# A formal MDA approach for mobile health systems

Val Jones, Arend Rensink, Theo Ruys, Ed Brinksma and Aart van Halteren.

Department of Electrical Engineering, Mathematics and Computer Science
PO Box 217, 7500 AE Enschede, The Netherlands
v.m.jones@utwente.nl

**Abstract.** M-health systems are safety critical systems intended for use by the public and are therefore characterized by especially strict requirements relating to safety, security, correctness, reliability, adaptability and user friendliness. This position paper proposes a methodology which realizes the MDA approach by utilizing formal methods to support verification, validation and transformation. The objective is to investigate the use of MDA enriched by formal methods to define a generic, evolvable architecture for m-health services which facilitates the rapid development and deployment of high quality adaptable m-health services.

## 1 Introduction

Currently available m-health systems range from simple alarm functions through patient monitoring functions to complete disease management systems. These systems tend to be closed, proprietary systems targeted at a single health condition or physiological measurement. Our vision is of an open and generic m-health service platform which can support an unlimited and evolving range of m-health devices and services including applications requiring high speed high bandwidth transmission and sophisticated analysis and interpretation of time-oriented clinical data [1], [2]. Such an m-health platform should support any combination of functionality sets allowing services to be customized to the needs of the individual at a certain point in time. It should also be accept on-the-fly upgrades to existing applications as well as completely new services. The service platform must therefore be (hardware and software) platform independent, flexible and adaptable.

The approach proposed here is a realisation of the MDA approach using formal methods to provide a sound foundation for the rapid development of mobile health systems. Formal methods are applied to support validation (by prototyping, model checking and formally based testing) and model transformation. The resulting methodology is expected to yield a robust software engineering approach for the development of mobile health services and applications.

The concept arises out of work undertaken in European projects including two FP5 IST Take Up Actions, MobiHealth (IST-2001-36006) and XMOTION (IST-2001-36059), which were completed in 2004. The research also draws on work at the University of Twente on model checking and on automatic test generation, implementation and execution. In the MobiHealth project a prototype health BAN (Body Area Network) was developed and trialled in various clinical settings. Many

research issues arising from the experience gained are investigated in various new projects including the Dutch FREEBAND projects A-MUSE and AWARENESS and European initiatives MOSAIC (FP6-IST-2003-2 004341) and the Ambient Intelligence_at_Work initiative of the IST New Working Environments Unit. This paper discusses one of the lines of research arising, relating to software engineering methodologies. The approach proposed targets the rigorous development of a generic architecture for evolvable mobile health systems.

## 2 The m-health vision

### 2.1 Body Area Networks for healthcare

Body Area Networks [3], [4], [5] combined with wireless communications give a technology platform for realising the m-health vision. We define a *BAN* as a network of wearable devices which communicate amongst themselves (intra-BAN communication) and which may also communicate externally with a remote location (extra-BAN communication). A BAN consists of a mobile base unit or *MBU* (a central processor and gateway performing computation and external communication functions) and a set of devices. The MBU could be a PDA or a smart phone.

Specialising this concept, an *m-health* BAN is defined as a network of wearable medical devices which communicate amongst themselves (eg via Bluetooth) and externally (eg. via GPRS or UMTS) with a remote healthcare location such as a hospital system, a medical call centre or a doctor's mobile system. Examples of medical devices which may be incorporated into a BAN are sensors (e.g. electrodes for measuring ECG, EMG or EEG) and actuators (for example controlling implanted drug delivery systems or pacemakers). There may be any number of different specialisations of the health BAN. A specialisation can be thought of as an extension of the generic health BAN by equipping it with a certain (set of) device set(s) and the associated software. An example would be a BAN for insulin dependent diabetes patients. The diabetes BAN could include two devices: a blood glucose monitor (sensor) controlling an implanted insulin pump (actuator). The diabetes management application could include of a set of distributed functions running locally on the BAN or remotely, or a mixture of the two. The distributed nature of the execution should be hidden from the user. Several specialisations of a health BAN have been trialled during the MobiHealth project [6], [7], [8], [9], [10].

### 2.2 Special requirements of m-health systems

Mobile healthcare systems for patients are safety critical systems intended for (possibly unsupervised) use by the public. These systems are therefore characterized by strict requirements relating to *safety, security, correctness, reliability* and *user friendliness*. In addition, the prospect of large scale deployment of m-health systems in the community brings requirements for *scalability*, *run-time adaptation* (eg. in response to changing network conditions) and *dynamic evolvability*. Finally, m-health

systems should be based on a *generic architecture*. We need robust methodologies to support the development of such safety critical systems. Here we focus on *correctness*, *evolvability* and *genericity* properties.

## 3 The approach

The objective is to contribute to the rigorous development of a software architecture which is able to support a variety of future BAN-based m-health services. The intention is to apply OMG's Model Driven Architecture™ (MDA) [11], [12], [13], augmented by formally-based software engineering methodologies and tools, to the m-health application domain. MDA is selected because it addresses the complete development life cycle and promises portability, cross-platform interoperability, and platform independence. In particular it is selected to support *genericity* and *evolvability* of the architecture and *domain specific modelling*. In our application of MDA to m-health we emphasise the need for formality and make explicit the activities of verification and validation. MDA is thus enhanced with formal methods in order to support the critical *correctness* requirements of health systems. Formal methods will be used to support verification (by model checking) and validation (by model-based testing) of critical properties, and to test equivalence between models and implementations. Model checking enables verification of logical consistency and correctness properties of a specification and detection of a variety of errors and undesirable characteristics such as deadlocks and race conditions. Together with formal testing, model checking can give a high degree of confidence in the correctness of the design and implementation (ie of PIM, PSM and code).

### 3.1 Combining MDA and formal methods

Figure 1 depicts a concept space for instantiation of the MDA approach, showing some candidate formalisms and implementation environments, and the role of meta-models and model transformation in deriving implementations.

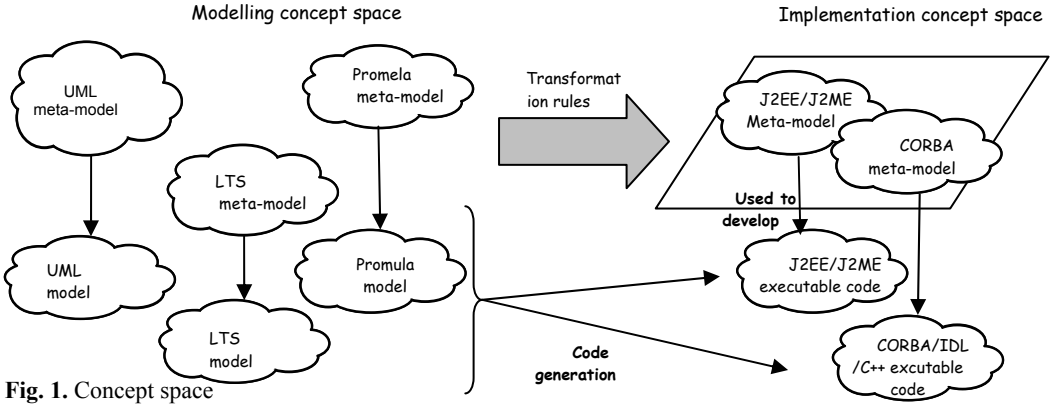


**Fig. 1.** Concept space

The MDA approach is applied by developing a Platform Independent Model (PIM) and transforming it to one or more Platform Specific Models (PSMs) targeted at specific implementation environments. Applications are derived from the PSMs for those specific platforms. Model transformation refers to meta-models (models of the source and target languages/environments).

Complete proofs of correctness are demonstrably not feasible for realistic sized systems; however, we propose to use formal methods within the MDA framework to establish a high quality software production process which can give high levels of confidence in the correctness of the designed system. Formal validation techniques used include early prototyping (model execution by simulation); model verification; and model-based testing of implementations. The guidelines of [14] will be followed so that the formal verification is performed in a controlled and reproducible way.

Modelling is performed using executable formal or semi-formal languages (e.g., UML, OCL, *me too*). Verification approaches include model checking [15] with tool support (e.g. SPIN [16], [17]); for validation we use model-based testing (automatic test generation and execution [18] using tools such as TORX [19]) and rapid prototyping (e.g. the *me too* approach [20]). Possible implementation approaches include the transformation approaches of [21], [22], [23] and model transformation [24].

### 3.2 Some anticipated challenges

Although promising a usable software development process targeting interoperability, reusability and portability, MDA raises some interesting challenges, including:

1. How to represent the dynamic aspects of systems?
2. How to address what we may call the "Lossy transformation" problem; when the expressive power of the source language exceeds that of the target?
3. How to establish preservation of semantic properties - a problem made more intractable where the source or target language of a transformation lacks an explicit formal semantics?
4. How far can we go with auto generation of implementations from models?

### 3.3 Some proposed solutions

We will consider alternative formalisms to represent behaviour (e.g. process calculus and models based on generalised transition systems) in order to address problem 1 above. As well as UML we consider other more formally defined languages (including but not confined to the UML related OCL) in order to detect problem 2 and to address problem 3. (Adding alternative formalisms to the MDA repertoire implies development of meta-models, transformation definitions and (possibly) additional tools.) Problem 4 refers to the point that automatic generation of complete applications remains an unreachable goal. Generally parts of an implementation must be hand crafted. We propose to investigate how model transformation using formal methods can be applied where possible and then augmented by judicious use of principled software engineering techniques for development and validation of the

remainder. A practical and scalable example which can form part of the solution is verification of the implementation by application of test suites automatically generated from the (platform independent) model.

Figure 2 shows one possible instantiation of our approach. An m-health application is modelled in UML, yielding a PIM (Platform Independent Model). Critical properties derived from the requirements are expressed formally (eg. as assertions). The UML model is transformed into a PROMELA model. The resulting PROMELA model together with the properties are input to the SPIN model checker, which verifies that these properties are met by the PROMELA model. So a degree of formal verification is achieved by model checking applied to the Promela version of the PIM. In this example, the target is a Java implementation. Applying model transformation again, a Java PSM is generated from the PROMELA PIM and Java code is derived from the Java PSM. A test suite is automatically generated from the PROMELA PIM using the test generation and execution tool TORX. The test suite is applied not to the model but to the Java implementation, providing formal validation by checking behavioural equivalence between model and implementation.

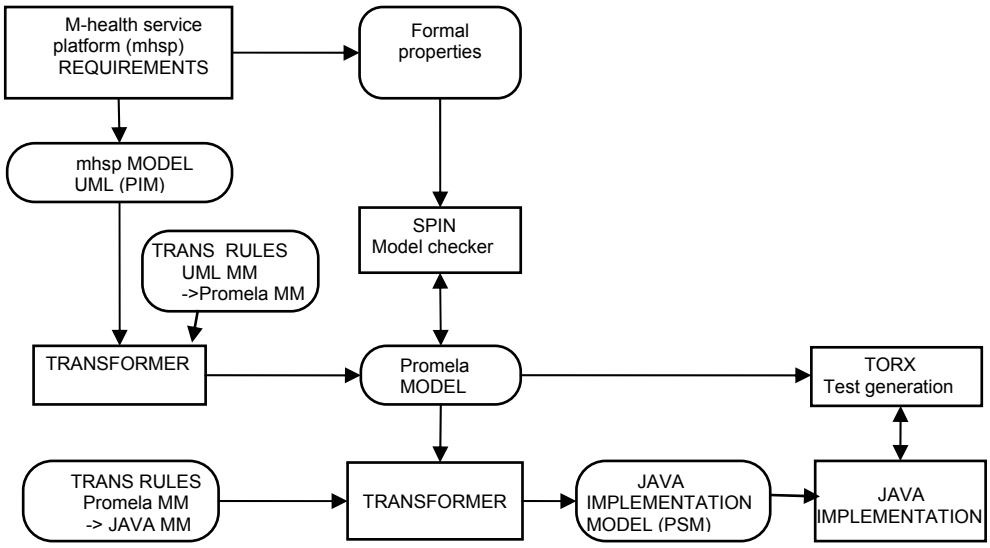


**Fig. 2**. One instantiation of the approach

We postulate a "Transformer": a generic model transformation tool which accepts a set of transformation rules mapping language A to language B, and a model in language A, and automatically produces a model in language B which is behaviourally equivalent to the source model. Since as yet we have no such "omnipotent transformer" guaranteeing correctness preservation, we still need formal validation of the implementation by model-based testing.

Other possible instantiations of the approach will result, for example, from use of different modeling formalisms (eg. *me too* plus process calculus), or because different implementation platforms (eg Symbian) or languages (eg C#, SQL) are targeted, or by

substitution of different validation methods (eg prototyping and/or simulation in place of model checking).

## 4 Discussion

The scientific focus of the proposed research lies in the investigation and advancement of software engineering methodologies for the development of domain specific services. This is achieved by testing theoretical developments from software engineering and formal methods against a real and complex engineering problem from the m-health domain and by instantiating the MDA approach for that domain. It is hoped that the research will increase understanding of the following issues:

- What are the real engineering challenges encountered by developers of distributed m-health services?
- How can we best model, validate and implement a software infrastructure that can be deployed in a distributed m-health service environment?
- What properties must a generic m-health architecture have in order to persist and support m-health product families (synchronic variation) and evolution of m-health products and services (diachronic variation)?
- How far can a fusion of the software engineering approaches of MDA and formal methods address these engineering challenges?
- Where are the boundaries between domain specificity and genericity (of models, model transformations and solutions)?

By exercising the chosen methods and tools on realistic m-health applications, we expect to derive a domain specific architecture for m-health services and a formally-based instantiation of the MDA approach. Hence the expected outputs include:

- A high level architecture for m-health services
- An MDA-oriented methodology for design and development of m-health services
- A proof of concept in the form of one or more applications of the methodology through to implementation. This will include models, meta-models, model transformations and prototype implementations.

The concept is in an early stage of development. Feedback from the MDA community is welcomed. It is hoped that through the proposed research we can make a contribution to MDA activities (eg via QVT [25]). Some of the QVT proposals are amenable to formalisation. It has been noted (eg. [26], [27]) that the theory of graph transformation appears to be especially suitable for the purposes of model transformation. If model transformation is defined on a formal footing, one can also expect to carry over formal verification results from one model to another.

## References

1. Shahar, Y., and Musen, M.A. (1993). RÉSUMÉ: A temporal-abstraction system for patient monitoring. *Computers and Biomedical Research* **26**, 255–273. Reprinted in van Bemmel,

J.H., and McRay, A.T. (eds) (1994), *Yearbook of Medical Informatics 1994*, pp. 443–461, Stuttgart: F.K. Schattauer and The International Medical Informatics Association.

2. Shahar, Y., and Musen, M.A. (1996). Knowledge-based temporal abstraction in clinical domains. *Artificial Intelligence in Medicine* **8** (3), 267–298.

3. Zimmerman, T.G., 1999, 'Wireless networked devices: A new paradigm for computing and communication', *IBM Systems Journal*, Vol. 38, No 4.

4. Van Dam, K, S. Pitchers and M. Barnard, 'Body Area Networks: Towards a Wearable Future', Proc. WWRF kick off meeting, Munich, Germany, 6-7 March 2001; http://www.wireless-world-research.org/.

5. Schmidt, R., 2001, *Patients emit an aura of data*, Fraunhofer-Gesellschaft, www.fraunhofer.de/english/press/md/md2001/md11-2001_t1.html

6. Jones, V. M., Bults, R. A. G., Konstantas, D., Vierhout, P. A. M., 2001a, Healthcare PANs: Personal Area Networks for trauma care and home care, *Proceedings Fourth International Symposium on Wireless Personal Multimedia Communications* (WPMC), Sept. 9-12, 2001, Aalborg, Denmark, http://wpmc01.org/, ISBN 87-988568-0-4

7. Dimitri Konstantas, Val Jones, Richard Bults, Rainer Herzog, *MobiHealth – innovative 2.5 / 3G mobile services and applications for healthcare*, 11th IST Mobile Summit 2002, Thessaloniki, May 2002.

8. Dimitri Konstantas, Val Jones, Richard Bults and Rainer Herzog, "MobiHealth - Wireless mobile services and applications for healthcare*", International Conference On Telemedicine - Integration of Health Telematics into Medical Practice*, Sept. 22nd-25th, 2002, Regensburg, Germany.

9. Widya, A. van Halteren, V. Jones, R. Bults, D. Konstantas, P. Vierhout, J. Peuscher, 2003. Telematic Requirements for a Mobile and Wireless Healthcare System derived from Enterprise Models. *Proceedings IEEE ConTel 2003: 7th International Conference on Telecommunications, June 11-13, 2003*, Zagreb, Croatia.

10. Nikolay Dokovsky, Aart van Halteren, Ing Widya, BANip: Enabling remote healthcare monitoring with Body Area Networks, International Workshop on scientiFic engIneering of Distributed Java applIcations, November 27-28, 2003, Luxembourg, LUXEMBOURG.

11. Object Management Group, MDA website, http://www.omg.org/mda/

12..MDA Guide Version 1.0.1, © 2003, OMG, omg/2003-06-01, http://www.omg.org/docs/omg/03-06-01.pdf

13. Anneke Kleppe, Jos Warmer, Wim Bast, (2003) *MDA Explained: The Model Driven Architecture™: Practice and Promise*, Addison Wesley Professional.

14. Theo C. Ruys and Ed Brinksma, Managing the Verification Trajectory, *Software Tools for Technology Transfer* (STTT), 4:2, Feb. 2003, pp.246-259.

15. Theo C. Ruys, (2001), *Towards Effective Model Checking*, PhD Thesis, University of Twente, Enschede, The Netherlands,March 2001.

16. G.J. Holzmann,(1991) Design and Validation of Computer Protocols, Prentice Hall, New Jersey, 1991, ISBN 0-13-539925-4.

17. G.J. Holzmann, (2003) The Spin Model Checker: Primer and Reference Manual, Addison-Wesley, ISBN 0-321-22862-6.

18. E. Brinksma, (1999). Formal methods for conformance testing: Theory can be practical! In N. Halbwachs and D. Peled, editors, *Computer Aided Verification (CAV)*, volume 1633 of *Lecture Notes in Computer Science*, pages 44-46, Trento, July 1999. Springer.

19. J. Tretmans and A. Belinfante. Automatic testing with formal methods. In *EuroSTAR'99: 7th European Int. Conference on Software Testing, Analysis & Review*, Barcelona, Spain, November 8-12, 1999. EuroStar Conferences, Galway, Ireland.

20. Alexander H and Jones V (1990). *Software Design and Prototyping using* me too. London: Prentice Hall International. ISBN 0-13-820259-1.

21. Correctness Preserving Transformations for the Early Phases of Software Development; *T. Bolognesi, D. De Frutos, R. Langerak, D. Latella.I,IN* Bolognesi T, van de Lagemaat J and Vissers C.A. (ed), *LOTOSphere: Software Development with LOTOS*, pp. 348-368, Kluwer Academic Publishers, 1995.

22. Jones V (1995). Realization of CCR in C, In Bolognesi T, van de Lagemaat J and Vissers C.A. (ed), *LOTOSphere: Software Development with LOTOS*, pp. 348-368, Kluwer Academic Publishers, 1995.

23. Jones VM (1997) *Engineering an implementation of the OSI CCR Protocol using the information systems engineering techniques of formal specification and program transformation*. University of Twente, Centre for Telematics and Information Technology Technical Report series no. 97-19. ISSN 1381-3625.

24. Bézivin, J., Dupé, G., Jouault, F., Pitette, G., Rougui, J.E.: *First experiments with the ATL model transformation language: Transforming XSLT into XQuery.* OOPSLA 2003 Workshop, Anaheim, California, October 27, 2003

25. OMG MOF 2.0 Query / Views / Transformations Request for Proposals. URL: http://www.omg.org/cgi-bin/doc?ad/2002-4-10

26. Sabine Kuske, Martin Gogolla, Ralf Kollmann, Hans-Jörg Kreowski, An Integrated Semantics for UML Class, Object and State Diagrams Based on Graph Transformation. In Butler et al., *Integrated Formal Methods, Third International Conference*, LNCS 2335, Springer 2002, pp. 11-28.

27. T. Mens. Conditional graph rewriting as a domain-independent formalism for software evolution. In Nagl et al, editors. *Applications of Graph Transformations with IndustrialRelevance,* volume 1779 of Lecture Notes in Computer Science. Springer-Verlag, 2000, pages 127–143.