPOWER project's next phase. In addition, controlled experiments using students at

Norwegian universities will be conducted in 2008 using an improved MPOWER Toolchain as intervention.

Concluding remarks

The results from statistical analyses and subjective comments from the respondents indicate that MDSD tools must be perceived useful and should be easy to use. Tool performance does not have a direct effect on MDSD use, although business analysis, traceability and code generation were found useful. It is especially important that MDSD tools are stable and provide complete and correct artefacts.

Using a survey based on established models for technology acceptance should be extended with qualitative approaches including free text feedback and possibly also interviews.

Acknowledgements

The authors would like to thank the MPOWER Consortium for participating in the survey and the European Commission for funding the MPOWER project (034707) under the IST programme.

References

[1] M. Jørgensen and K. Moløkken-Østvold, "How large are software cost overruns? A review of the 1994 CHAOS report," *Information and Software Technology*, vol. 48, pp. 297-301, 2006.

[2] K. Molokken and M. Jorgensen, "A review of software surveys on software effort estimation," *Empirical Software Engineering, 2003. ISESE 2003. Proceedings. 2003 International Symposium on*, pp. 223-230, 2003.

[3] Standish Group International, "The Chaos Report," Standish Group International 1994.

[4] C. W. Krueger, "Software reuse," *ACM Computing Surveys (CSUR)*, vol. 24, pp. 131-183, 1992.

[5] M. Shaw, "Abstraction techniques in modern programming languages," *IEEE Software*, vol. 1, pp. 10-26, 1984.

[6] J. Miller and J. Mukerji, "MDA Guide Version 1.0.1," Object Management Group (OMG) omg/2003-06-01, 2003-06-13 2003.

[7] T. Stahl and M. Völter, *Model-driven software development: technology, engineering, management*. Chichester: Wiley, 2006.

[8] M. Staron, "Adopting Model Driven Software Development in Industry–A Case Study at Two Companies," *Proceedings of the MoDELS/UML conference*, 2006.

[9] A. MacDonald, D. Russell, and B. Atchison, "Model-driven development within a legacy system: an industry experience report," *Software Engineering Conference, 2005. Proceedings. 2005 Australian*, pp. 14-22, 2005.

[10] A. Kleppe, *MDA Explained: The Model Driven Architecture™: Practice and Promise*, 2003.

[11] M. Guttman and J. Parodi, *REAL-LIFE MDA: Solving business problems with model driven architecture*: Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 2006.
[12] B. Selic, "The pragmatics of model-driven development," *Software, IEEE*, vol. 20, pp. 19-25, 2003.
[13] R. France and B. Rumpe, "Model-driven Development of Complex Software: A Research Roadmap," *International Conference on Software Engineering*, pp. 37-54, 2007.
[14] E. Coiera and E. J. Shovenga, "Building a Sustainable Health System," in *IMIA Yearbook of Medical Informatics 2007: Biomedical Informatics for Sustainable Health Systems*, vol. 46, *Methods of Information in Medicine, Supplement*, A. Geissbuhler, R. Haux, and C. Kulikowski, Eds.: Schattauer Publishers, 2007, pp. 250.
[15] The MPOWER Consoritum, "MPOWER Website," vol. 2007, 2007.
[16] G. C. Moore and I. Benbasat, "Development of an instrument to measure the perceptions of adopting an information technology innovation," *Information Systems Research*, vol. 2, pp. 192-222, 1991.
[17] F. D. Davis, R. P. Bagozzi, and P. R. Warshaw, "User Acceptance of Computer Technology: A Comparison of Two Theoretical Models," *Management Science*, vol. 35, pp. 982-1003, 1989.
[18] V. Venkatesh and F. D. Davis, "A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies," *Management Science*, vol. 46, pp. 186-204, 2000.
[19] I. Ajzen, "The theory of planned behavior," *Organizational Behavior and Human Decision Processes*, vol. 50, pp. 179-211, 1991.
[20] T. Dybå, N. B. Moe, E. M. Mikkelsen, S. Ict, and N. Trondheim, "An empirical investigation on factors affecting software developer acceptance and utilization of electronic process guides," *Software Metrics, 2004. Proceedings. 10th International Symposium on*, pp. 220-231, 2004.
[21] S. J. Mellor, *MDA Distilled: Principles of Model-Driven Architecture*, 2004.
[22] S. Ambler, *The Object Primer : Agile Model-Driven Development with UML 2.0*: Cambridge University Press, 2004.
[23] Sparx Systems, "Enterprise Architect," vol. 2007, 2007.
[24] S. Johnston, "UML 2.0 Profile for Software Services," vol. 2007: IBM, 2005.
[25] T. Erl, *Service-Oriented Architecture Concepts, Technology, and Design*. Crawfordswille, Indiana, USA: Prentice Hall, 2006.
[26] S. Sharma and A. Rai, "CASE deployment in IS organizations," *Communications of the ACM*, vol. 43, pp. 80-88, 2000.
[27] R. Likert, "A Technique for the Measurement of Attitudes," *Archives of Psychology*, vol. 140, pp. 1-55, 1932.
[28] T. Dybå, "An Instrument for Measuring the Key Factors of Success in Software Process Improvement," *Empirical Software Engineering*, vol. 5, pp. 357-390, 2000.
[29] J. Miller, "Statistical significance testing--a panacea for software technology experiments?," *Journal of Systems and Software*, vol. 73, pp. 183-192, 2004.
[30] S. Walderhaug, M. Mikalsen, G. Hartvigsen, E. Stav, and J. Aagedal, "Improving Interoperability in Healthcare Using Model Driven Software Develoment for Healthcare," presented at MEDINFO Brisbane, Australia, 2007.
[31] J. Mansell, A. Bediaga, R. Vogel, and K. Mantell, "A Process Framework for the Successful Adoption of Model Driven Development," *LECTURE NOTES IN COMPUTER SCIENCE*, vol. 4066, pp. 90, 2006.
[32] E. Rios, T. Bozheva, A. Bediaga, and N. Guilloreau, "MDD Maturity Model: A Roadmap for Introducing Model-Driven Development," *LECTURE NOTES IN COMPUTER SCIENCE*, vol. 4066, pp. 78, 2006.