A Product-Service System Proposal for Agile Modelling Method Engineering on Demand: ADOxx.org

Nesat Efendioglu[1], Robert Woitsch[2], Wilfrid Utz[3] and Damiano Falcioni[4]

Abstract: The importance of Modelling Method Engineering is equally rising with the importance of domain specific modelling methods and individual modelling approaches. In order to capture the most relevant semantic primitives that address domain specifics needs, it is necessary to involve both, method engineers as well as domain experts. Due to complexity of conceptualization of a modelling method and development of regarding modelling tool, there is a need of a guideline-, corresponding tools- and services-bundle supporting actors with different background along this complex process. Based on practical experience in business, more than twenty EU projects and other research initiatives, this paper introduces a product-service system to support the conceptualization of a modelling method on demand. The proposed product-service system is introduced and evaluated by three EU-funded research projects in the domain of multi-stage manufacturing, e-learning and cloud computing as well as additionally by an in-house development project in the area of decision modelling extensions. The paper discusses the evaluation results and derived outlooks.

Keywords: Meta-modelling, Modelling Method Design, Agile Modelling Method Engineering, Conceptualization, Modelling Method Engineering, Servizitation.

1 Introduction

The importance of Modelling Method Engineering is equally rising with the importance of Domain Specific Conceptual Modelling Methods and individual modelling approaches. In addition to existing (de-facto-) standards (e.g. Business Process Modelling Notation (BPMN), Decision Model and Notation (DMN), Case Management Model and Notation (CMMN)), a growing number of groups around the world design their individual modelling-methods (in accordance with the definition of such a method by [KK02] [Ka15]) for a variety of application domains.) The engineering of such applicable modelling tools as a result of the conceptualization process of modelling methods, is complex, especially when considering the mapping of the entire spectrum from language artefacts and corresponding functionality to concrete implementable and deployable modelling tool capabilities.

[1] BOC Asset Management GmbH, Innovation Group, Operngasse 20B, Vienna, 1040, nesat.efendioglu@boc-eu.com
[2] BOC Asset Management GmbH, Innovation Group, Operngasse 20B, Vienna, 1040 robert.woitsch@boc-eu.com
[3] BOC Asset Management GmbH, Innovation Group, Operngasse 20B, Vienna, 1040, wilfrid.utz@boc-eu.com
[4] BOC Asset Management GmbH, Innovation Group, Operngasse 20B, Vienna, 1040, damiano.falcioni@boc-eu.com

This is often seen as necessary, when model-based approaches are transferred in new application domains and hence require adaptations for modelling methods. A simple sample can demonstrated using the well-known standard for business process BPMN. Although BPMN can be used to design an administrative process, such as sending an invoice, it cannot be used to design a simple production process like producing a chair. The successor relation that indicates that one activity follows the other does not have properties like distance to the station, which is not necessary when sending an invoice, but is of utmost importance, when producing a chair. When analysing more complex scenarios like a car manufacturer shop floor (we faced in projects DISRUPT [DISRUPT17b], GO0DMAN [GO0DMAN17]), the adaptation requirements for a modelling language like BPMN becomes quite complex. Hence, providing well-known model-based approaches requires the adaptation by e.g. introducing the concept "distance" between two activities. On the one hand, in order to capture the most relevant semantic primitives that address domain specific needs, it is necessary to involve both the method engineers as well as domain experts; on the other hand for success of process for generation of modelling tools from design to deployment, it is necessary to enable knowledge exchange all stakeholders with varying backgrounds.

Challenging question is, how to support the generation of modelling tools that can range from a minor adaptation like the one introduced above, till the complete realisation of totally new modelling approach like a cyber threat modelling for cloud computing .

Today, there are different approaches, guidelines, tools and practices for the conceptualization of modelling methods and development of concrete modelling tools available that do not consider or support the full spectrum of the design and collaborative development of a modelling method, which unavoidably leads to limitations in the conceptualization of it [HKW13]. There is a need of a guiding framework, corresponding tools and services supporting method engineers along the complex conceptualization process taking all phases into consideration and ensuring collaboration among stakeholders involved in the process. Karagiannis proposes in [Ka15] the Agile Modelling Method Engineering (AMME) framework and authors of [ADOxx17b] propose the Modelling Method Conceptualization Process that based on AMME, guides the method engineers during conceptualization. The work at hand proposes a product-service system, whose core introduced in [EWU16], which supports agile modelling method engineering and conceptualization process. The product-service system (PSS) so-called "ADOxx.org PSS" aims to enable (1) efficient development lifecycle to produce professional modelling tools, (2) re-use of existing modelling method snippets/fragments, (3) re-use of existing platform functionalities (meta-functionality), (4) Involvement of domain experts in design of modelling method, (5) meta-model merge patterns to integrating mechanisms and algorithms, (6) multi-tool/standard merge mechanisms and finally (7) sustainability of results, while services supporting method engineering along conceptualization process taking Modelling Method Conceptualization Environment as basis.

Moreover the work evaluates the product-service system in four modelling method engineering cases, within three European Research projects, and one additional in the context of an in-house research project,

The remainder of the paper is structured as follows: Section 2 introduces the product services system and outlines Modelling Method Conceptualization Environment and Section 3 presents shortly Modelling Method Conceptualizations Services around Modelling Method Conceptualization Environment as product. Section 4 presents evaluation cases and results, while section 5 concludes the paper and gives an outlook on future work.

2 Modelling Method Conceptualization Environment

Having roots in software engineering, like in agile software development, during the modelling method engineering, involved stakeholders need procedures, structures and supportive tools allows high iterative process with as less as possible bureaucracy, and offers agile value and follows principles in Agile Manifesto [Pr17].

AMME is proposed in [Ka15] to support modelling method engineering during propagation and evolution of modelling requirements. The OMiLab Lifecycle [OMILab17] instantiates AMME and defines the internal cycle of a modelling method conceptualization with five phases; (1) “Create” as a mix of goal definition, knowledge acquisition and requirements elicitation activities that capture and represent the modelling requirements; (2) “Design” specifies the meta-model, language grammar, notation and functionality as model processing mechanisms and algorithms; (3) “Formalize” aims to describe the outcome of the previous phase in non-ambiguous, formal representations with the purpose of sharing results within a scientific community; (4) “Develop” produces concrete modelling prototypes and finally (5) “Deploy/Validate” involves the stakeholders in hands-on experience and the evaluation process as input for upcoming iterations.

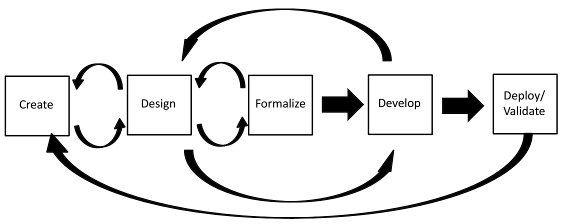

Fig. 3. Modelling Method Conceptualization Process

Due to the involvement of several stakeholders with varying knowledge backgrounds, perspectives and different objectives, in the conceptualization of a modelling method, the authors of [EWK15] propose so-called Modelling Method Conceptualization Process (as depicted in **Fig. 3**) by adding additional feedback channels into the OMiLab Lifecycle between: (1) Create and Design, to prove, if the designed modelling language covers the identified application scenarios and considers the identified requirements; (2) Design and Formalize to ensure formal approval of modelling language, as well as (3) Design and Develop - to improve modelling language in earlier stages before it is released and deployed. The work at hand introduces a new version of so-called "Modelling Method Conceptualization Environment toolbox that initially has been introduced in [EWU16] and that instantiates the above process and supports method engineers during each phase. The only exception is that of the "Create" phase, as this part is regarded as the most creative phase and standard tools and methods (also in some cases pen and paper can be the most appropriate tools) shall be freely selected. Modelling Method Conceptualization Environment proposes a combination of tools in sense of Integrated Development Environment (IDE), such as the Modelling Method Design Environment (MMDE, available at [LearnPAd17b]) for the Design, the ADOxx Library Development Environment (ALDE) and ADOxx, for Formalize and Develop, 2.3 Adoxx.org Build, Test and Deployment Services (available at [ADOxx17a]) for Deploy/Validate Phases.

As depicted in **Fig. 4**, typical life-cycle / usage scenario would be during the create phase domain experts and method engineers come together, identify requirements for modelling method, in design phase method engineers with tight collaboration of domain experts specifies the meta-model, language grammar, notation and processing functions on MMDE, as method engineers formalize design of modelling method collaboratively and commit on ALDE, developer(s) based on that formalization implements concrete modelling toolkit prototype within ALDE and ADOxx Development Toolkit. Developer(s) uploads the prototype into ADOxx.org build server, semi-automatic service behind starts completeness check, building installation package, testing of installation package and optionally deploy it on selected developer to download the toolkit, test it, validate by community members get feedback from them or the build services sends a link to owner to download and/or share the modelling toolkit. It is worth to mention that one of the objectives is to provide loosely coupled tools, so involved actors have the flexibility to decide to use one, a combination of tools from the toolbox or even use other appropriate tools of their choice, (e.g. method engineer uses MMDE during the Design Phase, but formalize the modelling method design with mathematical models or use another development tool during the Develop Phase and deploys them at the Developer Spaces and enable validation).

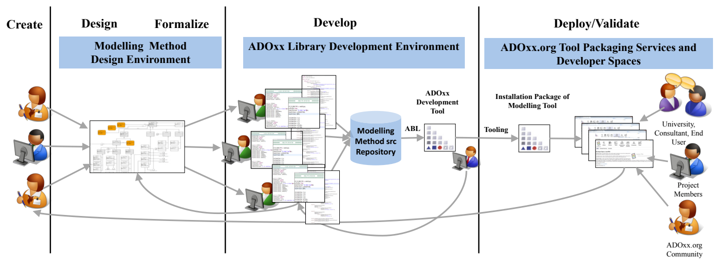
Create
Design
Formalize
Modelling Method Design Environment
Develop
ADOxx Library Development Environment
Modelling Method src Repository
ABL
ADOxx Development Tool
Deploy/Validate
ADOxx.org Tool Packaging Services and Developer Spaces
Installation Package of Modelling Tool
Tooling
University, Consultant, End User
Project Members
ADOxx.org Community

Fig. 4. Life-cycle within Modelling Method Conceptualization Environment from Users' Perspective

In the following sub-sections current abilities of the tools from the environment are shortly presented

2.1 Modelling Method Design Environment.

The Modelling Method Design Environment (MMDE) is itself a modelling tool to design other modelling methods. MMDE has been implemented based on lessons learned and the experience of the authors, who have been involved in modelling method/tool development activities in more than 20 EU research projects for varying domains. Based on these lessons learned and the results of a state of the analysis that we discussed in [EWK15], UML [OMG17] has been identified as a fitting starting point. Hence, the MMDE takes a subset of UML and extends it with required concepts and functionalities in order to overcome the following challenges **(Ch)**, which are identified by [EWK15] based on a state of the art analysis about specification of conceptual modelling methods:**Ch1,** "Requirements" model type is implemented that allows the elicitation of requirements, specifying their status as well as dependencies among them. The described requirements in this model type can be referenced to (a) all the modelling classes modelled in the related model type "Meta-Model" classes, (b) graphical notation (concrete syntax) definitions modelled in the "Graphical Notation" model type, (c) the "Modelling Stack" definition and (d) to the functionalities modelled in "Mechanisms & Algorithms" models. For **Ch2,** we extend the class diagram from UML with concepts, so method engineers can differentiate between class and relation class as well as relate different meta-models (-fragments) with each other using "Weaving" techniques as they are introduced in [EWK15][Wo14]. The modularization and layering of modelling language is essential to avoid complexities during the design of domain specific modelling methods [Se11] [KHW11]. Hence, we propose representation of the modelling stack as the "Meta-models Stack model type allowing method engineers to differentiate meta-models in sense of different model types that target different fragments of the system. In order to target **Ch3** and specify a proper graphical representation (concrete syntax) of each concept in a meta-model, we introduce another model type called "Graphical Notation" model type(allows definition of concrete syntax of model types with specifying graphical representations for each constructs in meta-models. This model type allows the description of graphical representations either assignment of vector graphics code written in GraphRep Language [ADOxx16] or with the assignment of concrete images including a description of the functionalities in the notation (e.g. attribute-value dependent visualization, context related views) In order to target **Ch4** to define the applicable modelling technique as steps and corresponding results we propose a model type called "Modelling Procedure" model type". The Modelling Procedure Model Type allows method engineers to define the steps with their required inputs and produced outputs, as well as the sequence of steps based on input – output relationships, in order to introduce case specific proper usage of their modelling method. Based on this procedural view. Further details about MMDE can be found in [EWK15].

2.2 ADOxx Library Development Environment.

The ADOxx Library Development Environment (ALDE) aims to enable formalization and parallel development of modelling tools libraries based on the designs deriving from Design Phase, merging different libraries and ensuring maintainability. As an experimental prototype ALDE uses the Resource Description Framework (RDF) as a format for data interchange [W3C17a] ALDE is a development environment based on the Eclipse IDE [Eclipse17a] and includes a meta²model defined in RDFS, the ALDE vocabulary. Having the vocabulary and utilizing Apache Ant® as a build mechanism [Apache17], the environment enables the definition of the transformation processes from ADOxx Library Languages to RDF and vice versa. Moreover ALDE serializes libraries in an arbitrary RDF format; for the prototypical realization RDF Turtle [W3C17b] has been used and includes the RDF XTurtle Editor developed by [AKSW17]. Having libraries in RDF Turtle format and a RDF Turtle Editor available, method engineers can adapt declaratively and script libraries collaboratively using standard functionalities of source-code management systems. Merging several libraries or integration of parts of libraries in each other becomes possible. The most important improvement is the integration of an ADOXX-JAVA-MM-DSL, which is developed based on feedbacks coming from evaluation cases of previous versions. The ADOXX-JAVA-MM-DSL is a framework that creates several abstraction layers over the ADOxx Library Language (ALL) format, the ADOxx internal language that describes a meta-meta-model. Every layer simplifies and adds features to the bottom one. The framework give so the possibility to operate and easily perform modification on a meta-model without dealing with its complexity. In order to assure that, an internal structure is managed that represents the ALL structure of the meta-model. This internal structure can be loaded from an existing ALL meta-model and can be exported as ALL as well. All the constraints and rules present in the ALL syntax are managed, so the framework can guarantee that only syntactically valid ALL meta-models will be loaded and generated.

2.3 Adoxx.org Build, Test and Deployment Services.

Adoxx.org Build, Test and Deployment Services [ADOxx17a] are web-based services that allow method engineers of the ADOxx community to build verified, professional and installable distribution packages that can be distribute to interested stakeholders. The service combines and validates all available inputs, integrates all elements, compiles the necessary artefacts and signs the outcomes and creates the actual installer as a download archive. As a collaboration space for the development and deployment phases, the concept of “Developer Spaces” has been introduced in ADOxx.org [ADOxx17b]. These spaces enable sharing of intermediate/release results, discussing development resources from all pre/past phases in the form of source code, snippet, examples and distribution packages with the community.

3 Modelling Method Conceptualization Services

In addition to the conceptualization tools described in the previous chapter, an appropriate support services are foreseen to support the modelling method engineers. The services are provided on the ADOxx.org portal, supporting a community of more than 1.300 modelling method engineers world-wide. The services are provided as follows;

1. **Download:** For the download, ADOxx.org provides the Meta Modelling Platform ADOxx in combination, Installation Instructions, Frequently Asked Questions, Startup-Package as well as a set of more than 30 available application libraries, which can be used to start with.
2. **Get Started:** For getting started, ADOxx.org provides important readings, provides a Forum that is structured according active communities, lists tutorial and training events that are offered free of charge, provides tutorial material for both the students – in form of guide samples and slides – as well as for the trainer – in form of a trainer handbook and offers tutorial videos and webinars.
3. **Development and Support:** For the development, ADOxx.org provides aforementioned tools and additional developer utilities, 3rd parties add on like but not limited to simulation, documentation, dashboards or Web-APIs. A collection of 200 graphical representations that introduce the major elements conclude the development support.
4. **Community:** For collaborative development within the ADOxx.org community a map is provided indicating the ten laboratories – nine in Europe, one in Asia, indicating the hot spots of developers, the participating research institutes, a set of 24 modelling tools as a result of [KMM16], and development spaces that enable a collaborative development and enable the use from solutions and tools from other projects.

5. **Documentation:** A complete specification and documentation is offered, where each relevant element of the modelling method is (a) explained based on the corresponding theory, (b) introduced with hand-on samples, (c) demonstrated with real-world scenarios, (d) mapped to forum entries of the community and finally (e) supported with tools where possible.

The operation of this service centre is provided via the portal, social media like Twitter, Facebook and LinkedIn, or via email. In justified cases an onsite support is also provided, where either the method engineer is trained, supported or critical implementation steps are performed by the ADOxx.org service centre.

4 Evaluation

Given that usually each modelling method engineering case differs from each other in sense of complexity of domain, variety of aspects to be targeted, number of involved actors, to calculate quantified evaluation means is difficult, and – to best of our knowledge - there is no similar conceptualization environment, hence, it is difficult to bench-mark our proposal and quantify the evaluation and provide statistically objective results. On the other hand, the most important tangible and objective evaluation result would be deployed and ready to use modelling toolkits, specification of modelling methods and communication of community members as proofs of concept. Those proofs of concepts for each are online and freely accessible (with exception of in-house project case). The links to access those proofs of concepts for each case are provided under regarding sub-section below.

The conceptualization environment introduced above has been applied in four different cases for evaluation: three EU-funded research projects in the domain of multi-stage manufacturing, eLearning and cloud computing and additionally in an in-house development project, in the area of decision modelling extensions into business process management.

Case 1: Conceptualization of a Modelling Method for E-Learning: The FP7 project Learn PAd [LearnPAd17a] proposes a process-driven-knowledge management approach based on conceptual and semantic models for transformation of public administration organizations into learning organizations. Learn PAd proposes a model-driven collaborative learning environment. In this case, 4 domain experts and method engineers have been involved. In addition, two developer teams, each consisting of 4 developers worked on the implementation of the tool. The results of the conceptualization process of this modelling method can be found at Learn PAd Developer Space [LearnPAd17b]as well as the developed prototypes [LearnPAd17c] can be downloaded and feedback can be given.

Case 2: Conceptualization of Modelling Method for Cloud Computing: The H2020 project CloudSocket [CloudSocket17a] introduces the idea of Business Processes as a Service (BPaaS), where conceptual models and semantics are applied to align business processes with Cloud-deployed workflows [WU15] . In this case, 6 domain experts and method engineers have been involved, as well as two developer teams, one with 5 developers, the other one with 2 members. The results of the conceptualization process of this modelling method can be found at CloudSocket Developer Space [CloudSocket17b], as well as developed prototypes [CloudSocket17c] can be downloaded and feedback can be given.

Case 3: Conceptualization of Modelling Method for holistic Manufacturing System Management: The H2020 project DISRUPT [DISRUPT17a] deals with the integration of innovative technologies into a holistic manufacturing system and optimization of production flow. The DISRUPT projects needs a modelling method to describe manufacturing system from supply-chain level down to shop-floor level. In this case 2 domain experts, one requirement engineer and one method engineer have been involved. Preliminary results can be found on DISRUPT Developers Space [DISRUPT17b].

Case 4: Integration of existing BPMN and DMN Modelling Methods: The in-house project requires integration of an already implemented DMN Modelling Method into existing BPMN 2.0 realization as part of a commercial product. In this case, 3 domain experts and method engineers, and a team of two developers have been involved.
The evaluation process was enacted in the following steps: (1) Provisioning: the tools -of the toolbox have been provided to the stakeholders in the involved cases. (2) Team Formation: representatives, - of the stakeholders in the project created development teams consisting of domain experts and method engineers following the conceptualization process and developing tools individually. (3) Feedback Phase: individual results have been consolidated periodically through video conferences and workshops, constituting the evaluation results.

Feedback on MMDE; Pro: It is possible to specify requirements and dependencies among them as well as tracing them; (2) to define modelling language fragments and modules, (3) layering the modelling language with navigational constructs; (4) definition of syntax, semantic and assignment of notation (concrete syntax); (5) definition of weaving among construct in different meta-models; (6) assignment of (multiple-) graphical notation (concrete syntax); (7) explicit definition of modelling procedure; **Contra:** It is not possible to define application scenarios and use cases, and design results can be exchanged solely using ADOxx specific formats or as static content (image, PDF or HTML). Hence, double effort in the design and in the formalisation and or development is currently necessary; **Outlook:** The MMDE is currently updated, to offer an XML export, which then can be transformed into different formats like the one that is used for the ADOxx-Java-MM-DSL.

Feedback on ALDE; Pro: it is possible to transform libraries in a machine as well as human interpretable format, ability to use reasoning algorithms, due to standard semantic formats; reduces complexity to edit, merge and maintain libraries; **Contra:** To take over results from Design Phase require manual steps.; it re-quires different transformation scripts for different meta-modelling technologies (such as ADOxx, EMF); **Outlook:** The semantic-based verification of meta model is seen as a useful extension of the ADOxx-Java-MM-DSL, hence an integration will be experimented. However, we see the necessary skill level for the meta model developer currently as inappropriate and tend not to follow this path.

Feedback on ADOxx-Java-MM-DSL; Pro: It is possible to merge libraries and start libraries from scratch. Furthermore, the code base can be stored and versioned in a versioning system enabling several developers in parallel to work on one library. Built scripts enable the automatic generation and deployment of the tool; **Contra:** The current code maturity needs improvement and documentation, enabling also non specialists to handle the tool; **Outlook:** This tool will be further improved and tested in two EU-H2020 research projects and will consequently be taught at the ADOxx.org Training Days and Webinars to achieve the required maturity.

Feedback on ADOxx.org Tool Packing Services and Developer Spaces; Pro: It is possible to have an installation package to distribute to interested stake-holders, building your own community around the modelling method, and get feed-back from them; **Contra:** Setting up and handling issues of a certain Developer Space involves certain manual steps, such, as the interested stakeholder has to send an e-mail to the administrator with a request of an own Developer Space; **Outlook:** This tool packaging service will be stepwise opened, so that the developer can also include own software components, which are then composed into a single tool package.

5 Conclusion and Outlook

In this paper we propose a product-service system instantiating the Modelling Method Conceptualization Process, which supports method engineers to develop their own modelling method and corresponding modelling software on demand with following agile modelling method engineering principles. The product-service system has been evaluated through an analysis of four different cases: three EU research projects and one in-house project. The evaluation results put forward that having an approach, a corresponding product and service bundle following the idea of model-driven engineering is effective in terms of transferring knowledge from the analysis of requirements up to the development of solutions. Being two main tools, MMDE and ALDE, experimental prototypes that are at very early stage of development, lack of full integration or automatic data exchange ability, and the need of manual steps building Developers Spaces came out as major limitations of the product-service system. As an outlook the following items derived from the evaluation: (1) currently we are evaluating development more integrated Modelling Method Conceptualization Environment, (2) and working out service-offerings for demands regarding development in productive settings.

Acknowledgment

This work has been partly supported by European Union's Horizon 2020 research and innovation programme within the projects DISRUPT (www.disrupt-project.eu) under grant agreement No: 723541 and CloudSocket (www.cloudsocket.eu) under grand agreement no: 644690.

References

[ADOxx16] ADOxx.org "GraphRep" 2016 [Online. Available https://www.adoxx.org/live/graphrep [Accessed 24.January.2017]

[ADOxx17a] ADOxx.org, "Developer Community," 2017. [Online]. Available: https://www.adoxx.org/live/community. [Accessed 23.January.2017].

[ADOxx17b] ADOxx.org, "ADOxx.org Developer Spaces," 2016. [Online]. Available: https://www.adoxx.org/live/development-spaces. [Accessed 23.January.2017].

[AKSW17] The Research Group Agile Knowledge Engineering and Semantic Web (AKSW), University of Leipzig, "Xturtle," 2015. [Online]. Available: http://aksw.org/Projects/Xturtle.html. [Accessed 23.January.2017].

[Apache17] The Apache Software Foundation, "Apache Ant Download," 2016. [Online]. Available: https://www.apache.org/dist/ant/binaries/. [Accessed 23.January.2017].

[CloudSocket17a] CloudSocket Consortium, "CloudSocket Project," 2016. [Online]. Available: https://www.cloudsocket.eu/. [Accessed 23.January.2017].

[CloudSocket17b] CloudSocket Consortium, "CloudSocket Developer Space," 2015. [Online]. Available: https://www.adoxx.org/live/web/cloudsocket-developer-space/. [Accessed 15 July 2016].

[CloudSocket17c] CloudSocket Consortium, "CloudSocket Developer Space - Downloads," 2015. [Online]. Available: https://www.adoxx.org/live/web/cloudsocket-developer-space/downloads. [Accessed 23.January.2017].

[DISRUPT17a] DISRUPT Consortium, "DISRUPT Developers Space" [Online available https://www.adoxx.org/live/web/disrupt/] Accessed 24 January 2017

[DISRUPT17b] DISRUPT Consortium, "Project Overview", 2017 [Online Available http://disrupt-project.eu/about/overview] Accessed 24 January 2017

[Eclipse17a] Eclipse Foundation, "Eclipse IDE for Java EE Developers," 2016. [Online]. Available: http://www.eclipse.org/downloads/packages/. [Accessed 23.January.2017].

[EWK15] N. Efendioglu, R. Woitsch and D. Karagiannis, "Modelling Method Design: A Model-Drive Approach," in IIWAS '15: Proceedings of the 17th International Conference on Information Integration and Web-based Applications, Brussels, Belgium, ACM, 2015.

[EWU16] Efendioglu, N., Woitsch, R., & Utz, W. (2016). A Toolbox Supporting Agile Modelling Method Engineering: ADOxx.org Modelling Method Conceptualization Environment. In J. Horkoff, M. A. Jeusfeld, & A. Persson, *The Practice of Enterprise Modeling* (pp. 317-325), 9th IFIP WG 8.1. Working Conference, PoEM 2016, Skövde, Sweden, November 8-10, 2016, Proceedings, Springer

[FRK2012] H.-G. Fill, T. Redmond and D. Karagiannis, "FDMM: A Formalism for Describing ADOxx Meta Models and Models," in Proceedings of ICEIS 2012, Wroclaw, Poland, Vol. 3, Wroclaw, 2012, pp. 133-144.

[GO0DMAN 17] GO0D MAN Consortium,, The project "GO0D MAN: Agent Oriented Zero Defect Multi-Stage Manufacturing" [Online] Available at: http://go0dman-project.eu/ [Accessed 03 March 2017]

[HKW13] V. Hrgovcic, D. Karagiannis and R. Woitsch, "Conceptual Modeling of the Organisational Aspects for Distributed Applications: The Semantic Lifting Approach," in COMPSACW, IEEE, 2013.

[Ka15] D. Karagiannis, "Agile Modeling Method Engineering," in Proceedings of the 19th Panhellenic Conference on Informatics, Athens, Greece, ACM, 2015, pp. 5-10.

[KHW11] D. Karagiannis, V. Hrgovcic and R. Woitsch, "Model Driven Design for e-Applications: The Meta Model Approach," in Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, iiWAS11, Ho Chi Minh City, Vietnam, ACM, 2011, pp. 451-454.

[KK02] D. Karagiannis and H. Kühn, "Metamodelling platforms," in In Proceedings of the 3rd International Conference EC-Web 2002, Dexa 2002, France, Springer-Verlag, 2002, p. 182.

[KMM16] D. Karagiannis, H. C. Mayr, J. Mylopoulos, Domain-Specific Conceptual Modelling, Springer International Publishing, 2016

[Kü04] H. Kühn, "Methodenintegration im Business Engineering, PhD Thesis (in German)," University of Vienna, 2004.

[LearnPAd17a] Learn PAd Consortium, "The EU Project Learn PAd," 2014. [Online]. Available: http://www.learnpad.eu/. [Accessed 23.January.2017].

[LearnPAd17b] LearnPAd Consortium, "LearnPAd Developer Space - Downloads," 2015. [Online]. Available: https://www.adoxx.org/live/web/learnpad-developer-space/downloads. [Accessed 23.January.2017].

[LearnPAd17c] LearnPAd Consortium, "LearnPAd Developer Space," 2015. [Online]. Available: https://www.adoxx.org/live/web/learnpad-developer-space. [Accessed 23.January.2017].

[OMG17] Object Management Group (OMG), "Documents Associated With UML Version 2.0," 2005. [Online]. Available: http://www.omg.org/spec/UML/2.0/. [Accessed 23.January.2017].

[OMILab17] Open Models Laboratory (OMILab), "Idea and Objectives," 2015. [Online]. Available: http://austria.omilab.org/psm/about. [Accessed 23.January.2017].

[Pr17] Principles behind the Agile Manifesto [Online. Available http://agilemanifesto.org/iso/en/principles.html] [Accessed 24 January.2017]

[Se11] B. Selic, "The Theory and Practice of Modeling Language Design for Model-Based Software Engineering—A Personal Perspective," in Generative and Transformational Techniques in Software Engineering III, Springer Berlin Heidelberg, 2011, pp. 290-321.

[VK14]	N. Visic and D. Karagiannis, "Developing Conceptual Modeling Tools Using a DSL," in Knowledge Science, Engineering and Management, Sibiu, Romania, Springer, 2014, pp. 162-173.
[W3C17a]	W3C, "RDF-Resource Description Framework," 2014. [Online]. Available: https://www.w3.org/RDF/. [Accessed 23.January.2017].
[W3C17b]	W3C, "RDF 1.1 Turtle Terse RDF Triple Language,," 2014. [Online]. Available: https://www.w3.org/TR/2014/REC-turtle-20140225/. [Accessed 23.January.2017].
[Wo14]	R. Woitsch, "Hybrid Modeling: An Instrument for Conceptual Interoperability," in Revolutionizing Enterprise Interoperability through Scientific Foundations, Hershey, 2014, pp. 97-118.
[WU15]	R. Woitsch and W. Utz, "Business Process as a Service, Model Based Business and IT Cloud Alignment as a Cloud Offering," in ES 2015, Third International Conference on Enterprise Systems, Basel, Switzerland, 2015.