A Proposal for a Common Representation Language for MDE Artifacts and Settings

Fábio Paulo Basso*

Federal University of Rio de Janeiro, COPPE-PESC, RJ, Brazil
fabiopbasso@cos.ufrj.br, expected defense date: 07/2016

Abstract. Empirical evidences suggest the need for a common representation language to be used in the core of a Knowledge Base (KB) for Model Driven Engineering (MDE). The absence of a common representation for MDE Artifacts (components, libraries, metamodels and model transformations and settings), hampers reuse and collaboration in inter-organizational contexts. The state-of-art introduces MDE Artifacts in target software projects with Domain Specific Languages (DSL) for MDE Settings including integration of MDE Artifacts, embedded with concepts for Model Transformation Chain (MTC), Model-Driven Integration (MDI), shared model-based tools though Software as Service (SaaS), process patterns and others. This research abstract presents a proposal for a common representation language named RAS++, intersecting existing metaclasses from DSLs for MDE Settings and adding metaclasses from the Reusable Asset Specification (RAS) OMG standard. Thus, it is presented past and ongoing work in the RAS++ in a PHD proposal.

Keywords: Knowledge Base; Model Driven Engineering; Reusable Asset Specification; Common Representation Language

1 Introduction

Model-Driven Engineering (MDE) still need improvements both in processes and tools [1, 2]. Last year Mussbacher and others [3] pointed out that research is still needed to lead MDE engineers to find and integrate appropriate solutions for specific needs in MDE-based processes. As a long-term goal for the MDE community, some authors suggested that solutions for MDE should be shared on the web in a Knowledge Base (KB)/repositories [2], thus allowing the quick discovery and comprehension of appropriate MDE artifacts (model transformations, tools, metamodels, libraries, files, etc.) [4].

A KB stores physical files from MDE artifacts and abstractions (descriptive and technical) associated with these artifacts [5]. Descriptive-level information provide semantics for search and comprehension of the shared information in repositories [4, 6, 7]. Technical-level information is represented with some Domain Specific Languages (DSLs), proposed in the literature of the area to represent specificity from MDE Artifacts [8–11] and generalized in this paper as "DSLs for MDE Settings". These DSLs are used for integration, orchestration, adaptation and execution of MDE Artifacts.

A central component of a KB for MDE Artifacts is a "common representation language"(CRL) [3] that provide description for catalog and search to/from repositories and also technicalities expressed with DSLs for MDE Settings, such as Model Typing [12]. Although it is important for the advent of KBs for MDE to be used in inter-organizational contexts [3, 2], a common representation language is missing in the literature of the area. Thus, the requirements for this CRL should be discussed in details.

We propose a new representation language named RAS++ with concepts relevant for a common representation. RAS++ extends the Reusable Asset Specification (RAS) [13], an important standard proposed by the OMG to structure elements for reuse through instruction for integration and classification of MDE Artifacts in repositories. RAS++ fits to requirements for a CRL that we have been found in the literature. This work is organized as follows: Section 2 presents the motivation of this research and a proposal for PhD thesis is presented in Section 3. Section 4 introduces the main related works and Section 5 points out the main features of RAS++. Section 6 discusses the research method and progress and Section 7 presents the evaluation plan. Section 8 summarizes main conclusions on this research.

2 Motivation

Concepts that circle the MDE adoption are of interest of some companies whose core business is MDE [14], as illustrates Figure 1. In [15], we reported an effort to introduce model-based solutions in target contexts (Figure 1, box 3), where several artifacts for MDE (core assets) have been developed to support specific needs in target software projects [9]. MDE Artifacts (e.g., model transformations and tools) are applied in different contexts of target companies [16], thus reusable in inter-organizational contexts [5]. The reuse of these artifacts requires a representation language and tool implementation that make the construction of MDE Settings flexible [17]. This requires the use of representation languages as the FOMDA DSL [15] illustrated in Figure 1, box 2.

FOMDA DSL has been used to adapt, orchestrate and execute model transformation components in some software projects, similarly as many other DSLs for MDE Settings such as: a) Model Transformation Chains (MTCs) [9, 10]; b) Component models for model transformations (CBD of MTs) [17]; c) Model-Driven Integration (MDI) for model tool chains [11]; d) Orchestration of MDE Artifacts on the cloud through Software as Service (SaaS) [18]; e) Integrated Software Development Process (SDP) with MDE (MDE-SDP) [8], and; f) Process patterns for MDE, considering MDE Settings as process components shared on repositories [16].

So far, the existing tool support accomplished our needs. However, the current scenario [5, 19] is more challenging because target companies are also inter-

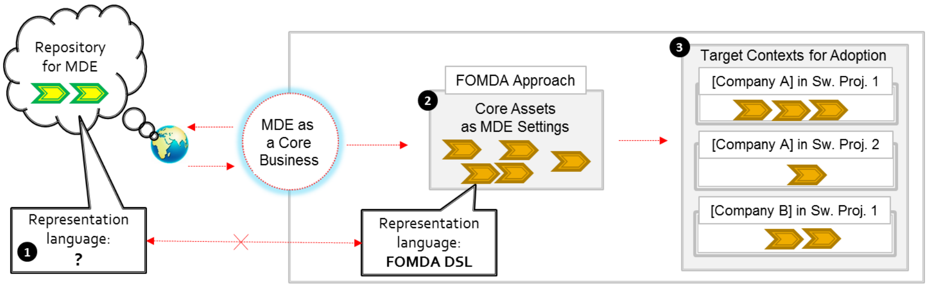

Fig. 1. Illustration of the Current Scenario for MDE as a Core Business

ested in MDE Artifacts developed around the world, as illustrated in Figure 1 (1). In this scenario, MDE Artifacts will be available in KBs [3] and must be searched [4], analyzed [6] and integrated [9] with the core assets illustrated in Figure 1 (2). Thus, the advent of a common representation language to be used in a KB, as shown in Figure 1 (1), can facilitate the implementation of this inter-organizational reuse scenario, as motivated by other researches in the area [4, 19, 5].

3 Proposal

Reuse steps: In order to implement this new scenario for reuse, we suggest that at least three steps shown in Figure 2 must be executed. The first step "Specification" is to specify MDE Artifacts in RAS++ (model in conformity with the "CRL"), allowing the storage of data associated with artifacts in a KB and retrieval in the same format. Then, software engineers/developers download these artifacts (not yet the physical content, but the information represented in RAS++ model) by searching the KB and comparing its features in the step "Acquisition". In the step "Transformation", model-to-model transformations together with a "deployment engine" will adapt an MDE Artifact for a specific format (model in conformity with a DSL X for MDE Settings). This would allow to implement the scenario motivated in Figure 1.

Issue: *No common representation for MDE Artifacts/Settings.* This is a key point in this research, because the lack of this language makes the reuse of MDE Artifacts in a global reuse scenario difficult [2, 3]. The literature of the area is rich with contributions for specific representation languages for MDE Settings. However, a more abstract and general representation is missing, which could promote the interchange of the information shareable between these DSLs (e.g., semantic for model-based operations, model transformation components and metamodels). The lack of a common representation language makes the concepts associated with MDE Artifacts replicated in each DSL for MDE Settings, which is bad to the advent of a common KB for MDE that must keep a uniformity for the stored information. This would allow, for example, to simplify the use of FOMDA DSL (focused in orchestration and adaptation of MT components) [15] together

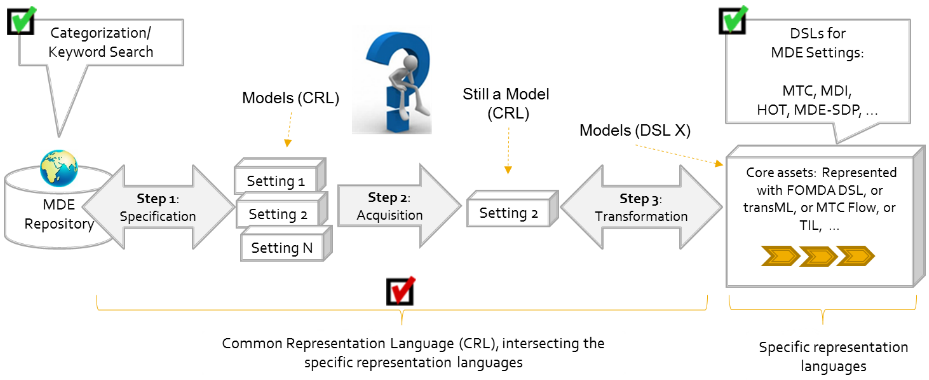

Fig. 2. A Scenario of Use of Information Associated with MDE Artifacts Through a Common Representation Language That Connects Repositories With DSLs for MDE Settings

with other DSLs such as TIL (focused in MDE tool integration) [11]. In other words, a common representation language could promote the complementarity of these specific representation languages.

Research Questions: *1) what are the requirements for a common representation language? 2) what is common between these DSLs for MDE Settings?* To the best of our knowledge, there is no answer in the literature of the area. Due to our previous experience with the FOMDA DSL [15], we assumed in [20] that some of the abstractions from MTCs should be used. The problem is that, besides DSLs for MTC, several others also allow to integrate MDE Artifacts in target contexts. Thus, a study that finds the requirements and the common abstractions would be a great contribution to the research and practice on MDE.

Requirements for a CRL: We have found in the literature some requirements for this common representation language. Some representations introduced in DSLs for MDE Settings are considered for adoption in practice [15], thus relevant for this language [3]. It is also important to consider that this language will be used on the core of a KB, which means that it should be flexible to represent any MDE Artifact such as those stored in the Repository for Model Driven Development - ReMoDD [4] (examples, data-sets and modeling pearls). The common language should facilitate the discovery and comparison of MDE Artifacts [2]. In this sense, another level of information is needed between the catalog of artifacts [6] and the representation of configurations between models and transformations [17]. Mohagheghi et al. agrees in this respect, concluding that establishing a bridge for this gap between technicalities and descriptive information is a key for success of MDE [1]. Thus, a CRL should be capable to represent information in descriptive and technical-level, bridging the gap between data stored in repositories and DSLs for MDE Settings.

4 Related Work

Knowledge Bases/Repositories. The ReMoDD [4], for example, makes use of only searches about some proposals published in some conferences such as MODELS, ECMFA, etc. Most information that should be explicit is available in documents, papers and tutorials, which requires long time to find and analyze adequate options. Other contributions are specific to represent properties of model transformations. Lúcio et al. provided standard taxonomy and metamodel to classify model transformations [6]. The search and retrieval of of model transformations from repositories based on source code annotations is proposed by Criado et al., providing rich query mechanisms based on OCL and integration with the GitHub [7]. Another recent contribution for reuse repository is the MDE Forge, allowing the storage of MDE Artifacts such as DSLs, model transformations and MTCs, allowing the automatic recommendation on the design of new MTCs and acquisitions through SaaS [5]. These approaches provide ways to classify MDE Artifacts in repositories, but the underlying representation language is not based on a standard specification. Differently, in RAS++ we are representing the information associated with MDE Artifacts following an OMG standard (RAS), which enables the interchange of information between component repositories and clients. This is an important feature for a CRL to implement the introduced scenario for "MDE as Core Business".

Assets. We can mention some extensions for RAS that represent information associated with software components [21–23]. These works propose to specify data related with components for application integration and service oriented architecture [21], feedback from users [22], component software license [23] and standard taxonomies. These extensions can also be applied to summarize information of some technical solutions for MDE. However, they are limited to represent technical information associated with MDE settings as the ones represented with RAS++.

Technicalities. Our current contribution analyzes abstractions needed in a common representation language, while our previous contributions in FOMDA DSL [15] analyzed requirements to support the execution and adaptation of MDE settings. Similarly to the FOMDA DSL, others are essential to introduce MDE in target contexts such as transML [10], MTC Flow [9], Bento DSL [17], TIL [11], , SPEM extensions [8] and process patterns [16]. However, they are limited to be used in the core of a KB due to its specificity and lack of rich structured descriptive data. Thus, RAS++ is complementary, more abstract and general.

Related work do not scales well in this new scenario for inter-organizational reuse. This way, RAS++ will present considerable contributions to enable its implementation.

5 Main Features In RAS++

In [20] we proposed RAS++, a new metamodel with concepts considered common to represent MDE Settings. In the following we discuss the main features in RAS++.

This metamodel is based on the Reusable Asset Specification (RAS) OMG standard [13]. An asset is: anything that provides reuse and value through reference (links), cataloged with standardized taxonomies, described by a set of properties and owning zero or more specifications about artifacts [24]. Assets are specifications that provide semantics for artifacts, usually represented in XML (MANIFEST file) with data associated with any sort of artifact that can be stored in reuse repositories. Through concepts related to semantics for reuse, assets are used to describe software components [23], application and domain models and, more recently, tools that help in the execution of software engineering tasks in the context of MDE [18]. Reusable assets (RAS) [13] can be stored in existing reuse repositories and retrieved in the same format, providing structures to support the reuse of MDE Settings and tasks to instruct the integration of these settings in target contexts. Thus, concepts introduced in asset specification languages such as RAS and Asset Management Specification (AMS) [24] are important for a common representation language.

Selective extension from the UML. Each metaclass in RAS++ extends at least the "Element" metaclass from a short version of the UML metamodel. This means that RAS++ supports the same light-height extensibility mechanism from the UML 2 (see an example in [25]), besides a standard heavy-height mechanism supported in RAS.

RAS++ intersects concepts from specific representation languages, some presented in [10, 17, 11, 18, 8, 16] for MTCs, MDI, SaaS, etc. These DSLs are designed to manage model transformations in a higher-level of abstraction than the model transformations rules implemented with ATL, QVT, ETL, etc. DSLs such as transML [10], MTC Flow [9], TIL [11] and FOMDA DSL [15] have been reported as important to introduce MDE in target contexts. Therefore, some concepts introduced in these proposals such as tasks/components, artifacts/parameters, metamodels and connectors/bindings are also of interest for a common representation language.

RAS++ increments existing works with structure for rich descriptive data connected with technicalities. After the study in [20], we noticed that other level of information is needed in between the catalog and instruction promoted by assets and the representation of technicalities of MDE Artifacts used by aforementioned DSLs. In this regard, Mohagheghi et al. concluded in 2009 that establishing a bridge for this gap between data used by technicians (e.g., MTCs) and by non-experts (e.g., description) is a key for the success of MDE [1]. In order to implement the scenario illustrated in Figure 2, bridging this gap is a need. Today, the existing DSLs are limited to connect rich descriptive information with technical data. For example, the use of catalog information associated with MDE Artifacts, as allowed in model transformation intents [6] (providing standard taxonomy for correct classification of model transformations in a KB), and in a MTC approach (e.g., Bento DSL [17]), is not enough to ensure that an MDE Artifact A (e.g., a DSL to design web front ends) is properly compared with another MDE Artifact B (e.g., a competing DSL). Recent surveys claimed that this is a problem because industry needs to compare features from

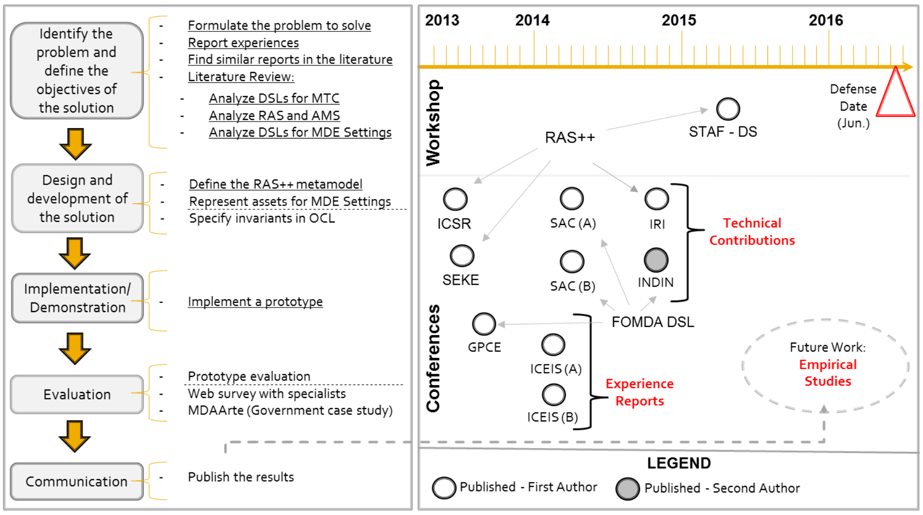

Fig. 3. Research methodology and progress

DSLs to decide the one that best fits for specific needs [2]. Thus, connecting rich descriptive data with technicalities of MDE Artifacts is also a requirement for a common representation language.

Research focus: In the current version of RAS++, we bridged this gap presented by Mohagheghi et al. [1], by introducing concepts that connect rich and structured descriptive data with technicalities from MDE Settings. This way, a study focused on concepts needed in a common representation language is proposed.

Expected contributions: A new representation language implemented with two prototypes, one to design assets and an other to transform assets into target DSLs. The result will allow to implement the scenario illustrated in Figure 2 with well accepted and common concepts for MDE Settings and with increments to the state-of-art in asset specifications.

6 Research Method and Progress

Figure 3 shows the research methodology and progress. Underlined text indicates a concluded activity, dashed text indicates ongoing activities and the other activities are to be started. To a follow-up of our progress, a grade of our contributions is presented in the right-part of Figure 3.

- **Identify the problem and define the objectives of the solution.** I started my PhD course with a previous industrial experience. I exposed some problems that hampered the introduction of MDE in target contexts to my research collaborators, who encouraged me to go ahead in research topic.

- **Formulate the problem to solve.** In some classes for software reuse solutions for issues such as the lack of a KB and a common representation language were surveyed. These studies resulted in two types of work: 1) those representing technicalities for MDE Settings and; 2) in an analytical comparison of asset representation languages, which complement the MTCs implemented with the FOMDA DSL. Based on these studies, we developed a preliminary version of RAS++ and a tool prototype, allowing to publish new works in conferences (ICSR and SEKE) [25, 26].
- **Report experiences on accomplishments and challenges for the scenario illustrated in Figure 1.** In parallel to RAS++, it was important to report benefits, limitations and drawbacks from our experiences in conferences (GPCE and ICEIS) [15, 27, 28].
- **Find similar reports in the literature.** Since 2013, the literature of the area on issues for the MDE adoption have been investigated. We found recent in surveys, experiences and reports common positions on issues that makes hard the reuse of MDE Artifacts considering a global scenario (inter-organizational). Thus, we have found only in recent publications the basis that supports our claims for the discussed issues, which suggests that this is a new and relevant research topic.
- **Literature review.**
  * **Analyze DSLs for MTC.** We looked for the state-of-art for MTCs to compare it with our experiences in [15]. In addition, it was presented new contributions to the FOMDA DSL in conferences (SAC and INDIN) [29–31]. We found common concepts and representations in related works. Moreover, we have found that the state-of-art is limited to accomplish the three steps (Specification, Acquisition and Transformation) illustrated in Figure 2.
  * **Analyze RAS and AMS.** We found in RAS the possibility to implement Step 1 (Specification). In the IRI conference [20] we presented the result of our extension for RAS, with the new meta-classes that give support for the technicalities found as common for MTCs. A new contribution focuses in the modeling descriptive information in assets represented with RAS and AMS.
  * **Analyze DSLs for MDE Settings.** This is a quasi-systematic literature review on MDE Settings. We also have found common concepts proposed in these DSLs non related directly with MTCs. A future work will present the result of our analysis with a classification used to define the relevant representations added to RAS++.

– **Design and development of the solution.** The RAS++ metamodel is constructed based on recent issues reported on the literature of the area. Literature review and experiences suggest that we are on the right way to bridge the gap between technical and descriptive information in a common representation that enable the implementation of the introduced reuse scenario. Ongoing works: 1) in order to create a database for evaluation through prototypes, many assets have been represented with the RAS++ and; 2) we are specifying OCL invariants associated with the RAS++ metamodel.

– **Implementation/Demonstration.** In order to validate RAS++ metamodel, we developed two prototypes: **a) An EMF-based designer tool** used to specify the assets, important for Step 1 (Specification); **b) Eclipse-based "RAS++ Deploy" plug-in** that aims at transforming RAS++ models into some of specific DSLs. These prototypes are continually being improved to support new meta-classes introduced in RAS++ metamodel and are discussed in [25, 26, 20].

7 Evaluation Plan

The evaluation plan considers the verification of the meta-classes introduced in RAS++ and evaluating the transformation from RAS++ assets to some of the existing DSLs for MDE Settings:

– **Web survey with specialists.** This study aims at understand some meta-classes proposed on the literature (e.g., metamodel, artifacts, workflow elements, filters, etc), evaluating what do people consider relevant to include in a common representation language for MDE Settings. Accordingly, we planned a web survey with questionnaire to find qualitative and quantitative data that highlight to the relevance of some concepts introduced in existing DSLs. Thus, this study will clarify which concepts should or should not be part of RAS++;
– **Evaluation through prototype.** This is a practical evaluation, based on implementations for the step 3 (Transformation). This practical evaluation will allows to transform MDE assets represented with RAS++ in target DSLs for MDE Settings.
– **MDArte (Government case study).** The last evaluation will be carried out in a real scenario for "MDE as Core Business", where a team from the company MDArte [14] will experiment the tool support developed in this proposal. MDArte company provides MDE services for some software development teams from the Brazilian government.

8 Conclusion

This work presented a proposal for a common representation language for MDE Settings named RAS++. In order to enable the implementation of a new scenario which needs the introduction of MDE Artifacts through Knowledge Bases (KBs), it was surveyed the literature to find similarities and differences between proposals that aim at represent MDE Settings. Therefore, this work highlights to main contributions from a PhD thesis proposal that add in RAS++ the support for syntax and semantics, associated with MDE Artifacts, in structures for reuse promoted by asset specifications.

References

1. Mohagheghi, P., Fernandez, M.A., Martell, J.A., Fritzsche, M., Gilani, W.: Mde adoption in industry: Challenges and success criteria. In Chaudron, M.R., ed.: Models in Software Engineering. Volume 5421 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2009) 54–59
2. Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., Heldal, R.: Industrial adoption of model-driven engineering: Are the tools really the problem? In: Proceedings of the 16th International Conference on Model Driven Engineering Languages and Systems. MODELS'13 (2013) 1–17
3. Mussbacher, G., Amyot, D., Breu, R., Bruel, J.M., Cheng, B.H., Collet, P., Combemale, B., France, R.B., Heldal, R., Hill, J., Kienzle, J., Schöttle, M., Steimann, F., Stikkolorum, D., Whittle, J.: The relevance of model-driven engineering thirty years from now. In: Model-Driven Engineering Languages and Systems. (2014) 183–200
4. France, R., Bieman, J., Cheng, B.: Repository for model driven development (remodd). In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 4364 LNCS (2007) 311–317
5. Rocco, J.D., Ruscio, D.D., Iovino, L., Pierantonio, A.: Collaborative repositories in model-driven engineering [software technology]. Software, IEEE **32**(3) (May 2015) 28–34
6. Lúcio, L., Amrani, M., Dingel, J., Lambers, L., Salay, R., Selim, G.M., Syriani, E., Wimmer, M.: Model transformation intents and their properties. Software & Systems Modeling (2014) 1–38
7. Criado, J., Martínez, S., Iribarne, L., Cabot, J.: Enabling the reuse of stored model transformations through annotations. In: International Conference on Model Transformations. (July 2015) 1–15
8. Polgár, B., Ráth, I., Szatmari, Z., Horvath, A., Majzik, I.: Model-based integration, execution and certification of development tool-chains. In: 2th Workshop on Model-Driven Tool & Process Integration (MDTPI). (2009) 35–46
9. Yie, A., Casallas, R., Deridder, D., Wagelaar, D.: Realizing model transformation chain interoperability. Software & Systems Modeling **11**(1) (2012) 55–75
10. Guerra, E., de Lara, J., Kolovos, D.S., Paige, R.F., dos Santos, O.M.: transml: A family of languages to model model transformations. In: 13th International Conference on Model Driven Engineering Languages and Systems. MODELS 2010 (2010) 106–120
11. Biehl, M., El-Khoury, J., Loiret, F., Törngren, M.: On the modeling and generation of service-oriented tool chains. Software & Systems Modeling **13**(2) (2014) 461–480
12. Vignaga, A., Jouault, F., Bastarrica, M.C., Brunelière, H.: Typing artifacts in megamodeling. Software & Systems Modeling **12**(1) (2013) 105–119
13. RAS: Reusable asset specification. av. at <http://www.omg.org/spec/ras/>. at june, (2014)
14. Monteiro, R., Assumpcao Pinel, R., Zimbrao, G., Moreira de Souza, J.: The mdarte experience: Organizational aspects acquired from a successful partnership between government and academia using model-driven development. In: International Conference on Model-Driven Engineering and Software Development (MODELSWARD). (May 2014) 575–586
15. Basso, F.P., Pillat, R.M., Oliveira, T.C., Becker, L.B.: Supporting large scale model transformation reuse. In: 12th International Conference on Generative Programming: Concepts & Experiences. GPCE'13 (2013) 169–178

16. Hebig, R., Giese, H., Stallmann, F., , Seibel, A.: On the complex nature of mde evolution. In: Proceedings of Model Driven Engineering Languages and Systems. MODELS 2013 (2013) 436–453
17. Cuadrado, J.S., Guerra, E., de Lara, J.: A component model for model transformations. IEEE Transactions on Software Engineering **40**(11) (Nov 2014) 1042–1060
18. Zhang, W., Moller-Pedersen, B.: Modeling of tool integration resources with oslc support. In: 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD). (Jan 2014) 99–110
19. Combemale, B., Deantoni, J., Baudry, B., France, R., Jézéquel, J.M., Gray, J.: Globalizing modeling languages. IEEE Computer, Institute of Electrical and Electronics Engineers **47**(6) (June 2014) 68–71
20. Basso, F.P., Werner, C.M.L., Oliveira, T.C.: Towards facilities to introduce solutions for mde in development environments with reusable assets. In: International Conference on Information Reuse and Integration. IRI'14 (2014) 195–202
21. Park, S., Park, S., Sugumaran, V.: Extending reusable asset specification to improve software reuse. In: Proceedings of the 2007 ACM symposium on Applied computing. SAC '07 (2007) 1473–1478
22. Hadji, H.B., Su-Kyoung, K., Ho-Jin, C.: A representation model for reusable assets to support user context. In: IEEE International Symposium on Service-Oriented System Engineering, SOSE '08. (2008) 91–96
23. Hong-min, R., Zhi-ying, Y., Jing-zhou, Z.: Design and implementation of ras-based open source software repository. In: Sixth International Conference on Fuzzy Systems and Knowledge Discovery, 2009. FSKD '09. Volume 2. (2009) 219–223
24. AMS: Asset Management Specification. Av. At <http://open-services.net/wiki /asset-management/oslc-asset-management-2.0-specification/>. At June, (2014)
25. Basso, F.P., Werner, C.M.L., Pillat, R.M., Oliveira, T.C.: A common representation for reuse assistants. In: 13th International Conference on Software Reuse. ICSR'13 (2013) 283–288
26. Basso, F.P., Werner, C.M.L., Pillat, R.M., Oliveira, T.C.: How do you execute reuse tasks among tools? a ras based approach to interoperate reuse assistants. In: 25th International Conference on Software Engineering and Knowledge Engineering. (2013) 721–726
27. Basso, F.P., Pillat, R.M., Rooz-Frantz, F., Frantz, R.Z.: Study on combining model-driven engineering and scrum to produce web information systems. In: 16th International Conference on Enterprise Information Systems. ICEIS'14 (2014) 137–144
28. Basso, F.P., Pillat, R.M., Frantz, R.Z., Rooz-Frantz, F.: Assisted tasks to generate pre-prototypes for web information systems. In: 16th International Conference on Enterprise Information Systems. ICEIS'14 (2014) 14–25
29. Basso, F.P., Pillat, R.M., Oliveira, T.C., Fabro, M.D.D.: Generative adaptation of model transformation assets: Experiences, lessons and drawbacks. In: 29th Symposium On Applied Computing. SAC'14 (2014) 1027–1034
30. Basso, F.P., Oliveira, T.C., Farias, K.: Extending junit 4 with java annotations and reflection to test variant model transformation assets. In: 29th Symposium On Applied Computing. SAC'14 (2014) 1601–1608
31. Paulon, A., Frohlich, A., Becker, L., Basso, F.: Wireless sensor network UML profile to support model-driven development. In: 12th IEEE International Conference on Industrial Informatics, At Porto Alegre, RS, Brazil. INDIN 2014 (2014) 227–232