e-Informatica Software Engineering Journal, Volume 7, Issue 1, 2013, pages: 7–14, DOI 10.5277/e-Inf130101

Comparison of MDA and DSM Technologies for the REA Ontology Model Creation

Zdeněk Meliš*, Jaroslav Žáček*, František Huňka*
*Faculty of Science, University of Ostrava
zdenek.melis@osu.cz, jaroslav.zacek@osu.cz, frantisek.hunka@osu.cz

Abstract

Using the ontology in information systems means an improvement of the possibility of solving tasks based on domain knowledge. The quality design of the ontological model is base of well-functioning system. In this area the most commonly used technology is MDA (Model driven development), which provides solid base for modeling language's metamodel definition. This article aims to compare this technology with a new approach to visual modeling – the DSM (Domain-specific modeling) technology. Using the narrow focus on specific domain and full code generation the DSM allows easy and rapid development of the ontological system. The REA (Resource–Events–Agents) ontology, intended for business processes modeling was used for comparison of these technologies.

1. Introduction

Currently the interest in developing information systems based on ontology is growing. With an increasing complexity of such systems, requirements for modeling principles with the high level of an abstraction to be able to transform created models to basic structures of an information system are growing [1]. The most commonly used technology is MDA that specifies functional details by a progressive model transformation. The DSM technology operates on a different principle. It allows creating models in a domain language and performs model validation and verification and full code generation.

First paragraphs describe the definition of the REA ontology and compared technologies – MDA and DSM, their principles and characteristics. The second part of the article deals with their comparison.

2. The REA ontology

According to [2] the ontology is a specification of a conceptualization. It is study of things that exists or can exist in explicit domain. A conceptualization means an abstraction and simplified view of the world. A specification means formal and declarative representation [3]. The ontology provides number of resources for intelligent systems, knowledge representation and knowledge engineering processes [4].

The REA is a concept for designing enterprise infrastructures based on ownership and their exchange. It is based on a concept of economic exchanges and conversions that increases company's value. According to [5] the ontology basis contains 5 parts, as you can see on Figure 1:

– Economic resource – elementary economic resource that company wants to plan, monitor and control. This resource can include raw material, money, work, labor, etc.

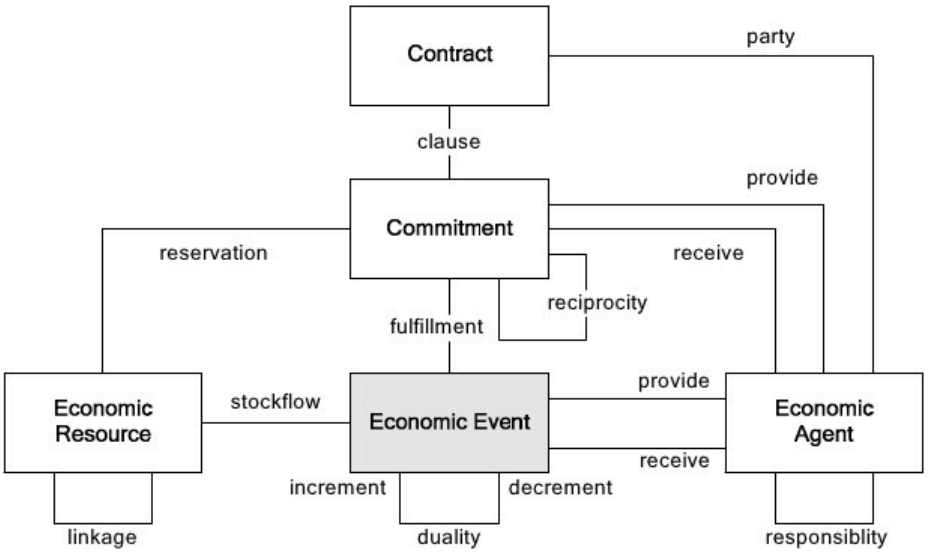

Figure 1. Fundamental REA concepts [5]

– Economic agents – an individual, group or company that controls economic resource and cooperate with other economic agents. An example can be customer, seller, employer, company etc.
– Economic event – represents economic change of resource (an increment or a decrement). This change can be immediate or long-term. An example can be work, using of services, renting, etc.
– Commitment – promise or obligation to perform an economic event in future.
– Contract – a set of commitments and rules (e.g. what happen when the commitment is not fulfilled)

Figure 1 shows fundamental model of the main concepts of REA ontology and links between them. One of the most important links is duality that answers us the question why an economic event occurs.

2.1. Levels distribution of the REA model

We can extend a basic view level by adding other entities to obtain an additional functionality. The REA ontology can be divided according level of an abstraction [6]:

– Value system level – the highest level of an abstraction. It describes the resources flow between the enterprise and its business partners.
– Value chain level – it divides an enterprise into strategically important activities. The enterprise gains a competitive advantage by doing these activities cheaper and better than competitors [7]. The view of this level describes resources flow between individual business processes.
– Model level – models describes transformation one economic resource into other, more valuable for enterprise. Figure 2 shows some concepts of model level and their links.
– Task specification level – the lowest level of an abstraction, it is an application of the model and contains raw company's data.

Generally the REA model level is divided into 2 levels according functionality [5]:

– Operational level is the basic skeleton of model. It describes events that already happened. Basic semantic abstractions of operational level are exchange, conversation and the value chain. Exchange and conversation increase an enterprise value and the value chain describes connection of various REA models into chain directly or indirectly contributing creation of desirable features of the final product or service. That final product can be exchanged for a more valuable resource with other economic agents.

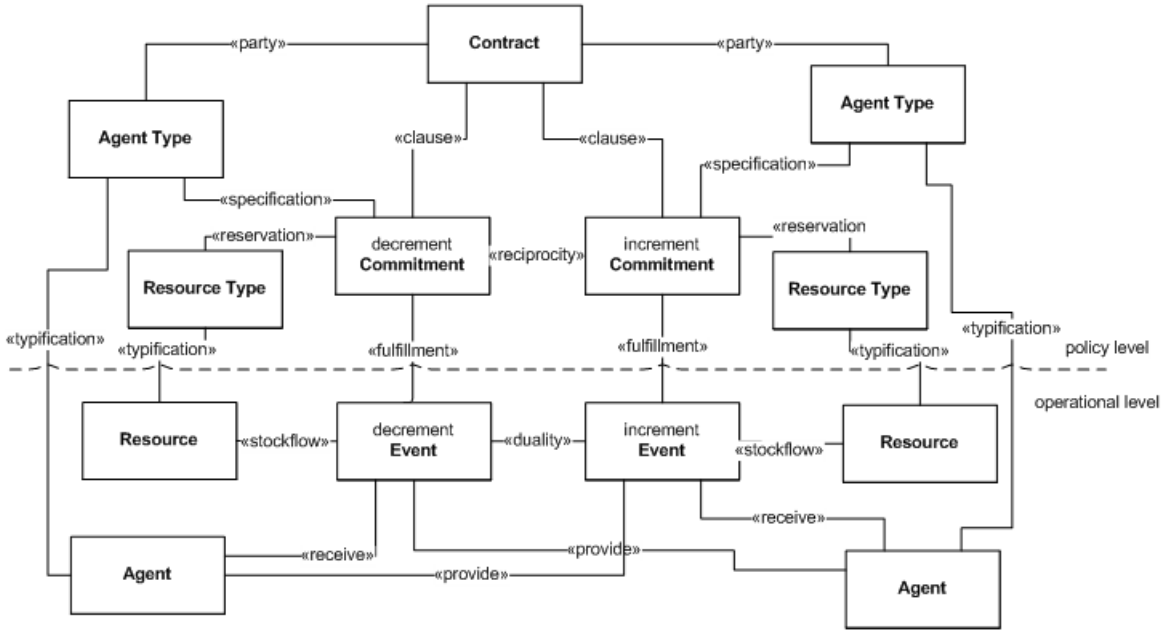

Figure 2. Example of business process model

– Policy level is an extension of operational level. It contains semantic abstractions describing what could, should or shouldn't happen such as group, type, contract, etc.

Figure 2 shows an example of business process model described by REA model level. It contains basic concepts of operational level and some semantic abstractions of the policy level.

2.2. Usage of the REA framework

Traditional business processes modeling approaches, such as flow chart, data model, use case, IDEF0, etc. use general concepts that are inappropriate because of low model specificity and therefore they cannot detect economical errors and make automation. The REA ontology uses specified concepts increasing amount model information while maintaining the model simplicity.

The REA model has internal rules for verifying the model consistency thus prevents creating of incorrect links. The result of this verification is the model, which is responding to an answer why the enterprise performs some activity and hence why economic events happened. This is a significant difference and a big advantage of the REA ontology over other traditional model solutions.

Another feature of the REA ontology is simplicity and understandability of models for ordinary users working with them. The model is also precise enough for automation [5].

3. Model Driven Architecture

MDA is a specification of OMG consortium (Object managment group) used for model driven software development. Model is simplified view of reality that defines formal set of elements to describe an objective and a purpose of development.

The reason for creating the MDA standard was an effort to increase the level of abstraction. Anytime in the history increasing level of abstraction led to increasing productivity.

3.1. MDA architecture

MDA is based on four-layer architecture (see Figure 3). The highest layer M3 is meta-metamodel. It is an abstract language and a framework for defining, specifying, designing and managing technologically independent metamodels and

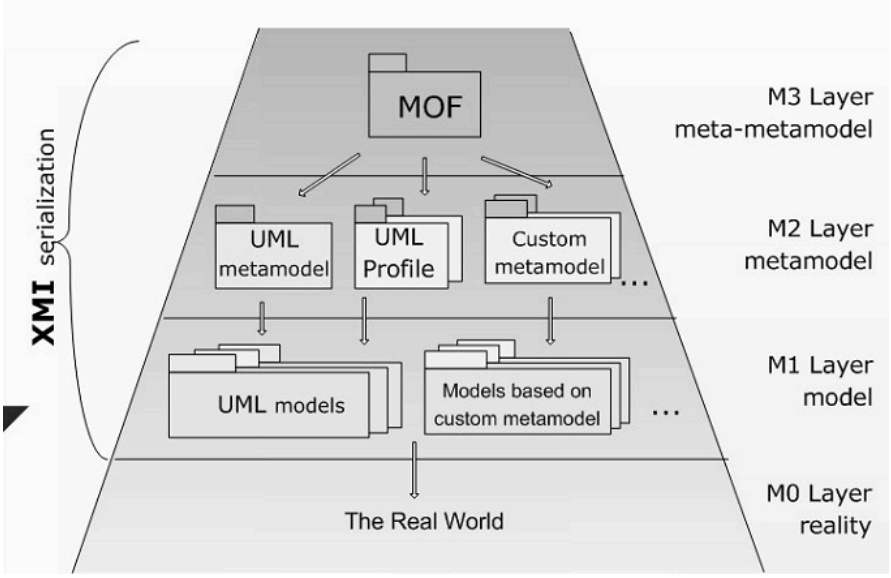

Figure 3. MDA architecture [3]

serves as the base for defining modeling language. The most commonly used language is Meta-Object Facility (MOF).

The second layer M2 contains all metamodels and specifications defined by top layer. In this layer there is usually defined UML language and its concepts (e.g. classes, associations, ...). The third layer M1 includes real world elements represented by metamodel concepts. The lowest layer M0 contains things of real world modeled in M1. This layer can include instances of concepts defined in M1 layer, or specific or abstract instances of real world.

3.2. MDA levels of abstraction

MDA defines 4 levels of an abstraction:

- CIM (the computational–independent model) – contains basic domain model with low level structure details. It models basic system requirements and basic business processes.
- PIM (the platform-independent model) – it is a model containing more details than CIM, but it is still free from technological details. Theoretically the PIM is assumed to be executed on a technologically independent virtual machine. This model describes the most of a system behavior.
- PSM (the platform-specific model) – in the PSM details such as the code structure for selected platform are generated, and constructs of the final language and details enabling code generation are completed (usually automatically).
- Code – the source code can be understood as the model of concrete realization on the platform.

3.3. Ontology infrastructure based on the MDA

The OMG Company tried to create an ontology system based on MDA to approach ontology systems to common programmers.

Figure 4 shows the ontology-based infrastructure made by MDA. The highest layer M3 contains basic MOF defining meta-metamodel. The M2 layer contains basic UML ontological profile describing basic modeling language's constructs and Ontology Definition Metamodel (ODM), which includes basic ontology concepts. This language is based on OWL, which is the result of the evolution of ontology languages. This layer forms a logical layer of the semantic web and allows mapping from ODM to OWL using XMI and XSL format (based on XML). The M1 layer contains general models based on higher layer

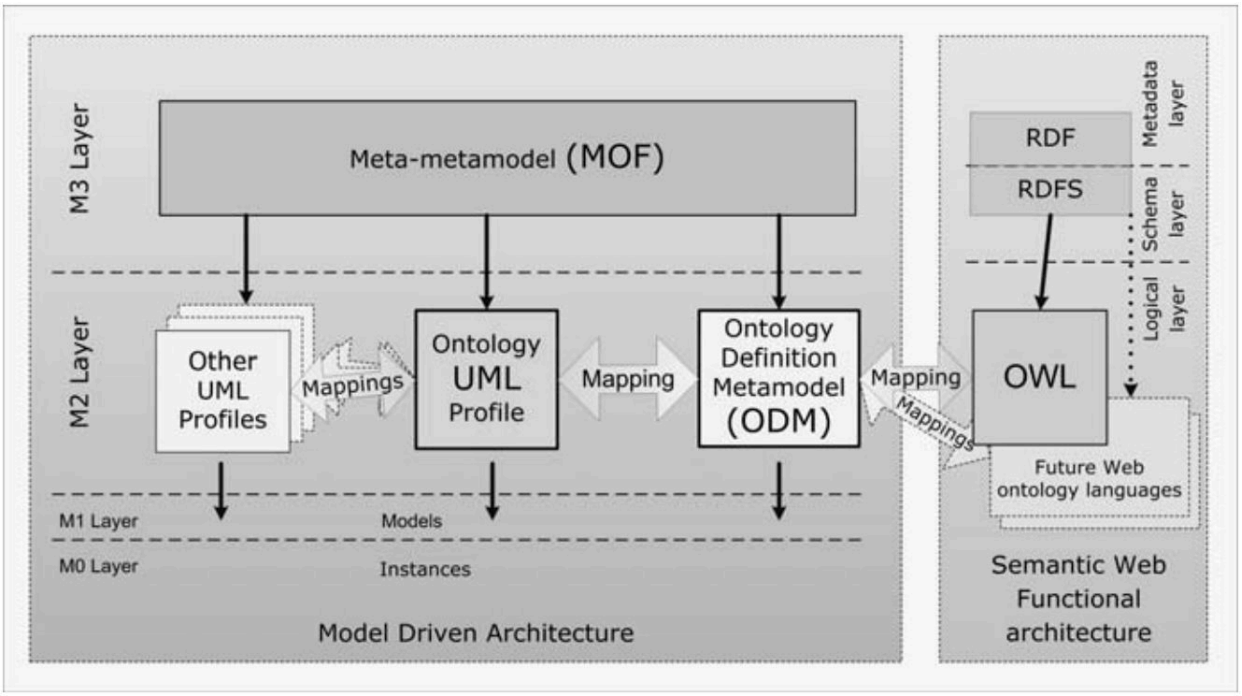

Figure 4. MDA based ontology infrastructure [3]

metamodels and M0 contains model instances and raw data.

MDA technology does not distinguish the type of the ontology - the general transformation process can be applied to any ontology, including the REA. The highest layer is MOF, M2 layer includes the REA ontology UML profile and M1 layer consists individual models.

4. Domain-specific modeling

Domain-specific modeling (DSM) is a software engineering methodology for software development [8]. The main focus of DSM is automated development in one narrow specific domain. The advantage of narrow focus is possibility to use domain terms in language including its graphical notification. DSM increases the level of an abstraction above code concepts eliminating the need for mapping of domain concepts into another language and reduces mapping errors [9].

4.1. DSM architecture

The DSM architecture contains 3 parts: the language, the generator and the domain framework. The language comes directly from a problem domain and provides an abstraction for solving domain problems. Individual domain concepts can form language objects, links, properties or submodels. The language itself has two parts. The syntax specifies language concept structure and uses a grammar to perform model validation and verification at the language level. The semantic defines the meaning of individual elements.

The generator performs model transformation to pre-defined structure, usually the source code which is complete and immediately executable without any modifications. The generator can produce documentations, metrics, statistics, prototypes, tests and more.

The lowest layer is the domain framework that performs multiple functions. It reduces complexity of code, eliminates duplicity in code, de-

fines an interface for generator, provides integration with existing systems, etc.

4.2. Ontology infrastructure based on the DSM

The figure 5 shows detailed DSM mapping of the ontology system. The vertical part of diagram shows the hierarchical ontological structure where Upper Ontology defines abstract layer of concepts usable in all domains, Core Ontology defines concepts usable in similar domain, Domain Ontology defines specific domain concepts and Application Ontology defines concepts that are modified for DSL mapping [10].

Unlike MDA, the DSM technology is directly dependent on the modeled ontology. Due the strict dependence on the domain, procedure of creation architectural layers of modeling system is different for each ontology.

The language of the REA ontology is defined by the UML profile and contains basic entities, relationships rules that allow validation and verification of the model. The target platform is hidden from user and no other action is required for creating a functional model so the system is able to work with an instance level of model. The core of such system is the generator, performing a translation of the model into the target platform code. Making of such generator is very difficult, expensive and time-consuming and requires domain expert participation. Once the generator is created, models creation is very easy even for nontechnical people.

5. Comparing MDA and DSM principles

5.1. MDA result

The MDA is based on incremental transformations that specify model details through all abstraction levels. This approach allows transformation of any REA ontology model independently on used entities, because transformation process is not affected by the applied semantic abstractions. The MDA structure allows easy transformation of the REA ontology language to any other language (e. g. to OWL language for semantic web support) without modification of the modeling system's core. Very helpful function may be the learning ability - the model learn transformation details set by a model creator and reuse them at the next transformation of the model with the same combination of used entities.

The big problem of this approach is that a model creator must have programming skills to be able set up required model details. Although the learning ability can be helpful, it can never provide a full model transformation into source code. Also the transformation between different languages can lead to implementation errors during mapping and therefore it request higher testing demands and resources.

5.2. DSM result

DSM is based on narrow domain focus. The language of model comes directly from REA ontology. That allows performing validation and verification check and the model eliminates the need for mapping between any languages by using domain terms. Models are well readable for people working with REA ontology and they do not need any programming skill to create an executable application from model.

The biggest problem of this approach is limited number of semantic abstractions that the generator can contain. REA ontology contains a lot of different semantic abstractions and some of them replace some other, some of them cannot be used together etc. Therefore it is not possible to implement all combinations of them into generator. Another problem is that REA allows creating new semantic abstractions and every modification of domain language requires modification of generator and its recompilation and redistribution. Generator recompilation can cause older model incompatibility.

5.3. Comparing results

Development procedure of ontology systems using these technologies is different and each of

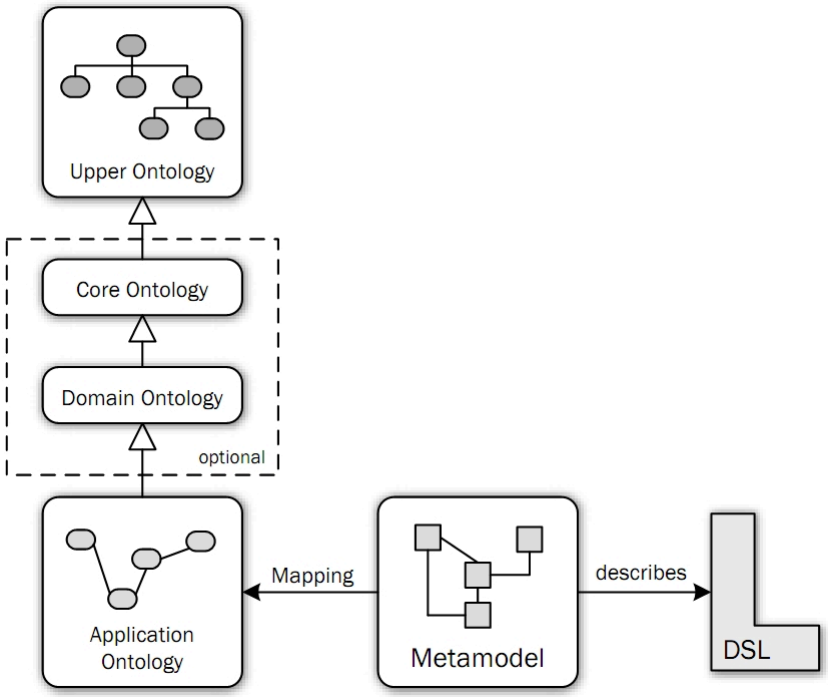

Figure 5. Ontological hiearchy [10]

them has positives and negatives. To model the general ontology, which assumes with possible changes in the structure, it is better to use the MDA technology but for the cost of no user friendly development and higher testing and resources requirements. In case of MDA each transformation between two languages has quadratic complexity $O(n^2)$ whereas the DSM model transformation has linear complexity $O(n)$ [10].

If only few semantic abstractions will be used (for example only one type of model will be created by application) it is much better to use DSM that has many advantages for REA ontology development. Nowadays there are some DSM modifications for supporting language changes without generator recompilation such as Expandable DSM generator [11].

The cost of development by MDA technology is equalized over whole developmental period. The DSM has largest costs at beginning of development, when the generator is built and subsequent development of applications based on the REA ontology has minimal costs because of higher abstraction level and thus increased productivity.

5.4. Extendable DSM generator

The basic idea of this approach is using some kind of pug-in system for extending the functionality of the generator. The modeling tool contains only basic semantic abstractions and all additional semantic abstractions are added through plug-ins (containing script for generator), which can be distribute via web services so modeling tool can automatically download and install needed packages. The solution of extendable DSM generator is described in [11].

6. Conclusion

Comparison of MDA and DSM technologies showed the fundamental differences and ways to using. Generally we can say that the REA ontology is better supported by DSM technology because it offers a lot of benefits such as using of the domain knowledge, the model validation and verification, low testing requirements and more. Due to architectural structure the DSM usage is limited by the narrow specific domain with the constant ontology and the limited count of

semantic abstractions. For that reason it is better to use the MDA technology if frequent changes of semantic abstractions of the REA ontology are expected. If large amount of modifications is not expected, but the ontology is not constant, the DSM extensible generator allowing changes of some semantic abstractions can be used [11].

Before the development begins it is necessary to perform basic analysis how will be the final system used and then choose which technology is better to that particular case.

Acknowledgement

The paper is supported by the grant reference no. 6141 provide by IGA Faculty of Science University of Ostrava.

References

[1] C. Chang and L. Ingraham, *Modeling and designing accounting systems: using Access to build a database*. John Wiley & Sons, Inc., 2007. [Online]. http://books.google.cz/books?id=NfzAAAAMAAJ

[2] T. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, Vol. Volume 5, No. 2, 1993, pp. 199–220, iSSN 1042-8143.

[3] D. Gasevic, D. D., and D. V., *Model Driven Architecture and Ontology Development*. New York: Springer Berlin Heidelberg, 2006.

[4] B. Bennett and C. Fellbaum, *Formal ontology in Information Systems*. IOS Press, 2006.

[5] P. Hruby, *Model-driven design using business patterns*, ser. Springer eBooks collection: Computer science. New York: Springer-Verlag, 2006.

[6] P. Hruby, M. Hucka, F. Huňka, J. Kasik, and D. Vymetal, *Víceúrovňové modelování podnikových procesů (Systém REA)*, ser. Series on Advanced Economic Issues. Ostrava: VŠB-TU Ostrava, 2010.

[7] H. Sedlackova and K. Buchta, *Strategicka analyza*. C H Beck, 2006.

[8] M. Fowler, *Domain-specific Languages*. Addison Wesley Longman, Inc., 2010.

[9] G. Guizzardi, L. Ferreira Pires, and M. van Sinderen, "On the role of domain ontologies in the design of domain-specific visual modeling langages," in *Proceedings of the 2nd OOPSLA Workshop on Domain-Specific Modeling Languages*, J. Tolvanen, Ed. Birmingham, USA: University of Alabama at Birmingham, 2002, pp. 25–38. [Online]. http://doc.utwente.nl/66750/

[10] M. Brauer and H. Lochmann, "Towards semantic integration of multiple domain–specific languages using ontological foundations," *Lecture Notes in Computer Science*, Vol. Volume 5021/2008, No. 2, 2008, pp. 34–48, available at WWW: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.9973.

[11] J. Žáček, Z. Meliš, and F. Huňka, "Extendable domain specific modelling generator based on web services," *ECON*, Vol. Volume 19, 2011, pp. 98–104.