

A framework for evaluating the quality of modelling languages in MDE environments

Author:
Fáber D. Giraldo

Advisors:
Óscar Pastor
Sergio España

September
2017



Centro de Investigación en Métodos
de Producción de Software



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



DOCTORAL THESIS

A framework for evaluating the quality
of modelling languages in MDE
environments

Author:
Fáber D. GIRALDO

Advisors:
Dr. Óscar PASTOR
Dr. Sergio ESPAÑA

*A thesis submitted in fulfillment of the requirements
for the degree of PhD in Computer Science*

in the

Research Center on Software Production Methods (PROS)
Department of Computer Systems and Computation (DSIC)

September 29, 2017

Declaration of Authorship

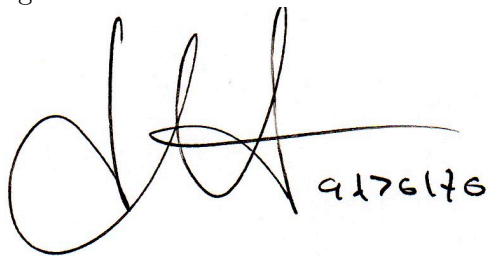
I, Fáber D. GIRALDO, declare that this thesis titled,

“A framework for evaluating the quality of modelling languages in MDE environments”

and the work presented in it are my own. I confirm the following:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- When I have consulted the published work of others, this is always clearly attributed.
- When I have quoted from the work of others, the source is always given. With the exception of these quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Signed:



9176176

Date:

September 29, 2017

Book cover jointly designed by Fáber D. Giraldo and Dinamo Production Multimedia (direccion@dinamopro.com).

Latex template from Steve Gunn (<http://users.ecs.soton.ac.uk/srg/softwaretools/document/templates/>) and Sunil Patel (<http://www.sunilpatel.co.uk/thesis-template/>).

Note: the *BibLatex* package was not used because of its incompatibility with UTF 8.0 special characters. Instead, the *natbib* package was used.

External reviewers

Hernán Astudillo Rojas
(Universidad Técnica Federico Santa María, Chile)

Manuel Ortega Cantero
(Universidad de Castilla - La Mancha, Spain)

Antonio Vallecillo Moreno
(Universidad de Málaga, Spain)

Thesis defence committee

President:

Vicente Pelechano Ferragud
(Universitat Politècnica de València, Spain)

Members:

Hernán Astudillo Rojas
(Universidad Técnica Federico Santa María, Chile)

Antonio Vallecillo Moreno
(Universidad de Málaga, Spain)

Secretary:

Ernesto Pimentel Sánchez
(Universidad de Málaga, Spain)

Substitute members:

Manuel Ortega Cantero
(Universidad de Castilla - La Mancha, Spain)

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Abstract

Escola Tècnica Superior d'Enginyeria Informàtica
Department of Computer Systems and Computation (DSIC)

PhD in Computer Science

A framework for evaluating the quality of modelling languages in MDE environments

by Fàber D. GIRALDO

This thesis presents the *Multiple Modelling Quality Evaluation Framework* method (hereinafter *MMQEF*), which is a conceptual, methodological, and technological framework for evaluating quality issues in modelling languages and modelling elements by the application of a taxonomic analysis. It derives some analytic procedures that support the detection of quality issues in model-driven projects, such as the suitability of modelling languages, traces between abstraction levels, specification for model transformations, and integration between modelling proposals. MMQEF also suggests metrics to perform analytic procedures based on the classification obtained for the modelling languages and artifacts under evaluation.

MMQEF uses a taxonomy that is extracted from the Zachman framework for Information Systems (Zachman, 1987; Sowa and Zachman, 1992), which proposed a visual language to classify elements that are part of an Information System (IS). These elements can be from organizational to technical artifacts. The visual language contains a bi-dimensional matrix for classifying IS elements (generally expressed as models) and a set of seven rules to perform the classification. As an evaluation method, MMQEF defines activities in order to derive quality analytics based on the classification applied on modelling languages and elements. The Zachman framework was chosen because it was one of the first and most precise proposals for a reference architecture for IS, which is recognized by important standards such as the ISO 42010 (612, 2011).

This thesis presents the conceptual foundation of the evaluation framework, which is based on the definition of *quality* for model-driven engineering (MDE). The methodological and technological support of MMQEF is also described. Finally, some validations for MMQEF are reported.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Resum

Escola Tècnica Superior d'Enginyeria Informàtica
Departament de Sistemes Informàtics i Computació (DSIC)

PhD. en Informàtica

Marc de treball per a l'avaluació de la qualitat de llenguatges de modelatge en entorns MDE

by Fàber D. GIRALDO

Aquesta tesi presenta el mètode MMQEF (*Multiple Modelling Quality Evaluation Framework*), el qual és un marc de treball conceptual, metodològic i tecnològic per avaluar aspectes de qualitat sobre llenguatges i elements de modelatge mitjançant l'aplicació d'anàlisi taxonòmic. El mètode deriva procediments analítics que suporten la detecció d'aspectes de qualitat en projectes model-driven com ara: idoneïtat de llenguatges de modelatge, traçabilitat entre nivells d'abstracció, especificació de transformació de models, i integració de propostes de modelatge. MMQEF també suggereix mètriques per executar procediments analítics basats en la classificació obtinguda pels llenguatges i artefactes de modelatge avaluats.

MMQEF fa servir una taxonomia per a Sistemes d'Informació basada en el framework Zachman (Zachman, 1987; Sowa and Zachman, 1992). Aquesta taxonomia proposa un llenguatge visual per classificar elements que fan part d'un Sistema d'Informació. Els elements poden ser artefactes associats a nivells des organitzacionals fins tècnics. El llenguatge visual conté una matriu bidimensional per classificar elements de Sistemes d'Informació, i un conjunt de set regles per executar la classificació. Com a mètode d'avaluació MMEQF defineix activitats per derivar analítiques de qualitat basades en la classificació aplicada sobre llenguatges i elements de modelatge. El marc Zachman va ser seleccionat a causa de que aquest va ser una de les primeres i més precises propostes d'arquitectura de referència per a Sistemes d'Informació, sent això reconegut per destacats estàndards com ISO 42010 (612, 2011).

Aquesta tesi presenta els fonaments conceptuals del mètode d'avaluació basat en l'anàlisi de la definició de qualitat en l'enginyeria dirigida per models (MDE). Posteriorment es descriu el suport metodològic i tecnològic de MMQEF, i finalment es reporten validacions.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

*Resumen*Escola Tècnica Superior d'Enginyeria Informàtica
Departamento de Sistemas Informáticos y Computación (DSIC)

PhD. en Informàtica

Marco de trabajo para la evaluación de la calidad de lenguajes de modelado en entornos MDE

by Fàber D. GIRALDO

Esta tesis presenta el método MMQEF (*Multiple Modelling Quality Evaluation Framework*), el cual es un marco de trabajo conceptual, metodológico y tecnológico para evaluar aspectos de calidad sobre lenguajes y elementos de modelado mediante la aplicación de análisis taxonómico. El método deriva procedimientos analíticos que soportan la detección de aspectos de calidad en proyectos model-driven tales como: idoneidad de lenguajes de modelado, trazabilidad entre niveles de abstracción, especificación de transformación de modelos, e integración de propuestas de modelado. MMQEF también sugiere métricas para ejecutar procedimientos analíticos basados en la clasificación obtenida para los lenguajes y artefactos de modelado bajo evaluación.

MMQEF usa una taxonomía para Sistemas de Información basada en el framework Zachman (Zachman, 1987; Sowa and Zachman, 1992). Dicha taxonomía propone un lenguaje visual para clasificar elementos que hacen parte de un Sistema de Información. Los elementos pueden ser artefactos asociados a niveles desde organizacionales hasta técnicos. El lenguaje visual contiene una matriz bidimensional para clasificar elementos de Sistemas de Información, y un conjunto de siete reglas para ejecutar la clasificación. Como método de evaluación MMEQF define actividades para derivar analíticas de calidad basadas en la clasificación aplicada sobre lenguajes y elementos de modelado. El marco Zachman fue seleccionado debido a que éste fue una de las primeras y más precisas propuestas de arquitectura de referencia para Sistemas de Información, siendo ésto reconocido por destacados estándares como ISO 42010 (612, 2011).

Esta tesis presenta los fundamentos conceptuales del método de evaluación basado en el análisis de la definición de calidad en la ingeniería dirigida por modelos (MDE). Posteriormente se describe el soporte metodológico y tecnológico de MMQEF, y finalmente se reportan validaciones.

Acknowledgements

One single page is insufficient to thank the many people who supported this thesis. However, if you are not on this page, you can be sure I will thank you in person as soon as possible.

I would like to thank COLCIENCIAS (Colombia) for funding this work through the Colciencias Grant call 512-2010 Bicentennial Generation.

I declare my admiration and respect for Professor Óscar Pastor. It was an honor and pleasure to work under his tutelage and leadership. When I first listened to his discourse about quality in MDE, I greatly admired his mastery and total conviction about the use and promising future of the conceptual modelling field. I also realized that there are so many challenges to be addressed in the MDE quality area. I hope to have met his expectations regarding these challenges (at least, partially). The Peter Chen award of the ER Conference 2016 is well-deserved recognition for his magnificent career.

One of the most cheerful, bright, and charismatic people I have ever met is Professor Sergio España. It was a pleasure to participate in our meetings to discuss this work, most of which were usually accompanied by the smoothest coffee in the world. Sergio is a great researcher and a very good friend. His enthusiasm and joy are contagious. This work is also for him. (Note: Sergio, please do not think your email will be able to rest after this work!!).

I would like to thank the people of the PROS Research Center of the Universitat Politècnica de València (Spain), especially Ana Ciudad for her collaboration and logistic support. This work has been supported by the Spanish Ministry of Science and Innovation project PROS-Req (TIN2010-19130-C02-02), the Generalitat Valenciana Project IDEO (PROMETEOII/2014/039), the European Commission FP7 Project CaaS (611351), and ERDF structural funds. A special mention goes to Suzanne Ruehle for her kind and strict English reviews. I enjoy each meeting with her because she shares her knowledge in the friendliest way.

I would also like to thank my academic and research home: the SINFOCI group at the University of Quindío (Colombia), especially William Joseph Giraldo, the leader of the group, for inspiring and supporting my career as a researcher. Special thanks go to Dr. Alfonso Londoño Orozco (President of the University of Quindío between 2007-2015), Dra. Patricia Landázuri (Vice-chancellor of Research at University of Quindío between 2007-2015) and Dr. José Fernando Echeverry Murillo (Dean of the Faculty of Engineering between 2007 – 2015, and currently President of the University of Quindío) for their valuable support in the starting stages of this thesis.

I would remiss not to give special mention to my students who are convinced of this work and apply it (or have applied it) in their undergraduate and master projects. They are: Mónica M. Villegas, Manuel A. Pineda, Carlos A. Carvajal, and Héider O. Rivera (from the Master in Engineering program), Mauricio Gallego, Érica M. Reyes, César A. Cataño, Juan D. Fernández, Jhonatan Arcila, Christian G. Cachaya, Luisa F. Arango, Fabián D. Osorio, Jorge Gaitán Portillo, Raúl Rivera, Jesús Onofre, María F. López, Hernán D. Restrepo, and Santiago Rueda (from the System and Computer Engineering undergraduate program at the University of Quindío).

Contents

Declaration of Authorship	iii
Abstract	ix
Resum	ix
Resumen	xi
Acknowledgements	xv
1 The research context	1
1.1 Problem statement	1
1.2 Overview of the solution	3
1.3 Research methodology	3
1.3.1 Research questions	4
1.3.2 The research roadmap	4
2 Literature review about quality in MDE	7
2.1 Introduction	8
2.2 Quality issues in MDE	10
2.2.1 Evolution and limitations of the MDA standard	10
2.2.2 A literature review about models and modelling language quality trends	13
2.2.3 Results	16
2.2.4 Identified categories of the definition of <i>quality</i> in MDE	19
2.2.5 Adaptive sampling	24
2.2.6 Other findings	28
2.2.7 Discussion	31
2.2.8 The relationship between quality in MDE and V&V	32
2.3 A mismatch analysis between industry and academy field	32
2.3.1 Literature review process design	33
2.3.2 Detected categories for industrial quality issues	36
MDA is not enough	37
Implicit questions derived from the MDE adoption itself	37
Tools as a way to increase complexity	37
Organizational support for the adoption of MDE	38
2.3.3 Detected categories for academic/research quality issues	38
Hard operationalization of model-quality frameworks	38
Defects and metrics mainly in UML	39
Specificity in the scenarios for quality in models	39
Software quality principles extrapolated at modelling levels	39

2.3.4	Findings in the literature review of mismatch	40
2.4	The sufficiency of current quality evaluation proposals	45
2.5	Open challenges in the evaluation of the quality of modelling languages	47
2.5.1	Using multiple modelling languages in combination	49
2.5.2	Assessing the compliance of modelling languages with MDE principles	50
2.5.3	Explicitly using abstraction levels as quality filters of modelling languages	51
2.5.4	Agreeing on a set of generic quality metrics for modelling languages	51
2.5.5	Including model transformations in the modelling language quality equation	52
2.5.6	Acknowledging the increasing dynamics of models	54
2.5.7	Streamlining ontological analyses of modelling languages	55
2.5.8	Incorporating modelling language quality as a source of technical debt in MDE	56
2.6	Conclusions	57
3	MMQEF: the conceptual and methodological framework	61
3.1	Introduction	61
3.2	The research problem	62
3.2.1	Conceptual approaches for addressing quality issues in MDE	63
3.2.2	The reference taxonomy	64
3.2.3	The support of the taxonomy for MDE quality issues	66
3.3	The MMQEF method	67
3.3.1	Purposes and preconditions	67
3.3.2	Method components	68
3.3.3	Cooperation principles	69
	Structure	69
	Roles	69
3.3.4	The main quality evaluation method components	69
3.3.5	MDE quality analytics derived	71
	Inference analysis derived from the Zachman taxonomy	72
	Derived supports for MDE quality evaluation	73
3.3.6	Tool support	76
3.3.7	An example application of the MMQEF method	76
3.4	Preliminary trade-off analysis of the MMQEF method	83
3.4.1	Main implications of the MMQEF method	83
	Why taxonomy analysis?	83
	Why this taxonomy?	84
	Is it the only taxonomy? Why not a quality ontology instead?	84
3.4.2	Feasibility of the classification procedure for quality purposes	85
3.4.3	The use of the taxonomic structure itself	85
3.4.4	MMQFE and other quality frameworks for MDE	88
3.5	Conclusions	89

4	The formal and technological support of the MMQEF method	91
4.1	Introduction	91
4.2	The taxonomic analysis and the MMQEF method	93
4.2.1	The formal support of the taxonomy for the quality evaluation analysis of modelling artifacts	96
4.2.2	The Zachman framework as a taxonomic theory	98
4.2.3	Related works	99
4.3	The FCA method	100
4.3.1	The FCA support for the taxonomic analysis	101
4.4	The EMAT Tool	103
4.4.1	Why is another FCA tool needed? Is EMAT another FCA tool?	103
4.4.2	The taxonomic evaluation procedure using EMAT	103
4.4.3	How should a resulting lattice be interpreted ?	105
4.4.4	Other complementary functions	106
4.4.5	EMAT architecture and future vision	107
4.4.6	A trade-off analysis of the EMAT tool	109
4.5	MMQEF and EMAT in practice	110
4.5.1	Quality analysis of the UML and BPMN modelling languages	110
4.5.2	Quality analysis of the OO-Method and CA integration .	114
4.6	Discussion	117
4.6.1	Why another quality framework for MDE?	119
4.7	Conclusions	124
5	Theoretical validation of the taxonomy used by MMQEF	125
5.1	Taxonomies and Information Systems	126
5.2	Is the Zachman taxonomy the only one that classifies modelling languages?	128
5.2.1	Results	130
5.3	The previous use of the taxonomy to classify modelling languages	134
5.4	Conclusions	136
6	Empirical Evaluation of MMQEF	137
6.1	Introduction	137
6.2	Background	139
6.2.1	Problem Statement	139
6.2.2	Research objective	139
6.3	Design of the empirical validation	139
6.3.1	Context	139
6.3.2	Experimental units	140
6.3.3	Experimental material	143
6.3.4	Tasks	145
6.3.5	Hypotheses, parameters, and variables	146
6.3.6	Analysis procedure	147
6.3.7	Execution - deviations	148
6.4	Analysis	148
6.4.1	Descriptive analysis	148

6.4.2	Testing of hypotheses	152
	Analysis of the MEM variables for MMQEF	163
6.5	Discussion	170
6.5.1	Evaluation of results and implications	170
6.5.2	Threats to Validity	170
6.5.3	Inferences	173
	The selection of practical methods.	173
	Detected reasons for choosing quality methods	173
	<i>Representations</i> as an important source for quality evaluation procedures.	174
	The need for a modelling context	175
	Perceived independence of the quality proposals	175
6.5.4	Lessons learned	175
	The improvement of the procedure for making inferences in MMQEF	175
6.5.5	The improvement of the documentation for MMQEF	176
6.5.6	Improving the process of selection and characterization of participants	176
6.5.7	Validating specific features of quality methods for MDE individually instead of all together	176
6.6	Conclusions	177
6.6.1	Summary	177
6.6.2	Impact	177
6.6.3	Future work	178
6.7	Raw data	178
7	Final conclusions and the forthcoming research roadmap	181
7.1	The formulation of application scenarios for MMQEF	181
7.2	The support of MMQEF to address the quality issues reported from industrial contexts	182
7.3	The consolidation of operative support for the MMQEF	182
7.4	The consolidation of a set of metrics for modelling languages	183
7.5	The management of interaction issues in modelling languages (HCI of modelling languages)	183
7.6	The promotion of MMQEF as a <i>Type V Theory</i> for IS	184
8	Final considerations	185
8.0.1	Derived publications	185
	Journal article	186
	Book chapter	186
	Proceedings in conferences	186
	Technical reports	187
8.0.2	Research collaborations	187
A	A multiple modelling languages quality scenario	189
A.0.3	Application of multiple models	191
	Business modelling models	192
	System models	194

A.0.4	The first signs of quality problems	196
	Semiotic clarity	197
	Perceptual Discriminability	199
	Semantic Transparency	199
	Visual Expressiveness	200
	Complexity Management	200
	Dual Coding	200
	Graphic Economy	201
A.0.5	Limitations of the selected approach to evaluate the quality of the models of the modelling scenario	201
B	The process-delivery diagram (PDD) specification for the MMQEF method	205
	Bibliography	215
	Index	228

List of Figures

1.1	Design Science scheme applied to the research project.	5
1.2	Roadmap of this thesis.	6
2.1	Summary systematic review protocol performed.	14
2.2	Summary of identified studies that offer definition about quality in modelling languages, in response to RQ2.	17
2.3	Evidence of taxonomy elements in the identified studies for quality definition.	29
2.4	Summary of studies that offer validations, usage reports, and operationalization approaches.	29
2.5	Summary of the literature review protocol performed.	34
2.6	Percentage distribution of identified works by type.	40
2.7	Percentage of quality industrial issues detected.	43
2.8	Percentage of quality academic/research issues detected.	44
2.9	Summary of the intentions found in the analyzed works.	44
2.10	Proposed order for model transformations.	53
2.11	Global distribution of quality issues in industrial and academic/research contexts.	58
3.1	Summary of the reference taxonomy.	64
3.2	Overview of the MMQEF method.	70
3.3	The main blocks of activities of the MMQEF method in the PDD convention.	71
3.4	Metrics for analytic reasoning derived using the GQM approach	74
3.5	The EMAT tool for the MMQEF method.	75
3.6	Summary of the application of the MMQEF method to the CDD methodology of the CaaS project.	77
3.7	Taxonomic analysis of the diagrams of the CDD methodology.	78
3.8	Taxonomic analysis of the information extracted from diagrams of the CDD methodology.	79
3.9	Taxonomic analysis of the metamodel of the CDD methodology.	80
3.10	The FCA lattice obtained from the classification shown in Fig. 3.7 and 3.8.	81
3.11	The FCA lattice obtained from the classification shown in Fig. 3.9.	82
4.1	Summary of the support provided by the reference taxonomy for quality evaluation in MDE contexts.	94
4.2	Zachman taxonomy matching the <i>Backbone taxonomy</i> (Guarino and Welty, 2000)(Welty and Guarino, 2001).	98

4.3	Summary of the FCA method (with figures taken from (Wolff, 1993)).	101
4.4	Example of the taxonomic evaluation for the UML and BPMN modelling languages performed in the EMAT tool.	104
4.5	Example of a lattice generated automatically in EMAT.	106
4.6	Example of an ontological analysis supported by the EMAT tool (extracted from (Ruiz et al., 2014)).	107
4.7	Architecture of the EMAT tool.	108
4.8	Summary of the application of the MMQEF method on UML and BPMN modelling languages.	111
4.9	Taxonomic analysis of the diagrams of the UML and BPMN modelling languages.	111
4.10	Taxonomic analysis of the modelling elements of the UML and BPMN modelling languages	112
4.11	Lattice generated from the classification shown in Fig. 4.9.	113
4.12	Lattice generated from the classification shown in Fig. 4.10.	114
4.13	Summary of the application of the MMQEF method to the integration of the OO-Method and the CA methods.	116
4.14	Taxonomic analysis of the diagrams of the OO-Method and CA methods.	116
4.15	Lattice generated from the classification shown in Fig. 4.14.	117
5.1	Summary of the systematic review protocol performed.	130
5.2	Summary of the systematic review protocol performed.	134
6.1	Summary of the tasks that were involved in the validation.	145
6.2	Resulting binomial distributions for the answers associated to T, DoU, FQI, NQI, QID, and NI by the professionals.	155
6.3	Resulting binomial distributions for the answers associated to T, DoU, FQI, NQI, QID, and NI by the students.	156
6.4	Quartile analysis for the <i>MEM Perceived Ease of Use</i> dimension.	168
6.5	Quartile analysis for the <i>MEM Intention to Use</i> dimension.	168
6.6	Quartile analysis for the <i>MEM Perceived Usefulness</i> dimension.	169
A.1	Current self-assessment process diagram for the University of Quindío (partial view).	192
A.2	The conventions used for the flowchart adaptation at the University of Quindío.	192
A.3	Business modelling models (I)	193
A.4	Business modelling models (II)	194
A.5	System models (I)	195
A.6	System data model (partial view).	196
A.7	Diagram example for the rationale of an architecture decision.	197
A.8	Examples of software products obtained from conceptual models.	197
A.9	Principle of Semiotic Clarity: there should be a 1:1 correspondence between semantic constructs and graphical symbols	198
A.10	Comparison between the symbols used at the University of Quindío and the symbols used in the semantic construct of the flowcharts	198

A.11	Symbols used in the university that meet the <i>semantically immediate / semantically opaque</i> categories.	199
B.1	Association of activities of the <i>Determine the organization of the modelling language</i> block with concepts of the MMQEF metamodel.	206
B.2	Association of activities of the <i>Identify the explicit traces that support the navigation between abstraction levels / viewpoints</i> block with concepts of the MMQEF metamodel.	209
B.3	Association of activities of the <i>Identify the capacities for model transformations</i> block with concepts of the MMQEF metamodel.	210
B.4	Association of activities of the <i>Find the mechanism for integration</i> block with concepts of the MMQEF metamodel.	211
B.5	Association of activities of the <i>Define suitability issues</i> block with concepts of the MMQEF metamodel.	212

List of Tables

2.1	Constrast of model-driven key terms between published MDA guides (part I)	11
2.2	Constrast of model-driven key terms between published MDA guides (part II)	12
2.3	Evaluation scheme applied for model quality studies in accordance with RQ2 (Giraldo et al., 2014).	16
2.4	Summary of the query results in the scientific databases (October 2016).	17
2.5	Summary of identified studies about quality in MDE - part I (updated to October 2016).	20
2.6	Summary of identified studies about quality in MDE - part II (updated to October 2016).	21
2.7	Sampling of categories' authors in quality frameworks for MDE (Part I).	25
2.8	Sampling of categories' authors and reference authors or works in quality frameworks for MDE (Part II).	27
2.9	Expected classification approaches for the identified studies.	28
2.10	Evaluation scheme for the formalism level of the studies.	28
2.11	The works found in the review of the mismatch between the research field of modelling language quality evaluation and the actual MDE practice in industry (Part I). Last update: October 2016.	41
2.12	The works found in the review of the mismatch between the research field of modelling language quality evaluation and the actual MDE practice in industry (Part II). Last update: October 2016.	42
2.13	Quality issues detected for the multiple modelling language scenario.	46
2.14	Summary of the categories definition about quality in MDE contexts (November 2016).	57
3.1	Percentage of participants in the experiment who answered the empty taxonomy survey.	86
3.2	Percentage of answers with no relation to the scope of the taxonomic cells.	86
3.3	Percentage of answers in which UML was detected.	87
3.4	Percentage of answers in which BPMN was detected.	87
3.5	Percentage of answers in which other modelling alternatives were detected.	87

4.1	Results of the survey applied in the first validation of the EMAT tool.	109
4.2	Support of the MMQEF method for the quality levels, goals, and means of the SEQUAL framework (I).	122
4.3	Support of the MMQEF method for the quality levels, goals, and means of the SEQUAL framework (II).	123
5.1	Evaluation scheme applied to the identified studies to evaluate IS with taxonomies.	127
5.2	Summary of studies found and useful studies about taxonomies for evaluating IS (updated: September 2016).	127
5.3	Reference works used to limit the EA frameworks for this review.	129
5.4	Summary of studies found and useful studies (updated: September 2016).	130
5.5	Summary of the primary studies that report EA frameworks with classification of modelling language information (Part I).	132
5.6	Summary of the primary studies that report EA frameworks with classification of modelling language information (Part II).	133
5.7	Summary of studies found and useful studies (updated: September 2016).	135
5.8	Summary of the studies that report the classification of modelling languages with the Zachman taxonomy.	135
6.1	Example of quality issues expected to be reported by the professionals who participated in the empirical validation.	141
6.2	Example of quality issues expected to be reported by the students who participated in the empirical validation.	142
6.3	Paired-comparison design for the validation of MMQEF.	143
6.4	Likert sentences associated to MEM variables (the <i>Perceptions</i> and <i>Intentions</i> dimensions).	144
6.5	Expected treatments for the independent variable.	146
6.6	Dependent variables identified for the validation.	147
6.7	Percentages of knowledge and application of MMQEF key terms.	149
6.8	Resulting levels of knowledge and application of MMQEF terms for the participant professionals.	150
6.9	Resulting levels of knowledge and application of MMQEF terms for the participant students.	151
6.10	Information about knowledge and use of modelling languages and DSLs from the professionals.	153
6.11	Information about knowledge and use of modelling languages and DSLs from the students.	154
6.12	Comparison of dependent variables associated to the hypotheses.	157
6.13	Number of participants who reported issues similar to those that were projected in Tables 6.1 and 6.2.	159
6.14	Statements extracted from participants during the validation with <i>positive comments for MMQEF</i> (Part I).	160
6.15	Statements extracted from participants during the validation with <i>negative comments for MMQEF</i> (Part II).	161

6.16	Statements extracted from participants during the validation with <i>positive comments associated to other quality methods</i> for MDE (Part III).	161
6.17	Statement extracted from participants during the validation with <i>negative comments for other quality methods</i> for MDE (Part IV).	162
6.18	Summary of the identified comments with positive/negative perceptions of the quality methods in the validation.	163
6.19	Comparison of favorable cases for MMQEF regarding the Paired-comparison design.	164
6.20	Summary of the Likert responses of the professionals for MMQEF.	166
6.21	Summary of the Likert responses of the undergraduate students for MMQEF.	167
6.22	H statistic values obtained from the <i>Kruskal-Wallis</i> test for the MEM dimensions.	169
6.23	Analysis of <i>validity threats</i> for the empirical validation (Part I).	171
6.24	Analysis of <i>validity threats</i> for the empirical validation (Part II).	172
6.25	Reported reasons for choosing the alternative method during the empirical validation.	174
6.26	Sources of information for detecting quality issues reported by the participants.	174
A.1	Visual variables of the flowchart notation used at the University of Quindío	200
B.1	Description of activities related to the <i>Determine the organization of the modelling language</i> block.	206
B.2	Description of activities related to the <i>Identify the explicit traces that support the navigation between abstraction levels / viewpoints</i> block (Part I).	207
B.3	Description of activities related to the <i>Identify the explicit traces that support the navigation between abstraction levels / viewpoints</i> block (Part II).	208
B.4	Description of activities related to the <i>Identify the capacities for model transformations</i> block.	209
B.5	Description of activities related to the <i>Find the mechanism for integration</i> block.	211
B.6	Description of activities related to the <i>Define suitability issues</i> block.	212
B.7	Concept table for the MMQEF method in PDD convention (I).	213
B.8	Concept table for the MMQEF method in PDD convention (II).	214

To my beautiful family (Noralba, Danilo, Claudia, and little Sebas) because without their motivation and company this work could be not possible.

Chapter 1

The research context



One of the main challenges in the model-driven engineering (MDE) initiative is the management and integration of languages and models formulated to support multiple views during the development of Information Systems (IS). Modelling languages create and use models that represent materialized views on *concerns* of an IS in accordance with rules defined by viewpoints. In this way, it is possible to mitigate the problems associated with the management of transversal features of an information system (France and Rumpe, 2007a).

Generally, the considerations addressed by languages and models are: *i)* business concerns; *ii)* non-functional features derived from quality attributes; *iii)* new paradigms for software construction (e.g., aspects, collaboration, requirements characterization, etc.); and *iv)* functional and logical concerns.

Currently, there is a proliferation of languages (with their abstract and concrete syntax and their semantics) as well as proposals that emerge for the purpose of managing specific views or perspectives of an IS. There are proposals that define a broad set of symbols and concepts but which are too specific to be of practical general application by academic, research and industrial communities. There are also proposals based on excessively stereotyped UML, which limit the expressiveness or meaning of the models to the stereotyped classes, and/or modifications (or additions) of UML symbols. New UML based notations may not fully satisfy the meaning-meaningful relation that is associated with a specific domain. Therefore, people who designed a notation of this style should be able to transmit the meaning of the concept to be expressed.

This thesis presents a proposal for defining the foundations of an evaluation framework to be applied on modelling languages that are used in MDE projects, for the purpose of determining the *quality* of these languages in the management and technical implementation of an IS according to the views (stakeholders) involved and the main features of the MDE paradigm itself.

1.1 Problem statement

MDE proposes modelling languages as the new abstraction units; hence, the introduction of a new language in an MDE environment should be as easy as

creating a new class in a Java project (Visser and Jos, 2007). Frequently, in MDE projects, it is possible find several proposals of languages, models, notations and tools that manage specific concerns that belong to multiple views of an IS. However, in practice many of these proposals are not applicable due to problems detected in their integration with a previous set of IS models. There are also some MDE initiatives whose domains have metamodels associated to them, but their representation is made in the UML language by stereotyping or modification to the original set of UML elements.

The adoption of MDE approaches has guided the development of a large number of model-driven initiatives. Although MDE emphasizes the use of models as the primary artifacts of an engineering construction process (mostly for software construction), it causes a conceptual divergence in the support of specific views and/or concerns belonging to an IS. This phenomenon is strengthened by the lack of (semantic) support offered by UML or other traditional languages.

Despite the development of metamodels, reference architectural frameworks, and ontological frameworks, has been recognized and widely reported the inability to consistently model all related and inherent views in an IS using a single metamodel or a single notation. The work of (Romero et al., 2009) shows how a single metamodel can only be feasible if the granularity and abstraction level of the viewpoints are similar, which is impossible to guarantee in a typical MDE scenario, which often has many viewpoints.

Because of the increasing collection of modelling languages and notations some methods have been proposed to assess the quality of modelling languages. Some proposals provide guidelines for designing languages based on principles drawn from semiotics and cognitive theory. The rationale behind these proposals is that models are a means to express conceptions about some phenomena, to reason about such conceptions, and to communicate them to others.

Although these methods emphasize the importance of the relationship between the concepts of the modeled concern with respect to the language used, the effort required to formalize semantic definitions creates a high cognitive load for those involved in an MDE process. Besides, these frameworks do not consider the most relevant features of the MDE itself in their formulation. This can be explained as a natural consequence of the many (divergent) interpretations of MDE that result from attempts to create new languages framed in MDE without having rationale support (specific interpretations of MDE). There are so many ways to adopt an MDE approach that it is not possible to establish general conclusions about MDE itself (Cabot, 2013).

In addition, the identified guidelines and frameworks do not evaluate the quality of models from dimensions such as *mapping* or translation between models (even models that belong to the same viewpoint of an IS), nor do they evaluate successful experiences originating from the massive application of a modelling technique in a particular MDE environment.

Most of the reported works about quality in the MDE field do not cover the quality of modelling languages from a MDE perspective, e.g., they do not explain how multiple proposals for managing multiple views in a MDE scenario can co-exist. In (Fettke et al., 2012), the authors highlight that the term *quality* in models does not have a consistent definition, and *it is defined, conceptualized,*

and operationalized differently based on the discourse of each previous research proposal. Works like (Vallecillo, 2010) propose an integration method for multiple languages supported by a reference framework (RM-ODP), but they do not specify how to evaluate the sufficiency, convenience or deficiency of these languages as such in a model-driven scenario.

1.2 Overview of the solution

The quality evaluation method proposed in this research was conceived as a conceptual, methodological and technological framework for the evaluation of modelling languages, whose purpose is to assess one set of languages/models regarding its incorporation and adoption capabilities in a MDE environment. This framework must also establish the capacity of languages to support automation and software generation.

The existence of several languages in an IS model-driven project could generate evidence about those languages that overlap and model IS aspects in a redundant way, or conversely, some of the IS aspects might not be covered by any language. Both situations involve a risk for MDE projects. They influence the adoption of model-driven methods and tools. Therefore, when the languages and tools are established accordingly, it will favor the adoption of model-driven initiatives. When the framework is used it will be possible to optimize the selection of languages. When the framework is applied in a model-driven project, the development time will be reduced and the resources used will be optimized.

When the framework can be applied, the following questions (among others) can be answered:

- Does the model describe more information than is really needed?
- Are the language and notation in accordance with MDE?
- Do the models allow traceability to be performed?
- Is there any aspect of the IS not covered by the identified models?
- Can the models generate fully functional software?
- Does the model cover a specific view of the IS?
- Is it possible to show whether the identified models are for declarative or mapping purposes?

1.3 Research methodology

The main goal of this research is to *formulate a method for the evaluation of the quality of modelling languages used within an MDE project*. This work aims to verify whether it is possible to generate a framework for the evaluation of modelling languages so that it can determine how a language is structured from the MDE viewpoint (i.e., if the language supports views, abstraction levels, integration capabilities, and if it is possible to generate full functional software from

the language(s) under review). The framework must indicate what is missing or what is not necessary in order for a language in an MDE environment to be used correctly.

1.3.1 Research questions

The research was focused on resolving the following questions:

- (RQ1) What problems are there in model-driven projects related to the selection of languages?
- (RQ2) When a set of modelling languages is selected to be used in combination in an MDE project, are there methods for evaluating the suitability of the set of languages?
- (RQ3) What is the set of concepts that are required to be modelled in a model-driven project?
- (RQ4) Propose a method for the evaluation of the quality of a set of languages used jointly within a model-driven project.
- (RQ5) What advantages/disadvantages are obtained by the application of the proposed method?

Through the proposed evaluation framework, the language designer or language engineer (Kleppe, 2008) can determine/evaluate in a practical way if a given language (with its associated artifacts) has the capability to do the following: create models from metamodel, manage views, viewpoints, and perspectives, and to use the potential capabilities of integration offered by the language. This integration would be with other languages used in MDE environments that support domains in accordance with the existing perspectives (at similar or different abstraction levels presented in these environments).

1.3.2 The research roadmap

This thesis uses the *Design Science* guideline proposed in (Wieringa, 2009) for the purpose of defining, managing, and differentiating the *practical* and *knowledge* problems throughout the project. In our research, *knowledge problems* (the *research cycles* - *RC* of Fig. 1.1) identify existing knowledge about IS construction under the model-driven paradigm and the scope and applicability of our proposed framework in MDE contexts. A *practical problem* (*engineering cycle* - *EC* of Fig. 1.1) is the formulation of the quality evaluation framework using MDE and IS reference architectures.

Our engineering cycle started with the identification of the involved stakeholders by means of a literature review of the academic/research field in MDE and reports of the adoption of this paradigm in industrial contexts. For this case, we considered people who are involved in model-driven projects, such as language users and method engineers. This identification includes expectations about the use of combinations of languages for developing IS under model-driven principles.

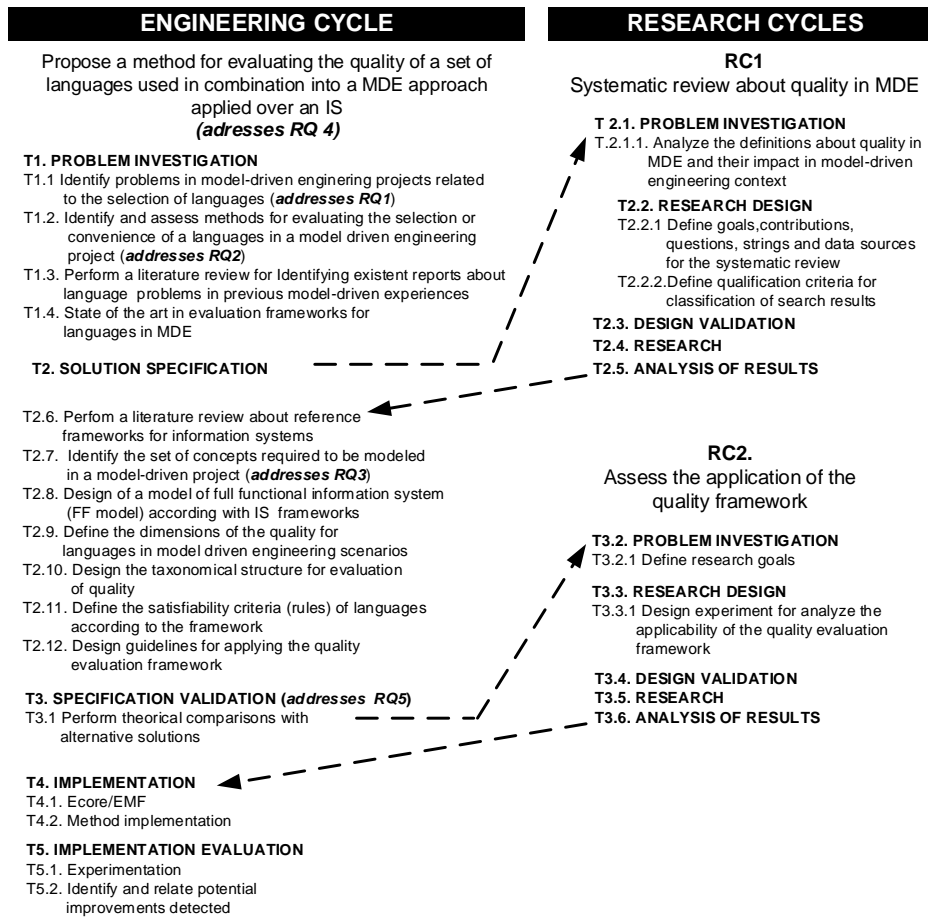


FIGURE 1.1: Design Science scheme applied to the research project.

RQ1 and *RQ2* were answered by performing a systematic review about the concept of quality in model-driven engineering in order to identify the most representative trends of quality in MDE. For *RQ1*, the problem issues were detected in the adoption of modelling languages in projects that follow the model-driven paradigm and previous evaluation frameworks were identified. For *RQ2*, the scope of the evaluation procedures for each evaluation framework was reviewed.

RQ3 was answered through a review about conceptual frameworks for IS, in which a reference architecture for IS was chosen. From this framework, the main conceptions about elements of an IS with MDE features were identified. *RQ4* was answered by the specification of the framework for the evaluation of languages in MDE contexts with its respective use guidelines. *RQ5* was answered by theoretical comparisons and a controlled experiment in which the proposed method was formally evaluated.

Following the research methodology that is summarized in Fig. 1.1, the structure of this thesis is depicted in Fig. 1.2, which presents the roadmap of the

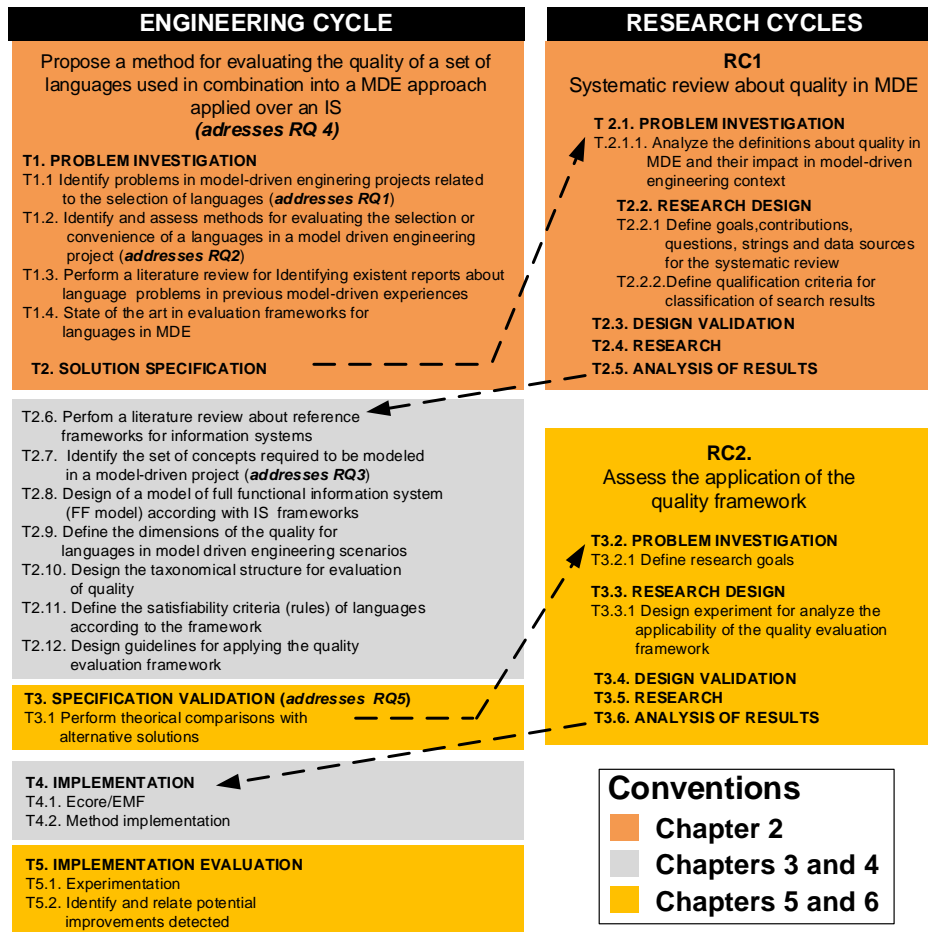


FIGURE 1.2: Roadmap of this thesis.

performed research in order to obtain the evaluation framework for modelling languages in MDE environments.

Chapter 2

Literature review about quality in MDE



The virtue of *quality* is not itself a subject; it depends on a subject. In the software engineering field, quality means good software products that meet customer expectations, constraints, and requirements. Despite the numerous approaches, methods, descriptive models and tools, that have been developed, a level of consensus has been reached by software practitioners. However, in the model-driven engineering (MDE) field, which has emerged from software engineering paradigms, quality continues to be a great challenge since the subject is not fully defined. The *use* of models alone is not enough to manage all of the quality issues at the modelling language level.

In this chapter, we present the current state and some relevant considerations regarding *quality in MDE*, by identifying current categories in quality conception and by highlighting quality issues in real applications of the model-driven initiatives.

We identified sixteen categories in the definition of quality in MDE. From this identification, by applying an *adaptive sampling* approach, we discovered the five most influential authors for the works that propose definitions of quality. These include (in order): the OMG standards (e.g., MDA, UML, MOF, OCL, SysML), the ISO standards for software quality models (e.g., 9126 and 25000), Krogstie, Lindland, and Moody. We also discovered families of works about quality, i.e., works that belong to the same author or topic.

Seventy-three works were found with evidence of the mismatch between the academic/research field of quality evaluation of modelling languages and actual MDE practice in industry. We demonstrate that this field does not currently solve quality issues reported in industrial scenarios. The evidence of the mismatch was grouped in eight categories, four for academic/research evidence and four for industrial reports. These categories were detected based on the scope proposed in each one of the academic/research works and from the questions and issues raised by real practitioners.

We then proposed a scenario to illustrate quality issues in a real information system project in which multiple modelling languages were used. For the evaluation of the quality of this MDE scenario, we chose one of the most cited and influential quality frameworks; it was detected from the information obtained in

the identification of the categories about quality definition for MDE. We demonstrated that the selected framework falls short in addressing the quality issues. Finally, based on the findings, we derive eight challenges for quality evaluation in MDE projects that current quality initiatives do not address sufficiently.

2.1 Introduction

Conceptual models are the main artifacts for handling the high complexity involved in current information system (IS) development processes. The cognitive nature of the models natively supports all of the issues that are derived from the presence of several stakeholders/viewpoints, abstraction levels, and organizational challenges in an IS project. The Model-driven Engineering (MDE) is a software engineering paradigm that promotes the use of conceptual models as the primary artifacts of a complete engineering process. MDE focuses on the business and organizational concerns so that technological aspects are the result of operations over models via transformations or mappings.

An underlying foundation for working with models was proposed in the first version of the Model-driven Architecture (MDA) specification of the Object Management Group (OMG, 2003). Here the basic principles for working and managing models were defined. These can be summarized in two main features: the specification of three *abstraction levels*¹ (Computation-Independent Model - *CIM*, Platform-Independent Model - *PIM*, and Platform-Specific Model - *PSM*), and the definition of the model transformation operations. However, the increase in the number of communities of model-driven practitioners and the lack of a common consensus regarding model management (due to conceptual divergences from practitioners) has produced challenges in the usage and management of models. The MDA 1.0.1 specification has become insufficient to address these challenges (see Section 2.2.1). Paradoxically, some of the derived challenges were formulated in IS frameworks prior to the official release of MDA specification.

One of the most critical concerns for the model-driven paradigm is the difficulty of its adoption in real contexts. Several reports have pointed out issues in model-driven adoption that are related to the misalignment between the model-driven principles and the real context (Burden et al., 2014)(Whittle et al., 2013)(Whittle et al., 2014). Some of these include the overload imposed by the model-driven tools, the lack of traceability mechanisms, and the lack of support for the adoption of model-driven strategies in organizational/development processes. Evidences from model-driven works and real applications suggest symptoms of quality assessment over models. In (Giraldo et al., 2014), the authors demonstrated the wide divergence in quality conception for MDE.

This chapter presents a three-year process to review the literature about the conceptualization of quality in MDE. Unlike other reviews on the same topic (most of which are summarized in (Goulão et al., 2016)), we focus on the identification of explicit definitions of quality for MDE, as well as the perception of quality in model-driven projects from real practitioners and its associated support in the academic/research field. This focus is important considering that,

¹Referred to *Viewpoints* in the original specification.

in the Engineering field, *high quality* is determined through an assessment that takes an artifact under evaluation and checks whether or not it is in accordance with to *its specification* (Krogstie, 2012e). Due to the specific features of the MDE paradigm, it is necessary to establish the impact of the MDE specification on the current initiatives of quality for this paradigm.

This chapter presents the current state of quality conception in model-driven contexts, presenting several factors that influence it. These include the subjectivity of the practitioners, the misalignment between the real application in model-driven scenarios and the research effort required, and the implications that quality in model-driven scenarios must be considered as part of an integral quality evaluation process. This chapter builds upon previous works by the authors (Giraldo et al., 2014, 2015a) and makes the following contributions:

- i)* An analysis of the quality issues detected for both academic/research contexts and industrial contexts is performed in order to determine if current research works on quality in MDE meet the requirements of real scenarios of model-driven usage. This analysis was performed through a structured literature review using *backward snowballing* (Wohlin, 2014) on scientific publications and grey literature (non-scientific publications).
- ii)* A demonstration of quality in MDE issues is presented in a real scenario. This demonstration shows that current proposals of quality in MDE do not cover quality issues that are implicit in IS projects, such as the suitability in multiple-view support, the organizational adoption of modelling efforts, and the derivation of software code as a consequence of a systematic process, among others.
- iii)* A set of challenges that must be considered and addressed in model-driven works regarding quality and the identified categories and industrial/research alignments is presented. This set is derived from the literature reviews and should be integrally considered by any quality evaluation proposal in order to guide model-driven practitioners in how to detect and manage quality issues in MDE projects.

The remainder of this chapter is structured as follows: Section 3.2.1 describes quality in MDE contexts and includes an extension of a previous systematic literature review (Giraldo et al., 2014) to identify the main categories of quality conceptualization in MDE to date. Section 2.3 shows the results of a literature review to determine the mismatch between the quality conceptions in research and the quality conceptions of industrial practitioners and communities of model-driven practitioners. Section 2.4 presents a real example where multiple modelling languages are used to conceive and manage a real Information System. This real scenario highlights quality issues on modelling languages and also the insufficiency of a quality evaluation proposal in MDE for revealing quality issues in the analyzed scenario. Section 2.5 describes some of the challenges that quality in MDE evaluation must address based on the reported findings and evidence. Finally, Section 6 presents our conclusions.

2.2 Quality issues in MDE

2.2.1 Evolution and limitations of the MDA standard

The model-driven paradigm does not have a common conception; instead, there are a plethora of interpretations based on the goals of each model-driven community. The most neutral and accepted reference for model-driven initiative is the MDA specification which reflects the OMG vision about model-driven scenarios. It serves as a common reference for roles and operations in models.

Even though the MDA guide 1.0.1 (OMG, 2003) has been a key specification for model-driven contexts, its lack of updates over a decade has contributed to the emergence of new challenges for model-driven practitioners. Each of these challenges has been addressed by individual efforts and initiatives. Also, this guide did not provide an explicit definition about quality in models and modelling languages despite the definition of key concepts (Tables 2.1 and 2.2) for using models as the main artifacts in a software/system construction process.

The MDA guide 2.0 (OMG, 2014b) released in June 2014 takes into account some of the current model challenges, including issues such as *communication*, *automation*, *analytics*, *simulation*, and *execution*. The MDA guide 2.0 defines the implicit semantic data in the models (which is associated with *diagrams* of models) to support model management operations. Although the MDA 2.0 guide essentially preserves the basic principles of model usage and transformation, it also complements the specification of some key terms and adds new features for the management of models. Tables 2.1 and 2.2 show the differences in some of the key modelling terms between MDA 1.0 and MDA 2.0. One of the most important refinements of MDA 2.0 is the explicit definition of model as *information*.

The MDA guide 2.0 attempts to address current model challenges, including quality assessment of models through *analytics of semantic data* extracted from models (*model analytics*). However, this specification does not prescribes *how* to perform analytics of this kind or quality assessment of models.

Clearly, the refinement of key concepts that is presented in Tables 2.1 and 2.2 (depicted in bold) demonstrates that the MDA guide 2.0 attempts to tackle new challenges that are implicit in modelling tasks. However, this effort is not sufficient considering that the MDA guide does not specify how to identify and manage semantic data derived from models; this guide is only a preliminary (or complementary) descriptive application of model-driven standards.

In addition most of the current challenges for the model-driven paradigm have only been proposed since the emergence of previous information system frameworks by researchers. In fact, IS frameworks such as FRISCO (Falkenberg et al., 1996) (from IFIP²) define key aspects for the model-driven approach. These include the use of models themselves (conceptual modelling), the definition of information systems, and the use of information system denotations by representations (models), the definition of computerized information system, and the *abstraction level zero* by the presence of processors. FRISCO gives MDA an opportunity to consider the communicative factor which is commonly reported as a key consequence of model use (Hutchinson et al., 2011b). In 1996,

²International Federation for Information Processing - www.ifip.org

TABLE 2.1: Contrast of model-driven key terms between published MDA guides (part I)

Key term	MDA guide 1.0.1 (2003)	MDA revision guide 2.0 (2014)
System	We present the MDA concepts in terms of an existing or planned system. That system may include anything: a program, a single computer system, a combination of parts of different systems, a federation of systems (each under separate control), people, an enterprise, a federation of enterprises. Much of the discussion focuses on software within the system.	A system is a collection of parts and relationships among these parts that can be organized to accomplish some purpose.
Model	A model of a system is a description or specification of that system and its environment for a certain purpose. A model is often presented as a combination of drawings and text. The text can be in a modelling language or in a natural language.	A model in the context of MDA is information that selectively represents some aspect of a system based on a specific set of concerns. The model is related to the system by an explicit or implicit mapping. A model should include the set of information about a system to which it belongs.
Modelling language		The structure, terms, notations, syntax, semantics, and integrity rules that are used to express a model.
ViewPoint	A viewpoint on a system is a technique for abstraction using a selected set of architectural concepts and structuring rules in order to focus on specific concerns within that system.	A viewpoint specifies a reusable set of criteria for the construction, selection, and presentation of a portion of the information about a system, addressing particular stakeholder concerns.
View	A viewpoint model or view of a system is a representation of that system from the perspective of a chosen viewpoint	A view is a representation of a particular system that conforms to a viewpoint.

TABLE 2.2: Contrast of model-driven key terms between published MDA guides (part II)

Key term	MDA guide 1.0.1 (2003)	MDA revision guide 2.0 (2014)
Abstraction		Abstraction deals with the concepts of understanding a system in a more general way; said in more operational terms, with abstraction one eliminates certain elements from the defined scope . This can result in introducing a higher level viewpoint at the expense of removing detail. A model is considered to be more abstract if it encompasses a broader set of systems and less abstract if it is more specific to a single system or a restricted set of systems
Platform	A platform is a set of subsystems and technologies that provide a coherent set of functionality through interfaces and specified usage patterns, which any application supported by that platform can use without concern for the details of how the functionality provided by the platform is implemented.	A platform is the set of resources on which a system is realized . This set of resources is used to implement or support the system.
Transformation	Model transformation is the process of converting one model to another model of the same system	Transformation deals with producing different models, viewpoints, or artifacts from a model based on a transformation pattern . In general, transformation can be used to produce one representation from another, or to cross levels of abstraction or architectural layers.

FRISCO suggested the need for harmonizing modelling languages and presented the *suitability* and *communicational* aspects for the modelling languages. Communication between stakeholders is critical for harmonization purposes. It allows important quality issues to be discussed from different views (Shekhovtsov et al., 2014). FRISCO also suggested relevant features for modelling languages (*expressiveness, arbitrariness, and suitability*).

These kinds of FRISCO challenges produce new concerns for model-driven practitioners. For example, *suitability* requires the usage of a variety of modelling languages and *communication* requires the languages to be compatible and harmonized. Since *suitability* concludes that a diversity of modelling languages is needed, the differences between modelling languages (due to this diversity) are unjustified.

MDA was the first attempt to standardize the *model-driven* paradigm, by defining three essential abstraction levels¹ for any model-driven project and by specifying model transformations between higher/lower levels. Even though MDA has been widely accepted by software development communities and model-driven communities, the question about the ability of MDA to meet the actual MDE challenges and trends remains a pending issue.

Generally, despite the specification of the most relevant features for models and modelling languages, the lack of a specification about *when something is in MDE* is evident. This is relevant in order to be able to establish whether or not model-based proposals are aligned with the MDE paradigm beyond the presence of notational elements. There is no evidence of a quality proposal that is aligned with MDE itself.

2.2.2 A literature review about models and modelling language quality trends

In the RCIS 2014 conference, we first presented the preliminary results of a *Systematic Literature Review* (SR) that was performed over 21 months, with the goal of identifying the main trends in quality definition in MDE (Giraldo et al., 2014). This review is ongoing since we are attempting to demonstrate the diversity in the resulting definitions, including the most recent ones.

Fig. 2.1 summarizes the SR protocol that was performed, which follows the Kitchenham guidelines (Kitchenham and Charters, 2007) for ensuring a rigorous and formal search on this topic. As is depicted in Fig. 2.1, the protocol was enriched with an *adaptive sampling* approach (Thompson and Seber, 1996) in order to find the *primary authors* on quality in MDE (see Section 2.2.5).

This SR addressed the following research questions:

- *RQ1*: What does *quality* mean in the context of MDE literature?
- *RQ2*: What does it mean to say that an artifact conforms to the principles of MDE?

While the main research question is RQ1, question RQ2 focuses on the fulfillment of the term *model-compliance*, i.e., whether or not the identified works have artifacts that belong to the model-driven paradigm. For this analysis, we

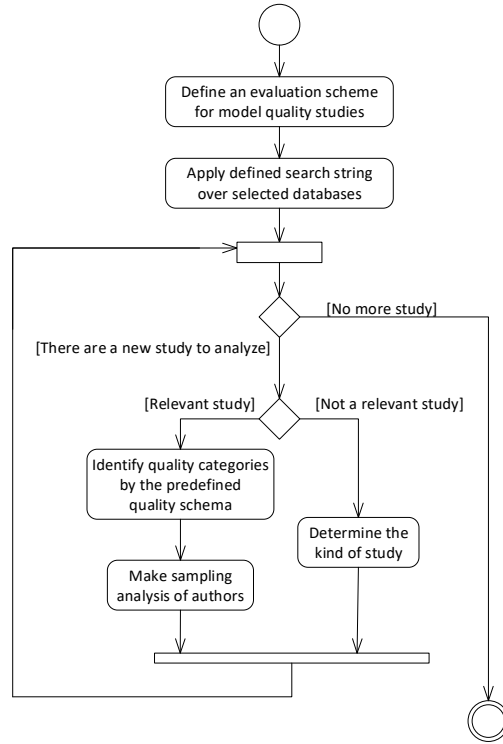


FIGURE 2.1: Summary systematic review protocol performed.

considered modelling artifacts such as models and modelling languages. From RQ1, we derived the search string depicted as follows:

$$\begin{aligned}
 &Quality \wedge (Language \vee Model * \vee Modelling\ language \vee Modelling \vee Notation) \\
 &\wedge (Model - driven * \vee MDD \vee MDA \vee Model - driven\ Architecture \\
 &\quad \vee Model - driven\ development \vee Model - based \vee MDE)
 \end{aligned}$$

The population of this work is made up of the primary studies published in journals, book sections, or conference papers, where an explicit definition about quality in model-driven contexts can be identified. The date range for this work includes contributions from 1990 until now. In order to identify these primary studies, we defined the search string that is presented above. All logical combinations were valid for identifying related works about quality in model-driven contexts. This search string was operationalized according to several configuration options (advanced mode) of each search engine. The information about the selected studies (bibliographical references) was extracted directly from the search engine.

The main sources of the studies were:

- Scientific databases and search engines such as ACM Digital Library, IEEE Explore, Springer, Science Direct, Scopus, and Willey. These include conference proceedings and associated journals.

- Indexing services such as Google Scholar and DBLP.
- Conference Proceedings: CAISE, ER (Conceptual modelling), RCIS, ECMFA, MODELS, RE, HICSS, ECSA, and MODELSWARDS.
- Industrial repositories such as OMG and IFIP.

For this review process, a minimal set of criteria was defined in order to include/exclude studies. These are as follows:

Inclusion criteria:

- Studies from fields such as computer science, software engineering, business, and engineering.
- Studies whose title, abstract and/or keywords have at least one word belonging to each dimension of a search string (*what, in which, and where*).

Exclusion criteria:

- Studies belonging to fields that differ from computer science, software engineering, model-driven engineering, and conceptual modelling (e.g., biology, chemistry, etc.).
- Studies whose title/abstract/keywords do not have at least two dimensions of the search string' configuration.
- Studies related to models in areas/fields that differ from software construction and enterprise/organizational views (e.g., water models, biological models, VHDL models, etc.).
- Studies related to artificial grammars and/or language processing.
- Studies not related to MDA/MDE/ technical spaces (Bézivin and Kurtev, 2005) (i.e., data schemas, XML processing, ontologies).

Due to the variety of studies, a classification schema was defined in order to differentiate and analyze them. Here, RQ2 plays a key role in this literature review because the evaluation of the *model-driven compliant* feature allow us to focus on the main artifacts of the modelling processes: models and modelling languages. Quality definitions are different for both artifacts. In fact, the SEQUAL framework (maybe the most complete work about quality in MDE) defines separately the *quality of models* (Krogstie, 2012b) and the *quality of modelling languages* (Krogstie, 2012c). The first definition is based on seven quality levels (*Physical, Empirical, Syntactic, Semantic and Perceived Semantic, Pragmatic, Social, and Deontic*). The second definition is based on six quality categories (*Domain appropriateness, Comprehensibility appropriateness, Participant appropriateness, Modeller appropriateness, Tool appropriateness, and Organisational appropriateness*).

All of the detected studies were analyzed using the questions in Table 2.3, which were defined in accordance with RQ2. We have resolved all of the questions

TABLE 2.3: Evaluation scheme applied for model quality studies in accordance with RQ2 (Giraldo et al., 2014).

Question		Responses
Does it offer a definition about quality?		Yes, No
<i>If the study does</i>	<i>Is the quality definition about models?</i>	Yes, No
	<i>If the definition is about models</i>	What kind of representations are referenced? Diagram, text
	<i>Is the quality definition about languages?</i>	Yes, No
	<i>If the definition is about languages</i>	What artifacts are referenced? Concrete syntax, abstract syntax, semantic

that this table contains for quality studies detected. These questions identify whether or not quality studies address the scope of the *MDE compliant* feature. For studies that do not offer a quality definition, we identify the type of proposed study based on previous categories detected in our research.

2.2.3 Results

Table 2.4 presents the results of the search string applied in the databases. A second debugging process was necessary to discard studies that appear in the search results but that do not contribute to this research. This new review was made using the abstracts of the studies. These studies were considered to be *not pertinent* for this research despite their presence in the results of the search on academic databases. These works show words that are defined in the search string according to the inclusion criteria defined above; however, they do not explicitly provide any method/definition about quality in MDE and the support for multiple modelling languages. In fact, works of this kind appear as results of the search string, but they cover other topics that are aligned with model-driven approaches. We also discarded repeated studies that appear in the results of searches on multiple databases. Our analysis was made on 176 relevant studies. A summary of the analysis is presented in Fig. 2.2.

This debugging is particularly important because it reflects the broad implications involved in the terms *model* and *quality*. Although these discarded works are model-driven compliance, they reflect the ambiguity that *model-driven compliance* represents (even without full MDA compliance), so the mere existence of models may be criteria enough to determine compliance with the model-driven paradigm. Also, the generality in the use of the terms *model* and *quality* in

³with Computer Science discipline / Engineering subdiscipline

⁴with Business Management discipline / SWE subdiscipline

⁵with Computer Science category

TABLE 2.4: Summary of the query results in the scientific databases (October 2016).

Source	Total search findings	Useful studies
ACM	48	7
IEEE	68	10
Springer ³	659	59
Springer ⁴	641	
ScienceDirect	166	9
Wiley	371	7
Scopus ⁵	784	84
Total selected papers		176

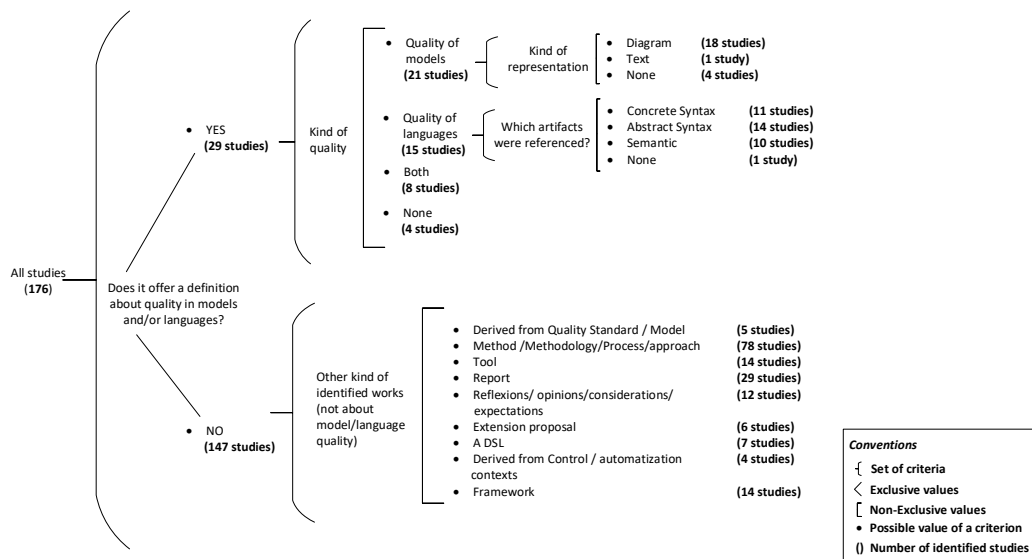


FIGURE 2.2: Summary of identified studies that offer definition about quality in modelling languages, in response to RQ2.

the software engineering context and related areas is demonstrated, producing a diversity of works to support initiatives under those terms as a result.

During the analysis of the 176 primary studies reviewed, we checked whether each paper offered an explicit definition of quality, or at least if the study provided a conceptual framework that would allow a definition of quality to be derived as a result of the application of some theory. Therefore, from the 176 detected studies, we detected 29 studies (16.48% of the target population) that provide a definition of quality in model-driven contexts. The number of papers that provide a definition of quality is relatively low with respect to the number of identified and debugged studies. This indicates that the quality concept leads to works where quality is the result of the application of a specific approach. In those cases quality is reduced to specific dimensions (e.g., metrics, detection of defects, increased productivity, cognitive effectiveness, etc.).

Of the 29 studies that provide definitions about quality, 21 studies (11.93%

of all studies) offer a definition in terms of quality of models. Eighteen of these studies (10.23%) present the quality of models in terms of diagrams (mostly UML), and only one study (0.57%) defines the quality of textual models. In addition, 15 of the 29 quality studies (8.52%) offer a definition of quality at the modelling language level, of which 11 studies (6.25%) mention quality at the concrete syntax level, 14 studies (7.95%) at the abstract syntax level, and 10 studies (5.68%) at the language semantics level. Of the 29 quality studies, 8 studies (4.55%) were detected in which the quality definition is shared between models and modelling languages. Similarly, we detected 4 other studies (2.27%) whose definitions of quality do not consider model or language artifacts. These studies are associated to *Category 1* presented in Section 2.2.4, which proposes a quality model for a quality framework for a specific model-driven approach.

On the other hand, 147 studies were detected (83.52% of total identified studies) that do not provide an explicit definition of quality in model-driven contexts. The presence of these studies is a consequence of specific model-driven proposals formulated to promote specific works on specific aspects of quality such as methodological frameworks, experiments, processes, etc. Of these works:

- Five studies (2.84%) present specific adoptions of standards such as ISO 9126, ISO 25010, descriptive models such as CMMI[®], and approaches such as Goal-Question-Metric (GQM) to support the operationalization of techniques applied in model-driven contexts (including model transformations).
- Seventy-seven of the 176 identified studies (44.32%) have proposed methodologies to perform tasks in model-driven contexts that are commonly framed in quality assurance processes (e.g., behavioral verification of models, performance models, guidelines for quality improvement in the transformation of models, OCL verifications, checklists, model metrics and measurement, etc).
- Fourteen studies (7.95%) report tools that are built to evaluate and/or support the applicability of specific quality initiatives in model-driven contexts.
- Twenty-nine studies (16.48%) are about designed experiments or empirical procedures to evaluate quality features of models that are mostly oriented towards their understandability.
- Twelve studies (6.82%) reported specific dissertations about quality procedures in model-driven contexts such as data quality, complexity, application of agile methodology principles, evaluation of languages, etc.
- Six studies (3.41%) are works that extend predefined model-driven proposals such as metamodels, insertion of constraints into the complex system design processes, definition of contracts for model substitutability, model-driven architecture extension, etc.
- Seven studies (3.98%) propose domain-specific languages (DSL) for specific tasks that are related to model management or model transformations.

- Four studies (2.27%) report model-driven experiences in industrial automation contexts where models become useful mechanisms to generate software with a higher level of quality which is defined as the presence of specific considerations at the modelling level previous to the software production.
- Fourteen studies (7.95%) define frameworks for multiple purposes such as measuring processes, quality of services, enrichment of languages, validation of software implementations according to their design, etc.

The existence of these studies indicate that the terms *quality* and *model* are often used as pivots to highlight specific initiatives that cover only certain dimensions of quality and MDE.

2.2.4 Identified categories of the definition of *quality* in MDE

In this research, a *category* is a set of established practices, activities, or procedures for evaluating the quality of models, regardless of any formality level and the modelling languages involved. According to RQ1, a summary of the defined trends is presented in Tables 2.5 and 2.6⁶. The trends reflect the grouping of the quality works identified.

- **Category 1 - Quality model for MDWE:** This quality model defines and describes a set of quality criteria (usability, functionality, maintainability and reliability) for the model-driven web approach (MDWE). The model also defines the weights for each element of the quality criteria set, and the relation of the elements with the user information needs (MDE, web modelling, tool support and maturity).
- **Category 2 - SEQUAL framework :** This is a semiotic framework that is derived from the initial framework proposed by Linland et al. Quality is discussed on seven levels: *physical*, *empirical*, *syntactic*, *semantic*, *pragmatic*, *social*, and *deontic*. The way different quality types build upon each other is also explained.
- **Category 3 - 6C framework:** These works propose the 6C quality framework, which defines six classes of model quality goals: *correctness*, *completeness*, *consistency*, *comprehensibility*, *confinement*, and *changeability*. This framework emerges as a grouping element that contains model quality definition and modelling concepts from previous works such as Lindland, Krogstie, Sølvsberg, Nelson. and Monarchi.
- **Category 4 - UML guidelines:** In this work the quality of a model is defined in terms of style guide rules. The quality of a model is not subject to conformance to individual rules, but rather to statistical knowledge that is embodied as threshold values for attributes and characteristics. These thresholds come from quality objectives that are set according to the specific needs of applications. From the quality point of view, only deviations

⁶The following conventions are used in the *Type of study* column of Tables 2.5 and 2.6: BC [Book Chapter], CP [Conference Proceeding], JA [Journal Article], WP [Workshop Proceeding], T [Thesis], M [Monograph].

TABLE 2.5: Summary of identified studies about quality in MDE
- part I (updated to October 2016).

ID Study	Year	Type of study	Study	Validated?	Operationalized	Quality definition
1	2011	BC	(Escalona et al., 2011)	NO	NO	<i>Category 1</i>
2	2011	BC	(Espinilla et al., 2011)	YES	YES	Quality model for MDWE
3	2010	CP	(Domínguez-Mayo et al., 2010)	NO	YES	
4	2011	JA	(Domínguez-Mayo et al., 2011)	NO	NO	
5	2012	BC	(Krogstie, 2012b)	NO	YES	
6	2012	BC	(Krogstie, 2012f)	NO	YES	SEQUAL framework
7	1995	CP	(Krogstie et al., 1995)	NO	YES	
8	2009	JA	(Mohagheghi et al., 2009a)	NO	NO	<i>Category 3</i>
9	2008	M	(Mohagheghi and Dehlen, 2008a)	NO	NO	6C framework
10	2007	WP	(Mohagheghi and Aagedal, 2007)	NO	NO	
11	2009	BC	(Hindawi et al., 2009)	YES	NO	<i>Category 4</i> UML guidelines
12	2007	BC	(Lange, 2007b)	NO	YES	<i>Category 5</i> Model size metrics
13	2010	CP	(Amstel, 2010)	NO	NO	<i>Category 6</i>
14	2005	T	(Merilinna, 2005)	YES	YES	Quality in model transformations
15	2011	JA	(Grobshtein and Dori, 2011)	YES	NO	
16	2012	JA	(Chaudron et al., 2012)	NO	NO	<i>Category 7</i> Empirical evidence
17	2005	WP	(Lange and Chaudron, 2005)	YES	YES	about the effectiveness of modelling with UML
18	2010	JA	(Cruz-Lemus et al., 2010)	NO	NO	<i>Category 8</i> Understandability of UML
19	2008	JA	(Heymans et al., 2008)	NO	YES	<i>Category 9</i> Application of model quality frameworks
20	2003	M	(Atkinson et al., 2003)	NO	NO	<i>Category 10</i>
21	2013	WP	(Mijatov et al., 2013)	NO	YES	Quality from structural design properties
22	2014	WP	(López-Fernández et al., 2014)	YES	YES	<i>Category 11</i>
23	2013	WP	(Le Pallec and Dupuy-Chessa, 2013)	YES	YES	Quality of metamodels
24	2007	JA	(Morais and da Silva, 2015)	NO	NO	<i>Category 12</i> Formal quality methods

from these values will lead to corrections; otherwise, the model is considered to have the expected quality. While the style guide notifies the user of all rule violations, non-quality is detected only when the combination of a set of metrics reach critical thresholds.

TABLE 2.6: Summary of identified studies about quality in MDE
- part II (updated to October 2016).

ID Study	Year	Type of study	Study	Validated?	Operationalized	Quality definition
25	2014	JA	(Heidari and Loucopoulos, 2014)	NO	NO	<i>Category 13</i>
26	2015	BC	(Reijers et al., 2015)	NO	NO	Quality factors of business process models
27	2015	CP	(Maes and Poels, 2007)	YES	NO	<i>Category 14</i> Quality procedures derived from IS success evaluation framework
28	2012	CP	(Sayeb et al., 2012)	NO	YES	<i>Category 15</i> A quality patterns catalog for modelling languages and models
29	2015	JA	(Challenger et al., 2015)	YES	NO	<i>Category 16</i> An evaluation framework for DSMLs that are used in a specific context

- **Category 5 - model size metrics:** Quality is defined in terms of model size metrics (MoSMe). The quality evaluation considers defect density through model size measurement. The size is generally captured by the height, width, and depth dimensions. This already indicates that one single size measure is not sufficient to describe an entity.
- **Category 6 - quality in model transformations:** The work presented in (Amstel, 2010) defines the quality of model transformation through internal and external qualities. The internal quality of a model transformation is the quality of the transformation artifact itself. The quality attributes that describe the internal quality of a model transformation are: understandability, modifiability, reusability, modularity, completeness, consistency, and correctness. The external quality of a model transformation is the quality change induced on a model by the model transformation. The work proposes a direct quality assessment for internal quality and an indirect quality assessment approach for external quality, but only if it is possible to make a comparison between the source and the target models.

Other work that is associated to this trend is presented in (Merilinna, 2005). This work proposes a specific tool that automates the quality-driven model transformation approach proposed in (Matinlassi, 2005). To do this, the authors propose a procedure that consists of the development of a rule description language, the selection of the most suitable CASE tool for making the transformations, and the design and implementation of a tool extension for the CASE tool.

In addition, in the work presented in (Grobshtein and Dori, 2011) quality is a consequence of an OPM2SysML view generation process, that uses an algorithm with its respective software application. Thus, quality is defined as the effectiveness and fulfillment of faithfully translating OPM to SysML.

- **Category 7 - empirical evidence about the effectiveness of modelling with UML:** The identified works do not provide a definition for quality in models; it contains a synthesis of empirical evidence about the effectiveness of modelling with UML, defining it as a combination of positive (benefits) and negative (costs) effects on overall project productivity and quality. The work contributes to the quality in models by showing the need for quality assurance methods based on the level of quality required in different parts of the system, and including consistency and completeness dimensions as part of quality assurance practices as a consequence of the communicational purposes of (UML) models.
- **Category 8 - understandability of UML:** This is an empirical study that evaluates the effect that structural complexity has on the understandability of the UML statechart diagram. The report presents three dimensions of structural complexity that affect understandability. The authors also define a set of nine metrics for measuring the UML statechart diagram structural complexity. This work is part of broad empirical research about quality in modelling with UML diagrams where works like (Piattini et al., 2011) can be identified.
- **Category 9 - application of model quality frameworks:** This is an empirical study that evaluates and compares feature diagrams languages and their semantics. This method relies on formally defined criteria and terminology based on the highest standards in engineering formal languages defined by Harel and Rumpe, and a global language quality framework: the Krosstie'SEQUAL framework.
- **Category 10 - quality from structural design properties:** Quality assurance is the measurement of structural design properties such as coupling or complexity based on a UML-oriented representation of components. The UML design modelling is a key technology in MDA, and UML design models naturally lend themselves to design measurement. The internal quality attributes of relevance in model-driven development are structural properties of UML artifacts. The specific structural properties of interest are coupling, complexity, and size. An example is reported in (Mijatov et al., 2013) where the authors propose an approach to validate the *functional correctness* of UML activities by the executability of a subset of UML provided by the fUML standard.
- **Category 11 - quality of metamodels:** Works of this kind specific languages and tools to check desired properties on metamodels and to visualize the problematic elements (i.e., the non-conforming parts of metamodels). The validation is performed over real metamodel repositories. When the

evaluation is done, feedback is delivered to both MDE practitioners and metamodel tool builders.

- **Category 12 - formal quality methods:** This trend is related to the ARENA formal method reported in (Morais and da Silva, 2015) that allows the quality and effectiveness of modelling languages to be evaluated. The reported selection process was performed over a set of user-interface modelling languages. The framework is a mathematical formula whose parameters are predefined properties that are specified by the authors.
- **Category 13 - quality factors of business process models:** In (Heidari and Loucopoulos, 2014) the authors proposed the QEF (Quality Evaluation Framework) method to assess the quality of business processes through their models. This method could be applicable to any business process notation; however, its first application was reported in BPMN models. The framework relates and measures business process quality factors (like resource efficiency, performance, reliability, and etc) that are the inherent property of a business process concept and can be measured by quality metrics.

In this trend, the SIQ framework (Reijers et al., 2015) is also identified for the evaluation of business process models. Here, three categories for evaluating models are distinguished: syntactic, semantic, and pragmatic. By this, there is an inevitable association of SIQ with previous quality frameworks such as SEQUAL (*Category 2*) and some works of Moody; however, the authors clarify that the SIQ categories are not the same as those that were previously defined in the other quality frameworks. The authors show how SIQ is a practical framework for performing quality evaluation that has links with previous quality frameworks. SIQ attempts to integrate concepts and guidelines that belong to the research in the BPM domain.

A complete list of works around quality for business process modelling is presented in (de Oca et al., 2015). This work reports a systematic review for identifying relevant works that address quality aspects of business process models. The classification of these works was performed by the use of the CMQF framework (Nelson et al., 2012), which is a combination of SEQUAL and the Bunge-Wand-Weber ontology.

- **Category 14 - quality procedures derived from IS success evaluation framework:** The authors in (Maes and Poels, 2007) proposed a method to measure the quality of modelling artifacts through the application of a previous framework of Seddon (Seddon, 1997) for evaluating the success of information systems. The method proposes a selection of four related evaluation model variables: Perceived Semantic Quality (PSQ), Perceived Ease Of Understanding (PEOU), Perceived Usefulness (PU) and User Satisfaction (US). This method is directly associated with a manifestation of the *perceived semantic quality* (*Category 2*) described in (Krogstie et al., 1995).

- **Category 15 - a quality patterns catalog for modelling languages and models:** The authors in (Sayeb et al., 2012) propose a collaborative pattern system that capitalizes on the knowledge about the quality of modelling languages and models. To support this, the authors introduce a web management tool for describing and sharing the collaborative quality pattern catalog.
- **Category 16 - an evaluation framework for DSMLs that are used in a specific context:** the authors in (Challenger et al., 2015) formulate a specific quality evaluation framework for languages employed in the context of multi-agent systems (MAS). Their systematic evaluation procedure is a comparison of a modelling proposal with a hierarchical structure of dimension / sub-dimension / criteria items. The lower level (criteria) defines specific MAS characteristics. For this trend, *quality* is a dimension that has two sub-dimensions: the general DSML assessment sub-dimension (with criteria such as domain scope, suitability, domain expertise, domain expressiveness, effective underlying generation, abstraction-viewpoint orientation, understandability, maintainability, modularity, reusability, well-written, and readability) and the *user perspective* sub-dimension (with criteria such as developer ease, and advantages/disadvantages). Both sub-dimensions are addressed by qualitative analysis; it is assumed that this type of analysis is performed with case studies that are designed with experimental protocols.

2.2.5 Adaptive sampling

Using the principles of the *adaptive sampling* approach defined in (Thompson and Seber, 1996), we analyzed the identified papers in order to explore clustered populations of studies about quality in models. We made a review of the bibliographical references of each study detecting *reference authors* or *works* (i.e., previous studies formulated before the publication of the analyzed study that have been cited in the quality studies identified). We established the *reference authors* or *reference works* as those who have been referenced by at least two quality studies detected of different authors.

To do this, we defined Tables 2.7 and 2.8, where the rows refer to the authors of the identified quality studies and the columns contain the referenced authors or works. A link in the (i,j) cell on Tables 2.7 and 2.8 (the color black in the cell fill) indicates that the author of the j column has influenced the authors of the i row; so that the i -author(s) cite the j -author(s) in the quality study(ies) that were analyzed.

TABLE 2.7: Sampling of categories' authors in quality frameworks for MDE (Part I).

Studies ⇑ Authors ⇓	<i>Authors of identified studies</i>																						
	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6 Study 13	Cat. 6 Study 14	Cat. 6 Study 15	Cat. 7	Cat. 8	Cat. 9	Cat. 10 Study 20	Cat. 10 Study 21	Cat. 11	Cat. 12	Cat. 13 Study 25	Cat. 13 Study 26	Cat. 14	Cat. 15	Cat. 16			
Cat. 1	█																						
Cat. 2		█																					
Cat. 3			█																				
Cat. 4				█																			
Cat. 5					█																		
Cat. 6 Study 13						█																	
Cat. 6 Study 14							█																
Cat. 6 Study 15								█															
Cat. 7									█														
Cat. 8										█													
Cat. 9											█												
Cat. 10 Study 20												█											
Cat. 10 Study 21													█										
Cat. 11														█									
Cat. 12															█								
Cat. 13 Study 25																█							
Cat. 13 Study 26																	█						
Cat. 14																		█					
Cat. 15																			█				
Cat. 16																					█		

In the Table 2.7 the columns (or j -authors) correspond to the same authors of quality studies; this was intentionally done in order to show the influence of authors on the analyzed quality studies. Table 2.7 shows that Krogstie (*Category 2*) is the author that has had the most influence on the quality works analyzed. His work influences 50% of the identified quality studies, followed by Lange (*Category 5*) with 31.3%. Two special cases occur in the columns of Krogstie and Mohagheghi (*Category 3*); they appear as authors of identified quality papers, but they were cited by other works that were not detected in the searches of the academic databases. We wanted to highlight the other works of the authors that influence the analyzed studies.

Table 2.7 also shows the studies that are referenced, created, or influenced by works of the same author. These studies do not affect other authors or proposals for quality in models. However, Table 2.7 also shows quality communities of researchers on topics such as model metrics and guidelines mainly applied over UML. Works led by Lange, Chaudron and Hindawi contribute to the consolidation of these research communities. This community phenomenon was originally reported in (Budgen et al., 2011), and is described in works like (Lange and Chaudron, 2006; Lange et al., 2003; Lange and Chaudron, 2005; Lange et al., 2006). In fact, the works of Lange presented in (Budgen et al., 2011) suggest that most model quality problems are related to the design process, which shows that a conflict arises with all viewpoint-based modelling forms, and not just UML.

In the Table 2.8 the columns represent other authors or works which were identified in the review of the bibliographical references for each quality study. As Table 2.8 shows, the OMG specifications and ISO 9126 standard are the most important industrial references that influence the formulation of quality studies.

The OMG specifications were cited by 68.8% of the authors of identified trends. The OMG specifications that were most cited by authors were MDA⁷ specification followed by UML, MOF, OCL, and SysML specifications. Evidence of the adoption of the OMG standard suggests that the works are MDA compliant, but this does not necessarily means an explicit adoption or alignment to the MDA initiative itself. The ISO standards (cited by 50% of the works) are used to support quality model proposals on the taxonomy composed by features, sub-features, and quality attributes. It is even useful for evaluation purposes. This kind of adoption excludes the quality dimensions that are involved in the ISO standards (quality of the process, internal quality, external quality, and quality in use).

Linland's quality framework (Linland et al., 1994) is one of the reference frameworks that is most frequently used and cited by the authors of the primary studies (43.75%). This framework was one of the first quality proposals formulated, and it takes into account the syntactic, semantic, and pragmatic qualities regarding goals, means, activities, and modelling properties. The Krogstie quality framework (an evolution of Linland's framework) is recognized as being the work that has most influenced contemporary works about the quality of models. In the case of Krogstie and Moody (cited by the 31.25% of the works), the authors of the analyzed studies cited early papers where they began to present the first versions and applications of their approaches. Finally, it is important

⁷<http://www.omg.org/mda/>

TABLE 2.8: Sampling of categories' authors and reference authors or works in quality frameworks for MDE (Part II).

	<i>Reference authors or works</i>					
Reference Authors \Rightarrow Studies Authors \Downarrow	(Lindland et al., 1994)	(Moody et al., 2002; Moody and Shanks, 2003; Moody, 2005, 2006)	OMG	ISO/IEC (2001)	(Kitchenham et al., 2002; Kitchenham and Charters, 2007)	(Harel and Rumpe, 2000, 2004)
<i>Category 1</i>						
<i>Category 2</i>						
<i>Category 3</i>						
<i>Category 4</i>						
<i>Category 5</i>						
<i>Category 6</i> <i>Study 13</i>						
<i>Category 6</i> <i>Study 14</i>						
<i>Category 6</i> <i>Study 15</i>						
<i>Category 7</i>						
<i>Category 8</i>						
<i>Category 9</i>						
<i>Category 10</i> <i>Study 20</i>						
<i>Category 10</i> <i>Study 21</i>						
<i>Category 11</i>						
<i>Category 12</i>						
<i>Category 13</i> <i>Study 25</i>						
<i>Category 13</i> <i>Study 26</i>						
<i>Category 14</i>						
<i>Category 15</i>						
<i>Category 16</i>						

to highlight the references to Kitchenham's works to support the application of systematic review guidelines and analysis in procedures on empirical software engineering.

2.2.6 Other findings

For this research, it was particularly important to identify the presence of taxonomy approaches in the primary studies that were found. The reports about validation, use, and operationalization of the primary studies were also identified in order to analyze the quality framework that is proposed in this thesis. Therefore, two additional schemas for the 176 relevant studies were: expected classification approaches, i.e., taxonomy (Table 2.9); and the formalism level (Table 2.10).

TABLE 2.9: Expected classification approaches for the identified studies.

Question		Responses
Does the study show evidence of any classification approach?		Yes, No
<i>If the study does</i>	What kind of approach? (one or more)	Quality Framework, Guideline (orientation), Evaluation Procedure, Tools for evaluating quality

TABLE 2.10: Evaluation scheme for the formalism level of the studies.

Question		Responses
Does the study report any validation?		Yes, No
<i>If the study does</i>	Validation through Non-Empirical Evaluation Techniques	Metamodelling, Metrics Approach
	Empirical Evaluation Techniques	Survey, Laboratory Experiment, Case Study, Action Research
Does the study report a usage scenario?		Yes, No
<i>If the study does</i>	Kind of usage	Under conditions of practice, Academic
Does the study report an operationalization?		Yes, No
<i>If the study does</i>	Kind of operationalization	Metrics, Procedure

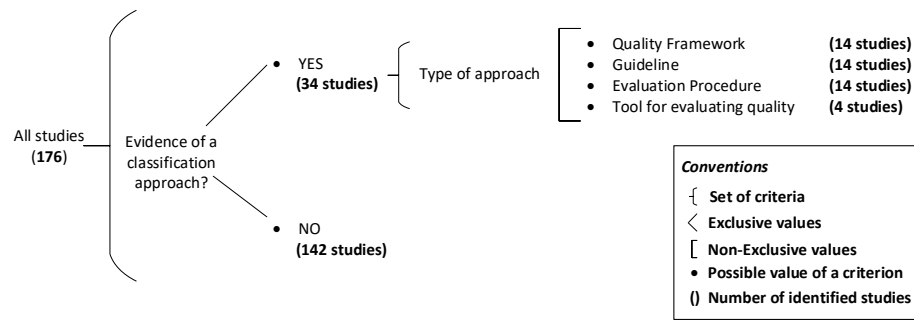


FIGURE 2.3: Evidence of taxonomy elements in the identified studies for quality definition.

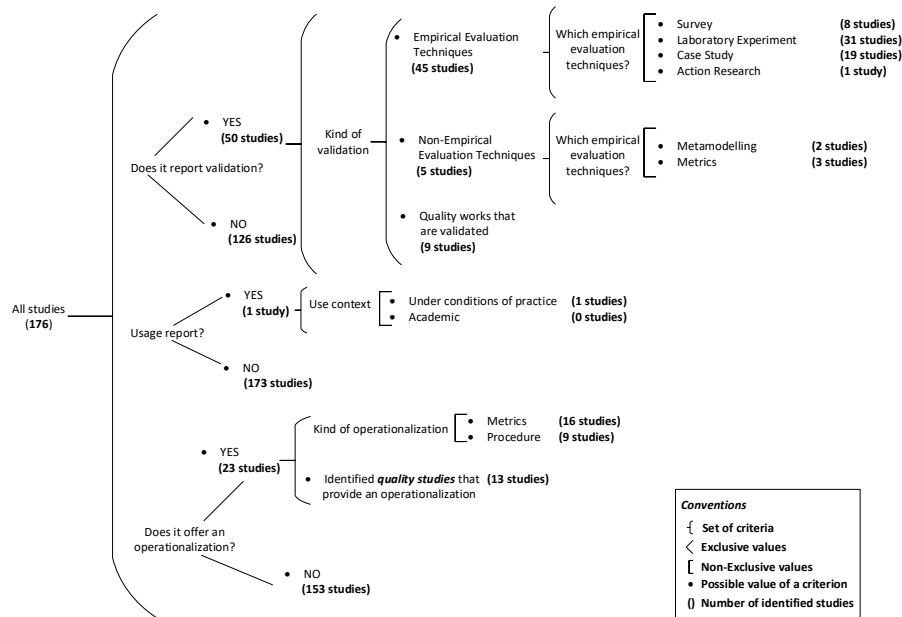


FIGURE 2.4: Summary of studies that offer validations, usage reports, and operationalization approaches.

As Fig. 2.3 shows, 34 of the 176 detected studies (19.32%) report the presence of a classification approach at the model level. The identified classification elements are *quality frameworks*, *guidelines/orientation*, *evaluation routines*. These classifications approaches were reported in 14 studies (7.95%). In addition, the *tools for evaluating quality* approach was reported in 4 studies (2.27%). Of these 34 studies, 17 belong to the 29 studies that report a definition of quality in MDE. In the 17 quality studies, 9 works (5.11%) define a quality framework for models, 10 works (5.68%) have guidelines, 5 works (2.84%) establish an evaluation procedure, and only 2 works (1.14%) present a tool for supporting a quality initiative. No taxonomic structures were found in any of the analyzed works.

Fig. 2.4 depicts the results for the validation, use, and operationalization of

the studies. With regard to the validation of the analyzed studies, 50 of the 176 works (28.41%) report it. The reported validations are empirical evaluation techniques (in accordance with (Siau and Rossi, 1998)) which include the following: laboratory experiments (31 studies – 17.61%), case studies (19 studies – 10.80%), surveys (8 studies – 4.55%), and an *action-research* approach (1 study – 0.57%). Of the 29 quality studies, 9 works (5.11%) report validation through laboratory experiments, two of which report the implementation of software to support the specific approaches formulated in the studies. The information provided in the studies is not sufficient to determine the level of formalism of the experiments performed. Non-empirical validation techniques were also reported. Three of the 176 studies (1.70%) report validation by metrics, and two studies (1.14%) report validation by metamodelling. Of the 29 quality studies, two works report validation through metrics and one work through metamodelling.

With regard to use, of the 176 studies, only 1 non-quality work (0.57%) explicitly reports its use in an industrial condition of practice (in non-academic contexts)(Monperrus et al., 2008b). No quality studies report their use. With regard to operationalization, twenty-three (13.07%) studies report the operationalization of their approaches, 16 quality studies (9.09%) report a kind of operationalization by metrics, and 9 studies (5.11%) propose specific procedures for applying their approaches.

In addition, as a consequence of the searches performed, an identification of studies belonging to the same authors or topics was made. These were sets of related works with specific approaches for evaluating quality in models such as model metrics, defect detections, cognitive evaluation procedures, checklists, and other works about quality frameworks. For our research, this distinction is particularly important because of their presence in the search results; however, most of them do not contribute a formal definition for quality in models. Instead, they focus on specific topics that are considered in quality strategies.

The identified families are the following:

- Understandability of UML diagrams (Piattini et al).
- SMF approach (Piattini et al).
- NDT (University of Sevilla Spain)
- SEQUAL Framework (Krogstie)
- Constraint - Model verification (Cabot et al., and others) (Chenouard et al., 2008)(González et al., 2012)(Tairas and Cabot, 2013).
- fUML (Laurent et al., 2013)(Mayerhofer, 2012).
- OOmcFP (Pastor et al.) (Marín et al., 2010)(Marín et al., 2013)(Panach et al., 2015a).
- 6C Framework (Mohagheghi et al.).

These families show how the interpretation of quality is reduced to specific proceedings or approaches in a way similar to mismatches or limitations on the

term *software quality*. Because of this, some authors like (Piattini et al., 2011) suggest the need for more empirical research in order to develop (at least) a theoretical understanding of the concept of quality in models and modelling languages.

2.2.7 Discussion

Section 2.2.4 answered *RQ1* (the meaning of quality in the MDE literature). The obtained trends of quality were classified in accordance with the schema that was defined in Table 2.3 (derived from *RQ2*). Despite the many model-driven works, tools, modelling languages, etc., the concept of quality has only been ambiguously defined by the MDE community. Most quality proposals are focused primarily on the evaluation of UML for many varied interests and goals.

Works about quality in MDE are limited to specific initiatives of the researchers without having applicability beyond the research or specific works considered. This contrasts with the relative maturity level of quality definitions such as the one presented in Section 2.2.2 (SEQUAL framework).

The low number of works on quality and the diversity of quality trends reflect specific quality frameworks and the respective communities that support these quality concept. The high number of results in the searches performed indicates misconceptions about quality due to the wide spectrum of model engineering in terms of its ease of application (any model can conform to MDE), and the lack of mechanisms to indicate when something is in accordance with MDE.

There are many definitions on quality in models in the literature, but, there is also dispersion and a general disagreement about quality in MDE contexts; this is demonstrated by multiple trends in the quality in MDE presented in Section 2.2.4.

MDE requires a definition of quality that is aligned with the principles and main motivations of this approach. Extrapolation of software quality approaches alone are insufficient because we move from a concrete level (code production, software quality assurance activities) to a higher abstract level to support specific modelling domains.

Traditional evaluations of UML are not enough for a full understanding of quality in models; UML is oriented to functional software features and also, is an object-oriented modelling approach. UML is the defacto software modelling approach, but the evaluation of quality models in terms of UML excludes the overall spectrum of MDE initiatives. Quality evaluation of cognitive effectiveness could restrict the overall quality in models to the diagram and notational levels.

The quality proposals analyzed do not consider how to reduce the complexity added by the model quality activities (experiments, changes in syntax and semantics, evaluation of quality features of a high level of abstraction, etc).

The quality evaluation trends reported does not take into account the implications at the tool level. Tools are a particularly important issue because a language can be explained by its associated tool. New challenges related to the tools that support MDE initiatives have emerged; an example can be seen in (Köhnlein, 2013). In the proposals, tools are limited to validation cases without further applicability beyond the proposal itself. Also, the lack of reports about

the validation and use of the quality proposals demonstrates the level that they were formulated in preliminary stage of research.

2.2.8 The relationship between quality in MDE and V&V

Verification and validation procedures (commonly referred to as V&V) are key strategies in the software quality area for avoiding, detecting, and fixing defects and quality issues in software products. These procedures are applied throughout all the lifecycle of the software product before its release.

MDE also takes advantage of V&V procedures by applying them in modelling artifacts (i.e., languages, models, and transformations) in order to find issues before the generation of artifacts such as source code or other models. One of the most representative examples in the MDE literature of V&V procedures is the MoDEVVa⁸ (Model Driven Engineering, Verification and Validation) workshop of the ACM/IEEE MODELS conference.

Thirteen of the sixteen categories of quality in MDE are associated to specific V&V procedures in MDE reported by the authors, highlighting the studies reported in (Mijatov et al., 2013) - *Category 10* - and (López-Fernández et al., 2014) - *Category 11* - which appear in the proceedings of the MoDEVVa workshop (MoDEVVa 2013 and MoDEVVa 2014, respectively). Three categories (2, 3, and 15) provide guidance for evaluating quality in modelling artifacts. Works of these categories must be interpreted in order to be applied in specific evaluation scenarios.

2.3 A mismatch analysis between industry and academy field

Quality in models and modelling languages has been considered in several ontological IS frameworks even before the formulation of the model-driven architecture (MDA) specification by the Object Management Group (OMG), as mentioned above. The ISO 42010 standard (612, 2011) defines that the architecture descriptions are supported by models⁹, but it recognizes that the *evaluation of the quality* of the architecture (and its descriptions) is the subject of further standardization efforts.

The survey artifact proposed in the CMA workshop of the MODELS conference¹⁰ presents a set of key features for all modelling approaches, considering issues related to the modelling paradigm involved, the notation, views, etc. This is a valuable effort to harmonize the study of the modern modelling approaches, which suggest higher features to analyze in modelling languages. However, some key issues such as usability, expressiveness, completeness, and abstraction management (which are key in ontological frameworks) are poorly described. The

⁸Current version of the MoDEVVa workshop available in <https://sites.google.com/site/modevva/>. Previous versions can be accessed in <https://sites.google.com/site/modevva/previous-editions>.

⁹For ISO 42010, the *architecture* of a system is the *essence* or *fundamentals* of it expressed through models.

¹⁰<http://www.cs.colostate.edu/remodd/v1/sites/default/files/ComparisonCriteria-v3.pdf>.

support for transformations between models, the role of tools in a model-driven context, and the diagrams as main interaction mechanism between models and users also require better descriptions..

The above evidence demonstrates quality in MDE is not an unknown factor for the adoption of model-driven initiatives in real contexts, e.g., software, IS, or complex engineering development processes. Therefore, the consideration and/or use of the MDE paradigm in industrial scenarios is an important source for detecting quality issues, taking into account that it would impact the adoption of model-driven initiatives. It is also important to identify the support of the current MDE quality proposals for the model-driven industrial communities and practitioners.

For this reason, we performed a complementary literature review in order to find evidence of the *mismatch* between the research field of modelling language quality evaluation and actual MDE practice in industry. In (Giraldo et al., 2015a), we presented the preliminary results of a literature review. This search is currently ongoing.

2.3.1 Literature review process design

We have performed a structured literature review using the *backward snowballing* approach. It has been demonstrated that it yields similar results to search-string-based searches in terms of conclusions and patterns found (Jalali and Wohlin, 2012), and we did not want to miss valuable grey literature¹¹ in the results. Grey literature is not published commercially and is seldom peer-reviewed (e.g., reports, theses, technical and commercial documentation, scientific or practitioner blog posts, official documents), but it may contain facts that complement those of conventional scientific publications.

Fig. 2.5 summarizes the literature review protocol that was performed. This literature review is an extension of a previous systematic review reported in Section 2.2.2. The *snowballing* sampling approach helps to identify additional works from an initial reference list. This list was obtained from an initial keyword search. We use the snowballing procedure reported in (Wohlin, 2014) to address the following research questions:

- *RQ1*: What are the main issues reported in MDE adoption for industrial practice that affect modelling quality evaluation?
- *RQ2*: What is the focus of works on modelling quality evaluation in the corresponding research field?
- *RQ3*: Does the term *model quality evaluation* have a similar meaning in both the industrial level and the academic/research level?
- *RQ4*: Is there a clear correspondence between industrial issues of modelling quality and trends in the identified research?

¹¹Grey literature refers to documents that are not published commercially and that are seldom peer-reviewed (e.g., reports, theses, technical and commercial documentation, scientific or practitioner blog posts, official documents). It may contain facts that complement those of conventional scientific publications.

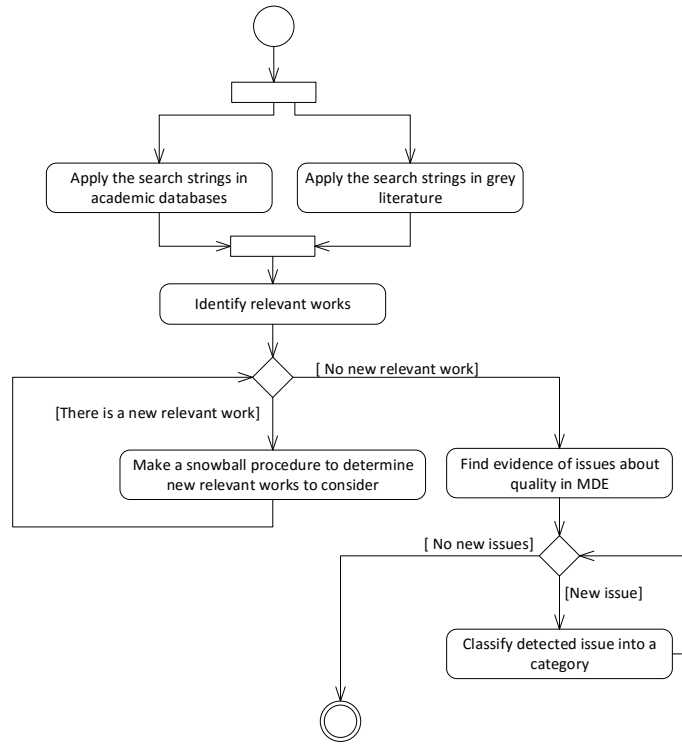


FIGURE 2.5: Summary of the literature review protocol performed.

Our snowballing search method was performed as follows:

1. The initial searches were done on scientific databases and search engines such as Scopus, ACM Digital Library, IEEE Explore, Springer, Science Direct, and Willey¹². These include conference proceedings and associated journals. We used search strings depicted as follows:

$$(MDE \vee Model-driven*) \wedge (real\ adoption \vee adoption\ issues \vee problem\ report)$$

2. For the resulting works, we chose articles that show explicit reports about the applicability of the MDE paradigm in real contexts.
3. For those relevant works, quality issues were identified, and their reference lists were reviewed to find related works on reporting quality issues. This iteration was made until no new works were identified.
4. To complement the quality issues detected, we analyzed web portals of software development communities, such as blogs, technical web sites, forums, social networks, and portals accessed from Google web search, using similar strings regarding previous scientific database searches. Our goal

¹²Currently, search engines such as Scopus could reference other main databases, but we preferred to check the above-mentioned databases to avoid the loss of valuable reports.

was to identify model quality manifestations from software practitioners who work with specific technical and business constraints.

Several inclusion/exclusion criteria were applied on the search results to identify relevant works for our analysis. These criteria are as follows:

Inclusion criteria

- Works where an explicit manifestation of quality on a model-driven issue were included and presented. Examples of these manifestations are *model transformation tool problems*, *misalignment of model-driven principles with specific business concerns*, *skepticism of the model-driven real application and sufficiency*, among others.
- Reports that include an approach to identify model-driven issues in real applications (e.g., interviews with people that perform roles within an IS project, questionnaires, or description about real experiences).
- Works that relate (and/or perform) a literature review approach on the applicability of model-driven approaches in real scenarios.
- For non-academic works (web portals), we checked the impact and quality of the posted information. This was done by reviewing the forum messages, the academic references used, and the level of the community that supports those portals in terms of technological reports, conference-related mentions, and participants' profiles.
- For non-academic works (web portals), we checked the link between authors and participants with well-known companies that report the application of model-driven approaches (e.g., MetaCase, Mendix, Integranova and etc.), and academic/industrial conferences related to model-driven and IS topics (e.g., CodeGeneration Conference, RCIS, CAiSE, MODELS, and etc.).

Exclusion criteria

- Works that report application cases of model-driven compliance approaches or initiatives (notations, application on a specific domain, guidelines, etc.), but whose main focus is the promotion of those specific approaches, without considering the collateral effects of their application.

Each included work was analyzed in order to find quality evidence (i.e., explicit sentences) in the adoption of the model-driven approach reported. Because of the kind of works detected and the level of formality of their sources, it was necessary to access the full content of each work, in order to determine the relevance of each contribution regarding the expectations formulated in our research questions. Despite the common terms used in the search strings, we only accepted works based on the MDE applicability report.

More information about reported quality issues can be found in the technical report available in (Giraldo et al., 2016a). This report presents all the works with their associated statements that support the detected quality issues. During the review of these issues, we found that quality evidence could be categorized as follows:

Industrial issues (RQ1)

- Industrial issue 01: *Implicit questions derived from the MDE adoption itself.*
- Industrial issue 02: *Organizational support for the MDE adoption.*
- Industrial issue 03: *MDA not enough.*
- Industrial issue 04: *Tools as a way to increase complexity.*

Academic/research issues (RQ2)

- A/R issue 01: *UML as the main language to apply metrics over models and defect prevention strategies.*
- A/R issue 02: *Hard operationalization of model-quality frameworks.*
- A/R issue 03: *Software quality principles extrapolated at modelling levels.*
- A/R issue 04: *Specificity in the scenarios for quality in models.*

Sections 2.3.2 and 2.3.3 describe in depth the above categories related to *RQ1* and *RQ2*, respectively. Section 2.3.4 presents the results of the mismatch related to *RQ3* and *RQ4*.

2.3.2 Detected categories for industrial quality issues

In response to *RQ1*, in the following, we present four categories that we defined for grouping the sentences of industrial quality issues. In (Giraldo et al., 2016a), 240 quality sentences are reported from industrial sources. These affect the perception of model-driven initiatives, and, therefore, their quality. Each category groups sentences of several sources that share a common quality issue. These categories were used to facilitate the analysis of the industry-academy mismatch.

The *MDA is not enough* category groups the sentences that report the lack of the MDA specification to resolve questions in the use and application of models and modelling languages (see Section 2.2.1). The *Implicit questions derived from the MDE adoption itself* category groups sentences in which open questions remain unresolved when a model-driven initiative (with its associated set of languages, models, transformations, and tools) is applied in a specific context.

The *Tools as a way to increase complexity* category groups the sentences that report explicit problems in the use and application of model-driven tools (e.g., tools based on the Eclipse EMF-GMF frameworks and associated projects). Tools are the main mechanism for creating and managing models by the application of modelling languages. Finally, the *Organizational support for the MDE*

adoption category groups the sentences that report issues in the organizational adoption of model-driven initiatives.

In the following, we describe each category in more detail:

MDA is not enough

As a reference architecture, MDA provides the foundation for the usage and transformation of models in order to generate software using three predefined abstraction levels. A definition of quality in models that is supported in the alignment with MDA would not be enough. This is because the compliance with the guidelines of this architecture is the minimum criterion expected for the management of models and it must be implicitly supported by current tools and model-driven standards.

A real consequence of this MDA insufficiency is presented in (Hutchinson et al., 2014). The authors show the lack of consensus about the best language and tool as being a pending issue that is not covered in the MDA specification. This issue affects real scenarios where a combination of languages is used to support specific industrial tasks. The model-driven community have recognized the lack of structural updates of the MDA specification in the last decade, which produces imprecise semantic definitions over models and transformations (Cabot). The MDA revision guide 2.0 (OMG, 2014b) released in June 2014 preserves these issues.

Implicit questions derived from the MDE adoption itself

This covers concerns about the suitability of languages and tools (Hutchinson et al., 2014)(Staron, 2006), new development processes derived from MDE adoption (Hutchinson et al., 2014), MDE deployment (Hutchinson et al., 2011a), the scope of the MDE application (Aranda et al., 2012; Whittle et al., 2014), and implicit questions about how and when a MDE approach is applied, e.g., *when and where to apply MDE ?* (Burden et al., 2014), and *which MDE features mesh most easily with features of organizational change? which create most problems?* (Hutchinson et al., 2011a). The correct usage of the modelling foundation in current modelling approaches is also questioned (Whittle et al., 2014).

Tools as a way to increase complexity

The absence of support for MDE tools and the lack of trained people require that great effort be made to adapt to the context of the organization with probably less than optimum results (Burden et al., 2014). This issue leads to problems with the followings: customization, tailoring, and interoperability among modelling tools (Burden et al., 2014; Mohagheghi et al., 2013b), management of traceability with several tools (Mohagheghi et al., 2013b), the high level of expertise and effort required to develop a MDE tool (Burden et al., 2014)(Mohagheghi et al., 2013b), tool integration (Baker et al., 2005)(Burden et al., 2014)(Mohagheghi and Dehlen, 2008b)(Mohagheghi et al., 2013a), the dissatisfaction of MDE practitioners with the available tools (Tomassetti et al., 2012), the lack of technological maturity of the tools (Mohagheghi et al., 2013a), the scaling of

the tools to large system development (Mohagheghi and Dehlen, 2008b), poor user experience (Mohagheghi et al., 2009c), too many dependencies for adopting MDE tools (Whittle et al., 2013), and poor performance (Baker et al., 2005).

Organizational support for the adoption of MDE

This category represents issues that are related to commitments, costs especially training (Hutchinson et al., 2014), resistant to change (Aranda et al., 2012), the alignment and adaptation of MDE with *how* people and organizations work (Burden et al., 2014)(Whittle et al., 2014), and organizational decisions based on diverging expert opinions (Hutchinson et al., 2011b).

The main concern of these works is the misalignment between the model-driven principles and the organizational elements. Most of the works on model-driven compliance are related to technical adoption, such as modelling tools, model-transformation consistency, and the incorporation of models in software development scenarios. However, due to the lack of an explicit model-driven process, organizational issues may not be able to be completely managed in a model-driven approach, by final model users.

2.3.3 Detected categories for academic/research quality issues

In response to *RQ2*, we propose another four categories in order to group the focus of the works on quality evaluation in the academic/research field. Seventy-one issues from this field were reported in (Giraldo et al., 2016a). The categories reflect the intention of the researchers in the model-driven field for managing quality issues. These are as follows:

Hard operationalization of model-quality frameworks

High abstraction and specific model issues influence the operationalization of model quality frameworks (i.e., the instrumentation of a framework by a software tool). Therefore, quality rules or procedures may not be fully implemented by operational mechanisms such as XSD schemas, EMF Query support, etc. In (Störrle and Fish, 2013) present an attempt to make operational the *Physics of notations* evaluation framework (Moody, 2009), however this operationalization (and any similar proposal) could be ambiguous as a consequence of the lack of precision and detail of the framework itself.

An example of model quality assurance tools as reported in (Arendt and Taentzer, 2013) where an operational process for assessing quality through static model analysis is presented. Instead of having an operational model quality framework, a quality framework like *6C* (Mohagheghi et al., 2009a) has been used as a conceptual basis for deriving a quality assurance tool.

The lack of full operationalizations of model quality evaluation frameworks shows that model evaluation is still more an art than science (Nelson et al., 2005), and that current specifications to evaluate quality in models and modelling languages continue to be complex procedures for language designers and final model users.

Defects and metrics mainly in UML

Most of the quality proposals in models focus their effort on the applicability of metrics in UML models and the definition of guidelines to detect and avoid defects in UML diagrams. This trend is a direct consequence of the limitation of the model-driven paradigm in UML terms.

Limitations are based on the specific model-driven vision of OMG. This promotes the model-driven approach in UML, which offers a set of modelling notations that cover multiple aspects of business and systems modelling. MDA also promotes the UML extension using profiles by tailoring the core UML capabilities in a unified tooling environment (OMG, 2003, 2014b).

However, this vision contrasts with the low incidence of UML as the main artifact in software and IS development processes. Clear and recent evidence is reported in (Petre, 2013), where the main trend regarding the use of UML among a group of software experts was *No Usage (No UML)*; the second representative trend was *UML models were useful artifacts for specific and personal tasks, but these were discarded after explanatory tasks were completed*. A very low number of participant experts mention UML in code-generation tasks.

Ambiguity in UML persists due to the specific meanings and interpretations that model practitioners applied to it. This ambiguity directly affects the full adoption of UML as a standard for software and information systems development communities. Also, there is no link between the quality issues reported in UML with the standardization effort of UML by OMG. The complexity in the UML formal specifications contributes to the confusion of model-driven practitioners.

Specificity in the scenarios for quality in models

The most relevant works in this issue have a specific focus from which the quality of models are defined. The quality frameworks formulated in (Krogstie, 2012d; Lindland et al., 1994) have a semiotic foundation due to the use of signs in the process of the domain representation. Other works like (Mohagheghi and Dehlen, 2008a; Mohagheghi et al., 2009a) propose desirable features (goals) for models. Some proposals are specific to the scope of the research performed (e.g. (Domínguez-Mayo et al., 2010)).

Some of the classical procedures for verifying the quality of conceptual models are related to the cognitive effectiveness of notations (generally UML models). In this way, quality motivations are limited to an evaluation (and probably intervention) process on a notation.

Software quality principles extrapolated at modelling levels

Within the MDE literature there are proposals that extrapolate specific approaches for evaluating software quality at model levels, which are supported by the fact that *MDE is a focus of software engineering*. Some of the reported software quality approaches include the usage of metrics, defect detection in models, application of software quality hierarchies (in terms of characteristics, sub-characteristics and quality attributes), best practices for implementing high

quality models and model transformations. There is even a research area that is oriented to the evaluation of the usability of modelling languages (Schalles, 2013), where the usability in diagrams is prioritized as the main quality attribute of models.

The main motivation for this extrapolation is the level of *relative maturity* of the software quality initiatives. In (Moody, 2005) the author suggests the formulation of quality frameworks for conceptual models based on the explicit adoption of the ISO 9126 standard, because of its wide usage in real scenarios and the fact that this standard makes recognizable the properties of a product or service. In (Kahraman and Bilgen, 2013) authors present a set of artifacts that are formulated to support the evaluation of domain-specific languages (DSLs). These instruments are derived from an integration of the CMMI model, the ISO 25010, standard and the DESMET approach. The success of a DSL is defined as a combination of related characteristics that must be collectively possessed (by combining practices from CMMI and ISO 25010 hierarchy). Proposals of this kind assume that there is an existing relation among organizational process improvement efforts, their maturity levels, and the quality of DSL's.

Software quality involves a strategy for the production of software that ensures user satisfaction, absence of defects, compliance with budget and time constraints, and the application of standards and best practices for software development. However, software quality is a ubiquitous concern in software engineering (Abran et al., 2013), and therefore, in the MDE context, additional effort is required for the adoption of the MDE approach.

2.3.4 Findings in the literature review of mismatch

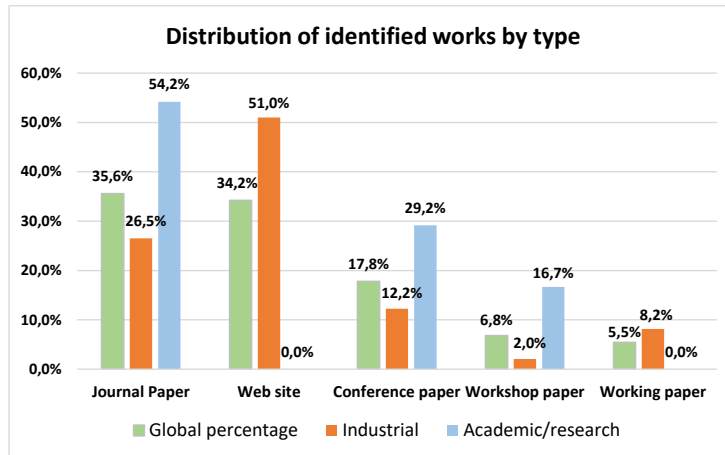


FIGURE 2.6: Percentage distribution of identified works by type.

For this literature review journal papers were the main source of quality issues for both contexts (industrial and research), as shown in Fig. 2.6. However, for the *industrial* context, specialized websites (grey literature) make significant contributions to the quality from a practitioners' perspective. We found 49

TABLE 2.11: The works found in the review of the mismatch between the research field of modelling language quality evaluation and the actual MDE practice in industry (Part I). Last update: October 2016.

	MDE Quality evidence in industry	MDE Quality evidence in academic & research
<i>Industrial issue 01</i>	(Vara and Marcos, 2012; Cuadrado et al., 2014; Hutchinson et al., 2014; Bertrand Portier, 2009; Haan, 2008; Whittle et al., 2014; Dubray, 2011; Linders, 2015; Tone, 2010; Finnie, 2015; Klinke, 2008; Vallecillo, 2014; Hebig and Bendraou, 2014; Brown, 2009; DenHaan, 2009, 2011b; Pierson, 2007; Cabot, 2009; Brambilla, 2016; OMG, 2016; Mohagheghi et al., 2009c; Igarza et al., 2012; Clark and Muller, 2012; Whittle et al., 2015; Aranda et al., 2012; Baker et al., 2005; Whittle et al., 2013; Cuadrado et al., 2014; Hoang, 2012; Platania, 2016; Fournier, 2008)	(Agner et al., 2013; Panach et al., 2015a; Davies et al., 2006; Mussbacher et al., 2014; Quintero and Muñoz, 2011; Quintero et al., 2012; Poruban et al., 2014; Bruel et al., 2015; Picek and Strahonja, 2007)
<i>Industrial issue 02</i>	(Brambilla and Fraternali, 2014; Lukman et al., 2013; Hutchinson et al., 2014; Torchiano et al., 2013; Mohagheghi et al., 2013b; Bertrand Portier, 2009; Whittle et al., 2014; Dubray, 2011; Klinke, 2008; Kulkarni et al., 2010; Vallecillo, 2014; Igarza et al., 2012; Aranda et al., 2012; Baker et al., 2005; Quora, 2014)	
<i>Industrial issue 03</i>	(Brambilla and Fraternali, 2014; Lukman et al., 2013; Bertrand Portier, 2009; Haan, 2008; Dubray, 2011; Corneliussen, 2008; Klinke, 2008; Kulkarni et al., 2010; Vallecillo, 2014; Brown, 2009; Mora et al., 2006; Ortiz et al., 2013; Clark and Muller, 2012; Hutchinson et al., 2014; Platania, 2016; Krill, 2016; Quora, 2015a,b)	(Mussbacher et al., 2014; Singh and Sood, 2009)
<i>Industrial issue 04</i>	(Cachero et al., 2007; Torchiano et al., 2013; Haan, 2008; Linders, 2015; Corneliussen, 2008; Klinke, 2008; Kulkarni et al., 2010; Vallecillo, 2014; DenHaan, 2009, 2011b; Pierson, 2007; Brambilla, 2016; Igarza et al., 2012; Ortiz et al., 2013; Whittle et al., 2015; Hutchinson et al., 2014)(DenHaan, 2010, 2011a; Marín et al., 2014; Aranda et al., 2012; Baker et al., 2005; Whittle et al., 2013; Cuadrado et al., 2014; Hoang, 2012; Quora, 2015b)	(Teppola et al., 2009; Mussbacher et al., 2014; Quintero and Muñoz, 2011; Quintero et al., 2012; Poruban et al., 2014; Bruel et al., 2015; Singh and Sood, 2009)

TABLE 2.12: The works found in the review of the mismatch between the research field of modelling language quality evaluation and the actual MDE practice in industry (Part II). Last update: October 2016.

	MDE Quality evidence in industry	MDE Quality evidence in academic & research
<i>A/R issue 01</i>		(Agner et al., 2013; Davies et al., 2015; Nugroho, 2009; Gorschek et al., 2014; Laguna and Marqués, 2010; Wehrmeister et al., 2014; Dijkman et al., 2008; Picek and Strahonja, 2007)
<i>A/R issue 02</i>		(Fabra et al., 2012; Teppola et al., 2009; Molina and Toval, 2009; Nugroho, 2009; Kessentini et al., 2013; Dijkman et al., 2008; Vallecillo, 2010)
<i>A/R issue 03</i>		(Panach et al., 2015b; Molina and Toval, 2009; Kolovos et al., 2008)
<i>A/R issue 04</i>		(Panach et al., 2015b; Fabra et al., 2012; Van Der Straeten et al., 2009; Vallecillo, 2010)

industrial works and 24 academic/research works; the analysis was made on a total of 73 works.

To answer *RQ3*, Tables 2.11 and 2.12 present the identified works classified in the categories described in Section 2.3.1. The found mismatches show that model-driven practitioners perceive quality of models and modelling languages in different ways. It greatly depends on the application context where modelling approaches are used.

Fig. 2.7 shows the percentage of quality issues detected in the industrial works analyzed. From a real software engineering perspective, there is an initial assumption about the high degree of impact related to model-driven tools and its consequences on development and organizational environments. However, for the industrial works analyzed, we detected the *implicit questions derived from the MDE adoption itself* issue as being the first concern of quality regarding the applicability of models and modelling languages. This issue is derived from the great ambiguity about when something is in MDE (or when something is MDE compliant) and also from the open questions generated in the application of models.

Clearly, industrial publications show a marked trend when discussing the deficiency, consequences, and support of the modelling act itself before using of specific modelling tools. In addition, quality issues related to the tools are evident in the detected works. Beyond the consequences of the application of model-driven initiatives, tools become a key artefact in perceiving, measuring and managing quality issues in modelling languages, taking into account concerns related to organizational, interactional, and technical levels.

The results in Fig. 2.8 highlight the presence of academic and research works that address industrial issues such as *implicit questions derived from the MDE*

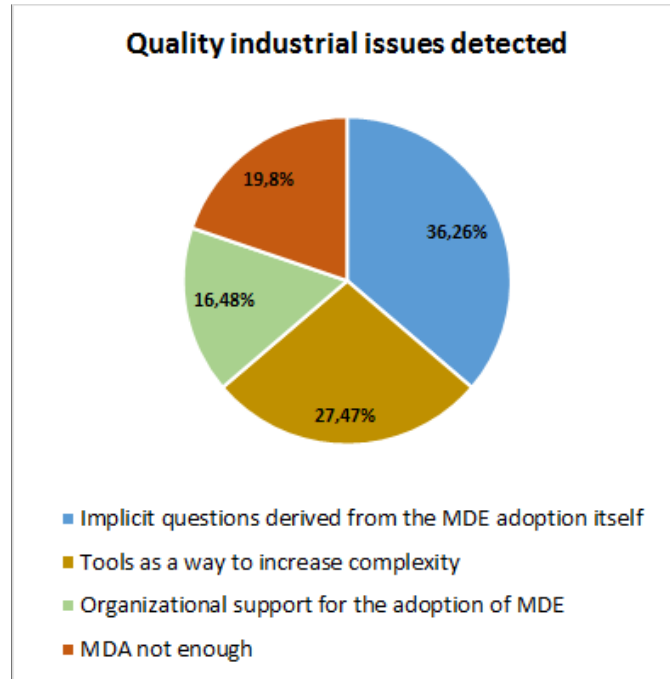


FIGURE 2.7: Percentage of quality industrial issues detected.

adoption itself and *tools as a way to increase complexity*. Some statements from academic and research sources show an alignment with industrial issues. However, in Fig. 2.8, the percentage of works that address industrial issues is lower than the sum of the percentages of works that promote specific interests of researchers in this field. It shows that model-driven researchers tend to focus on theoretical works; thus, these industrial issues are not interesting or relevant to model-driven researchers. This lack of research support increases the conceptual and methodological gaps for the real application of model-driven initiatives and promotes confusion in the model-driven paradigm.

An example of this theoretical emphasis of researchers is the relative proximity of the issue of *implicit questions derived from the MDE adoption itself* of the industrial category and the issue of *defects and metrics mainly on UML* of the academic/research category. There are many efforts that target the quality management of models through the intervention of modelling practices in UML as the defacto language for software analysis/design. There is clearly a gap between these quality trends and the reports about the real usage and applicability of UML, as in the study reported in (Petre, 2013).

Academic/research works also consider the inherent complexity involved in achieving concrete tools from theoretical quality frameworks for models and languages due to the high level of abstraction involved in them. In contrast, industrial works do not report specific quality issues that are related to the academic/research categories. Therefore, for answering *RQ4*, the above evidence demonstrates a very significant difference between the perceptions and

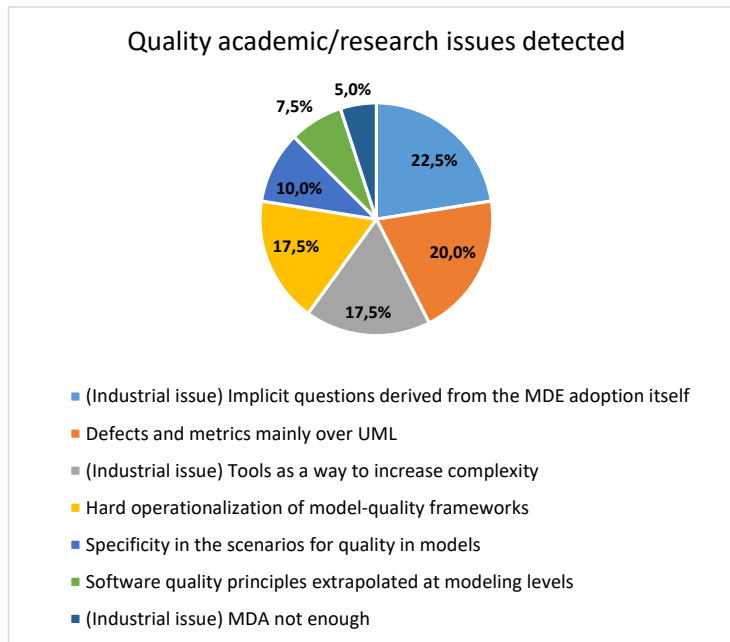


FIGURE 2.8: Percentage of quality academic/research issues detected.

efforts regarding quality in modelling languages and models for industrial and academic/research scenarios. This issue gap between industrial and academic communities requires a method that resolves the problems in industry that are not covered by the current methods.

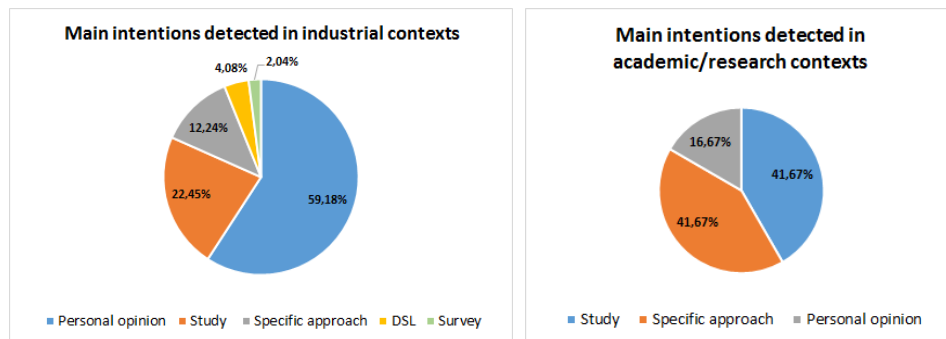


FIGURE 2.9: Summary of the intentions found in the analyzed works.

In the academic-research and industrial contexts, the subjectivity and the particularities of the application scenarios play an important role in the derivation of quality issues in model-driven initiatives. Fig. 2.9 shows the main intention of the analyzed works, depending on whether the work was written for academic/research purposes or for industrial purposes. These intentions refer to personal opinions, studies, or approaches. The main sources for the industrial

context are *opinions* and *interactions in web sites* reported in the grey literature. This is valuable considering that these resources show real experiences of attempts to use model-driven initiatives in real software projects.

In the academic and research field, there is a strong trend (41.67% of reported works) towards specific model-driven initiatives promoted by practitioners. Among these initiatives are DSL, model-driven approaches, operations on models (e.g., searching over models, establishing the level of detail of models), and specific considerations for model transformations (e.g., BPMN models to petri nets). Although several modelling language quality issues were extracted from formal studies performed by researchers, it is important to note how quality issues also serve as *excuses* (or *pivots*) for promoting specific model-driven initiatives.

In summary, the current academic/research methods have not solved quality issues for MDE reported in the industry (Section 2.3.2). It seems that researchers have not yet addressed these problems satisfactorily¹³. Therefore, we consider it necessary to list the open challenges and to define (in a greater depth) the research roadmap proposed in (Giraldo et al., 2015a) in order to cover these issues comprehensively. Thus, in Section 2.4 we show a real scenario in which quality issues associated to the Sections 2.3.2 and 2.3.2 are depicted. Afterwards, in the Section 2.5 we present a set of challenges we inferred from the evidences related in both above literature reviews.

2.4 The sufficiency of current quality evaluation proposals

In this section, we present a scenario for multiple application modelling languages. The case presented in this section was a finished project that had been previously developed by the authors, the implementation of an information system for institutional academic quality management. In this IS project, quality issues were empirically demonstrated. Quality evaluation methods were not used during the execution of this model-driven project. The full specification of the case is presented in Appendix A.

The objective of this scenario is to demonstrate that the application of an existing quality method has not revealed all of the modelling quality issues of the project, despite the execution of the analysis as a post-mortem task. For this empirical study, we have chosen the *Physics of Notations - PoN* - (Moody, 2009), the most widely cited modelling language quality evaluation framework available in the literature. We show that, despite having many useful features, this framework is insufficient to cover all the needs that arise when evaluating the quality of (sets of) modelling languages in MDE projects. The identification of these uncovered needs serves as additional input for the definition of a research roadmap in Section 2.5.

A *post-mortem* analysis was performed to evaluate the quality of a set of modelling languages that were employed in the project (Flowchart, UML, E/R, and

¹³Some attempts and efforts have been made such as (Mussbacher et al., 2014), but quality issues continue to be open challenges.

TABLE 2.13: Quality issues detected for the multiple modelling language scenario.

Id	Description of the quality issue	Detected?	PoN principle
I1	All business process activities are included in the same diagram. This hinders model comprehensibility, especially in the case of domain experts.	Yes	Complexity management.
I2	For the modelling language that was used for modelling the business processes (Flowchart), 81.25% of its symbols were not used.	Yes	Semiotic clarity - Symbol Deficit.
I3	The original semantics of some modelling elements was altered by the domain experts.	Yes	Semiotic clarity - Symbol Excess.
I4	The interpretation of business processes models was affected by the low number of symbols used for modelling them.	Yes	Perceptual Discriminability.
I5	Some modelling elements have a meaning that is not generally correct.	Yes	Semantic Transparency semantically opaque.
I6	A lack of guidance was detected to denote the process flow of the diagrams.	Yes	Visual Expressiveness.
I7	Excessive textual representation in the business diagrams.	Yes	Dual coding
I8	The graphical notation used for business process models is too minimalist.	Yes	Graphic Economy
I9	The Flowchart language is not suitable for modelling business processes or complex systems because of its simplicity.	No	
I10	Final users of the IS suggest the use of BPMN to model the institutional academic quality management process.	No	
I11	The analysts decided to use a UML profile for business process modelling when there are modelling languages that are better suited for this purpose. In fact, in later stages of the project, domain experts happened to discover BPMN and contested the initial language selection.	No	
I12	A decoupling between models from organizational and system levels was found.	No	
I13	There are no mechanisms to manage traceability of modelling artifacts.	No	
I14	Extra effort was required to translate non-UML system models to specific platforms.	No	

architecture languages). Section A.0.3 presents the models that were obtained in the project. Each one of the PoN principles was applied to the obtained models in the project to determine whether or not the models meet the PoN principles. Section A.0.4 presents the results of the quality assessment with the PoN framework.

Table 2.13 summarizes the detected quality issues in the proposed scenario. Although it is true that the application of the PoN framework allows quality issues in the modelling scenario to be detected, other critical quality issues were not detected by this method. PoN meets its goals of analyzing the concrete syntax of the modelling languages under evaluation. However, other quality issues appear for factors such as multiple modelling languages, different abstraction levels, several stakeholders, and viewpoints.

One single quality framework may be insufficient to integrally address all quality issues in MDE projects. Even though there are guidelines to support the application of existing individual quality methods which avoid subjective criteria that influence the final results of the analysis for PON (e.g., (da Silva Teixeira et al., 2016)), there are no systematic guidelines for using quality methods for MDE in combination.

2.5 Open challenges in the evaluation of the quality of modelling languages

Sections 3.2.1, 2.3, and 2.4 presented the problems and questions that remain regarding the evaluation of quality issues in the MDE field. Current phenomena for model-driven applicability, use, and the associated quality issues create several challenges that impact the adoption of the model-driven paradigm. Here it is not enough to evaluate quality from a prescriptive perspective as is proposed for most of the identified quality categories of Section 2.2.4. Any quality evaluation method in models and modelling languages requires the incorporation of the realities regarding MDE itself.

These realities are not unfamiliar to the model-driven community. In the following, we have highlighted the terms and sentences that represent them in bold. They were taken from recognized sources that provide definitions about models. A quick overview of some classical model definitions reveals the presence of *subject* as a fundamental element of the model itself. This is valid for the unified axiom of model as concept in order to understand a subject or phenomenon in the form of description, specification, or theory:

- OMG MDA guide 1.0 (OMG, 2003): A model of a system is **a description or specification of that system** and its **environment** for a certain **purpose**. A model is often presented as a combination of drawings and text. The text may be in a **modelling language** or in a natural language. Model is also a **formal specification** of the **function, structure and/or behavior** of an application or system.

- OMGA MDA guide 2.0 (OMG, 2014b): **A model is information** that selectively represents an aspect of a system based on a specific **set of concerns**. A model should include the **set of information** about a system that is within.
- ISO 42010-2011 (612, 2011): A model can be anything: a model can be a **concept** (a mental model), or a model can be a **work product**. Every model has a **subject**, so **the model must answer questions about this subject**.
- Stanford Encyclopedia of Philosophy (SEP) (Hodges, 2013): A model is a construction of a **formal theory** that describes and explains a **phenomenon**. You model a system or structure that you plan to build by writing a **description** of it.

A conceptual foundation for the model-driven approach was established for the information system community before the formulation of MDA itself, taking into account the main challenges (see Section 2.2.1). ISO 42010 established the importance of the *viewpoint*, *view*, *model kind*, and *architectural description* concepts. Also, the term *correspondence* must be used in the specification of model transformations. Specifically, FRISCO presents the *suitability* and *communicational* aspects for the modelling languages and the need for harmonization of modelling languages. The communicative factor is commonly reported as a key consequence of model usage (Hutchinson et al., 2011b).

In addition, the *subject* of modelling includes quality issues as presented in Section 2.3. The subjective usage of model representations, the freedom to formulate model-driven compliance initiatives, and the wide applicability of models for any IS-supported domain, requires an underlying support to analyze models and all the artifacts that modelling languages provide in order to model any IS phenomena. This rationale must consider the key premises on which the model-driven context was promoted. These become the main input for any model analytics process, in a way that is complementary to previous model quality evaluation frameworks.

The research roadmap of (France and Rumpe, 2007b) was (and continues to be) widely accepted by model-driven practitioners due to their explicit skepticism about the MDE vision and its related problems (including quality evaluation issues). Other roadmaps as presented in (Kolovos et al., 2013; Mohagheghi et al., 2009c; Rios et al., 2006; Vallecillo, 2010) address specific concerns about MDE applicability, with informal considerations about its the adoption in real scenarios, and lack of relation to any IS foundations. These quality issues that we have described in Section 2.3 show a gap between the real application of MDE and its foundational principles.

Because of the divergence of the quality definition in MDE, the lack of support from the academic/research field for practitioners of the model-driven initiatives, and the diverse interpretations by the different research communities, we have deduced a set of challenges that any quality evaluation method for MDE should consider in order to assess quality from a MDE viewpoint (i.e., taking into account the main realities that govern this paradigm). We consider that a

required rationale for quality evaluation in model-driven initiatives must address the following critical challenges in the MDE paradigm itself:

2.5.1 Using multiple modelling languages in combination

This reality is inherent to IS development where multiple views must be used to manage the concerns derived from stakeholders. Each view could have its associated language, and in the same way, one language could support several views of the information system. In this case, if L is the set of all the modelling languages $\{l_1, l_2, \dots, l_n\}$ used to support the views (and viewpoints) in a IS project and Q is the assessment of quality for MDE, then $Q\{L\} \neq Q\{l_1\} \cup Q\{l_2\} \dots \cup Q\{l_n\}$.

Several questions are derived from IS feature: the *suitability* of the languages used to model and manage a specific view, the *coverage* level of the modelling proposals, the relevance and pertinence regarding the specific intention of modelling, and the degree of utility of a modelling language by virtue of the stakeholder concerns under consideration.

Even though, the evaluation of these features heavily depends on subjective criteria, their consideration is mandatory to be able to support modelling and integration approaches on views within a model-driven project (with their respective implications). Subjectivity is intrinsic to the model-driven paradigm, and although an absolute truth in model-driven will not be possible, its consideration facilitates the consolidation of model management strategies in model-driven environments. These quality questions are the essential for information systems.

The treatment of the *multi*-factor concept is not a new topic in the MDE community. It has been considered in previous MDE challenges as reported in (Van Der Straeten et al., 2009). However, the percentage of works that propose a method to manage the *multi* modelling phenomenon is very low (Giraldo et al., 2014) and these do not provide a computerized (operational) tool for model-driven practitioners.

The *multiple* feature in models and information systems (and its derived quality implications) inherently leads to the analysis of the capabilities provided by modelling languages to represent an IS phenomenon adequately and to integrate it with other proposals that cover others IS concerns. The current information systems foundations provide the required inference tools to contrast the capabilities of modelling languages to support the *multiple* feature.

In Section 2.2.3, the percentage of identified works that consider quality evaluation methods for multiple languages is low (4.02%). It shows the minor impact of quality in model proposals on the management of complex information system developments, which contain multiple views and viewpoints supported by conceptual models.

The works that consider evaluation over a set of modelling languages (Krogstie, 2012e,f; Mohagheghi et al., 2009a) present two theoretical evaluation frameworks whose operationalizations are not clear (i.e., any evaluation procedure could be too abstract for the MDE community especially for people from software development contexts). However, their works are a very important advance in the foundation of a body of knowledge for quality in MDE. The evaluation of

multiple modelling languages remains an open issue. Evidence can be found in different reports of the application of some quality works. Generally, these reports present the evaluation of a single modelling language. The evaluation of multiple languages is empirically deducted.

2.5.2 Assessing the compliance of modelling languages with MDE principles

There is a general consensus about the MDE concept as the promotion of models as primary artifacts for software engineering activities (Di Ruscio et al., 2013; González and Cabot, 2014), and as the presence of model transformations that refine abstract/concrete modelling levels. However, due to the generality of this consensus, an initiative may be model-driven without a strict fulfillment of the minimum aspects necessary for real applicability with technological support (e.g., notations without an associated abstract syntax, stereotyped elements of common modelling languages, or modelling proposals with specific intentions and poor adoption by model-driven practitioners).

Despite the specification of the most relevant features for models and modelling languages, there is a lack of specification about *when something is in MDE* (Section 2.3.2); this must be established if model-based proposals are aligned with the MDE paradigm beyond the presence of notational or textual elements. There is no quality proposal that is aligned with MDE itself (i.e., a quality approach that defines a validation procedure to determine whether or not a model-driven initiative meets the MDE core features). Although intuitively one could consider that it all boils down to the extent to which a specific model-driven method meets the core MDE features, literature on quality has not explicitly covered in detail what it means to be aligned with MDE and whether the quality of this alignment can be measured.

It is arguable that the existence of methods claiming to be model-driven that do not actually fulfill the MDE paradigm influences the stakeholders perception of the MDE paradigm itself. For instance, an alleged method might not fulfill expectations, and these negative experiences might end up being generalized to the paradigm itself. This can be a factor that hinders the adoption of MDE approaches and contributes to open issues such as the ones covered in Section 2.3.2.

The definition about *when something is in MDE* or *when something is MDE compliant* must take into account critical concerns beyond the simple usage of models or textual and graphical representations. This includes the alignment of the model with a modelling purpose (in a way similar to the multidimensional views in IS development), the explicit association with an abstraction level (principle that is introduced by MDA), the conceptual support of modelling languages through metamodels, and the capabilities provided by the modelling artifact to integrate with other modelling initiatives and to support models transformations, mappings, and software generation.

In this way, quality (Q) can be defined as the operation $Q = \{L, E\}$, where L is a set $\{l_1, l_2 \dots l_n\}$ of one or more modelling languages in a MDE project and E represents a MDE environment (i.e., the set of the concerns in a MDE

project such as the one described above). Therefore, determining Q implies that $\forall l \in L$, l satisfactorily meets (or addresses) E .

2.5.3 Explicitly using abstraction levels as quality filters of modelling languages

This challenge is a consequence of the MDA specification where three abstraction levels (Computation-Independent Model (CIM), Platform-Independent Model (PIM), and Platform-Specific Model (PSM)) were explicitly proposed in order to clarify and define the usage and scope of models with regard to their intention and closeness with business, system or technical levels.

Abstraction levels act as the reference element to evaluate the convenience of modelling proposals. Harmonization of modelling initiatives within model-driven projects should be supported in information provided by the abstraction levels. Other quality features such as suitability, coverage, communication, integration capacities, and mapping support can be analyzed (possibly predicted) by the explicit presence of abstraction levels. Abstraction levels should not have ambiguous concepts. Theoretical frameworks such as FRISCO provide definitions about computerized information systems and the *abstraction level zero* through the presence of *processors*. In this way, the lower abstraction level is framed around technological boundaries where information is processed.

Abstraction levels are a critical approach for understanding information systems and defining the alignment of model-driven initiatives with business, system, or technical scenarios within an IS architecture (in accordance with the MDA specification). The abstraction levels make the use of modelling techniques explicitly, so that a posterior inference process can determine the suitability of the modelling proposal.

The abstraction level challenge includes a discussion about the convenience of the model-driven architecture and its support through the *instanceOf* relation. This relation occurs between layers, not inside them. No other relations are permissible. This is a constraint artificially imposed without any philosophical or ontological arguments.

The lack of a common consensus about when something is model-driven compliant (challenge 2.5.2) favors the emergence of self-denominated model-driven initiatives without a formal analysis beyond notational proposals that are supported by a problem context that justifies their formulation. The explicit presence of abstraction levels within a model quality evaluation procedure allows the *convenience* of any model-driven compliance initiative to be taken into account based on the rules and prescripts of each level. For example, decisions about the practical implications for using UML at business levels could be addressed and contrasted against the implications of the model semantics and the scope of the business level.

2.5.4 Agreeing on a set of generic quality metrics for modelling languages

The applicability of metrics and measurement processes in models has been used to rate specific elements that are associated to model-driven projects. This

includes the presence of defects (Marín et al., 2013); the size of diagrams (commonly UML diagrams)(Lange, 2007a); model transformations(van Amstel et al., 2009); metamodels (Monperrus et al., 2008a)(Pallec and Dupuy-Chessa, 2013); metamodels with controlled experiments (Yue et al., 2010); etc. Since these application are very specific, works of this kind can only be starting points to define operationalization of specific quality efforts.

Reports about metrics in models present the intention of applying metric approaches derived from software quality works. However, the quality features presented above (Section 2.2.4) do not include certain associated metrics. The usage and applicability of metrics is highly subjective. Consequently, it is not important to discern which specific field of the model-driven paradigm is the most appropriate to identify and implement metrics (e.g., metrics on notations, metrics for the use of models, metrics on metamodels, metrics for a specific modelling language).

Most of the identified works define metrics for models. We recommend metrics for modelling languages. Some works also define metrics for subsets of languages, e.g., metrics that are specified by metamodeling (López-Fernández et al., 2014). However, the scope of these metrics is limited. Therefore, we consider metrics that can be applicable to any modelling language or sets of modelling languages.

The most important contribution of the metrics should be to consolidate the essential aspects of model management in order to establish a set of core modelling features that can be used. This is a challenge given the large size of the model-driven paradigm compared to traditional software development projects. From a MDE pure viewpoint a set of metrics is required to measure the derived features and issues in the information systems modelling process itself.

Thus, approaches such as Goal-Question-Metric (GQM) or other metric-related techniques can be useful for deriving metrics from the goals associated to the modelling act itself (independent of the degree of subjectivity presented). Modelling goals should be aligned with information systems architectural principles over specific individual considerations derived from the application of specific model-driven approaches.

2.5.5 Including model transformations in the modelling language quality equation

Model transformations are critical in model-driven contexts. Modelling languages are often the source or target of model transformations. It is critical to ensure that the modelling languages are appropriate for this purposes.

Transformations constitute the full manifestation of the power of conceptual models in terms of managing the complexity associated with the multiple views and deriving artifacts from the same subject under study. Works such as (Van Amstel, 2010) present new quality features for the transformations. These are derived from a *transformation process* itself (i.e., the rationale of the transformation and its consequences beyond the mere usage of model transformation languages).

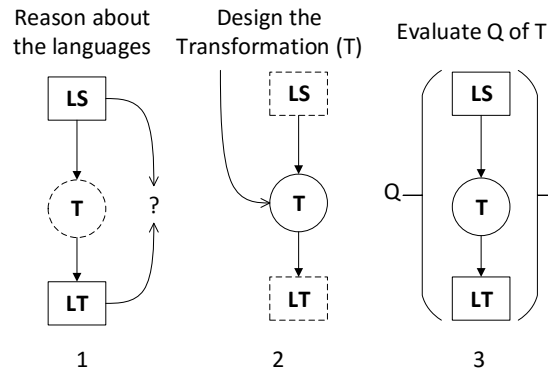


FIGURE 2.10: Proposed order for model transformations.

Some current works propose methods for evaluating the quality of transformation of languages. We think it is important to consider the opposite way, i.e., given the goal of defining a transformation from one modelling perspective to another either horizontally (endogenous) or vertically (exogenous), we need methods to evaluate whether or not the choice of source/target modelling languages is appropriate. This idea is also considered in (da Silva, 2015). The author claims that models must be defined in a consistent and rigorous way; therefore, a *certain level of quality* is required so that models might be properly used in transformation scenarios.

For existing works, a pre-selection of the languages is assumed so that the appropriateness of the transformation is evaluated. However, there are no mechanisms for reasoning whether or not the languages are appropriate. Fig. 2.10 presents an appropriate order for transformations. It includes reasoning about the languages as the first step, then the design of a transformation, and, finally, the quality evaluation.

The inherent complexity of transformations must be tamed by a process, where the main features of the transformation can be identified and managed. Modelling transformation languages cannot provide full support to phenomena derived from issues such as the following: transformations between modelling languages in the same abstraction level; influence of traceability in the transformation; and implications of information carried in traceability models (Galvão and Goknil, 2007); addition of information in mappings models; and differences between mapping and transformation models.

Orientations about model transformations as presented in (Mens and Gorp, 2006) consider the mappings and transformation to be a managed process, where activities such as analysis, design, implementation, verification, and deployment can be performed. Both alternatives (mapping and transformation) must be considered in accordance with the MDA principles (the basis for the general consensus around the model-driven initiative).

All decisions about transformations should not be delegated exclusively to the model transformation language employed; it is an artifact of the model transformation process itself. In addition, semantic rules in models (expressed by

Object Constraint Languages - OCL - for example) require supporting information about considerations for their translation. Addressing the question about *when the conversion under analysis is a mapping model or transformation model* must be the initial activity and orientation of the process itself.

2.5.6 Acknowledging the increasing dynamics of models

Taking advantage of the *context of use* of to the *semiotic dimension* of *pragmatics*, in (Barišić et al., 2011), the authors propose the evaluation of the productivity of domain experts through experimental validation of the introduction of DSLs. This is a key issue because it considers the *quality in use* characteristic for DSLs, so quality in model-driven context transcends beyond the *internal quality* presented in (Moody, 2005). Using usability evaluation, the authors provide some traces for the cognitive activities in the context of languages based on user and tasks scenarios.

Unfortunately, experiments of this kind only consider element languages (except the *representation*) as the natural consequence and interface between the syntax, semantics, and users. *A representation must reflect the semantics of the language*, i.e., implicitly the semantics could be derived from the representations. With this term, we considered both diagrams and textual instances of the modelling languages from the perspective of their users.

The MDA revision guide 2.0 promotes this challenge by presenting analytic procedures that are performed once the *data semantics behind the diagrams* are captured¹⁴. MDA 2.0 prescribes the capture of models in the form of required data for operations such as querying, analyzing, reporting, simulating, and transforming (OMG, 2014b).

There is more evidence that models are no longer static representations of realities. The dynamics in models is increasing in MDE environments. By dynamics we refer to interaction with the elements of the model, the navigation through structures of related models, the simulation of behavior (e.g., GUI models), queries on models, etc.

This dynamics is not usually considered by frameworks for the evaluation of quality in modelling languages. However, it is important for the essential management and use of models in MDE projects. Ignoring it can lead to problems in the final system. Therefore, we believe this challenge must be explicitly considered as part of the quality of (sets of) modelling languages in MDE environments.

Most of the modern proposals about semantics management in model-driven context are too formal and empirical for the community. The lack of an appropriate treatment for representations promotes the presence of modelling tools that do not have the appropriated tools support for modelling purposes (only representations without any association to the semantics). A modelling language can be considered *good* if its associated tool implicitly explains and supports its semantics.

¹⁴The MDA specification particularly promotes the *diagram* term. It can be inferred from previous OMG proposals for managing diagrammatic representations of languages based on arcs and nodes.

2.5.7 Streamlining ontological analyses of modelling languages

The reported methods for evaluating quality in models and modelling languages include artifacts such as guidelines, ontological analysis, experimentation, and usability evaluation. Ontological analysis is one of the approach that is most reported to evaluate modelling languages regarding concrete conceptualizations. Works such as (Becker et al., 2010; Opdahl and Henderson-Sellers, 2002; Siau, 2010) give some examples of evaluation processes with the BWW ontological model applied over UML and DSLs respectively. In (Costal et al., 2011), the enhancement of the expressiveness of the UML was proposed based on the analysis using the UFO ontology. The authors in (Ruiz et al., 2014) use an ontological semiotic framework for information systems (FRISCO) as pivot to integrate two modelling languages; ontological elements are used to relate and support the integration between concepts of both languages.

While it is true that ontological guidance provides a powerful tool to help in the understandability of models (Saghafi and Wand, 2014), ontological analysis includes procedures at philosophical levels which may not be accessible (or interesting) for all of the model-driven community. These analyses are performed by method engineers who have a general vision about the implications of modelling languages in model-driven projects. However, most of the model-driven community are final users of modelling languages, so their interests are focused on the applicability of languages in a domain. An *agile* ontological approach is needed to facilitate analysis and reasoning about the applicability of modelling languages, according to the particular characterizations of the domain being modelled.

The term *agile* means the *real knowledge* about the modelling act in accordance with information systems principles. Agile approaches consider constant improvements, short iterations, and the exchange of knowledge and experience among team members (Silva et al., 2015). Current ontological analysis proposals on models and modelling languages limit their application to specific model-driven communities, which are interested in the evaluation of modelling approaches or the promotion of specific modelling proposals. In addition there are several information systems frameworks (not just ontological frameworks) which contribute their own individual conception of information systems. In order to promote ontologic reasonings about modelling implications, we propose an intermediate stage where a native IS neutral description can be used to classify modelling artifacts before starting the inference process with an information system ontology.

Another important advantage that an agile ontological analysis could offer to the model-driven community is its potential use to develop supporting material (orientation, guidelines, etc) for the correct application of modelling-related practices in real contexts. Some examples of practices are the choice of language, adequate usage of tools, management of traceability information in transformation processes, etc.

2.5.8 Incorporating modelling language quality as a source of technical debt in MDE

Most of the proposed frameworks for quality in models act upon specific model artifacts, abstract syntax, or concrete syntax. These frameworks do not consider the implications of the activities performed in models in terms of the consequences of the good practices that were not followed. This is a critical issue because model-driven projects have the same project constraints as software projects. The only difference is the high abstract level of the project artifacts and the new roles for to domain experts and language users.

The main concern of the term *technical debt* is the consequence of poor software development (Tom et al., 2013). This is a critical issue that is not covered in model-driven processes whose focus is specific operation in models such as model management and model transformations. A landscape for technical debt in software is proposed in (Kruchten et al., 2012) in terms of evolvability and external/internal quality issues. We think that model-driven initiatives cover all the elements of these landscapes since that authors such as (Moody, 2005) suggest models as elements of internal quality software due to their *intermediate nature* in a software development process. Researchers of the Software Engineering Institute (SEI) in (Schmidt) propose a further work that is related to the analysis and management of decisions concerning architecture (expressed as modelling software decisions) because it implies costs, values, and debts for a software development process. The integration between the model-driven engineering and technical debt has not been considered by practitioners of each area despite the enormous potential and benefits for software development processes.

Some of the quality issues reported in Section 2.3 show concerns about the consequences of model-driven applied practices (especially their formal manifestation as model-driven processes). However, unlike traditional software technical debt, the consequences of MDE activities could cover all of the abstraction levels involved, including business and organizational concerns.

The benefit of considering this challenge is twofold because this implies that model-driven processes must be formulated and formalized. In addition, a prior vision of the consequence of model-driven activities will avoid misalignments with the real application context. Most of the MDE applicability problems are generated by technical incidences in the MDE tools. The consequence of any model-driven activity should be measurable and quantified without waiting until the quality is impacted in an specific scenario.

Technical debt in model-driven contexts has begun to be considered by model-driven practitioners. An example is presented in (Izurieta et al., 2015), where the authors explore the improvement of the software architecture quality through the explicit management of technical debt during modelling tasks. In the opinion of the authors, taking the technical debt into account at modelling levels enhances the value added to MDE and it also promotes the progressive adoption of modelling by regular software practitioners.

2.6 Conclusions

The virtue of *quality* does not exist per se; it depends on the *subject* under consideration. In MDE contexts there are a plethora of meanings about quality in MDE as consequence of the multiple interpretations about the real scope of the MDE paradigm. This paradigm ranges from between the mere usage of conceptual models to specialized semantic forms. As a relatively young discipline, multiple conceptualizations of quality have not yet been acknowledged by model-driven communities and practitioners. The most critical consequence of this is the reported misalignment between the expectations of real industrial scenarios and the proposals that emerge from academia.

A greater number of quality concepts in model-driven projects have high abstraction level sources with concerns related to the act of modelling itself. Just as there is widespread belief that *good quality models should generate software artifacts with good quality*, there should be a standard conceptualization of the implications of *good quality models*. However, this conceptualization fails because the paradigm does not establish when something is MDE (or is in compliance MDE). For the model-driven case, the significant impact of subjectivity generates multiple efforts and works about the *quality* term, most of which do not address the real expectations, constraints, and requirements of real contexts.

TABLE 2.14: Summary of the categories definition about quality in MDE contexts (November 2016).

Contribution	Identified category
Specific quality frameworks	Category 1
	Category 2
	Category 3
	Category 13
	Category 14
UML	Category 4
	Category 5
	Category 7
	Category 8
	Category 10
Quality of transformations	Category 6
Concrete scenarios	Category 9
	Category 11
	Category 14
	Category 15
	Category 16

Through two formal literature reviews, we have shown several categories in the definition of quality for the MDE field. We have also analyzed the mismatch of quality evidence between industrial practitioners (and communities of model-driven practitioners) and academic researchers. Table 2.14 and Fig. 2.11

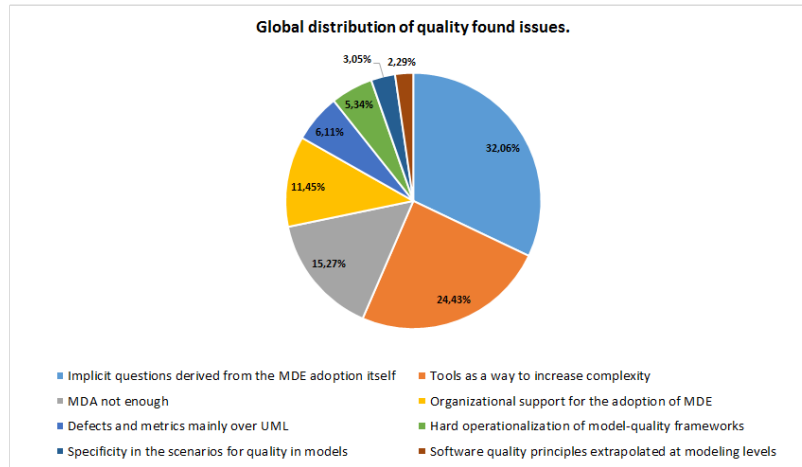


FIGURE 2.11: Global distribution of quality issues in industrial and academic/research contexts.

summarize the main findings of both reviews. Table 2.14 presents the categories identified in the definition of quality in MDE contexts, classifying them according to their main contribution for evaluation procedures. Sixteen definitions of quality for MDE context were detected. Fig. 2.11 summarizes the industrial-academic/research mismatch of model-driven quality issues. One hundred twenty-one issues were detected in the works found by grouping explicit statements that show quality problems.

Detected categories about quality in models are a strong basis from which to start the discussion on this topic. However, most of the MDE core features and challenges are discarded. These include the suitability of languages and their joint usage, the conformity to MDE, the management of abstraction levels, the granularity of models, etc. In Section 2.4, we showed how these quality issues emerge in real model-driven projects with the modelling only act itself because MDE projects are constrained to business, system, and technical concerns. For this reason, we claim that the model-driven community must pay attention to the challenges formulated in Section 2.5 in order to derive quality initiatives with an effective impact on the practitioners of the model-driven paradigm, who mostly come from traditional software development contexts.

The diversity of MDE-compliant works and the lack of a general consensus about MDE (possibly similar to the OMG MDA initiative) produce particular definitions about quality. As Krogstie states: *model quality is still an open issue* (Krogstie, 2012e); it will continue to be an open issue as long as the diversity of ideas about MDE persists. None of the identified categories establish when an artifact can be explicitly considered MDE compliant. Multiple categories confirm that the term *quality in models* does not have a consistent definition and it is defined, conceptualized, and operationalized in different ways depending on the discourse of the previous research proposals (Fettke et al., 2012). Fig. 2.11 shows the *implicit questions from the adoption of MDE itself* as being the main open issue in the perception of quality in MDE that still does not have a satisfactory

response due to the lack of consensus about the scope of the definition of *model-driven compliance*.

The software engineering field has specific standards, efforts, and initiatives that allow practitioners to reach agreements and consensus on the quality conceptualization in software projects. However, the MDE paradigm (which is born from software engineering methods) lacks consensual initiatives due to the multiple new challenges and categories that emerge in quality evaluation in MDE. We believe there must be a comprehensive consensus that takes into account the quality evaluation in MDE by using essential principles of information systems architectures that drive modelling actions and decisions.

Chapter 3

MMQEF: the conceptual and methodological framework



Modelling languages have proved to be an effective tool to specify and analyze various perspectives of enterprises and information systems. In addition to modelling language designs, works on model quality and modelling language quality evaluation have contributed to the maturity of the field. Currently, part of the software industry is slowly but steadily adopting model-driven engineering (MDE) after a well-deserved trough of disillusionment. Although consolidated knowledge on quality evaluation is still relevant to this scenario, in previous works, we have identified misalignments between the topics that academia is addressing and the needs of the industry in applying MDE, thus identifying some remaining challenges. In this chapter, we focus on the need for a method to evaluate the quality

of a set of modelling languages used in combination within a model-driven development environment. This chapter presents MMQEF (*Multiple Modelling language Quality Evaluation Framework*), describing its foundations, presenting its method components and discussing its trade-offs.

3.1 Introduction

When working with complex information systems (IS), developers must elicit, specify, and manage requirements from multiple stakeholders. It is often the case that several perspectives on information systems are combined (Frank, 2012). Conceptual models have proved to be a valuable tool to accomplish this. Methodologically speaking, the model-driven engineering (MDE) paradigm promotes the notion that conceptual models are the main artefacts during IS engineering. The quality of conceptual models is a key factor for the success of MDE development projects. To some extent, the quality of models is influenced by the quality of their corresponding modelling languages.

We conducted some systematic literature reviews that revealed the trends and the discrepancies in the area of model and modelling language quality evaluation (Giraldo et al., 2014, 2015a). More importantly, we found that the industrial practice of MDE has some characteristics that pose open challenges to

current modelling language evaluation methods. An important challenge comes from the fact that many MDE technological environments and projects rely on the combined use of several modelling languages that were not necessarily meant to be used together. Another critical issue is that models used in MDE projects are typically subject to (semi)automatic transformations, requiring the modelling languages to be appropriate for this purpose.

This chapter proposes a solution with the potential to address some of the open challenges. We present *MMQEF*, a method to evaluate the quality of a set of modelling languages used in combination within an MDE context. The core of the method is a classification procedure that uses a reference taxonomy of IS concepts. As a proof of concept, we selected the Zachman framework (Zachman, 1987), but the method could be applied using other IS architectures. We also describe the tool support that we developed and report on a practical application of the method and on an empirical validation, which proves that the Zachman framework is a good candidate for a reference taxonomy.

The remainder of this chapter is structured as follows: Section 3.2 describes the research problem; it also describes the theoretical background of the method for modelling language quality evaluation. Section 3.3 describes the method, explaining its main components and offering guidelines for practitioners. Section 3.4 addresses the applicability of the proposed method and provides rationale for some method design decisions. Section 4.7 concludes the chapter and outlines future work.

3.2 The research problem

Currently, the implementation of complex information systems uses conceptual models for taming the heterogeneity and multiplicity of views that are involved in this type of projects. *Concerns*, *models*, and *views* are essential components of an IS (612, 2011). However, reports about the adoption of the MDE initiatives indicate that MDE itself possesses open questions that affect its applicability, especially in organizational contexts. Reference (Giraldo et al., 2015a) previously reported these gaps and open challenges, based on the perceived information from industrial and academic evidence.

The freedom that MDE promotes for managing IS concerns leads to the formulation of alternatives at modelling levels without a precise rationale beyond the mere justification of the interests or needs of the authors. The adoption of MDE approaches have guided the development of many model-driven compliance initiatives. Although it emphasizes the use of models as primary artefacts of a software construction process, it causes a conceptual divergence in the support of specific views and/or concerns belonging to an IS. This phenomenon is strengthened by the semantics offered by the modelling languages because it can be too formal (complex) or at the other end (without).

A clear example of this type of conceptual divergence is found in the modelling act for software and IS architecture specifications, in which multiple modelling languages have been reported for managing concerns including UML, profiling, stereotypes, other modelling languages, and domain-specific languages (DSLs). Because there is no universal rule for modelling the concerns involved in

an architecture project, any mechanism for transmitting decisions and rationale are valid even if they are not formally specified.

Owing to the increasing collection of modelling languages and notations, some authors have proposed methods for assessing the quality of modelling languages. The rationale behind such proposals is that models are a means to express conceptions about some phenomena, to reason about such conceptions, and to communicate them to others. Reference (Giraldo et al., 2014) previously reported some identified quality evaluation frameworks for modelling languages, but the characteristics of MDE projects require additional features in the method. For example, in MDE projects, it is common for several modelling languages to be used in combination to specify different perspectives of a system. In such cases, the languages might overlap in expressiveness, or they could overlook some relevant concerns. Moreover, in MDE technological frameworks, the quality of modelling languages goes beyond the representational aspects: the language should be designed in a way that facilitates model transformations.

Current modelling language evaluation methods fail to identify the situations described above. These also do not consider the most relevant features of the MDE itself into their formulation. This is a consequence of the several and divergent interpretations of MDE. The lack of consensus regarding MDE produces attempts to add new notations and languages framed in MDE without the support of a rationale.

3.2.1 Conceptual approaches for addressing quality issues in MDE

An approach to address the model-driven issues of quality is the *ontological analysis* (i.e., the assessment of modelling language elements w.r.t the guidance provided by ontology frameworks for ISs). Although the question about whether *ontological guidance results in better models* is an open issue (Saghafi and Wand, 2014), ontologies for IS including the BWV and the UFO are commonly used to evaluate modelling languages in accordance with ontological constructs, establishing their completeness through a mapping process between modelling elements and ontological constructs. Other reported usages or examples of ontological analysis are the integration of modelling languages and the incorporation of modelling constructs inside previous proposals of modelling languages.

The main support provided by the ontological analysis approach is inference-based reasoning, in which the role of an *analyst* (or *designer*) of languages can assess the consequences of the constructs for a language in a specific representation of the real world that is interesting for users of ISs. Often quality is referred to as an ontology-based solution owing to the difficulty of distinguishing between the *problem space* scope and the *solution space* scope associated with the *model-based* and *ontology-based* approaches, respectively (Shekhovtsov et al., 2014).

Another conceptual tool reported in the analytics process in model-driven

contexts is *taxonomic analysis*. This tool provides guidance for specific modelling tasks through the classification of artefacts or approaches. The decisions on modelling artefacts¹ are made based on previous classification schemas. Examples of taxonomies applied in MDE contexts are: taxonomies for model transformations, taxonomies for MDE tools, taxonomies of model synchronization types, and taxonomies for managing the evolution of modelling languages.

Despite the high potential of taxonomic analysis for understanding the implications of model-driven practices, there are few reports of their use. In a way similar to ontological analysis, the process of taxonomic assessment can lead to subjective analysis owing to its focus on specific features of the model-driven paradigm. Ontological and taxonomic analysis approaches must be used in a complementary way. Mechanisms for inference and classification are valuable strategies for managing challenges in the adoption of the model-driven paradigm because of their support for reasoning over models.

3.2.2 The reference taxonomy

The *Zachman framework* (Zachman, 1987; Sowa and Zachman, 1992) (hereinafter called the *reference taxonomy*) is a taxonomy derived from linguistic and philosophical sources, whose main purpose is to support a process for rationalizing the systemic use of IS elements to define an enterprise solution. The rationale process uses a procedure of classification. This taxonomy was conceived as an architecture proposal for the description of ISs that identifies the essential elements in a holistic system, which will be deployed in an organization. It established the basis for (and also influenced) current relevant standards such as ISO 42010 (612, 2011) (*software and systems architecture descriptions*) and frameworks for enterprise architecture.

		Classifiers (viewpoints definitions)										
Perspectives (roles)		DATA What (Things)	FUNCTION How (Processes)	NETWORK Where (Location)	PEOPLE Who (People)	TIME When (Time)	MOTIVATION Why (Motivation)					
MDA Abstraction levels	SCOPE (Contextual) Planner	Computation-Independent Model (CIM)					Key things	Key process	Key locations	Key people	Key events	Key strategies
	BUSINESS MODEL (Conceptual) Owner	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	
	SYSTEM MODEL (Logical) Designer	Platform-Independent Model (PIM)					DSL cell	DSL cell	DSL cell	DSL cell	DSL cell	DSL cell
	TECHNOLOGY MODEL (Physical) Builder	Platform-Specific Model (PSM)					DSL platform cell	DSL platform cell	DSL platform cell	DSL platform cell	DSL platform cell	DSL platform cell
	DETAILED REPRESENTATIONS (Out-of-Context) Sub-Contractor	Code or physical implementations					Specific data implementation	Specific functionality implementation	Specific network implementation	Specific role implementation	Specific timing operationalization	Specific rules implementation

FIGURE 3.1: Summary of the reference taxonomy.

¹We use this term to refer to modelling languages and models generated from these languages.

Fig. 3.1 summarizes the main features of the taxonomy reference. Basically, it is a classification language with a bidimensional configuration (taxonomy structure) that constitutes the *notation* of this classification language. Rows are the *abstraction levels* involved in an IS project; they are represented as roles relevant to levels of the organization and business (domain) down to the level of the specific technology for the implementation. Columns are philosophical *classifiers* (Thalheim, 2011)(Thalheim, 2012) used to justify the elements involved in the construction of an IS regardless of the abstraction levels considered globally. The use of the classifiers is commonly reported in the IS literature as an *analytic* conceptual tool for supporting reasoning and decisions. Classifiers are elements that derive implicit rules on views of IS (i.e., *viewpoints*).

Combinations of rows and columns (abstraction levels and classifiers, respectively) produce cells with specific purposes. They have a basic and implicit model that must be fulfilled for any modelling initiative that covers these purposes; this is the *DSL cell*. Fig. 3.1 also depicts the alignment of the taxonomy with the MDA (OMG, 2003) architecture of reference according to a previous work reported in (Frankel et al., 2003). For the cells associated to the *Platform-specific Model* (the same *Technology model - physical* level of the taxonomy) a *DSL cell* is constrained by the implementation platform (e.g., models of Enterprise Java Beans for Java projects or models of Entity Framework for .Net platform); this is the *DSL platform cell*.

The combination of abstraction levels and classifiers gives taxonomic units (i.e., the cells in the grid) a unique meaning, scope, and intention, with associated *metaconcepts* that are specific for each cell. Some previous efforts have attempted to abstract the foundational concepts of the taxonomy using MOF-based metamodels and integration with IS foundational ontologies; however, there is no consensus. Despite this, these *metaconcepts* are implicit in the same specification of the taxonomy, so it is possible to infer the foundations of specific languages for each cell. In this way, the resulting classification is a comparison between the information of a modelling language (or information obtained from its instanced models) and the information contained in the metaconcepts of the cells.

The main feature of the reference taxonomy is its native support for the management of the semantics through its semantic bidimensional structure (Zachman, 2003), in which conceptual models involved in an IS development process can be classified. The MDA guide 2.0 (OMG, 2014b) focuses on the *semantic data* to perform model analytics procedures. However, it does not define any method to deduce these types of data in modelling artefacts. The semantics are defined w.r.t. a *semantic domain* and the mapping of the syntax within that domain. The semantic domain specifies the concepts that exist in the universe of discourse. This is a prerequisite for comparing semantic definitions (Harel and Rumpe, 2004). The reference taxonomy provides a semantic domain that abstracts the IS reality, and it allows the capture of decisions on the concepts that modelling artefacts must manifest. This implies that the meaning of the modelling artefact is recognized for analysis purposes.

The reference taxonomy per se does not define any relationship between elements of the framework. It defines only rules to classify information derived from

conceptual models. Thus, any inference analysis is derived from the *classification* of the information about conceptual models that were used in a modelling procedure with the essential classifiers expected by the framework itself (*things, processes, locations, people, events, and strategies*). The reference taxonomy and ontological frameworks for ISs are complementary philosophical tools owing to the common presence of an IS architecture definition.

The most relevant contribution of the taxonomy for MDE contexts is the support for *reasoning* (Burge et al., 2008) on models and modelling languages in IS contexts (i.e., the underlying reasoning for the creation and use of related artefacts). Designers of modelling languages have a tool for decision-making to justify the purpose and intentions of the specific modelling efforts considering the current plethora of IS methods. This directly impacts the modelling language harmonization.

3.2.3 The support of the taxonomy for MDE quality issues

Given its formulation, the reference taxonomy works with conceptual models to represent IS phenomena that result in combining *abstractions* with *viewpoints*. The reference taxonomy manages the quality in IS projects with conceptual models, by the accurate depiction of the relevant results from the classification act, the explicit treatment of the semantics, and the explicit consideration of transformations. The main goal of this conceptual tool is to relate conceptual things with representations on computers (Sowa and Zachman, 1992). The taxonomic framework is a result of the abstractions; in this way, it facilitates any reasoning over the MDA architecture, which promotes the software development by utilizing abstract hierarchical models.

The taxonomy itself does not propose any special procedure or methodology to evaluate models. However, owing to this modelling support, quality issues of current models can be addressed by analytics procedures aligned to the classification itself. Examples of quality issues that can be addressed by the framework are the following:

Separation of concerns: the taxonomy promotes the selection of subsets in which decisions are applied. Instead of taming the complexity using a global model of all of the cells, the taxonomy suggests that decisions must be made w.r.t. the scope of each cell.

Communication and suitability: the foundation of the taxonomy is the existence of a set of additives and complementary architectural representations (AR) for ISs. This is a critical issue: in the *MDA foundation model document* (Object and Reference Model Subcommittee of the Architecture Board, 2010), the metaclass *model* is associated explicitly with the term *architecture description* defined in the ISO 42010 standard (before the ISO 1471 standard). Thus, a reasoning mechanism is needed for discernment regarding the selection of a specific AR for an abstraction-viewpoint combination, considering the variety of notations, methodologies, and languages. This discernment process involves the professional communication among the stakeholders that participate in a modelling effort for an IS project. The framework gives the freedom to use any approach or rationale for this analysis.

Model integration capabilities: the co-existence of several modelling initiatives must be evaluated to develop an IS in a consistent and optimized way. It applies the analysis to the capabilities offered by modelling languages to address goals related to the cells in the taxonomy. This analysis of all modelling languages involved in an IS development project improves modelling efforts, avoiding duplication of modelling approaches (i.e., using multiple models in different modelling languages to represent the same information), identifying complementary goals according to specific requirements of the modelling approaches, and identifying IS concerns that are not covered by the employed languages.

Models transformations: one of the most important challenges formulated by the taxonomy is to ensure that model transformations take place as a direct consequence of the addition of information in the interaction of abstractions and viewpoints. The *model mapping* feature (mentioned in the MDA original guide) occurs in the progressive changes on models that cross from higher to lower abstraction levels. Information from the computer-independent platform (CIM) level are enriched with constraints associated with lower levels. Thus, this leads to sufficient information to facilitate its implementation in a technical environment. In a similar way, the transformation of information in two different columns must be justified to support the criteria of the designer of the language to derive models from different essential properties or viewpoints (e.g., *time-location*, *data-process*). An explicit rationale about *why* and *how* information from a column can derive (or generate, or support) information for another column is critical in the features of modelling languages. The *traceability* evidence can be derived from analytics over the information classified by the reference taxonomy.

The taxonomic structure gives a useful and valuable set of information required to model any phenomena inside the scope of an IS (Smith, 2013). The classification rules also guarantee consistency in any IS modelling activity.

3.3 The MMQEF method

MMQEF (*Multiple Modelling language Quality Evaluation Framework*) is a method to evaluate the quality of modelling languages and models using a reference taxonomy for Information Systems (Section 3.2.2). Following the template for documentation of components for methods proposed in (Sandkuhl and Stirna; Goldkuhl et al., 1998), in this section we show the main considerations of the MMQEF method. We also address some of the most common questions regarding the use of the reference taxonomy in our method.

3.3.1 Purposes and preconditions

The main purpose of our method is to evaluate the quality of modelling languages used in MDE scenarios for IS development (i.e., the capability of any model artefact and the user language to represent an IS concern in the most appropriate way). Quality is the degree to which a model and/or modelling language has specific attributes to support essential features of IS and their implicit relations.

Our evaluation method supports semantic inferences and reasoning derived from the use of modelling languages in an IS construction process. The classification mechanism is used as a conceptual tool for determining the degree of support that a modelling language offers to represent a specific concern of the IS reference architecture (this is the reference taxonomy).

This method should be used in a model-driven project where:

- There is one or more modelling language(s) to support the concerns and viewpoints associated with the IS.
- The IS project integrally covers the MDA levels (CIM-PIM-PSM), and the transformations and mappings must be supported by a rationale.
- There is interest to know the real support of a modelling language when it will be applied to manage any IS concern.

The precondition for this method is to know the use of its bidimensional structure and its associated rules. The taxonomic structure of the reference taxonomy is compliant with the MDA levels (Frankel et al., 2003). This previous knowledge is important to apply and support classification procedures and decisions. The levels of the MDA specification (called abstraction levels) must be supported by specific modelling proposals in accordance with their nature and intention. In this way, a classification task could fail if the subjective criteria of the analyst are not aligned with the specific constraints of each taxonomy level.

3.3.2 Method components

The main component of the method is the *reference taxonomy*, which has a structure, a set of rules, and operations over the elements under classification. The structure offers an approach to explicitly manage abstraction levels jointly with classifiers that derive viewpoints. Rules of the taxonomic structure define the semantics associated with the act of modelling itself, contrasting the elements for classification with essential features that are associated with the IS.

The taxonomy reference defines seven rules. These rules are applied as originally formulated in (Sowa and Zachman, 1992). The classification derived from this taxonomy requires the explicit consideration of the *technical issues implementation* of model artefacts under consideration. It is left to the discretion of the analyst/designer role to decide whether to consider the technical aspects. However, there is an explicit feature of the computational implementation that should sometimes be considered to demonstrate the feasibility of the modelling effort from a computational perspective.

We take the taxonomy and apply its specific procedures to contrast specific mechanisms to manage information that is offered by the modelling languages against the essential elements expected in the taxonomy by default.

The other required component is the information extracted from the modelling languages itself or information from models. This required information pertains to the real use of models/modelling languages (expressed in the elements with which the user of languages interacts directly - e.g., concrete syntax and

notations) and the semantics offered by the artefact under analysis. For this case, this type of semantic information is contained in MOF-based metamodels.

3.3.3 Cooperation principles

Structure

The evaluation method requires an organizational structure in which there is a global vision about the IS scope and goals. These structures have levels of architecture descriptions, crossing enterprise architecture, business architecture, and software architecture. These architectures are required by their implicit use, the presence of conceptual models, and the management of views/viewpoints that derive languages and methodologies to cover IS concerns.

Roles

Two roles are required for this evaluation method. The *modelling language analyst/designer* is responsible for proposing and/or conceiving some modelling language(s) to manage specific concerns in an IS project. Generally, this role could be assumed by architects, domain experts, or method engineers with a global vision level of the overall project. A *modelling language final user* is identified also; this role refers to the stakeholders or domain experts who will use the language based on their intentions and interests in the IS.

3.3.4 The main quality evaluation method components

To represent the main components and procedures for our method, Fig. 3.2 summarizes the method showing the inputs, outputs, and roles involved. The main inputs of the MMQEF method are the reference taxonomy, artefacts describing and clarifying the modelling language (i.e., language specifications such as metamodels, modelling guidelines, or even actual models that serve as examples), and the previous knowledge of the IS domain and the language itself. The output is the classification of modelling languages according to the organization of their information over the taxonomic structure. Inferred reasoning results from the classification itself.

The components of the method use the prior information about the domain (or knowledge of modelling tasks), and the information derived from the pre-conceptions from the participants in the modelling act (Bjeković et al., 2014). Those inputs are contrasted considering the information expected for the cells of the taxonomy. This means that the classification is made by the contrast of information, and the reference taxonomy defines the principles that underlie the classification from an IS perspective.

Associated procedures of the method that are more specific are depicted using the *process-delivery diagram* (PDD) approach formulated in (Brinkkemper et al., 1999; Weerd and Brinkkemper, 2009). Expanding the method depicted in Fig. 3.2, we find an *Activity Diagram* with five main blocks associated with the evaluation procedure (Fig. 3.3).

The activity diagram depicted in Fig. 3.3 is for readability purposes. Each block represents one specific evaluation procedure supported by the reference

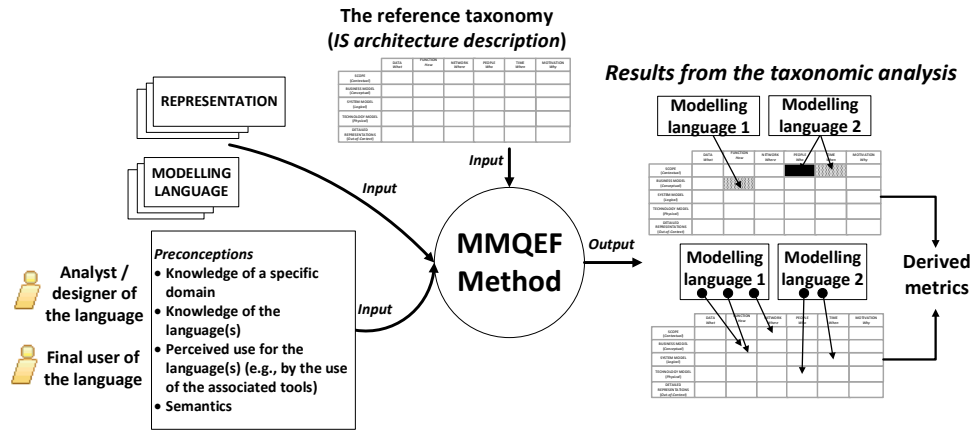


FIGURE 3.2: Overview of the MMQEF method.

taxonomy and the specific intention of the evaluation. A first set of activities is related to the *organization* that the language must have regarding the reference taxonomy, which in turn acts as a reference architecture for IS domains. Thus, this block checks for the architectural structure as an essential property provided by the modelling language to integrally manage the concerns of an IS.

The second block of activities checks for the support provided for managing the incremental evolution of the information under modelling. In this way, the classification of model elements and derived information generate a *navigation model* over the bidimensional structure of the taxonomy, where the traces (or information) are identified that support the progressive evolution of the modelled information.

The third block of activities in Fig. 3.3 verifies the capacities for the specification of *transformations* that the language should possess. It ensures that transformations can be conceived and occur from a semantic reasoning using the essential IS concerns of the classification, independent of constraints and rules imposed by languages and frameworks for model transformations.

A set of activities is considered for checking the *integration capacities* provided by the modelling language. The taxonomy serves to identify the IS concerns covered/not covered by the modelling language. For the uncovered concerns, activities verify whether the modelling language provides any mechanism for modelling these concerns by using one or more additional languages, and thus undertakes a full modelling effort that considers all relevant concerns for a specific IS project.

Finally, based on the individual classifications obtained for the modelling languages under evaluation, the *suitability* of modelling languages is analyzed if they share taxonomic concerns (cells) in their evaluation over the taxonomy. Individual results of the classification support the decisions about the most appropriate language for modelling a set of IS concerns.

With the activities related above, MMQEF gives a methodological orientation for deducing the quality evidence of modelling languages (from their properties) and modelling elements (from their use in an IS context); this evidence is

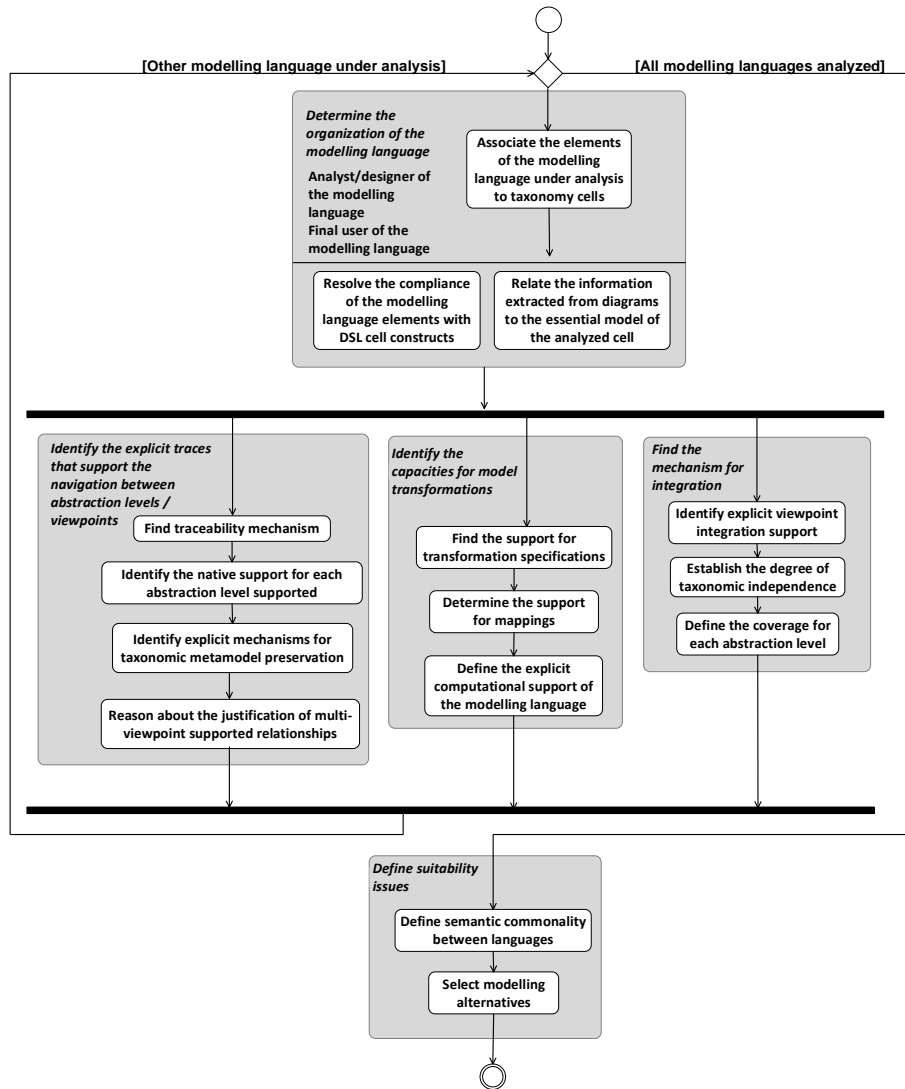


FIGURE 3.3: The main blocks of activities of the MMQEF method in the PDD convention.

derived from a classification procedure, which is the initial step of the method. MMQEF also provides a technical environment that enables inference of semantic deductions over the languages and elements under taxonomic analysis. Appendix B more thoroughly describes the activities and concepts associated with the MMQEF method following the PDD specification.

3.3.5 MDE quality analytics derived

Some quality analytics procedures over modelling languages and model elements can be inferred from the considerations presented above. These start with the

classification of modelling elements and artefacts, either in the form of information from modelling language features or modelling language instances (representations). The classification pertains to the goal of the modelling artefacts with the scope of each abstraction/viewpoint combination.

Inference analysis derived from the Zachman taxonomy

Ontologies are supported by taxonomic structures (Guarino and Welty, 2000). Thus, inference reasoning is a result of a previous classification activity. The reference taxonomy itself is not an ontology (Malik, 2009), but this framework has the taxonomic structure required to promote inference reasoning for the use of models and modelling languages in an IS development process.

In analytic procedures on model artefacts, data semantics come from two main sources: the core concepts of languages (metamodel) and the information derived from model representations. The first relates core language concepts with the essential modelling dimensions expected in each abstraction/column combination. This action allows deduction of the IS concerns for which the modelling language was formulated. This is an analytics procedure from the language engineer².

Moreover, the second is a classification from a language user perspective, in which the *representations* become the main artefact to manage the interaction between the stakeholder and the IS. The classifications of the representations pertain to the purpose or goal perceived by the language users and the information that it contains. Inferences about it helps the language engineer identify quality issues such as suitability, i.e., the harmonization of modelling initiatives for the design of languages without waiting until its implementation to find other modelling initiatives that address the same IS concern(s). This analysis implies that all stakeholders involved in the development of an IS must be identified before the formulation of models.

Any analytic procedure with representations requires information beyond the particular interpretation of the users of languages in accordance with their particular interpretation of the modelling elements. If diagrams are taken as *interaction means*, information on data semantics for analytics could be derived from models used in *model-based graphical user interfaces* tasks (Luyten et al., 2004)(Molina et al., 2012) (e.g., *user models, navigation models, presentation models, dialog models, mental models*, etc.). Thus, the *diagram* transcends from the *big picture* resource to an *information unit* with data about the expected/real intentional use. This is similar to the *pragmatic* of diagrams (Gurr, 2002) under the model-driven way, in which it is important to obtain additional information that helps in making reasoning inferences.

This idea differs radically from the current approaches for managing diagrams in model-driven environments, such as the OMG Diagram Definition (DD) specification (OMG, 2014a). The DD proposal focuses on defining a basis for modelling and interchanging graphical notations that have diagrams with node and arc styles. It establishes the graphical information of one diagram for which language users have control (Diagram Interchange - DI - i.e., position of nodes and

²According to the roles defined in (Kleppe, 2008)

line routing points) and how the shapes of the graphical language must be instantiated in a mapping from abstract syntax models and DI (Diagram Graphics - DG). However, although the spirit of DD is the formulation of a framework for other modelling language specifications to define their diagrams, this formulation pertains to graphical concerns exclusively. Mechanisms for reasoning over diagrams with an inference analysis focus are not considered.

Derived supports for MDE quality evaluation

Support for assessing the quality of models and modelling languages is derived from the classification of the reference taxonomy. This is particularly relevant for model-driven practitioners because the evaluation is made directly on the modelling act itself rather than some adapted software quality practice (Moody, 2005) as is commonly reported in the MDE literature. The evaluation support is as follows:

A rationale about the organizational impact of the model-driven initiative(s): This refers to the degree of complete support that a model-driven initiative offers for an IS concern framed into an organizational context. Its analysis involves procedures related to the checking of the alignment of model-driven initiative purposes with the organizational goals, the degree of effective support of model-driven tools at computational-independent levels, the degree of understanding and use of models by organizational users, etc.

The native IS architecture over organizational constraints powered by the taxonomy facilitates the reasoning about the applicability of a model-driven approach in any organizational environment that uses IS. However, this analysis is not generic, and the specific business features in which models will be applied (e.g., business capabilities) must be considered to guarantee the compliance between the business concerns and model-driven approaches.

Analysis and design of transformations: The taxonomic analysis provides support for identifying the evolution of the information and the semantic relations generated. Thus, analysts of the language can propose decisions and considerations about model transformations before their implementation on a model transformation language.

In the case of model mappings (evolution from the PIM to the PSM level within a specific viewpoint), the effort will focus on the preservation and evolution of the high-level concepts until their preliminary technical implementation in accordance with the generic model of each viewpoint. Thus, the relations between the conceptual information and constraints imposed by the lower-level abstractions and perspectives are evident.

For transformation between models of different columns and the same row, a semantic argument is necessary that supports the generation or derivation of models in different viewpoints and guarantees the complete depiction of the IS reality (Sowa and Zachman, 1992) from the row under analysis.

The taxonomic framework recognizes the *dependency* relation between cells in a given row, although these have relevant independence in their definition. These

are relevant for the design of model transformations, especially when they are considered model co-evolution features in model management platforms. Dependency relationships generate conceptual models of transformations, posteriorly implemented on a modelling language transformation.

In scenarios of transformation, their underlying rationale is the *traceability information* that semantically manages the evolution of models. This information must be explicitly available to design and implement model transformations. This feature enables formulation of a process for consideration from the conception until the implementation and deployment of model transformations (in an analogous way to a software development process), without an explicit dependence only on the model transformation languages.

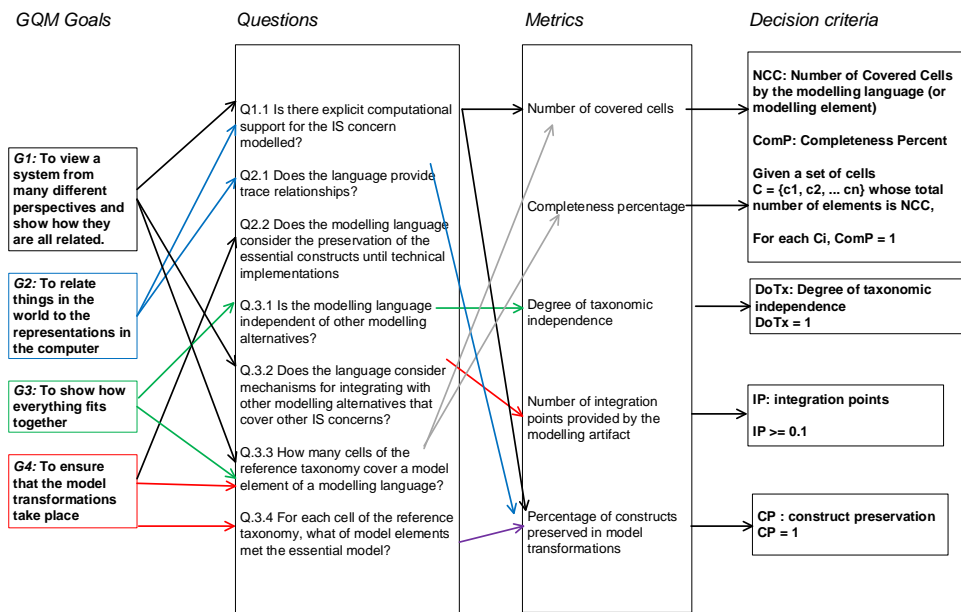


FIGURE 3.4: Metrics for analytic reasoning derived using the GQM approach

MDE metrics: Following the goal-question-metric approach (Basili et al., 1994)(Fig. 3.4), some metrics are formulated for supporting the quality evaluation of modelling languages with the reference taxonomy. For this, the goals were extracted directly from one of the original specifications of the taxonomic framework (Sowa and Zachman, 1992). These relate to the development and deployment of models over IS concerns, so these contribute to improving the applicability of a model-driven initiative. Questions define the scope of a modelling proposal, identifying other modelling proposals with similar purposes and anticipating consequences of the model initiative application. The derived metrics are as follows:

Number of covered cells (NCC): This refers to the total cells covered by the model element/modelling language. If covered cells belong to the same column,

the modelling artefact has an explicit trace intention. However, if cells are from two or more columns, the modelling artefact must provide a semantic mechanism to derive relations between the viewpoints under consideration according to the goals of the modelling act.

Completeness percentage (ComP): For each cell covered by a model artefact, the purpose of the artefact is compared with the scope of the cell to determine its degree of compliance. This is useful to find modelling proposals in which the intention is not originally aligned with the scope, but the modeller/modelling language designer decides to use it for any reason. This also reveals *profiling trends* for modelling elements. This metric is as follows: $\forall cell \in NCC, ComP = 1$.

Degree of taxonomic independence (DoTx): This refers to the degree to which a classification obtained for a modelling language (or its associated modelling elements) is *distinctive*, which means that it is unique and clearly differs from classifications for other languages. The covered cells for the modelling language or element allow the main characteristics and scope of the classification to be deduced. If two or more languages are detected in an IS project that share common cells, the independence of the language for managing a specific concern is questioned, so a suitability reason is required to choose the most appropriate alternative for modelling these concerns. This decision could use other metrics, such as the *Completeness percentage - ComP*.

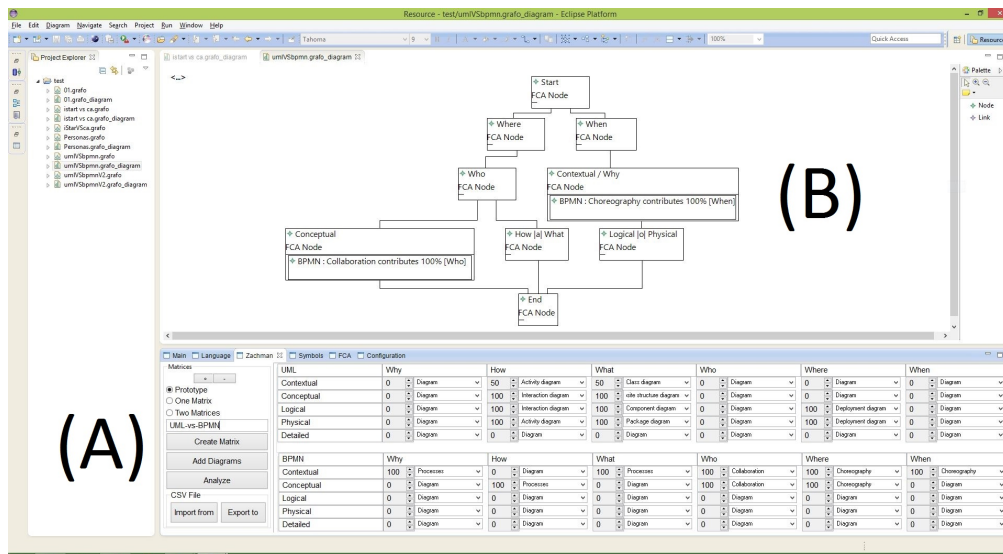


FIGURE 3.5: The EMAT tool for the MMQEF method.

Number of integration points provided by the modelling artefact (IP): an analysis over the integration capabilities of modelling artefacts must be performed to find the semantic relations that support the coexistence of modelling efforts. This feature contributes to the design of model transformations for purposes of managing multiple instances of the IS model.

There is no universal prescription for the coverage that a modelling language should possess for the different abstractions/viewpoints of an IS. However, if a modelling language defines integration points, this feature is an explicit recognition for previous modelling initiatives that could better model some IS concerns, so the modelling language under analysis can focus on a specific IS concern and specialize its particular modelling approach.

Another direct consequence of this analysis is *suitability decisions*. This contrasts the degree of completeness of each modelling initiative w.r.t. the covered scope. This type of consensus could optimize the application of modelling proposals. From this, the suitability percentage of specific modelling proposals is derived.

Percentage of constructs preservation in model transformations (CP): This relates to the degree of preservation of essential concepts during a high-low mapping until their technical implementation. The main goal of this metric is to avoid the loss of information at the crossroads between different abstraction levels / viewpoints so that the business concepts evolve until full technical implementation.

3.3.6 Tool support

To support the application of the MMQEF quality evaluation method, we developed EMAT (*Eclipse Modelling Analytics Tool*). It is an Eclipse plugin for the Eclipse Modelling Framework project (one of the main model-driven technical environments). This plugin was developed to operationalize the classification process and its obtained data.

EMAT is formally supported by the *Formal Concept Analysis* approach (Priss, 2006). Fig. 3.5 shows an example of a supported taxonomic analysis with the *classification* as input (Fig. 3.5 - part A), and the semantic relations as output (Fig. 3.5 - part B). Specific features and concerns regarding this tool will be described in further works. For now, our main intention is to relate the technological support provided for the MMQEF quality evaluation method, which is a critical issue because conventional modelling quality methods are difficult to implement (operationalize) in specific tools owing to their associated abstraction and theoretical background.

3.3.7 An example application of the MMQEF method

To demonstrate the applicability of our quality evaluation method, we use it to analyze the modelling languages used in a specific capability-driven development (CDD) scenario of the European project named *Capability as a Service in digital enterprises* (CaaS) (Stirna, 2013). The CaaS project proposes the CDD approach for digital enterprises to exploit the notion of *capability* as a means of design both for services and with services. Thus, CaaS elaborates an integrated approach consisting of methods, tools, and reusable best practices that allow companies to adjust their business services and information technology systems according to the changes in business context and technologies (Egido et al., 2014; Bērziša et al., 2015).

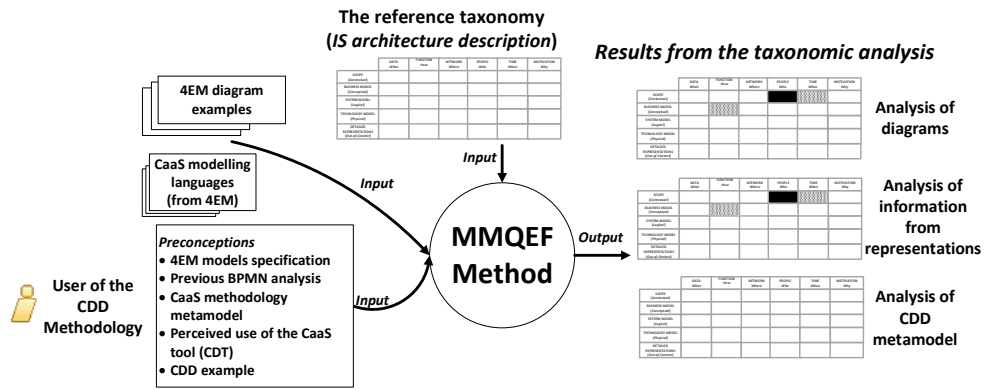


FIGURE 3.6: Summary of the application of the MMQEF method to the CDD methodology of the CaaS project.

The CDD methodology of the CaaS project uses modelling languages to represent enterprise designs, context models, and patterns. These are based on the EKD (Bubenko and Stirna, 2001) and 4EM (Sandkuhl et al., 2014a) approaches to Enterprise Modelling. Thus, CaaS uses *goal models*, *process models*³, *business rule models*, *concept models*, *actor/resource models*, and *technical component and requirement models*. The methodology also *considers* models for handling variability concerns. Those are models at the enterprise level, mitigating current limitations of specific platforms by the application of the CaaS approach. References (Quevedo et al., 2015; España et al., 2014) report an example of an application over SOA platforms.

To start the taxonomic analysis, we use the available information of the modelling languages and their use; this comes from the specification of the CDD methodology (which specifies a metamodel for it), the explanation about each involved model (taken from the 4EM specification (Sandkuhl et al., 2014a)), and examples of the CDD methodology as presented in (Centre, 2015). In addition, taking advantage of the particular method for modelling business processes chosen in the CDD methodology³, a previous analysis (reported in (Zhao et al., 2012)) that uses the reference architecture for analyzing the BPMN notation was also used. Fig. 3.6 gives a brief overview about the application of the MMQEF method over the CDD methodology.

One of the most classical reported ways to begin the taxonomic analysis is the association between diagrams and cells; this yields the classification shown in Fig. 3.7. However, to perform the activities related in Table B.1, the information contained in the example diagrams of the 4EM is also considered. This information enables precise compliance of the element of the involved languages with the cells of the taxonomy. This compliance is shown in Figure 3.8, in which the first key quality issues appear. For example, there is an overlap in the *scope-planner/why* cell between the *goal* concept associated with the *goal model* and the *technical components and requirements model* diagrams.

³In the CaaS project, processes are modelled using BPMN instead of the original proposal of 4EM (Sandkuhl et al., 2014a).

MDA Abstraction levels		WHAT (Things)	HOW (processes)	WHERE (location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	Scope (Planner)	Context Modelling					Technical Components and Requirements Model (TCRM) Goals Model(GM) Context Modelling
	Enterprise Model (Owner)	Concepts Model (CM) Context Modelling Adjustment Model	Business Processes Model (BPM) Variability Modelling - Context Modelling		Actors and Resources Model (ARM) BPMN	BPMN	Goals Model (GM) Business Rule Model (BRM) Technical Components and Requirements Model (TCRM) Context Modelling Adjustment Model
PIM	System Model (Designer)	Technical Components and Requirements Model (TCRM)					
PSM	Technology Model (Builder)						
Physical	Detailed Representation (Programmer)						
	Functioning (User)						

FIGURE 3.7: Taxonomic analysis of the diagrams of the CDD methodology.

A key issue detected from the obtained analysis of Figure 3.8 is the fulfilment of the *enterprise model - owner* row of the reference taxonomy, by the association of the modelling elements belonging to this particular CDD approach to each column of this row. This constitutes a complete model from the *enterprise model- owner* row according to the rules of the taxonomy framework (Sowa and Zachman, 1992). From this, it is possible to show how the CDD approach meets the organizational level regarding the scope of the CaaS project itself, but this row contains other quality issues. The suggested classification in the *enterprise model- owner/where* cell (extracted from ((Zhao et al., 2012)) is quite questionable owing to the semantic considerations of the modelling elements of the BPMN related to this cell (pool and message flows). The proposed use of both concepts is far from the original definition of these concepts in the BPMN specification ((OMG), 2011) (*participants and messages between them, respectively*). Thus, the *enterprise model- owner/where* cell should be empty owing to the lack of modelling support from the CDD approach for the *where* question, which is critical for managing specific implementation issues for an enterprise operational deployment. In addition, another quality issue originating from the previous BPMN classification proposal is in the *enterprise model- owner/who* cell, where the *workflow* concept is located, but it must be in the *enterprise model- owner/how* because this represents a business process. For the same cell, the *Organizational unit* element of the *Actors and resources model* diagram has no information about its physical location.

MDA Abstraction levels		WHAT (Things)	HOW (processes)	WHERE (location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	Scope (Planner)	Context Modelling: Context element.					Goal model diagram: Goal Goal model diagram: Problem Goal model diagram: Cause Goal model diagram: Constrain Technical Components and Requirements Model diagram: IS Goal Technical Components and Requirements Model diagram: IS Problem Context Modelling: Context indicator Context Modelling: Calculation
	Enterprise Model (Owner)	Concept model diagram: concept. Context Modelling: Measurable property. Context Modelling: Context Type. Context Modelling: Variation aspect. Context Modelling: Context Element Range. Context Modelling: Context Set Context Modelling: Variation point. Adjustment: classes IDA event Based Adjustment. Adjustment: classes IDA Scheduled Adjustment	BPMN Context Modelling: Business Process . Context Modelling: Process variant.	BPMN: Pools BPMN: Message Flows	Actors and Resources Model diagram: Individual Actors and Resources Model diagram: Role Actors and Resources Model diagram: Resource Actors and Resources Model diagram: Organizational Unit BPMN: workflow	BPMN: Events BPMN: Gateways	Goal model diagram: Goal Business rules mode diagram: rule Goal model diagram: KPI Technical Components and Requirements Model diagram: IS Goal. Context Modelling: Adjustment. Adjustment: classes KPI Calculation Adjustment: classes IDA calculation
PIM	System Model (Designer)	Technical Components and Requirements Model diagram: IS Requirement Technical Components and Requirements Model diagram: IS Functional Requirements Technical Components and Requirements Model diagram: IS Nonfunctional Requirements					
PSM	Technology Model (Builder)						
Physical	Detailed Representation (Programmer)						
	Functioning (User)						

FIGURE 3.8: Taxonomic analysis of the information extracted from diagrams of the CDD methodology.

Figure 3.8 shows that the CDD methodology provides conceptual support for the PIM abstraction level and no support for the PSM and technical implementation levels. This is a critical issue considering the full operationalization of the results from the methodology over real computational platforms. Although the *NCC* metric (section 3.3.5) demonstrates proper support of the approach for the CIM level (58.33% of CIM cells – 7/12; 83.33% for *enterprise model – owner* row, cells 5/6), no information about the following modelling elements is explicitly provided.

Regarding the activities show in Table B.2, some issues were found regarding the *traceability* and *navigation between abstraction levels*. In the *enterprise model – owner/how* cell, the *actor* modelling element must have explicit relationships with *goals*, *rules* and *processes*, but these are not formally defined; instead these are conditioned to the analyst’s criteria. Another quality issue is in the *system-what* cell, where there are no traceability relationships among the *business processes model*, the *actors and resources model*, and the *concepts model*, despite the explicit relations that they must have by the references proposed in the *requirement expressions*. In addition, *functional requirements* that are associated

with this cell must be clearly defined with reference to the *concepts model*, but these relations are not explicitly defined.

Table B.2 also shows that there is evidence of preservation of a construct between the *identify the explicit traces that support the navigation between abstraction levels / viewpoints* activity and the *goal* concept of the *goal model*, which is in the *CIM scope – planner / why* and *CIM enterprise model – owner / why* cells. This is a result of the proposed operationalization for the *goal* concept by *business rules* or business processes. The justification of the *multiple viewpoint* is not provided by the methodology; however, it is derived from the use of the models for covering the enterprise concerns.

MDA Abstraction levels		WHAT (Things)	HOW (processes)	WHERE (location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	(Scope) Planner				Resources		Context Indicator
	(Enterprise Model) Owner	Context set, Context Situation, Context element Range, context element value, KPI value, Context element type. Measurable property, Adjustment constant, capability, Context Element, Variation Aspect. Variation Point. capability delivery variation point. Process Variant Variation Point	Process, Process Variant, Capability Delivery Pattern (alias Pattern).				Goal, KPIs, Indicator Calculation, Context calculations, Adjustments, KPI calculations,
PIM	(System Model) Designer					ScheduledAdjustment Event Based Adjustment	
PSM	(Technology Model) Builder						
Physical	(Detailed Representation) Programmer						Capability Adjustment
	Functioning (User)						

FIGURE 3.9: Taxonomic analysis of the metamodel of the CDD methodology.

For the *Identify the capacities for model transformations* activities block of the MMQEF method (Table B.4), the capacities for transformation, mapping, and computational support of models are not covered for the CDD methodology, so these are delegated by the particular implementations of this CDD approach. For the *integration mechanisms* (Table B.5), the taxonomic analysis demonstrates that there is no full taxonomic independence between the *technical components and requirements model diagram* and the *goal diagram* owing to the closeness of the *goal* concept of both initiatives. This produces an overlap of the modelling tasks at the CIM level. The lack of *integration points* for each modelling language involved in the CDD methodology generates cases in which concepts from the two models are presented in the same diagram without explicit support beyond the diagrammatic integration purposes (e.g., *goals* of

the *goal model* and *rules* from the *business rules model* in an *enterprise model-owner/why* cell).

The classification reflects suitability evidence (activities shown in Table B.6). Most of the covered cells (77.77%) have at least two modelling languages that support them. In most cases, the analysis reflects that these languages are complementary regarding the semantic purpose (scope) of the cells. However, in the *CIM scope – planner / why* cell, a suitability decision is required to choose the most appropriate language that models the *goal* and *problem* concepts; in this cell two different alternatives were detected for modelling both concepts. However, the CDD methodology does not indicate which language is the most appropriate modelling language for covering both concepts.

In Fig. 3.9, the taxonomic analysis shows how most metaconcepts of the CDD methodology resolve concerns related to the data of the *context* and management of the *goals* at the *enterprise model- owner* level. For this reason, the taxonomic analysis differs from the previous analysis depicted in Fig. 3.7 and 3.8. In addition, support for managing the variability of business services is detected at the same level. Normally this explicit consideration is part of the *context model*; however, its application is on points of processes. Therefore, the resulting classification is associated with the *enterprise model – owner / how* cell for the semantic purposes of specifying variation scenarios.

Finally, Fig. 3.10 and 3.11 depict the obtained results for the taxonomic analysis in the EMAT tool (3.3.6). Four conventions are required for understanding the FCA outputs of the EMAT tool; these are as follows:

- */a/* an *abstraction level* is associated only with the *viewpoint*.
- */o/* a *viewpoint* is associated only with an *abstraction level*.
- *| a |* Two *viewpoints* come together when they are related to two *abstraction levels*.
- *| o |* Two *abstraction levels* come together when they are related to the same *viewpoints*.

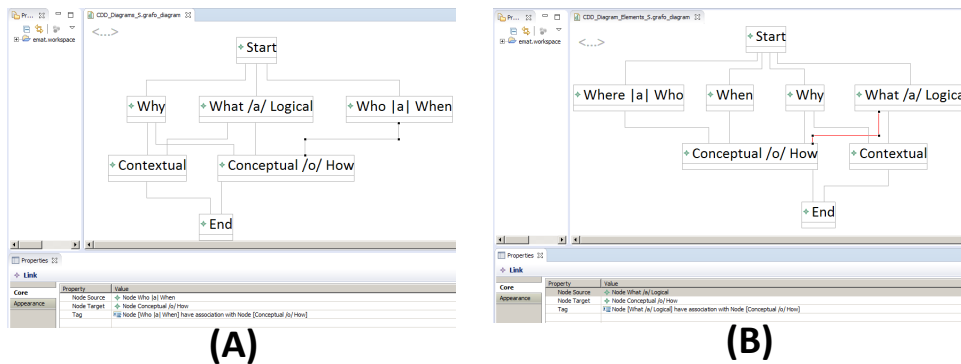


FIGURE 3.10: The FCA lattice obtained from the classification shown in Fig. 3.7 and 3.8.

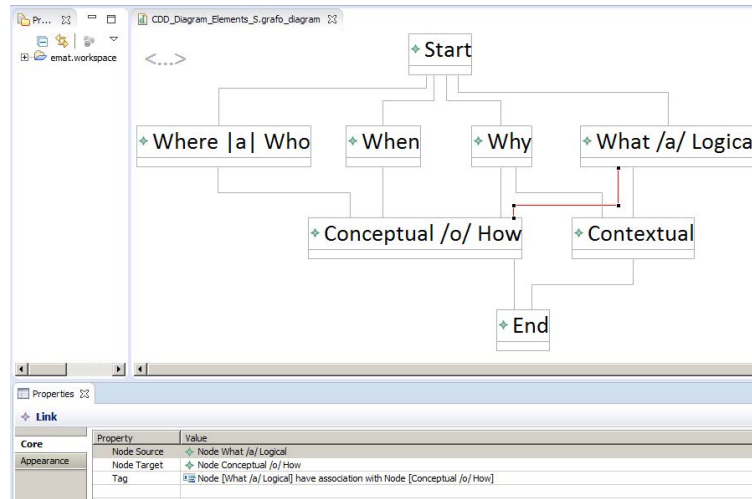


FIGURE 3.11: The FCA lattice obtained from the classification shown in Fig. 3.9.

Fig. 3.10 - part *A* presents the FCA lattice (or connected graph) for the classification previously shown in Fig. 3.7. The $\{What/a/Logical\}$ node depicts the support that the *what* viewpoint offers for the *PIM - system model* level (or *logical* level in the original description of the taxonomy); this is because the *what* column is the only one that provides a diagram for these levels.

The $\{Why\}$ node has relative independence because it is not sufficiently related to other viewpoints, which means that the diagrams that meet its associated cells (for the involved classifiers) are unique for this viewpoint. $\{Why\}$ has a shared attribute only with the $\{What/a/Logical\}$ viewpoint by the *Technical Components and Requirements Model (TCRM)* attribute. Thus, the $\{Contextual\}$ node (or the *CIM - Scope* level) is represented as an independent node with relationships with the $\{What/a/Logical\}$ and $\{Why\}$ nodes because these are the only ones that offer support for modelling it. The $\{Who | a | When\}$ node is generated for the tool to indicate that these viewpoints are related in the same abstraction level by sharing a common diagram. This node has semantic closeness with the $\{Conceptual/o/How\}$ node (*CIM - Enterprise model* level) because it covers the same abstraction level but without sharing a specific diagram between both nodes.

Fig. 3.10 - part *B* presents the FCA lattice resulting from the taxonomic analysis over the modelling elements of the CDD methodology shown in Figure 3.8. Two direct associations were obtained in the $\{Conceptual/o/How\}$ and $\{What/a/Logical\}$ nodes because the modelling elements of these cells contain modelling elements exclusively, which means that these elements are not associated with other rows.

The $\{Who | a | When\}$ node reflects an association at a conceptual level (*CIM - Enterprise model* level) between the *where* and *who* viewpoints because these both have a common abstraction level. The *why* and *when* nodes do not have a direct association with other abstraction levels, so these were processed as independent nodes.

The most relevant conclusion derived from the lattices is the full management of the *conceptual* row (the *CIM - Enterprise model* level) supported by the modelling approaches under analysis; this can be identified by the distance and number of input links to the *{Conceptual /o/ How}* node. Nodes with a combination of *abstraction levels* and *viewpoints* depict the existence of common modelling elements that derive a semantic closeness; depending on the semantic distance (the closeness between concepts of the modelling language under analysis) the FCA algorithm groups nodes or associates them by an explicit link. Fig. 3.11 also shows this support, but in this case, the nodes differ from those presented in Fig. 3.10 owing to the obtained classification of the metaelements of the CDD methodology.

3.4 Preliminary trade-off analysis of the MMQEF method

3.4.1 Main implications of the MMQEF method

The most representative drawback of this evaluation method is its high coupling with the reference taxonomy. Even if the taxonomy is justified as a holistic description of the essential elements of an IS, its use is tied to the subjective criteria of the analyst to meet with the expected conceptual elements of the taxonomy cells. Because of its neutral conception, the taxonomy does not have default modelling languages for its units. Instead, previous reports are identified that contain examples that attempt to clarify the expected models for each taxonomy unit. The MDA specification also does not prescribe default modelling languages for the MDA levels (it only promotes its UML support).

However, we propose taking advantage of these subjective criteria to promote reasoning on models, and thus justify the selected choices. The classification of modelling languages (and their associated artefacts) enables the verification of the sufficiency of a modelling approach to conceptually manage a specific IS concern.

Why taxonomy analysis?

Despite the great potential for inferences that ontologies provide to support reasoning on quality, they could influence (or alter) the essential features of modelling languages. In these approaches, modelling constructs must fix the ontological constructs, assuming that they are valid from the perspective of a specific community (but there are a plethora of ontologies). According to (Rosemann and Green, 2000), if a model (and thus a modelling language) is ontologically incomplete, the analyst/designer role will have to augment the model(s) to ensure that the final computerized information system adequately reflects that portion of the real world that is intended to be simulated.

The particularity in the analysis provided by IS ontologies contributes to the conceptual divergences in the model-driven community and the consolidation of isolated IS initiatives with their associated communities. The philosophical sufficiency of the ontologies is not questioned. However, if modeling languages must be evaluated using multiple ontological frameworks, it will probably lead to an overload in the formulation and evaluation of languages because the consistency

of the results of multiple ontological analyses is not guaranteed. In other words, for the same language, we would obtain multiple results for the same language due to the particular conceptions of each ontological framework.

Taxonomic analysis is a practical application of the classification as a fundamental mechanism for organizing knowledge. It addresses problems in conceptual modelling that are related to suitability issues (*which theory of concepts to use*) (Wand et al., 1995) and their derived quality phenomena. Classification provides the basis that helps to determine the *value* in an inference-based procedure (Parsons and Wand, 2008).

Why this taxonomy?

Following the considerations presented in (Parsons and Wand, 2008), the taxonomy is an explicit method for modelling semantics that describes the structure of knowledge (semantic domain) about things in the IS domain. The taxonomy defines a holistic description of an IS in an enterprise context, considering *computationally independent* aspects for their implementation in specific computerized platforms. It has an implicit mapping with the MDA levels (Frankel et al., 2003), so this supports any phenomenon regarding the formulation and usage of models. By default, the taxonomy uses models to cover the essential features of the ISs.

The taxonomy provides an IS architecture description with the essential models required to cover all real concerns involved in an IS project (from organizational to computerized issues). The taxonomy focuses on the information that must be managed by the artefacts provided in modelling languages under consideration.

For other taxonomy proposals for model-driven initiatives (section 3.2.1), the reference taxonomy has a more complete taxonomic scope because, by default, it considers viewpoints (and their resulting views) that are directly associated to business and technical levels where an IS will be deployed. Model operations (e.g., transformations) are results of the interaction of models within IS levels.

Classification is the main purpose of the reference taxonomy independent of its derived commercial uses. The taxonomy defines *classifiers* (i.e., the essential elements for each column that are required to conceive and produce a model from an *abstraction level/viewpoint* combination). Classifiers are set according to the conceptual modelling foundations presented in (Thalheim, 2011, 2012). MDE covers all domains of the IS architecture defined by the reference taxonomy (Brossard et al., 2011).

Is it the only taxonomy? Why not a quality ontology instead?

IS evaluation is quite concerned with taxonomies (Kautz and Nagm). The IS literature commonly reports the use of taxonomies to classify ISs according to their type, development process, deployment over specific computational infrastructures, analysis of internal components, and other categories. However, these activities fall into subjectivity and contextual factors.

Unlike most reported IS taxonomies, the reference taxonomy proposes classification using a set of essential elements that must be considered by an IS. This

means that classification of IS elements are set from an IS viewpoint. Few IS taxonomy proposals use modelling elements for the classification act itself; for example, (Lyytinen, 1987) suggested *language* as a component of a taxonomy to classify methodologies for IS development.

Both ontologies and taxonomies have business value based on their relationship to architectural models that support IS practices (Laware and Kowalkowski, 2005); however, the attempts to equate the reference taxonomy with ontological frameworks for IS fail owing to the difference between the main purposes of the approaches (to classify elements and make inferences respectively).

In turn, the efforts to formulate an *ontology for quality* increases the conceptual divergences regarding quality in model-driven contexts. Analogously to software quality models, previous quality models were proposed, each of which was valid for specific software communities; a consensus was reached when ISO proposed the 9126 standard (subsequently migrated to the current ISO 25000 model). A unique ontological model for quality in MDE cannot satisfy the open MDE challenges owing to the wide divergence regarding the scope of MDE, when an initiative is MDE compliant, and how MDE be aligned with current methods for IS construction.

3.4.2 Feasibility of the classification procedure for quality purposes

Currently, there are no reports about the explicit use of the taxonomy as a model evaluation tool that supports model analytics. However, there are some works that present the support of the taxonomy in modelling classification tasks. For example, (Kingston and Macintosh, 2000) makes suggestions about modelling techniques for a medical domain; these suggestions were made from the classifiers of the taxonomy. The authors also proposed the usage of individual perspectives of the taxonomy as a *user interfaces* for a knowledge distribution system. In the analysis over the Zachman framework reported in (Noran, 2003), the authors suggested a set of modelling languages to populate each cell of the taxonomy structure according to the intention, design, and needs of the specific tasks for each cell. The reference taxonomy was used in (Liao et al., 2015b) to classify the identified models for a specific domain under analysis. The reference taxonomy is often (and commonly) used to justify the scope of specific modelling initiatives regarding the scope of an IS holistic description.

3.4.3 The use of the taxonomic structure itself

One of the most critical challenges for the MMQEF method is the use of the taxonomic structure itself. Although, the taxonomic framework has been commonly reported in the IS literature, its application is conditioned to the subjective criteria of the analyst, according to the identified scope of each cell and the previous knowledge or experiences in modelling.

For this, a preliminary experiment was conducted with a population of 36 participants including final undergraduate students, Masters students and researchers with knowledge in software engineering projects, use of modelling languages, and information systems. Each participant was given a description of the

taxonomic framework with reference examples (extracted from (Frankel et al., 2003; de la Vara et al., 2007)) about how cells are expected to be met. In addition, an empty structure of the taxonomy was included in the survey so that participants filled out this structure with the modelling languages, modelling elements or computational infrastructures that they considered appropriate for the cells.

Table 3.1 presents the results obtained for this experiment, which indicate the percentage of participants who filled each cell with the names of the modelling languages or modelling elements that they knew. Clearly, a significant percentage is reached by the *What/PIM-designer* cell, in which common data model approaches were reported by participants.

TABLE 3.1: Percentage of participants in the experiment who answered the empty taxonomy survey.

MDA		WHAT	HOW	WHERE	WHO	WHEN	WHY
CIM	Planner	63.89%	69.44%	25.00%	30.56%	25.00%	38.89%
	Owner	38.89%	52.78%	13.89%	47.22%	16.67%	25.00%
PIM	Designer	75.00%	52.78%	25.00%	27.78%	13.89%	8.33%
PSM	Builder	55.56%	30.56%	13.89%	22.22%	16.67%	11.11%
	Subcontractor	25.00%	2.78%	5.56%	5.56%	5.56%	2.78%
	Functioning	16.67%	13.89%		2.78%	11.11%	

Based on the results of Table 3.1, an extra review was required to verify the information submitted by the participants for each cell of the taxonomy. Therefore, Table 3.2 presents the percentage of modelling approaches reported by the participants that are not consistent with the scope of the analyzed cells. For example, for the *Where/CIM-owner* cell, one of the five participants indicated the *Zachman* modelling approach; this also occurs in the *When/CIM-owner* and *When/PIM-designer* cells. Other examples of abnormal answers were found for the *Where/PIM-designer* and *Who/CIM-owner* cells, in which some participants reported the *unified process* as a modelling language.

TABLE 3.2: Percentage of answers with no relation to the scope of the taxonomic cells.

MDA		WHAT	HOW	WHERE	WHO	WHEN	WHY
CIM	Planner	26.09%	20.00%	22.22%	18.18%	11.11%	28.57%
	Owner	7.14%	10.53%	20.00%	17.65%	16.67%	22.22%
PIM	Designer	3.70%	5.26%	11.11%		20.00%	33.33%
PSM	Builder	20.00%	18.18%	40.00%	37.50%	33.33%	50.00%
	Subcontractor	33.33%		50.00%			
	Functioning	33.33%				25.00%	

Finally, we looked for the use of common modelling languages to populate the cells of the taxonomic framework. Tables 3.3 to 3.5 depict the obtained percentages for UML, BPMN, and other modelling alternatives, respectively.

The results obtained from this survey reflect a lack of knowledge (and consensus) about modelling artefacts for specific concerns of an IS project. Most

TABLE 3.3: Percentage of answers in which UML was detected.

MDA		WHAT	HOW	WHERE	WHO	WHEN	WHY
CIM	Planner	65.22%	60.00%	33.33%	54.55%	55.56%	57.14%
	Owner	85.71%	78.95%	80.00%	64.71%	33.33%	22.22%
PIM	Designer	96.30%	89.47%	88.89%	100.00%	60.00%	33.33%
PSM	Builder	85.00%	90.91%	60.00%	50.00%	66.67%	25.00%
	Subcontractor	77.78%	100.00%	50.00%	100.00%	50.00%	
	Functioning	66.67%	60.00%		100.00%		

TABLE 3.4: Percentage of answers in which BPMN was detected.

MDA		WHAT	HOW	WHERE	WHO	WHEN	WHY
CIM	Planner	8.70%	8.00%				
	Owner	7,14%	21.05%	20.00%	11.76%		11.11%
PIM	Designer				10.00%	20.00%	33.33%
PSM	Builder						
	Subcontractor						
	Functioning		20.00%				

TABLE 3.5: Percentage of answers in which other modelling alternatives were detected.

MDA		WHAT	HOW	WHERE	WHO	WHEN	WHY
CIM	Planner	8.70%	20.00%	44.44%	27.27%	33.33%	14.29%
	Owner	7.14%	5.26%		29.41%	66.67%	44.44%
PIM	Designer		5.26%	11.11%	20.00%	60.00%	33.33%
PSM	Builder	5.00%			12.50%		25.00%
	Subcontractor					50.00%	100.00%
	Functioning					75.00%	

of the percentages depicted in Table 3.1 are less than 50, which reflects the low number of participants that propose some modelling approach for specific IS concerns. In addition, subjective criteria are evident, reflected in answers featuring proposed approaches for specific cells based on the experience of the participant instead of the scope of the involved initiative.

3.4.4 MMQEF and other quality frameworks for MDE

The MMQEF framework takes advantage of taxonomic analysis to identify and evaluate quality issues for modelling elements, modelling languages, and models that define languages. The key premise is the use of an Information System (IS) reference architecture that proposes *classification* as the strategy for understanding all phenomena involved in an IS project. By default, *models* are the conceptual supports for these phenomena.

In this sense, MMQEF does not attempt to replace or create any previous method for evaluating quality in MDE. Conversely, MMQEF is a complementary approach that preserves and promotes previous quality frameworks because the classification act encourages quality considerations derived from sources such as semiotics and ontologies.

Current, MDE quality frameworks and guidelines could be too abstract to be applicable for model-driven practitioners (Mendling et al., 2010), most of whom are influenced by typical software development issues. MMQEF gives an operational framework, based on a conceptual, methodological, and technological support, for performing procedures of quality evaluation that are aligned with previous quality principles, for example, those prescribed in the SEQUAL framework (Krogstie, 2012b), one of the most relevant quality initiatives for model-based and model-driven communities.

SEQUAL and MMQEF are complementary owing to their constructivist vision of modelling as a consequence of the interaction among the IS domain, the modelling languages, and the stakeholders involved with their associated knowledge. A classification process discovers the *true nature* of things, describing relationships among objects that should generate hypotheses (Sokal, 1974). Through the classification and the proposed analysis procedure, the MMQEF method implicitly considers the main components of SEQUAL and their quality types (Krogstie, 2012b), except for the *empirical quality* type, which can be better supported by works on concrete syntax and visual notation design such as (Green and Petre, 1996; Moody, 2009). MMQEF also considers the categories of guidelines for the quality of modelling languages and the *pragmatic*, *social*, and *deontic* considerations for metamodels (Krogstie, 2012c).

With regard to other model quality initiatives, MMQEF allows the *6C* quality goals defined in (Mohagheghi et al., 2009a) to be identified and rationalized. Its methodological framework and the analytic support derived from it (discussed in Section 3.3.5) provide a conceptual infrastructure (from an IS reference architecture) for making precise argumentations directly from the modelling act over an IS.

Most current quality challenges for the model-driven paradigm come from previous IS frameworks such as FRISCO (Falkenberg et al., 1996) (December

1996). It defines key aspects for the model-driven approach: the use of models (conceptual modelling), the definition of information systems, the use of information system denotations by representations (models), the definition of computerized information system, and the *abstraction level zero* by the presence of processors (physical level).

FRISCO promotes the *communicative* factor as a key consequence of the use of models. FRISCO also suggests the need to harmonize modelling languages, presenting the *suitability* and *communicational* aspects for the modelling languages. Communication among stakeholders is critical for harmonization purposes because it allows them to discuss relevant quality issues from different views (Shekhovtsov et al., 2014). Therefore, FRISCO suggests relevant features for modelling languages (*expressiveness*, *arbitrariness*, and *suitability*).

These types of FRISCO challenges produce new concerns for model-driven practitioners. For example, suitability requires the usage of a variety of modelling languages and communication requires that these languages must be compatible and harmonized. Suitability concludes that a diversity of modelling languages is required, so the differences among modelling languages (due to this diversity) are unjustified.

MMQEF gives a methodological and technological framework that address these FRISCO challenges on modelling languages. The classification analysis enables identification of the purpose and use of the modelling languages involved in an IS project regarding the scope of cells. MMEQF also considers an explicit set of activities to rationalize the suitability of the languages, where decisions about this and harmonization are based on the coverage and completeness demonstrated in the reference taxonomy.

The presence of computational levels in the taxonomic analysis (abstraction level zero of FRISCO) binds the explicit computational support required for the modelling initiatives under evaluation; this ensures the coverage of IS concerns from the domain to computational levels with real deployment. In addition, the taxonomic analysis of MMQEF addresses and reflects the expressiveness provided by the languages, and it provides a practical approach to discuss the advantages/disadvantages of the modelling act on concerns into an IS. Communicational issues result from hypotheses generated in the clustering of modelling languages regarding the taxonomic structure and their use.

3.5 Conclusions

This chapter presents the main considerations of the MMQEF approach, which is a method to evaluate the quality of modelling languages and modelling artefacts used in the construction of information systems by classification procedures. We described a scenario of applicability of the MMQEF method, in which multiple modelling languages were used for considering IS concerns in the context of business services. We are currently validating the method with MDE practitioners; however, we have preliminarily presented a theoretical validation of our method, discussing the main implications that the method possesses (i.e., its high dependence on an IS reference architecture and its compliance with previous MDE quality initiatives).

Further works are required to detail the evaluation procedures, expanding the main components of the method to obtain more granular components in a native language for their operationalization as a navigable process (such as a SPEM process with tasks, steps, artefacts, orientation, etc., or a SEMDM from ISO/IEC 24744:2014 (for Standardization/International Electrotechnical Commission et al., 2014)). Additional effort is required to demonstrate and discuss the formal support of the FCA approach to operationalize the MMQEF method (Section 3.3.6) in a popular native model-driven environment such as Eclipse EMF. Another work is projected to contrast our proposed method with previous MDE quality initiatives to find points of convergence and validate the method in model-driven scenarios and to develop a more detailed trade-off analysis of our approach.

Chapter 4

The formal and technological support of the MMQEF method



The Model-Driven Engineering (MDE) paradigm promotes the use of conceptual models in information systems (IS) engineering and research. As engineering products, conceptual models must be of high quality, which applies to both conceptual models and the modelling language used to build them. Due to the several challenges, divergences, and trends for quality assessment and assurance in MDE context, one way to perform a quality evaluation process is to use an approach where the applicability and goals of modelling artifacts can be compared with respect to the essential principles of the development of IS.

This chapter derives formal and technological requirements for a modelling language quality evaluation framework with the potential to tackle some of the open MDE quality challenges. For this purpose, we propose using principles from an IS architecture reference as a *taxonomy* that is applied on the modelling languages and elements used in information system development in order to perform analytic procedures.

Through this chapter, we discuss how this taxonomy supports analytics that are in modelling languages for quality purposes by its management of the semantics. We also demonstrate that this taxonomy can be considered as a *formal context* for the application of the *Formal Concept Analysis* (FCA) method. Finally, a tool that operationalizes the taxonomic evaluation procedure and the FCA analytic method is presented.

4.1 Introduction

The design and development of Information Systems (IS) with conceptual models is highly dependent on the cognitive abilities and experience of the people that participate in the modelling tasks. When an analysis is made on a conceptual model, it is difficult to qualify or give an opinion about its associated quality.

One of the main problems here is the *conceptual divergence* in the use and applicability of modelling languages. The cognitive process involved in the modelling act influences their definition and application. This leads to a decoupling

between the initial goals of the languages and the real use that final users of the language give to them.

The key behind the analytics process for quality evaluation in Model-Driven Engineering (MDE) is the explicit management of the *semantics* dimension. The Model-Driven Architecture (MDA) guide 2.0 (OMG, 2014b) promotes the *semantic analytics of models* as procedures over semantics data for assisting in decision making, monitoring and quality assessment. These procedures include tasks such as validation, statistics, and metrics.

However, the current state of the model-driven initiatives does not relate the semantics of the modelling languages with the essential modelling features that are expected in an IS development process. The conceptual frameworks about the model-driven paradigm do not clearly answer whether or not any modelling artifact is in MDE. There is an open issue about the derivation and management of data about the semantics that is associated to quality analytic processes on modelling languages and elements.

It is possible to support analytics on quality issues of modelling languages from the semantics data. Examples of issues are the following: suitability, expressiveness, arbitrariness, support for communication, management of traceability, and systematic support for transformations of models.

This chapter proposes a method and a tool to support an analytic procedure on modelling languages and modelling elements. It belongs to a *taxonomic evaluation* method which uses a *taxonomy* that is extracted from an IS reference architecture. This method is formally supported by the application of the *Formal Concept Analysis* (FCA) approach.

We use FCA for explicitly identifying relationships between concepts of languages and modelling elements. This approach (by default) manages the semantics of these concepts through a graphical representation in which the semantics is expressed as *distances* and *grouping* between concepts. From this application of FCA, we also derive a tool whose implementation was made on the Eclipse Modelling Project (specifically the EMF and GEF frameworks). This project is recognized as being one of the most important technical environments for MDE.

The automation of the FCA method by a plugin for a native MDE development environment results in an appropriate alternative for the validation of quality in models and modelling languages. Our work makes the following contributions:

- We discuss the applicability of an IS reference architecture as a *taxonomy* in a MDE quality analytic procedure. We support it using an ontological validation of the reference taxonomy. The sufficiency of the reference architecture as a *taxonomic theory* is also presented.
- We present a quality tool for managing the semantics and making quality analytic procedures on modelling languages and elements.
- Two examples of quality evaluation on modelling approaches are described in order to show the applicability of our evaluation method in combination with its associated tool.

The remainder of this chapter is structured as follows. Section 4.2 shows the main features of the taxonomic evaluation method, highlighting the conceptual support of considering an IS reference architecture as a MDE taxonomy. Section 4.3 describes the FCA method and its application in the taxonomic evaluation of modelling languages. Section 4.4 describes the main features of the implemented tool. Section 4.5 presents the application of the quality evaluation method with the implemented tool for supporting the management of the semantics data that are obtained from the taxonomic analysis. Section 4.6 discusses the main features of the quality evaluation method, including its applicability with respect to previous methods for quality evaluation at model-driven levels. Section 4.7 concludes the chapter and outlines future work.

4.2 The taxonomic analysis and the MMQEF method

Taxonomies play an important role in the development of IS projects (Olivé, 2001). In (Laware and Kowalkowski, 2005), the authors describe the business value of taxonomies and ontologies in terms of their relationship with architectural models that support IS practices.

This work is based on a taxonomic analytic procedure that uses *taxonomy* that relies on the Zachman framework (Zachman, 1987) (Sowa and Zachman, 1992). This framework is one of the most relevant reference architectures for Information Systems. Despite its commercial applications in the enterprise architecture field, we focus only on the essential principles that define this framework around the classification of artifacts that belong to an IS in an organizational context. The taxonomy allows us to derive an analytic procedure for evaluating modelling languages and elements.

The framework is a two-dimensional logical structure in the form of a *matrix*. The *rows* represent the *abstraction levels* involved in an IS development process, which move from organizational towards specific computational implementations. The *columns* are philosophical questions (*what, why, how, where, when, who*), which are used to understand the role of the modelling artifacts that are presented in an IS project with respect to the viewpoints. The combinations of abstraction levels and questions generate *cells* in the matrix. Each cell is unique (i.e., each cell has only a scope), and each cell is independent of the others.

The taxonomy establishes seven rules (Sowa and Zachman, 1992) in order to perform the classification. These are as follows:

- R1* The columns do not have any specific order.
- R2* Each column has a basic (*simple*) model. This implies that there is an *essential concept* for each column that answers the *question* of its associated column. The basic model constitutes the generic metamodel for any column.
- R3* The basic model of each column is unique.
- R4* Each row represents a perspective that is unique and different .

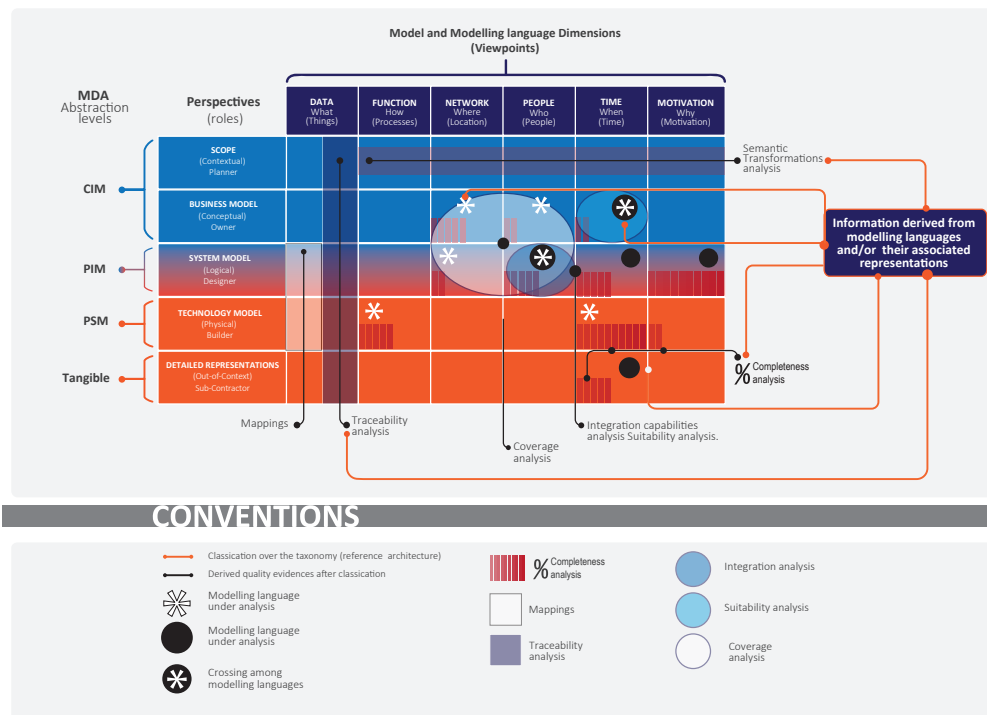


FIGURE 4.1: Summary of the support provided by the reference taxonomy for quality evaluation in MDE contexts.

R5 Each cell is unique.

R6 The integration of the models of the cells in a row constitutes a *complete model* of this row.

R7 The logic of the framework is recursive (i.e., the essential models can explain to itself), and each cell could be analyzed with the use of the entire taxonomy.

Fig. 4.1 summarizes the classification of artifacts of modelling languages and elements in the taxonomy. The Information that comes from modelling languages (such as semantic constructs) and modelling elements (e.g., data derived from diagrams) is classified with respect to the taxonomy structure depending on the purpose (or goal) of the modelling artifacts that is perceived by the analyst, and the information of the modelling artifacts that can be captured for the cells of the taxonomy.

The classification activity with the reference taxonomy and the derived analytic procedure that were described in Fig. 4.1 constitute the *Multiple Modelling Quality Evaluation Framework* method (MMQEF), which is described in previous publications. This method helps to find quality issues in modelling languages and elements according to their support for specific concerns and phenomena inside an IS.

Thus, *quality* for model-driven engineering could be defined as the degree to which a given modelling artifact meets an IS concern, considering its location

inside an abstraction level with a clear purpose (or viewpoint) and an explicit traceability for deriving technical implementations (as part of an IS development process).

The two-dimensional structure of the taxonomy gives a set of information that is useful and valuable for modelling and reasoning about any phenomena inside the scope of IS (Smith, 2013). The rules for classification guarantee the consistency for any IS modelling activity. The taxonomy recognizes the presence of different approaches and graphic representations that are appropriate for the cells, but it also recognizes that they are not completely adequate due to the different purposes that are addressed by each cell. For this reason, any attempt to fix this insufficiency without a systematic procedure (e.g., to arbitrarily join the formalisms of some cells) could lead to serious problems in the posterior design of the IS design; also, a sub-optimization of formalisms could be evident (Sowa and Zachman, 1992).

Another advantage for the taxonomy is the management of the model transformations as a *controlled process*, i.e., the transformations and mapping of models take place as a direct consequence of the addition of information in accordance with the interaction between abstractions levels and questions. The *model mapping* feature (which is mentioned in the first version of the MDA guide (OMG, 2003)) occurs as a result of the structural changes in the models that cross from higher to lower abstractions. Information that comes from the Computation-Independent model (CIM) level are enriched with constraints that are associated to lower levels so that it is possible to get enough information for the implementation of higher models in a technical (computational) environment.

Similarly, *transformations* between the information of two different columns must be sufficiently justified in order to support the derivation of models from different essential properties (e.g., *time-location*, *data-process*). Among the main features in the design of modelling languages, there must be an explicit rationale about *why* and *how* information from a column can derive/generate/support information for another column. In addition, the evidence of traceability can be obtained from the information classified in the taxonomy.

Fig. 4.1 also describes the main quality analytic procedures over modelling languages and elements that are supported by the taxonomy. The most important quality question that is managed by this taxonomy is the support of the *essential modelling* of IS concepts and their associated abstraction level. This is done by contrasting the modelling artifacts with the minimum information that is expected in each cell of the taxonomy.

The classification act requires the explicit rationale of the *technical issues implementation* of the model artifacts under consideration. The Platform-specific models (PSM) row of Fig. 4.1 expresses models that are transformed to any specific technological platform: programming languages, development environments, supporting platforms (frameworks, engines, APIs), hardware and network configurations.

Therefore, when the classification is performed, the associated technical details must be considered to ensure the full functional implementation of the IS (i.e., the transformation of the models to executable artifacts on computational

platforms). This is the implication of the *abstraction level zero* that the reference taxonomy defines in the lower row. This *functional implementation* feature must be considered at some point to indicate the feasibility of the modelling effort from a computational perspective.

The taxonomic analysis supports *quality inferences* for modelling languages and models elements¹. This distinction is important because quality evidence come from different sources regarding the type of artifact under analysis; for modelling languages, quality is based on their properties; and for modelling elements, quality is based on their derived and recognized use.

4.2.1 The formal support of the taxonomy for the quality evaluation analysis of modelling artifacts

The main feature of the reference taxonomy that is used by MMQEF is the classification of modelling artifacts that fit each other in a systemic way (Wegmann et al., 2008). Thus, the sufficiency of the taxonomy must be evaluated to determine the support of this feature and the derived reasoning about quality on modelling artifacts. To do this, we apply an ontological evaluation procedure proposed in (Siau and Rossi, 1998). In this procedure, the elements of the taxonomy are matched w.r.t. constructs from previous IS methods in order to verify the *essential notions* to be met by any well-constructed taxonomy. These IS methods are proposed in (Guarino and Welty, 2000)(Welty and Guarino, 2001).

The evaluation methods define four meta-properties for the understanding, comparison, and integration of taxonomies: *identity*, *unity*, *essence*, and *dependence*. The *identity* meta-property distinguishes a specific instance of a certain class from other instances of that class by means of a characteristic property which is unique for these instance. The *unity* meta-property distinguishes the parts of an instance from the rest of the world by means of a unifying relation that binds the parts, and only the parts together. The *essential* meta-property discusses which properties of an instance change / do not change over time, and how an instance can be reidentified after some time. Finally, the *dependence* meta-property asks about the several relations of the instances.

The reference taxonomy supports all four meta-properties as follows:

- The scope of each cell (by the classifiers that are extracted from the *abstractions-questions* combination) offers enough information to define an *identity* for a specific IS concern under modelling. This *identity* allows a modelling language artifact to be classified through an ontological reasoning that relates its intention with the identity information of the cell. Simultaneously, the whole/part notion is managed by this taxonomy through the rules for integrity in columns and rows (R2 to R6 of the taxonomy).
- *Unity* of the taxonomy at the column level is achieved by the adherence of the artifacts that are classified in the columns with the predefined generic model of each column (R2). For the rows of the taxonomy, *unity* is achieved

¹These refer to the specific elements that appear with the use of a modelling language, according to the definition presented in (Object and Reference Model Subcommittee of the Architecture Board, 2005).

by the alignment of each cell with the principles of each abstraction level in order to create a single row model that consists of multiple (and different) models of viewpoints (questions) with a consistent semantic link.

- For the *essential* meta-property, the classifiers of the taxonomy offer the evaluation mechanism required to identify the evolution of modelling language artifacts through the management of the inserted changes during the lifecycle of that artifact. This is the key factor in managing the traceability of models.
- For the *dependency* meta-property, the classifiers and rules of the framework define relations of dependency among the classified artifacts, which systematically harmonize with the independence of each cell.

In addition, the *backbone taxonomy* concept is defined in the IS methods used for the ontological evaluation of taxonomies. This is a set of special properties that are associated to a taxonomy for imparting structure in an ontology, facilitating human understanding, and enabling integration of knowledge. The *backbone taxonomy* is composed of the following concepts:

- *Formal roles*: these are properties that express the part that is played by one entity in an event, often exemplifying a particular relationship between two or more entities.
- *Material roles*: these are elements that inherit identity conditions from some type. They represent roles that are constrained to particular types of entities.
- *Phased sortals*: they correspond to a certain temporal phase of their instances.
- *Attributions*: these represent values of attributes (or qualities) like color, shape, etc.
- *Mixins*: these properties intuitively represent various combinations (disjunctions or conjunctions).

Fig. 4.2 presents the mapping of the reference taxonomy mentioned in our work with the *backbone taxonomy*. For this case, *formal roles* are mapped to the roles that are associated to the abstraction levels (originally defined as *perspectives*). The *material roles* are those that are involved in the modelling act itself (i.e., either defining/using/supporting a modelling language), such as business expert, system analyst, designer, architect, developer, ethnographer, security expert, etc. They are defined as roles that are associated with IS views and inherit from roles of abstraction levels.

The *phased sortals* concept refers to the different times in which analysis can be performed. For this case, we have temporal analysis when the modelling language elements cross between the MDA abstraction levels that the reference taxonomy supports. Another *phased analysis* occurs when the taxonomy is applied in posterior stages of the IS modelling process over time, e.g., to check the

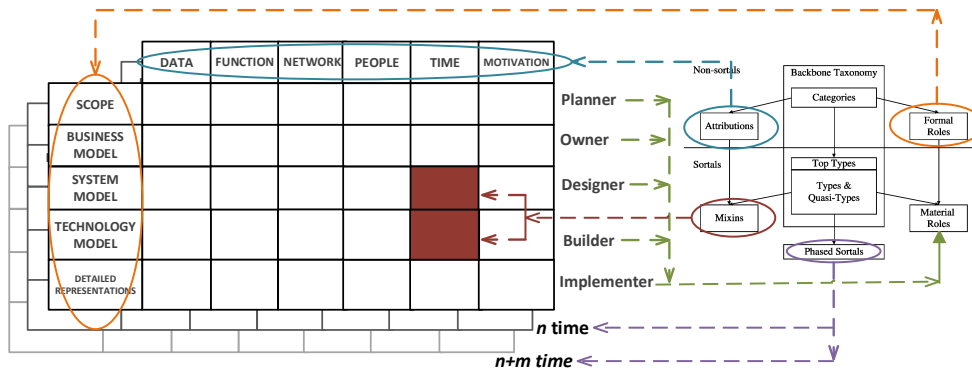


FIGURE 4.2: Zachman taxonomy matching the *Backbone taxonomy*(Guarino and Welty, 2000)(Welty and Guarino, 2001).

progress of the IS modelling and development (R7 of the taxonomy). *Attributions* are defined by the essential model of each column and the semantic coherence of the each row. Both elements define the attributes of the modelling artifacts under classification. *Mixins* are the combination of properties from abstractions and questions that are expressed as cells.

4.2.2 The Zachman framework as a taxonomic theory

While the above theoretical evaluations demonstrate the sufficiency of the taxonomy to derive reasoning and understanding, an analysis is required from a more taxonomical perspective to verify if the reference taxonomy that MMQEF uses could be considered as a *taxonomy theory* (i.e., a theory that prescribes how to classify objects of interest, explain similarities and differences among objects, and derive analysis).

To do this, we use a prescriptive framework formulated in (Muntermann et al., 2015) for determining whether or not the Zachman framework qualifies as a taxonomic theory. This is done through a three-condition procedure as follows:

- *Condition 01*: The candidate taxonomy must be formally represented, and it must meet a four-evaluation criteria analysis (*Usefulness, Clarity of Classification, Completeness and Exhaustiveness, and Expandability*).
- *Condition 02*: It must include components that describes the theory.
- *Condition 03*: It must provide a foundation to develop other theories.

To meet *Condition 01*, there are previous reports about the formal representation of the taxonomy; one example can be found in (Martin and Robertson, 1999). In addition, the above criteria are met as follows:

- *Usefulness*: the taxonomy is useful to identify and classify architectural representations (conceptual models) that relate concepts in an IS to the representations in underlying computational platforms. The taxonomical proposal is an IS architecture framework that takes advantage of architectural representations for understanding an IS.

- *Clarity of Classification*: the taxonomy clearly defines how to classify conceptual models and also defines the characteristics of each category in which models can be placed.
- *Completeness and Exhaustiveness*: the taxonomy defines the main abstraction levels and viewpoints for fully understanding an IS that is developed with conceptual models.
- *Expandability*: the taxonomy establishes a recursive logic that supports the classification for complex concepts.

To meet *Condition 02*, the main components of the theory (i.e., the bi-dimensional structure, and the rules and principles for classification) are described in (Zachman, 1987; Sowa and Zachman, 1992).

Condition 03 is met through the functions and analytic procedures that must be done to place modelling artifacts (modelling languages and elements) in the taxonomy, and, therefore, to derive quality inferences. The taxonomy provides the conceptual foundation to support the methodological and technological framework of the MMQEF method.

Because the Zachman-based taxonomy that is used in the MMQEF method satisfies the three conditions of (Muntermann et al., 2015), it can be considered to be a taxonomic theory with a conceptual foundation that is sufficient to support quality analytics at model-driven levels.

4.2.3 Related works

Some works previously used the taxonomic framework to reason about modelling languages. (Molina et al., 2014) presents a scenario in which the framework was used as a conceptual tool to systematically integrate a set of modelling languages that are used in a development process of a groupware software. In this way, the authors achieve a harmonization of modelling languages without detriment to the expectation of the participant roles. The main benefit of this harmonization for an IS project is the generation of computational platforms that cover the multiple expectations of the IS, each of which is modelled with the own resources of its associated viewpoints.

In (Frankel et al., 2003), a systemic combination between MDA and the reference taxonomy was proposed. MDA explicitly supports the taxonomy through the definition of the abstraction levels and the foundations of mappings in the top-down relations among these levels. The taxonomy complements MDA with the definition of the *modelling dimensions* (from the philosophical questions) that supports the mapping and transformations of models at conceptual levels, which are independent of their implementation on a specific model transformation language.

However, currently, there are no reports about the applicability of the reference taxonomy as a model evaluation tool that support inferences and analytics on models. Some works that propose the use of the taxonomic structure for modelling-related tasks are evident. For example, the authors in (Kingston and Macintosh, 2000) make suggestions about modelling approaches for a medical

domain; these suggestions are made by performing a classification of modelling alternatives in the taxonomy. The authors also propose the use of individual perspectives of the taxonomy as *user interfaces* for a knowledge distribution system. In the analysis with the taxonomy reported in (Noran, 2003) a set of modelling languages is suggested to populate each cell in accordance with the purpose of the languages and the specific tasks that are associated to each cell.

The support of the taxonomy for inferences at ontological levels has also been reported. (Kingston, 2008) describes how the taxonomy was used for managing multi-perspective modelling in an ontology development process. The resulting reasoning comes from the classification activity on the type of knowledge that is addressed. The R7 rule of the taxonomy (recursivity) was used in (Garner and Raban, 1999) to propose an IS context management approach that provides a dynamic validation of user requirements. The classifiers of the taxonomy were used by (de Graaf et al., 2014) to address an ontological approach in specific architecture scenarios.

A similar work about relations among viewpoints is reported in (Romero et al., 2009), where the authors take advantage of the RM-ODP viewpoints to propose a generic model-driven approach for the specification and realization of correspondences (relationships) among these viewpoints. However, unlike the reference taxonomy, RM-ODP focuses on the development of the architecture so that the rationale and tradeoffs of the architecture do not belong to the RM-ODP model (Tang et al., 2004).

Regarding the operationalization of analytic procedures for evaluating modelling languages, a work is reported in (Shuman, 2010) which proposed a checklist for reviewing issues of operational executable architectures. Thus, the features of modelling languages (diagrams or specific graphical constructs) were classified and measured according to the items that were extracted from the DoDAF framework. Unlike our work, this operationalization was proposed as a spreadsheet file and was not included in a MDE technical environment.

4.3 The FCA method

The *Formal Concept Analysis* (FCA) approach is a mathematical method for data analytics, representation of knowledge, and information management (Priss, 2006). It is based on the ordered set theory. FCA provides a natural method for defining concepts in a model so that these concepts are associated to others through shared features. The main goal of the FCA method is that *concepts* are all the pairs of *objects* and *attributes* that have a mutual dependence.

The foundational support of the *formal concept* is that a *concept* is determined by its *extension* (i.e., the collection of objects that are covered by this concept) and its *intention* (the set of properties or attributes that are included in this concept). Therefore, a *formal concept* is conformed in an incidence relationship called *formal context* between a set of objects and a set of attributes through any closing operator.

According to (Wolff, 1993), the mutual dependence between objects and attributes is defined as a *formal context* (a partially ordered triplet) $K = (O, A, I)$ where: O represents a formal set of objects (e.g., the classes in a software model);

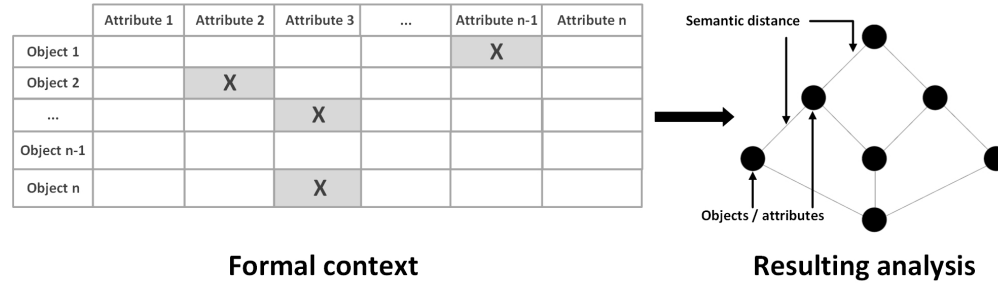


FIGURE 4.3: Summary of the FCA method (with figures taken from (Wolff, 1993)).

A is a set of attributes that may or may not have objects; and I is a relation of incidences that shows the association between an object and an attribute; I is defined as $I \subseteq O \times A$. I is expressed as a binary relation (o, a) , it is interpreted as *the o -object has the a -attribute*.

A *formal context* can be represented as a matrix with crossings, where the rows are O , the columns are A , and the incident relation I is a series of crossings between rows and columns. The FCA verifies the closeness between concepts (i.e., the *semantic distance*) by operations of containment and overlapping depending on the incidences that were found. This analysis receives a matrix of objects (rows) and attributes (columns) as input. If a specific feature is identified, a mark is assigned in the corresponding cell of the attribute that possesses it. The output of the analysis is a *concept lattice* (connected graph), where nodes are the concepts and attributes under analysis, and the lines represent the semantic distance between them. Fig. 4.3 summarizes the FCA method.

Previously, some works reported the applicability of FCA in the conceptual modelling field, e.g., as a mechanism for extraction of rules that supports the validation of *good models* (Richards, 2000), the learning of model transformations by rule extraction (Saada et al., 2012), the search of generalizations in models to improve their abstraction levels (Falleri et al., 2008), and attempts to synthesize models from constraints (She et al., 2014).

Other previous applications of the FCA method to IS topics have been reported, e.g., the analysis of social networks data (Freeman and White, 1993), the classification of software bugs (Borchmann et al., 2014), the representation of XML data (SăCărea and Varga, 2014), data integration (Liu and Li, 2014), the composition of services (Abid et al., 2015), and processes for building/refining ontologies (Bendaoud et al., 2008).

4.3.1 The FCA support for the taxonomic analysis

The FCA method provides an efficient approach for the derivation of inferences of any modelling artifact by using the relationships between artifacts of the modelling languages or elements. These inferences help to verify the fulfillment of modelling goals, and they facilitate the analytics for designers of languages and final users of languages.

The FCA method processes the grammar constructs provided by the modelling languages that are involved in an IS development process (semantic constructs, diagrams, etc.), generating a connected graph or *concept lattice* as the output, where the relations between elements are described. The resulting lattice derives inferences about the application of modelling languages.

As mentioned above, the FCA method is based on the set theory and the possible operations that can be realized on sets. FCA looks for similarities at the row and column levels and takes them as subsets in order to associate the higher number of ordered pairs (o, a) that have a marked relationship in the incidence matrix. These associations are achieved by a containment operation.

In order to operationalize the quality evaluation procedure mentioned in Section 4.2, we consider the reference taxonomy as a *formal context* where the (o, a) pairs are the incidences between the abstraction levels (rows) and the philosophical questions (columns). These incidences are derived when the information of modelling languages and elements is classified using the taxonomic structure.

In the classification of the modelling artifacts over the reference taxonomy, some types of association of concepts are identified:

- An object is the only that has an attribute in all the sets of ordered pairs that were submitted.
- Two or more objects have the same associated attributes, so that their semantic distance will be linked to the same node in the concept lattice.
- At the column level, there are two or more attributes that are contained in the same relationship associated to an object (or more).

Due to these associations, the rules of the taxonomy must be contrasted with the rules of the FCA method for analyzing concepts in order to harmonize the two set of rules for generating concrete analyses over the modelling artifacts. This was achieved by a modification the FCA original algorithm so that the taxonomic independence that is defined by each row of the reference taxonomy (R4 and R6) could be not combined during the FCA parsing procedure.

The FCA method receives the matrix with the two-dimensional structure of the taxonomy as input. Objects are the abstraction levels (*Contextual, Conceptual, Logical, Physical, and Detailed*), and attributes are primitive philosophical questions (*Why, How, What, Who, Where, When*). The incidences are the relations between objects and attributes in one or several cells of the taxonomic structure that were identified for some features of the modelling artifact under analysis.

In addition, to meet R5 of the taxonomy (*each cell is unique*), a *contribution value* was established to indicate the *percentage of completeness* that the modelling artifact contributes in answering the coverage of the abstraction level-philosophical question pair for a specific cell. The incidences are also marked using the element of the modelling language that contributes to covering a specific cell (e.g., a diagram). All of this information is used to generate the concept lattice.

4.4 The EMAT Tool

EMAT (*EMF Modelling Analytics Tool*) is a technological framework that is implemented to support the evaluation of quality in modelling languages and elements through the taxonomic analysis proposed in the MMQEF method. EMAT is a plugin for the Eclipse Modelling Project², specifically the Eclipse Modelling Framework (EMF). The plugin supports the analytic procedure with the reference taxonomy presented in Section 4.2 and the FCA method of Section 4.3 for the management of the semantics data derived from the taxonomic analysis.

4.4.1 Why is another FCA tool needed? Is EMAT another FCA tool?

In accordance with previous reviews of our work, we identify a common question about why it is necessary to develop another FCA tool, taking into account previous FCA tools (e.g., Concept Express³, and Toscana⁴). These types of FCA tools support analysis of objects/attributes in generic contexts. EMAT is not another FCA tool. It focuses on providing the required technical support to evaluate modelling artifacts through taxonomic analysis and operationalizing the generation of lattices that contain objects and attributes derived from information of the modelling artifacts.

EMAT implements the FCA support for the taxonomic analysis previously mentioned in Section 4.3.1. This FCA analysis considers the application of the seven rules of the reference taxonomy. The output of EMAT is not only a drawing of a connected graph, it is also a *conceptual model* of the semantic closeness among objects/attributes of modelling languages and elements. Quality inferences can be deduced and also automated from this model.

Unlike traditional FCA tools, EMAT works within a MDE technical environment. As a further consequence, we hope this integration promotes the reduction of the subjectivity criteria that are traditionally associated to the quality evaluation processes. EMAT allows model-driven practitioners to perform reasoning over shared semantic lattices.

4.4.2 The taxonomic evaluation procedure using EMAT

With respect to the conceptual integration proposed in Section 4.3.1, the EMAT tool supports the taxonomic analysis of modelling artifacts using the reference taxonomy. To do this, EMAT offers a view into the Eclipse working area to classify the elements of the modelling languages under analysis in each cell of the taxonomy.

For each language under evaluation, the analyst (i.e., the role that represents the designer of the language or the final user of the language) can indicate whether the elements of the language fit in a cell with its associated *coverage value*; it is a percentage between 0-100 that indicates the degree of coverage that these elements contribute to answering the specific question in the considered

²<https://eclipse.org/modeling/>

³<http://conexp.sourceforge.net/>

⁴<http://toscanaj.sourceforge.net/>

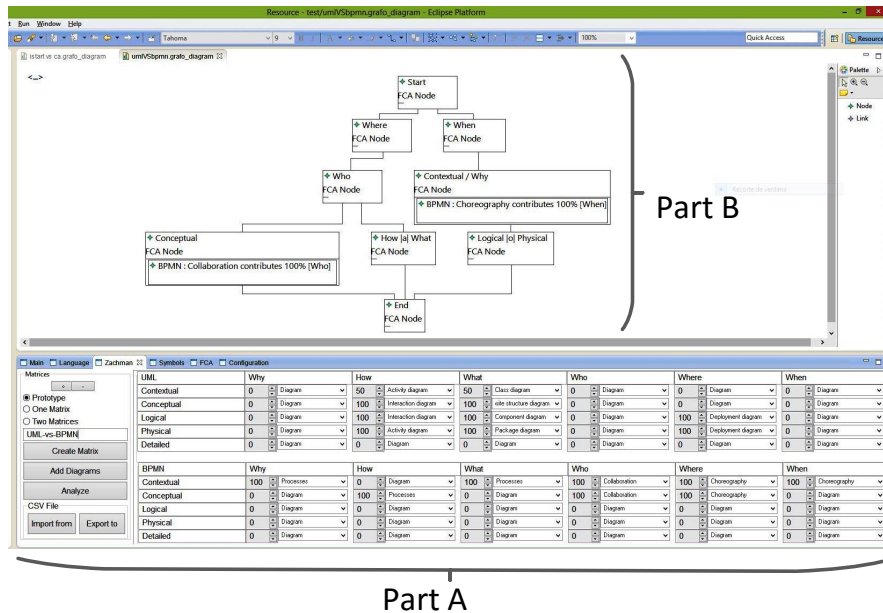


FIGURE 4.4: Example of the taxonomic evaluation for the UML and BPMN modelling languages performed in the EMAT tool.

cell. For each cell, two input values are generated (*modelling_language_element*, *percentage*) for the FCA algorithm; these values mark the *incidence* relationship in the matrix.

In the current version of the tool, a graphical interface was implemented to evaluate one or more modelling languages versus the reference taxonomy, i.e., a FCA analysis is performed by overlapping the matrixes (each one with the specific set of incidences for each modelling language under analysis). This evaluation produces relationships of containment in the resulting concept lattice. In this concept lattice, the contributions of each language for each cell are reported, differentiating which of the languages is closer to 100% of the contribution. This is the *completeness analysis* that was depicted in Fig. 4.1.

Fig. 4.4 presents an example of an analysis that was performed over the UML and BPMN modelling languages. For this case, we use the information from the diagrams associated to both languages, indicating the coverage provided by each diagram in each cell of the taxonomic structure (Fig. 4.4- *Part A*). Each diagram has a structure to classify it. Afterwards, a concept lattice (Fig. 4.4- *Part B*) is automatically generated as the result of the application of the FCA method with the modifications for preserving the taxonomic rules.

EMAT allows the configuration of options for the automatic generation of the concept lattice. Because of its mathematical foundations, the FCA properties must be fulfilled. Therefore, EMAT provides a configuration panel where the user can apply the FCA analysis by selecting between two alternatives, either fulfilling the essential modelling of the taxonomy or applying each FCA rule to each matrix. Each alternative generates a connected graph.

Finally, to support the associations reported in Section 4.3.1, we use three

conventions to identify them (Fig. 4.4 *Part A*). The / convention represents that the object is the only one that has an attribute (i.e., a single association between an abstraction level with a philosophical question or vice versa). The | *obj* | convention indicates that two or more abstractions have the same associated attributes. The | *att* | convention defines that two or more questions are associated with one or more abstractions. Depending on the selected configuration, the FCA analysis can be performed by taking into account R3 of the taxonomy (each column has a basic model), or applying the FCA method to look for the semantic similarities between columns reported from the classification of modelling artifacts.

4.4.3 How should a resulting lattice be interpreted ?

The graphical interpretation of the lattice in EMAT is equivalent to a normal FCA result, which provides a hierarchy for analyzing concepts from a set of objects and attributes that compose them. Each concept of the obtained hierarchy represents a set of objects that share the same values or meanings for a certain set of attributes.

The formal concepts are defined as a pair of a set of objects (*extension*) and another set of attributes (*intention*). The *extension* is all the objects that share the given attributes. The *intention* is all the attributes that share some given objects. Formal concepts can be ordered partially due to the containment relationship among their sets of objects and attributes. This order produces a hierarchized system in which *sub-concepts* and *super-concepts* appear. These are visualized as nodes and links.

EMAT applies the notion of *object*, *attribute*, *concept*, *sub-concept*, and *super-concept* to interpret an obtained lattice. This interpretation must be done top-down, from the start to end nodes, and taking into account that the attributes (i.e., the questions in the reference taxonomy) are those nearest to the start node, and the objects (i.e., the abstraction levels in the reference taxonomy) are nearest to the end node.

An object has attributes, and, in turn, an attribute could be in many objects. This produces a hierarchized lattice in which the *sub-concepts* are the highest nodes in the lattice, and the *super-concepts* are the lowest. The relations among nodes are binaries. They detect if a node has a link with another node, and if a node is a *super-concept* or a *sub-concept* with respect to another node and its associated position.

An additional consideration is needed in the taxonomic analysis because EMAT provides a *completeness percentage* to indicate the degree of support of a modelling artifact for a specific concept. This percentage is depicted as internal nodes inside a conceptual node.

Fig. 4.5 presents an illustrative lattice that was generated intentionally in EMAT, which has the *Physical*, *Why*, and *When* attributes, and the *Contextual* node that is an object. *Contextual* has some internal nodes that describe the relations between nodes.

Contextual and *When* have a relation of 50% that is covered through the BPMN Business Process diagram. This same relation is covered by the UML

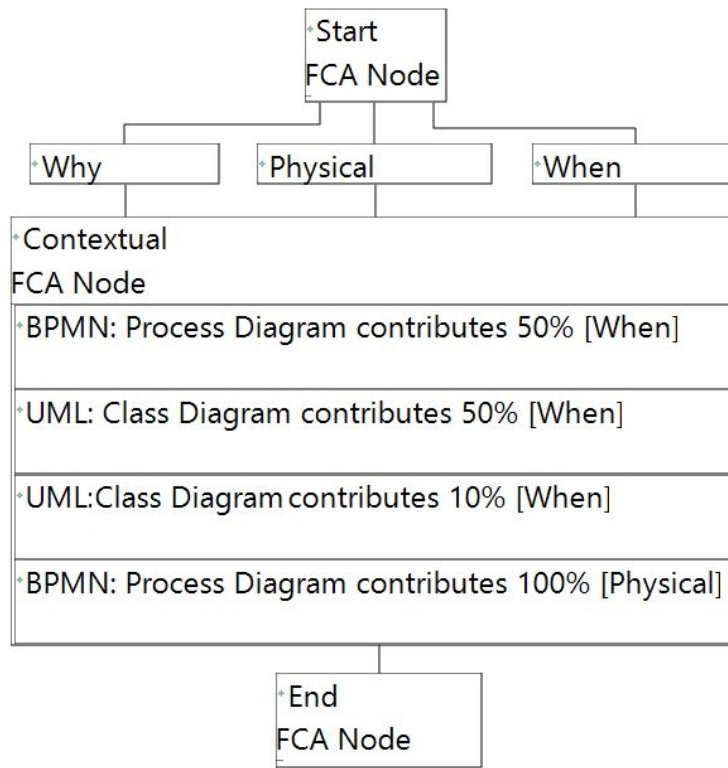


FIGURE 4.5: Example of a lattice generated automatically in EMAT.

Class diagram. Afterwards, a relation among the *Contextual* concept and the *Why* attribute is covered only by the UML Class diagram with a level of 10%. Finally, there is a relation between *Physical* and *When* with a level of completeness of 100% through the BPMN Business Process diagram. This relation appears inside the *Contextual* node because, in the classification with EMAT, the analyst indicated that these BPMN diagram support both levels (*Contextual* and *Physical*).

From the lattice shown in Fig. 4.5, it can be deduced that BPMN is the most appropriate language for this modelling task because its diagrams cover most of the levels and questions involved in the taxonomic analysis.

4.4.4 Other complementary functions

Another functionality that was implemented in the current version of the EMAT tool is the conceptual integration of modelling languages by the use of foundational ontologies for IS. For this case, the required incidences for the FCA method are the relationships between constructs of the modelling languages under analysis. These constructs are linked conceptually through an ontological construct that belongs to an IS ontology.

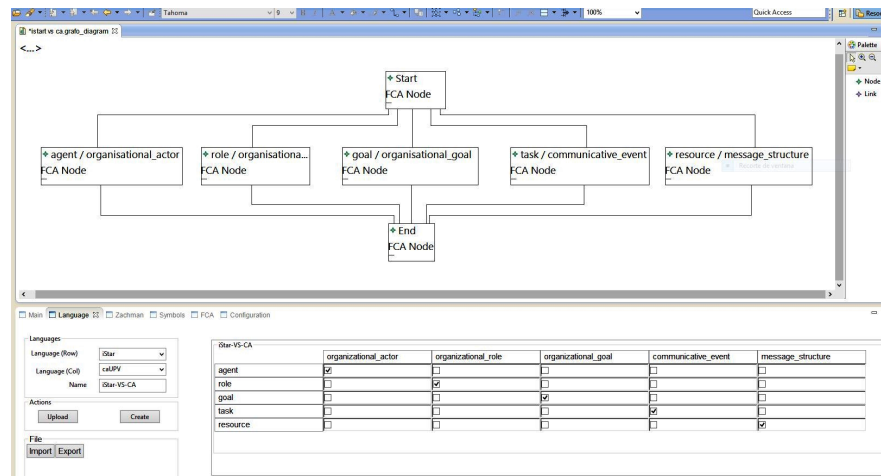


FIGURE 4.6: Example of an ontological analysis supported by the EMAT tool (extracted from (Ruiz et al., 2014)).

Fig. 4.6 presents an example of integration based on the case reported in (Ruiz et al., 2014). This work reported the integration of the i^* goal-oriented modelling method and the *Communicational Analysis*, which is a communication-oriented business process modelling method. This integration uses ontological concepts taken from the FRISCO framework for IS (Falkenberg et al., 1996) in order to find similar concepts in both languages (i.e., the constructs that share similarities with respect to the concepts previously defined in the reference ontology).

In this type of analysis, the FCA method can be used to support the association of constructs, and, hence, for determining the semantic closeness between them with the foundational concepts of the ontology. The analysis provides more precise information about the fulfillment of the constructs. The tool imports the constructs of the languages under analysis from the metamodel defined in the corresponding *ecore* file.

Finally, to reduce the complexity in the input of the matrixes for the taxonomic/ontological analysis, EMAT can load them from cvs files generated from Microsoft Excel, in which previously the matrixes were established. In addition, EMAT can display the nodes that were discarded in the FCA algorithm in the work area (i.e., those nodes without any relationship defined in the matrix of the formal context). This feature is useful for considering those abstractions/questions that were not covered by the languages under evaluation.

4.4.5 EMAT architecture and future vision

Fig. 4.7 describes the EMAT architecture. This work reports the implementation of *Part A* of Fig. 4.7. This part corresponds to the *analytics component* that receives a taxonomic analysis (the formal context) as input, and produces a FCA concept lattice in which nodes represent the concepts of the taxonomy and the elements of the modelling languages that satisfy these concepts. The lines of the

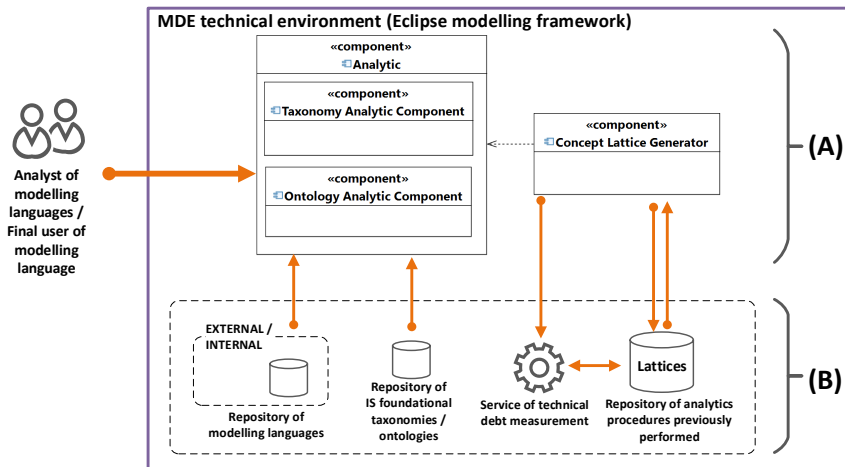


FIGURE 4.7: Architecture of the EMAT tool.

lattice represent the semantic closeness between concepts and elements under analysis.

With the release of the EMAT as a plugin for the EMF platform, we promote the evaluation of the quality of modelling languages through a technical modelling environments that is commonly used by model-driven practitioners.

Some visionary elements of the architecture are presented in Fig. 4.7, *Part B*. The information from modelling languages and elements will be extracted from existing specialized repositories. In the current version of EMAT, we extract the information of the modelling languages of the AtlantEcore Zoo repository from the AtlanMod research group from the Ecole des Mines de Nantes (France)⁵.

We plan to create a shared repository in which the results of the analytic procedures for quality evaluation of modelling languages can be available for model-driven practitioners. This shared information will be useful for taking decisions about the applicability of modelling languages for specific IS concerns and projects.

In the current version of EMAT, the resulting lattices are rendered directly over the work area of EMF (see Section 4.4.2). However, lattices are not only a visual output with graphical information of the semantics, they are also models that contain data about the semantic closeness of modelling languages and modelling elements. In this way, quality inferences (that are derived from the taxonomic analysis of MMQEF) are formally supported and potentially automatable by the application of some query analysis and/or formal methods.

Integration with a *technical debt* calculus service for models and modelling languages is also proposed. This service is currently a work in progress (Giraldo et al., 2015b). We propose that the semantic information derived from the analytic procedure can be used as metrics to derive technical debt analysis of modelling artifacts.

Finally, another repository is proposed for taxonomies and ontologies used by EMAT in the FCA analysis. This will be a parametrization feature of EMAT

⁵<http://web.emn.fr/x-info/atlanmod/index.php?title=Ecore>

TABLE 4.1: Results of the survey applied in the first validation of the EMAT tool.

Question	None	Low	Medium	Partially	High
Does the EMAT tool met the goals of the taxonomic analysis?	22%	22%	17%	17%	22%
Is the concept lattice resulting understandable?	26%	21%	16%	11%	26%
Is the use EMAT intuitive?	26%	26%	11%	11%	26%
Is EMAT useful?	22%	22%	22%	17%	17%
Is the analytics procedure complicated?	7%	22%	21%	21%	29%

that will allow the most convenient taxonomy/ontology to be selected for making a quality evaluation procedure for the analysts of modelling languages. The core EMAT functionality is the FCA analysis from the reference taxonomy of the Zachman Framework (Section 4.2). However, this repository will contribute to the operationalization of this type of analysis on modelling languages, and, therefore, it gives shared support to discuss and reason about the obtained analysis .

4.4.6 A trade-off analysis of the EMAT tool

Despite the potential of EMAT to address semantic data from modelling languages, there is a clear risk about its use due to the cognitive effort involved for the interpretation of the lattice obtained and the knowledge associated to the FCA approach. Both conditions are required by the model-driven practitioners that are interested in performing analytic procedures on modelling languages.

In order to determine the presence of these conditions in an evaluation scenario and the the usefulness, understandability, and further use of EMAT, a survey about the EMAT tool was developed. A group of nine model-driven practitioners was selected. The participants (researchers and Master’s students) were chosen in accordance with their previous knowledge in the use of model-driven technical environments such as EMF, metamodeling, generation of DSLs, and code-generation.

Before using EMAT, the participants received a lesson about the FCA method and the taxonomic analytic procedure. Afterwards, a small exercise was assigned to each participant in which the suitability of two modelling languages was questioned. In this lesson, an exercise for modelling a core business process of the University of Quindío was formulated, using the BPMN 2.0 and UML (including a profile for business modelling proposed in (Kruchten, 2000)). Within the considerations for the exercise, we promoted typical features of model-driven

projects such as the capacity of code-generation and the support of documentation for business experts. EMAT was used to perform the analytic procedure for both languages involved by using the reference taxonomy.

Finally, a survey was given to the participants to measure their first interaction with the EMAT tool. The responses were scored in a range from 1 (no impact) to 5 (high impact). Table 4.1 summarizes the results obtained. These results indicate initial skepticism; however, this skepticism was expected by the researchers due to the lack of previous contact of the model-driven practitioners with formal methods to manage semantics, and their preliminary knowledge about the IS reference architecture. The configuration options for the graphic rendering of the working environment also directly affected the expressiveness and understandability of the resulting concept lattices.

Quality challenges of this type on modelling languages may not be currently required for these participants due to the surrounding factors such as the relative size of their model-driven projects or the low complexity level in which their projects are developed. However, the results obtained give us important feedback on issues that must be addressed regarding the analytic method and tool and for their applicability to more complex IS development projects.

4.5 MMQEF and EMAT in practice

In this section, we present two demonstrations of the application of MMQEF and EMAT. Modelling scenarios were identified for both examples with more than one modelling language and their associated modelling elements. We apply MMQEF focusing on the use of EMAT and its generated lattices to perform quality inferences.

4.5.1 Quality analysis of the UML and BPMN modelling languages

Fig. 4.8 presents the application of the MMQEF method to analyze the UML and BPMN modelling languages. We chose these two languages due to the taxonomic analyses that were previously made separately for the two languages. These previous analyses used the reference taxonomy to classify the diagrams associated to each language. For example, the analysis on UML presented in (Frankel et al., 2003) was proposed from the first release of the MDA specification. The BPMN analysis described in (Zhao et al., 2012) evaluates the suitability of this language for linking the business with the information systems.

UML has been commonly considered the modelling language (by default) for software systems, focusing on functional features of these systems, and deriving objects that interact with each other to perform those expected functionalities. On the other hand BPMN is one of the most recognized approaches for modelling business process. Some BPMN platforms (or BPMN suites) support desirable features of modelling languages, such as the execution of models (business process models), the automatic generation of software code, and the native use for domain experts or business analysts.

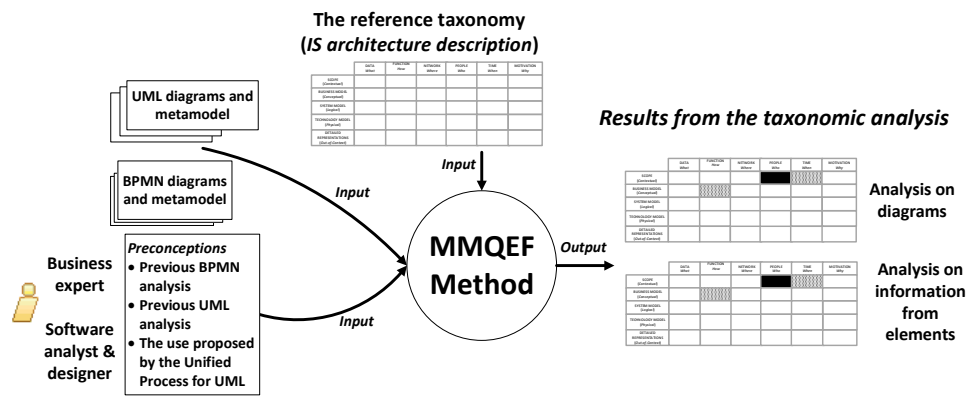


FIGURE 4.8: Summary of the application of the MMQEF method on UML and BPMN modelling languages.

MDA Abstraction levels		WHAT (Things)	HOW (processes)	WHERE (location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	Scope (Contextual)	UML: Package diagram (100%)					
	Enterprise Model (Conceptual)	UML: Class (Business classes) diagram (100%) UML: Use Case (Business Use cases) diagram (50%) BPMN: Business Process diagram (100%)	UML: Activity diagram (80%) UML: Business Object Model diagram (70%) BPMN: Business Process Diagram (100%)		UML: Use Case (Business Actors) diagram (70%) UML: Activity diagram (90%) BPMN: Collaboration diagram (90%) BPMN: Choreography diagram (50%)	BPMN: Choreography diagram (50%)	BPMN: Business Process diagram (100%)
PIM	System Model (Logical)	UML: Class (Persistent classes) (100%)	UML: State machine (80%) UML: Sequence diagram (60%) UML: Communication diagram (60%)	UML: Deployment diagram (100%)	UML: Sequence diagram (60%) UML: Communication diagram (60%)	UML: Sequence diagram (70%) UML: Communication diagram (70%)	
PSM	Technology Model (Physical)						
Physical	Detailed Representation (Detailed)						

FIGURE 4.9: Taxonomic analysis of the diagrams of the UML and BPMN modelling languages.

In this analysis, we also consider the guidance that was provided in the Unified Process (Kruchten, 2000) for the specific use of UML in the most representative disciplines of Software Engineering. This process was the first methodological prescription that proposes the use of UML in the construction of software systems. It also considers the use of an UML profile to perform *business modelling*, which is the discipline that prescribes a business engineering effort in order to derive software system requirements for software solutions that must fit into an organization. This consideration was formulated before the official release of the BPMN approach.

We made the classification of the modelling artifacts from the specifications

MDA Abstraction levels	WHAT (Things)	HOW (Processes)	WHERE (Location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	Scope (Planner)	UML: Package Diagram/Package (100%) UML: Package Diagram: Packageable element (200%)				
	Enterprise Model (Owner)	BPMN: Activity (100%) BPMN: Pool (50%) BPMN: Data object (100%) BPMN: Group (100%) BPMN: Task association (100%) BPMN: Type dimension (10%) BPMN: Task (100%) BPMN: Choreography task (100%) BPMN: Multiple Instances (100%) BPMN: Transaction (100%) BPMN: Nested/Embedded Sub-Process (100%) UML: Class diagram: Relations (100%) UML: Class diagram: Class (100%) UML: Use Case (Business Use cases) diagram: Use case (50%) UML: Use Case (Business Use cases) diagram: Relations (60%)	BPMN: Event (100%) BPMN: Activity (100%) BPMN: Gateway (100%) BPMN: Sequence Flow (50%) BPMN: Message Flow (100%) BPMN: Association (50%) BPMN: Message (100%) BPMN: Task (100%) BPMN: Coreography task (100%) BPMN: Process/Sub-Process (100%) BPMN: Collapsed Process/Sub-Process (100%) BPMN: Expanded Sub-Process (100%) BPMN: Collapsed Sub-Choreography (100%) BPMN: Expanded Sub-Choreography (100%) BPMN: Gateway control types (100%) BPMN: Sequence flow (100%) BPMN: Normal flow (100%) BPMN: Uncontrolled flow (100%) BPMN: Conditional flow (100%) BPMN: Exception flow (100%) BPMN: Message flow (100%) BPMN: Compensation association (100%)	BPMN: Data object (100%) BPMN: Fork (100%) BPMN: Join (100%) BPMN: Decision, Branching Point (100%) BPMN: Exclusive (100%) BPMN: Event-Based (100%) BPMN: Inclusive (100%) BPMN: Meeting (100%) BPMN: Looping (100%) BPMN: Activity Inoping (100%) BPMN: Sequence Flow Inoping (100%) BPMN: Association (100%) UML: Activity (Business workflow) diagram: Activity (100%) UML: Activity (Business workflow) diagram: Constraint (80%) UML: Activity (Business workflow) diagram: Actions (80%) UML: Activity (Business workflow) diagram: Node Decision (80%) UML: Activity (Business workflow) diagram: Node Bifurcation(80%) UML: Business Object Model diagram:Object (50%) UML: Business Object Model diagram:Relations (10%)	BPMN: Event (20%) BPMN: Sequence Flow (20%) BPMN: Flow dimension (100%) BPMN: Lane (100%) BPMN: Start (50%) BPMN: Intermediate task (70%) UML: Use Case (Business Use cases) diagram: Actor (100%) UML: Activity Partition: (70%) BPMN: OF-Page Connector (100%) UML: Activity (Business workflow) diagram: Control Flow (80%)	
PIM	System Model (Designer)	UML: Class diagram: Class (100%) UML: Class diagram: Relations (100%)	UML: State machine:State (60%) UML: State machine:Transition (80%) UML: Sequence: Messages lost or found (60%) UML: Sequence: Messages sent (60%) UML: Sequence: Constraint time (60%) UML: Communication: Object (80%) UML: Communication: Message (60%)	UML: Deployment: Node (100%) UML: Deployment: Instance (100%) UML: Deployment: Artifact (100%) UML: Deployment: Association (100%)	UML: Sequence: Actor (100%) UML: Communication: Actor (80%)	UML: Sequence: Life line (100%) UML: Communication: Life line (100%)
PSM	Technology Model (Builder)					
Physical	Detailed Representation (Programmer)					
	Functioning (User)					

FIGURE 4.10: Taxonomic analysis of the modelling elements of the UML and BPMN modelling languages

of both languages provided in the OMG web site⁶ and the previous analysis using the reference taxonomy. For this case, Fig. 4.9 presents the classification of the diagrams that are associated to the languages under analysis, and Fig. 4.10 presents the classification of some modelling elements that are relevant to this analysis.

These classifications were made in accordance with the perceived use of the modelling artifacts regarding the taxonomic cells (i.e., artifacts are classified based on the information that each cell can capture from them). The percentages associated to each element are assigned based on the perceived completeness of each modelling artifact for the specific cell.

The classification presented in Fig. 4.9 is the input for the EMAT tool. To begin the FCA analysis, EMAT looks for objects and attributes without any relation to incidence. In this case, the *Physical* and *Detailed* objects are eliminated. Then, EMAT looks for objects and attributes that are unique (i.e., those that have only one incidence). In this case, the *Contextual* object only has an incidence with the *What* attribute through the *UML package diagram*. This object is added to the *What* attribute using the /att/ label to indicate that *What* only has the *Contextual* object. This grouping produces the new concept [What/att/Contextual].

In the same way, the *Why* attribute has only one incidence with the *Conceptual* object through the *BPMN Business process* diagram. Therefore, this attribute is added to the *Conceptual* object using the /obj/ label to indicate that the *Conceptual* object only has the *Why* attribute. It produces the new concept [Conceptual/obj/Why]. This same procedure is performed for the *Where* attribute, resulting in the new [Logical/obj/Where] concept.

⁶Specifications available at <http://www.omg.org/spec/UML/2.5/> (UML), and <http://www.omg.org/bpmn/index.htm> (BPMN).

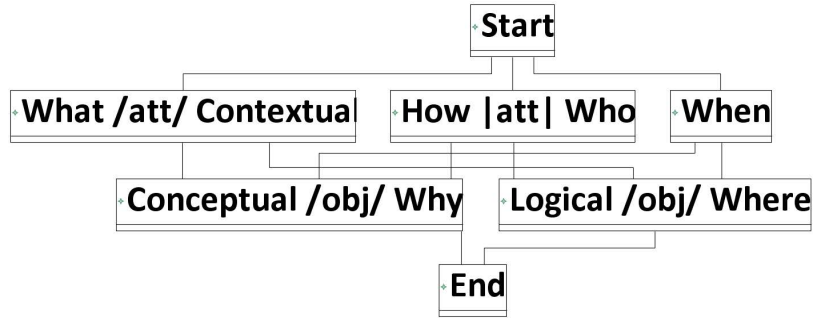


FIGURE 4.11: Lattice generated from the classification shown in Fig. 4.9.

Afterwards, EMAT looks for objects with common attributes and vice versa. This analysis considers the new concepts that were formulated in the previous step. If EMAT determines that two attributes are related to the same objects, it shows the */att/* label. Finally, EMAT looks for objects that contain other objects for the same (or fewer) attributes. These are the *super-objects*. The same is performed to look for *super-attributes*.

Finally, EMAT generated the lattice shown in Fig. 4.11. Following the interpretation process that is presented in Section 4.4.3, the following inferences are detected from the classification of diagrams:

- When the *What* viewpoint is defined, it also defines the *Contextual* level.
- When the *Contextual* level is defined, it also defines the *Why* viewpoint.
- When the *Logical* level is defined, it implicitly defines the *Where* viewpoint.
- The *How* and the *Who* viewpoints are indistinctly defined with the use of the involved modelling languages and their diagrams.
- The *When* viewpoint has not modelled consistently with respect to the other viewpoints and abstractions under analysis. In other words, the modelling of *When* is very different with respect to the other selected cells in the taxonomy.

EMAT applies a similar procedure for the classification of modelling elements shown in Fig. 4.10. This generates the lattice shown in Fig. 4.12. In this case, the consistency issues are in the *When* and *Conceptual* viewpoints.

From the taxonomic analysis on UML and BPMN, MMQEF infers the following quality issues that must be addressed for integrally modelling the main concerns of an IS project:

- BPMN does not cover the data modelling of business processes.
- The *Where* viewpoint is not integrally covered at the CIM levels for the two modelling languages.

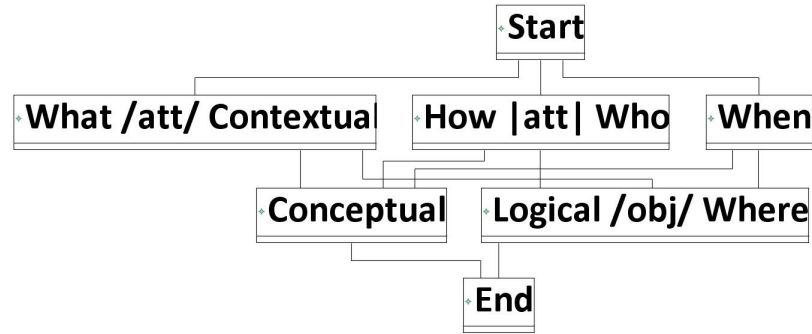


FIGURE 4.12: Lattice generated from the classification shown in Fig. 4.10.

- There is no explicit specification to manage the traceability from the *Conceptual* to the *Physical* abstraction levels. For this reason, decisions about the derivation of code or executable components are related to specific tools. The modelling languages do not explicitly define this generation.
- It is not clear whether or not the models associated to the *Conceptual* level have any relation to the models of the *Logical* level. It can be interpreted as an attempt to directly generate software infrastructure from *Conceptual* models, or a lack of alignment among these levels for modelling complementary concerns (from the perspective of an integral process of code generation based on the preservation of the main modelling constructs).
- Instead of competing initiatives, UML and BPMN could complement each other to model the *Conceptual* abstraction level and its associated view-points. BPMN can take advantage of the UML stereotype proposed in the Unified Process for covering complementary concerns at business levels, such as business goals, business rules, business workers, business key concepts, and entities.
- The *Time* viewpoint does not have explicit support. It is a complementary property of the modelling artifacts that were classified.
- The information associated to the *Who* in the taxonomic analysis does not allow the generation of specific user supports at lower levels (e.g., rules for access controls, users management, etc). Similar to the *Time* viewpoint, the information from modelling artifacts that satisfies the *Who* viewpoint relies on specific properties of modelling artifacts.

4.5.2 Quality analysis of the OO-Method and CA integration

In this section, we use the MMQEF method to evaluate the integration of two previous modelling methods that were generated from (and also supported by) the PROS Research Centre: the OO-Method (Pastor and Molina, 2007; Pastor et al., 2013), and the Communicational Analysis (CA) method (España et al., 2009; España Cubillo, 2012).

The OO-Method is a model-driven object-oriented software development method that generates complete applications from conceptual models. This method is a pioneer in the model-driven field, and one of the first proposals for obtaining software products from source conceptual schemas. The OO-Method uses a methodological process based on the object-orientation paradigm, in which its models (i.e., the object model, the dynamic model, the functional model, and the presentation model) contain the static, dynamic, and presentation properties of a modelled system. The OO-Method underlies the *Olivanova* framework, which defines a compiler of conceptual schemas and a model execution system. Currently, the OO-Method is technologically supported by the *Integranova* platform⁷.

CA is a communication-oriented requirements engineering method that is focused on the specification and modelling of the communicative messages and events that are associated to business process. It is possible to generate OO-Method models from these models. CA has an associated tool, the GREAT modeller (Rueda et al., 2015); this tool is based on Eclipse EMF and models business processes, their communicative events and messages, and generates OO-models through transformations of these communicative models.

The methodological integration of these two modelling methods was proposed in order to enrich the OO-Method with *requirement modelling* capabilities (González et al., 2011; Pastor and España, 2012). Thus, the OO-Method adds modelling support at the CIM level of the MDA specification (the Contextual and Conceptual levels of the reference taxonomy), complementing its native support for the PIM (Logical) and PSM (Physical) levels. This enrichment allows requirements and their associated transformations to conceptual schemes to be defined in an automated way. The integration of the OO-Method and CA is a clear demonstration of the *Requirements2Code* metaphor (Pastor et al., 2013), which uses a well-defined set of models and models transformations for this purpose.

Another feature that supports our analysis of the OO-Method and CA methodological integration is the evidence of previous reports where both methods were analyzed independently with the reference taxonomy. The resulting analyses were reported in (de la Vara et al., 2007) (for OO-Method) and (España Cubillo, 2012) (for CA).

To start the analytic procedure, we classify the modelling artifacts from both methods. These come mainly from their associated diagrams. Fig. 4.13 summarizes the application of the MMQEF method in this analysis. The inputs come from the conceptual specifications of the methods, their related publications, the previous individual classifications, and the information derived from their associated tools (Integranova and GREAT modeller, respectively).

Fig. 4.14 presents the obtained classification. Derived data from representations are placed in the cells in accordance with the information that can be captured by each one. Taking into account that the *Physical* level is covered by the code and infrastructure that is generated by the Integranova platform, the methodological integration has a coverage of 79.167% of the CIM-PIM-PSM levels, highlighting the coverage at CIM cells. The lattice of Fig. 4.15, which

⁷Tool available at <http://www.integranova.com/>.

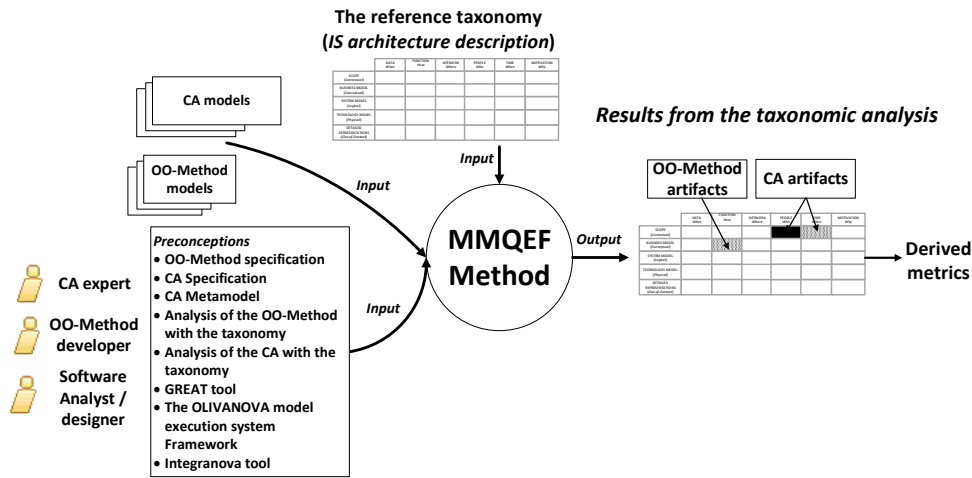


FIGURE 4.13: Summary of the application of the MMQEF method to the integration of the OO-Method and the CA methods.

MDA Abstraction levels		WHAT (Things)	HOW (processes)	WHERE (location)	WHO (people)	WHEN (time)	WHY (motivation)
CIM	Scope (Contextual)			CA: Organizational Network (Locations)	CA: Organizational units		CA: Organizational Goals (Business strategy)
	Enterprise Model (Conceptual)	CA: Business Object Glossary. CA: Message Structures (Analysis)	CA: Communicative Event Diagram CA: Message structures(Analysis)		CA: Organizational actors (Roles)	CA: Temporal Restrictions	CA: Business indicators (Communicative level)
PIM	System Model (Logical)	OO-Method: Functional Model OO-Method: Object Model OO-Method: Presentation Model CA: Message Structures (Design) CA: Logical Data Model.	OO-Method: Dynamic Model (State Transition Diagram, Object Interaction Diagram) OO-Method: Object Model OO-Method: Presentation Model CA: Message structures(Design) CA: Business Process Model (Physical events)		OO-Method: Presentation Model OO-Method: Object Model CA: Abstract Interface Model	OO-Method: Object Model	CA: Business indicators (Usage level) OO-Method: Presentation Model
PSM	Technology Model (Physical)	CA: Physical data model (platform specific schema)	CA: Business Logic Component Design		CA: Interface Component Design		CA: Business indicators (Operational level)
Physical	Detailed Representation (Detailed)						

FIGURE 4.14: Taxonomic analysis of the diagrams of the OO-Method and CA methods.

was generated automatically by EMAT, shows this coverage and the closeness semantics of the OO-Method and CA integration.

The integration of OO-Method and CA establishes a very complete modelling approach that models business concerns and guarantees their traceability to software platforms through the native mapping that is previously implemented in the OO-Method. This traceability takes attributes and properties belonging to the communicative messages and events and derives conceptual models at

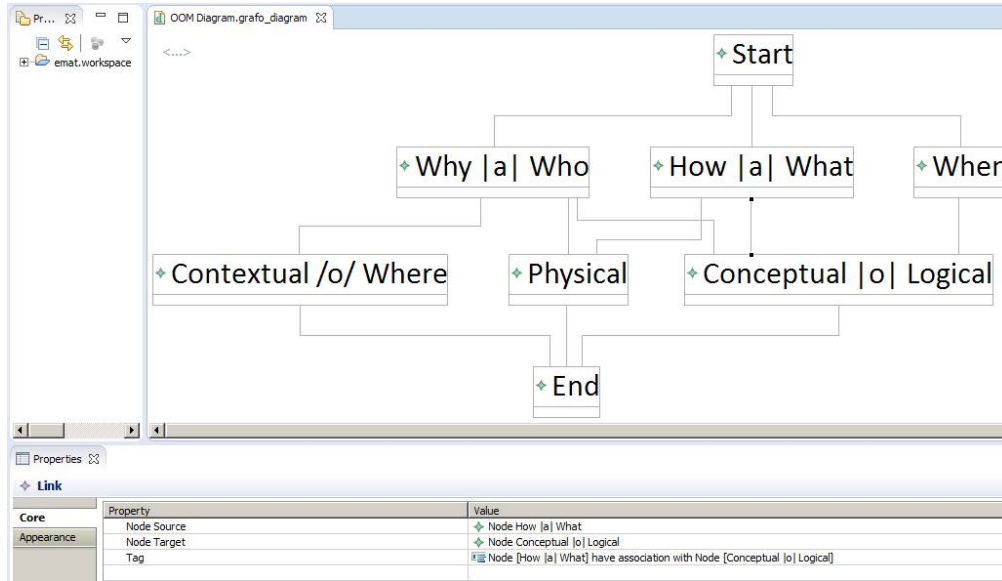


FIGURE 4.15: Lattice generated from the classification shown in Fig. 4.14.

the OO-Method level. Works such as (España et al., 2012) report previous applications of this derivation procedure. The traceability relations have the capacity to potentially support the generation of other OO-Method models such as the Presentation model.

However, the integration of the two modelling methods lacks coverage of the *Where* viewpoint, which is delegated to the specific constraints that are imposed by the code generation frameworks that are used by the technical environment of the OO-Method. However, taking advantage of the *Organizational network* diagram of the CA method, underlying models can be identified to address specific concerns associated to the *location* question. This would provide an interesting opportunity to model new IS phenomena regarding this viewpoint, such as service platforms, cloud deployment, micro-services, ubiquitous interfaces, etc.

4.6 Discussion

Currently, there are formal approaches for the management of semantics in MDE contexts (e.g., (Combemale et al., 2009; Boronat and Meseguer, 2008; Wolterink, 2009; Gargantini et al., 2009)). These proposals have a low adoption rate by model-driven practitioners due to their misalignment with the scenarios for applying model-driven initiatives (most of them are derived from traditional software development projects). We can also find an overload that affects MDE adoption because model-driven principles are reduced to concerns of the adoption of specific tools and their organizational deployment (Whittle et al., 2013).

Most of the current MDE challenges require methodological tools that are aligned with model-driven principles. Tools without conceptual support affect the adoption of MDE (which have been extensively reported). In addition to

technical support, methodological guidance is required in order to perform analytic reasoning on models.

One of the most relevant features of this guidance is the possibility to reason about *when* a given artifact is *model-driven compliant* (i.e., the assumption of whether or not a modelling artifact have a clear purpose for an IS concern, independently of notational justifications (e.g., a language profiling, any language with new graphical elements, and similar)).

The reference taxonomy provides a foundational architectural description of IS from an organizational perspective. It natively uses models to support all the issues (cells) that are required to fully understand an IS (from organizational levels to technical implementation levels). The reference taxonomy meets the conceptual model defined in the ISO 42010 standard (612, 2011) for architecture frameworks, architecture descriptions, architecture description languages, elements, correspondences, decisions, and rationale.

Following the ISO 42010 standard, the taxonomy provides an architecture description for IS and the associated architectural practices. We believe that there is a strong correlation between the model-driven paradigm and the IS principles that the taxonomy proposes. Therefore, the reference taxonomy allows the assessment of the current MDE quality challenges.

MMQEF provides the required support for the quality evaluation of modelling languages. This includes metrics for models and modelling languages, orientation derived from the analysis of the taxonomy, and information for making decisions for specific situations (e.g., a decision about the convenience or suitability of a specific modelling language). Some advantages of MMQEF are:

- The method is in compliance with recognized ontological frameworks and standards for IS due to the use of the reference taxonomy.
- An explicit and standardized *visual support* is provided to analyze modelling initiatives. This is similar to an ontological evaluation; however, instead of using a subjective conception of a domain, the method uses a universal IS architecture that has been accepted by the most recognized IS standards.
- It facilitates the formulation of metrics for the *coverage* of a model artifact or a modelling language in accordance with the cells involved in the different abstraction levels and modelling dimensions of the taxonomy. In addition, orientations and guidelines can be obtained based on to the associated reasoning and the intentions of specific modelling efforts or communities.
- This method gives conceptual and methodological support for evaluating the suitability of a modelling language in accordance with its purpose, the identified coverage, and the abstraction levels and modelling dimensions that are involved.
- The method also considers the information that can be obtained from the diagrams (e.g., as the resulting artifact from the notation and the

semantics of a modelling language) with respect to the intentions of use of the modelling artifact and the intentions of the modelling effort.

- The integration capacities that are offered by modelling languages can be explicitly identified.
- Support is obtained for the management of mappings, transformations, and the traceability relations between modelling artifacts. The taxonomy makes the type of the analyzed relation and the required specifications to support processes of models transformations explicit.
- The taxonomy defines the most granular elements of modelling (i.e., the minimal information that must be considered in a modelling effort). These granularity levels are related to the viewpoints (Henderson-Sellers and Gonzalez-Perez, 2010) (in this case, the philosophical questions).
- As a conceptual and methodological tool, MMQEF (through EMAT) provides a shared knowledge repository for reasoning, analyzing, and communicating quality issues on modeling artifacts and modelling languages, with their implications in a real IS project.

4.6.1 Why another quality framework for MDE?

Because MMQEF solves some open challenges regarding modelling language quality evaluation. In (Giraldo et al., 2015a) we listed some open pending challenges in the quality evaluation of modelling languages. MMQEF address each one of these challenges as follows:

- *Language/model according to MDE (MDE compliant)*: MMQEF allows to verify the purpose of the modelling languages under analysis by locating them into the specific cells of the reference taxonomy, and thus, identifying the association of the languages with the abstraction levels, the capacities of the languages to integrate with others, the support for models transformations, and the generation of concrete functional platforms.
- *Multiple modelling languages*: MMQEF evaluates the suitability of a set of modelling language to support specific IS phenomena. For each language under analysis, MMQEF determines the completeness, coverage and integration capacities provided by the languages.
- *Explicit management of abstraction levels*: the reference taxonomy that is used in the MMQEF method allows to explicitly consider the abstraction levels involved in an IS project, from the business to functional implementation level.
- *Metrics over models*: MMQEF defines five metrics with their associated decision criteria, to support reasoning about modelling languages previously classified in the reference taxonomy.

- *Models Transformations as a managed process*: in a models transformation process, the method allows to evaluate whether or not the chosen source-target modelling languages are appropriate.
- *Semantic in the diagram*: MMQEF allows to manage the artifacts that are associated to modelling languages, which result from the interaction of the users with the languages (i.e., the interaction with the models, the navigation through structures of related models, the simulation of expected behaviors, and queries over models).
- *Agile ontological analysis*: the method provides a precise and prescriptive set of task and activities to evaluate the quality on modelling languages.
- *The management of quality issues in modelling languages as technical debt evidence*: as it was reported in Section 4.4.5, we project to use the semantic models that are generated by the EMAT tool of the MMQEF method, as the input to calculate technical debt on modelling languages using previous services for the calculus and management of technical debt in software development environments.

Because It is not yet another framework, since it can be used in combination with previous approaches. MMQEF does not attempt to be another isolated framework; it can be used in combination with other frameworks. The analytic procedure (supported in the classification of modelling artifacts) allows quality dimensions and properties that are formulated in previous frameworks to be addressed. Most of these dimensions and properties are empirically evaluated with respect to the expectations and intentions of the authors of the frameworks or the analysts.

For example, Tables 4.2 and 4.3 (originally formulated in (Krogstie, 2012b)) present the support of MMQEF for the SEQUAL framework, which is one of the most important quality frameworks for the model-driven and model-based fields. These tables present a summary of the SEQUAL quality levels, goals, and means. These tables also contain the support of MMQEF to address these items of SEQUAL from our perspective and the descriptive precepts of SEQUAL. In the last column of two tables (MMQEF support), the color black indicates support of the MMQEF as a complete method, and light gray indicates support through the functions of the EMAT tool (previously described).

The reference taxonomy defines a conceptual artifact for reasoning about the application of modelling approaches to model IS concerns in an organizational context. MMQEF takes advantage of this reasoning to perform quality analytics at higher levels such as semantics, deontic, social, and pragmatics (including the understanding of both human and tool).

The quality analysis of MMQEF is framed within an organizational perspective. Therefore, the organizational adoption of modelling initiatives is also considered to be part of the quality evaluation assessment. Quality dimensions such as the goals of modelling, the explicit knowledge of the audience, and interpretations are easily addressed by MMQEF and reported in EMAT.

Other quality evaluation methods can also complement their analyses with the combined use of MMQEF and EMAT. For example, the quality goals proposed in the 6C approach (Mohagheghi et al., 2009b) (*Correctness, Completeness, Consistency, Comprehensibility, Confinement, and Changeability*) can be accurately expressed through their association with the taxonomic analysis and the metrics defined in MMQEF. Another example is the *Physics of Notations* (PoN) work (Moody, 2009), which can complement its notational scope with the semantic analysis of MMQEF. Thus, PoN cognitive principles such as the *Semantic transparency, Semiotic clarity, Cognitive integration, Cognitive fit, and Complexity management* can be easily analyzed with the support provided by MMQEF.

In addition, the MMQEF method proposes the EMAT tool to manage the data of the semantics that is derived from the taxonomic analysis. This is a clear advantage of MMQEF over other quality frameworks because they do not provide any concrete (native) tool to support their quality assessment.

TABLE 4.2: Support of the MMQEF method for the quality levels, goals, and means of the SEQUAL framework (I).

Quality type and characteristics	Means		MMQEF support	
	Beneficial existing quality	Model and language properties		Modelling techniques and tool support
Physical – internalizeability		Persistence	Database activities	
		Currency	Repository functionality	
		Availability		
Empirical – minimal error frequency	Physical	Comprehensibility appropriateness	Diagram layout	
		Aesthetics	Readability index	
			Refactoring	
Syntactic – syntactic correctness	Physical	Formal syntax	Structural metamodelling	
			Error prevention	
			Error detection	
			Error correction	
Semantic – validity	Physical	Domain appropriateness	Statement insertion	
		Participant appropriateness	Statement deletion	
Completeness	Syntactic	Modeller appropriateness	Behavioural meta-modelling	
		Language extension mechanisms	Meta-model adaptation	
		Formal semantics		
		Modifiability	Driving questions	
		Analyzability	Model reuse	
			Model testing and consistency checking	

TABLE 4.3: Support of the MMQEF method for the quality levels, goals, and means of the SEQUAL framework (II).

Quality type and characteristics	Means			MMQEF support	
	Beneficial existing quality	Model and language properties	Modelling techniques and tool support		
Pragmatic – comprehension	Physical	Operational semantics	Inspection		
	Empirical	Executability	Visualization		
	Syntactic		Modelling of intentions and other meta-data	Filtering	
Replusing					
Paraphrasing					
Explanation					
Execution					
Animation					
Perceived semantic-perceived validity	Physical	Variety	Simulation		
			Participant training		
Perceived completeness	Empirical Syntactic Pragmatic				
Social – agreement	Physical Perceived semantic	Inconsistency modelling	Model integration		
			Conflict resolution		
Deontic-goal validity	All	Traceability	Based on the specific type of modelling and goals of modelling		
			Tracedness	Adherence to standards	

4.7 Conclusions

In this chapter, we have presented an analytic procedure proposal for evaluating the quality of modelling languages and elements by the use of taxonomic analysis. The classification activity is used to answer when any modelling initiative is model-driven compliant (i.e., *when* is it in MDE) through its alignment with IS concerns defined in a reference taxonomy that is extracted from a recognized IS architecture description. With the FCA mathematical method, we demonstrated the formality of the reference taxonomy for supporting quality analytic procedures on modelling artifacts and managing the semantic data that is generated in these procedures. The EMAT tool for operationalizing these quality procedures was also reported.

Taking into account the feedback obtained from the preliminary validations, as further work, we are improving the tool by adding complementary visualization options to interpret the conceptual lattices obtained (e.g., radial graphs). In addition, we will populate the tool with more examples of taxonomic analysis in order to provide more precise guidance for potential users of the tool. The quality evaluation method by taxonomic analytics will be specified in more detail to demonstrate its advantages and its potential for MDE and IS communities and practitioners.

Chapter 5

Theoretical validation of the taxonomy used by MMQEF



One of the main inputs for the MMQEF method is the Zachman-based taxonomy. It was chosen because it works with conceptual models, relating conceptual things with representations on computers. In addition, the taxonomy provides a *reference architecture* for Information Systems (IS) by identifying the essential elements in a holistic system which will be deployed in an enterprise. The Zachman framework established the basis for (and influenced) current relevant standards such as ISO 42010 (Software and Systems Architecture Descriptions) and frameworks for Enterprise Architecture (EA). In addition, Zachman is compatible with the main terms and definitions described in the ISO 42010 specification for reference architectures.

Reference architectures provide reusable knowledge for managing specific purposes in the IS field. They guide the development of IS taking into account concerns from organizational to computational levels. Because of the *multiple stakeholder* feature of IS projects, multiple views and viewpoints are required with each one being addressed by one or more modelling language(s).

In MMQEF, quality evidence appears when the information of modelling languages is compared against the information that each cell in the taxonomy captures. These cells are a consequence of the abstraction level/viewpoint combination. The dependence of the MMQEF method with the taxonomy might be questionable, especially if other EA initiatives similar to Zachman propose a classification procedure. The EA scope is derived from the use of Zachman as one of the first holistic approaches to conduct an EA process (i.e., the analysis, design, planning, and implementation of an enterprise).

In order to justify using the Zachman-based taxonomy for the MMQEF method, three literature reviews were performed to identify the explicit application of taxonomies in the evaluation of information systems (Section 5.1), the potential use of other EA proposals as classification tools applicable to modelling languages (Section 5.2), and the previous works that use the taxonomy to classify modelling languages (Section 5.3), respectively. The reviews follow the protocol that was previously presented in Section 2.2.2 for the identification and classification of primary studies (i.e, the selection of sources, and the inclusion/exclusion criteria).

5.1 Taxonomies and Information Systems

MMQEF involves the taxonomic evaluation of IS. Taxonomies are conceptual tools for classifying elements in a specific context, in this case, the elements that come from modelling languages for building fully functional IS. The classification of objects and their relationships generates hypotheses that must be verified (Sokal, 1974). Classification is the initial step in the evaluation procedure proposed by MMQEF. It derives hypotheses about the purpose, scope, and use of modelling languages in the development of an IS.

Quality in the conceptual modelling field is not an unknown topic. It has been considered since the early stages of this paradigm, in which the key role that quality plays and the problems that are associated to its formulation are highlighted. For example, the authors in (Shanks and Darke, 1997) relate common (and current) problems in the definition of quality, such as its misunderstanding, specific purposes (regarding specific interests), unproductivity, lack of structure, and its description in the form of desirable features with overlapping properties and without an underlying theory.

In turn, it has been acknowledged that classification supports conceptual modelling due to its implications for the constructs used for modelling and the activity of creating models (Wand et al., 1995). Thus, quality issues for modelling languages, such as suitability, can be properly addressed by the organization of the knowledge about them. Works such as (Costagliola et al., 2002; Gemino and Wand, 2005) provide evidence of using taxonomic elements for reasoning about specific properties of modelling languages, even with the formulation of cognitive principles for guiding the classification as reported in (Parsons and Wand, 2008).

Classification and taxonomies have also been considered since the early stages of the IS field. Some relevant works on this topic consider languages as components of taxonomies for guiding the development of IS. A related work is proposed in (Lyytinen, 1987), in which the authors define a taxonomy of three main objects for systems contexts: technology, language, and organization. These objects are also considered in the taxonomy used by MMQEF. Other works in which elements of the reference taxonomy have been used are found in (Stockdale and Standing, 2006) for evaluation of IS using five of the six Zachman original classifiers (columns). Classifiers also have been used in (Prat et al., 2015) to organize a method for evaluation of artifacts in IS projects.

The MMQEF method specification, which includes the use a taxonomy for IS to derive quality issues of modelling languages, is compliant with (and allows determining) the taxonomic dimensions for the success of IS as previously defined in (DeLone and McLean, 1992). The method takes advantage of the classification schema for information entities that is provided by the Zachman taxonomy (Essien, 2015) and applies it to derive and manage quality issues that emerge in the information extracted from modelling languages. To address the lack of formalism in the IS evaluation process (as reported in (Kautz and Nagm, 2008)), MMQEF provides the formal support of the taxonomy for the management of quality issues.

A first review was performed to identify taxonomies that are used in the evaluation of IS with conceptual models and modelling languages. This search

was performed from April - September 2016. The search string was the following:

$$(\textit{Taxonomy} \wedge (\textit{evaluation} \vee \textit{evaluating}) \wedge \textit{information} \wedge \textit{systems})$$

A classification schema (Table 5.1) was applied to the selected studies to identify whether or not the identified taxonomies are based on the Zachman structure and whether or not they classify modelling languages. Table 5.2 presents the obtained results.

TABLE 5.1: Evaluation scheme applied to the identified studies to evaluate IS with taxonomies.

Question		Responses
Does the study mention a taxonomy to evaluate IS?		Yes, No
<i>If the study does</i>	<i>Is it the Zachman taxonomy?</i>	Yes, No
	<i>Is the taxonomy applied to modelling languages?</i>	Yes, No

TABLE 5.2: Summary of studies found and useful studies about taxonomies for evaluating IS (updated: September 2016).

Database	Found studies	Useful studies
Scopus	241	5
ScienceDirect	291	3
IEEE	294	4
Springer	260	3
Total	1086	15

Due to the high number of results for the use of the term *taxonomy* in this review, the selection of the papers was strict in order to focus on the studies that report an evaluation of IS by taxonomies. Therefore, fifteen studies were identified that initially met criteria of the search string. However, in the classification of the studies with the schema defined in Table 5.1, the following findings were observed:

- Only three works (Carter, 1986; Botchkarev and Andru, 2011; Prat et al., 2015) explicitly report a taxonomy for evaluating IS.
- In the works that report evaluation by taxonomy, none of them use the Zachman taxonomy.
- None of the fifteen studies report the evaluation of modelling languages as part of the evaluation of IS.

These findings show that taxonomies in IS are mostly used to promote specific interests. There is not a widely accepted taxonomy for IS. Instead, most of the taxonomies are specifically proposed in accordance with the purposes of the

authors. Despite the key role of modelling languages and conceptual models in the development of IS, they are not considered in the evaluation of IS. Therefore, the selection of the Zachman bi-dimensional taxonomy provides MMQEF with important support for covering all issues of IS since this taxonomy is also a reference architecture for IS that is integrally aligned with the foundational concepts defined in ISO 42010.

5.2 Is the Zachman taxonomy the only one that classifies modelling languages?

Current EA frameworks have different goals and purposes. The authors in (Crowder et al., 2016) summarize some of these purposes in four EA frameworks: the Zachman framework, The Open Group Architecture Framework (TOGAF), the Department of Defense Architecture Framework (DoDAF), and the British Ministry of Defence Architecture Framework (MODAF). That work also highlights that the Zachman framework has a *taxonomic purpose*. These EA frameworks are applied (and/or adapted) according to organizational needs.

Due to the implicit association of the Zachman taxonomy with the EA framework, another systematic review process was required to identify classification mechanisms in information systems analysis that can be applicable to modelling languages. This review was performed between June 2015 - September 2016. It verifies whether or not other EA frameworks can be used to classify information that is applicable to modelling languages, i.e., if there are other proposals framed as EA frameworks that have a classification structure similar to Zachman and support the classification of modelling languages. Fig. 5.1 summarizes the applied literature review protocol for this search.

Because of the many proposals of EA frameworks, a refinement process was applied to select those frameworks that were relevant to the review. To do this, some reference works were used (Table 5.3) to delimit the EA frameworks that were queried in the review process. As a result, the EA frameworks used in the review were restricted to: Zachman, Geram, Feaf, and Togaf. In addition, the RM-ODP framework was added because it is considered to be an ISO standard (ISO, 1998)¹.

The research question that addressed this review is as follows: *Are there other classification mechanisms for information of modelling languages that are derived from EA proposals and different from the Zachman framework?* The following search string is derived from this question: all logical combinations are valid to identify related works about quality in the model-driven contexts. Table 5.4 summarizes the obtained results from this search.

$$\begin{aligned} & ((Zachman \vee GERAM \vee FEAF \vee TOGAF \vee RM - ODP) \\ & \quad \wedge (Taxonomy \vee classifiers)) \vee \\ & (Enterprise \wedge architecture \wedge (taxonomy \vee (taxonomy \wedge classifiers))) \end{aligned}$$

¹Geram is also considered by ISO in the 15704 standard (ISO, 2000).

TABLE 5.3: Reference works used to limit the EA frameworks for this review.

Reference work	Year	EA Frameworks
(Krogstie, 2012a)	2012	Zachman framework, Generalised Enterprise Reference Architecture and Methodology (GERAM) , Architecture of Integrated Information Systems (ARIS), TOGAF, DoDAF
(Sessions, 2007)	2007	TOGAF, Zachman, the Federal Enterprise Architecture Framework (FEAF).
(Al-Nasrawi and Ibrahim, 2013)	2013	TOGAF, FEAF, Gartner framework.
(Leist and Zellner, 2006)	2006	Zachman, TOGAF, FEAF, ARIS.
(Tupper, 2011)		Zachman, TOGAF, FEAF, Gartner framework.

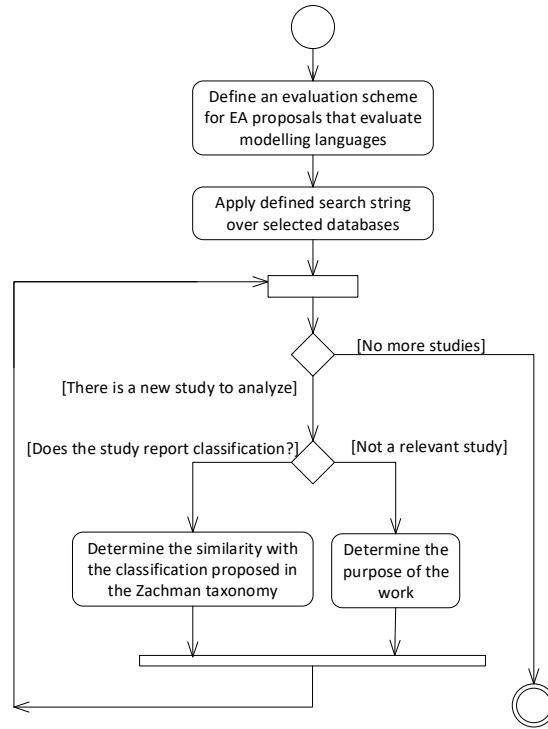


FIGURE 5.1: Summary of the systematic review protocol performed.

5.2.1 Results

TABLE 5.4: Summary of studies found and useful studies (updated: September 2016).

Database	Found studies	Useful studies
Scopus	99	10
ScienceDirect	367	15
IEEE	47	7
Springer	252	7
ACM	330	8
Total	1095	47

This systematic literature review began with 1095 papers and ultimately focused on 47 papers. This was due to the high number of studies that are related to the search terms (i.e., the names of the EA frameworks and the term *taxonomy*). Afterwards, we performed a comparative analysis according to the protocol depicted in Fig. 5.1 to determine whether or not the primary studies classify information from modelling languages. From this analysis, we obtained 7 studies (14.89% of the studies) that explicitly report classification. Tables 5.5 and 5.6 summarize the studies detected.

From these seven studies, a relevant contribution was found in (Whitman et al., 2001). This study takes advantage of the DoDAF framework to compare and evaluate modelling languages as part of the exploration of executable architectures, which are framed into three dimension: DoDAF elements, the formality of the modelling, and the implementations that can be achieved from the modelling language under analysis. A subset of eighteen DoDAF views were used to evaluate the languages. The authors make the evaluation operational through a table in which the information of modelling languages (either models, diagrams, or metamodel concepts) is evaluated against the *operational executable architecture applicability* criteria previously used in DoDAF. Unlike the MMEQF method, the evaluation of modelling languages that is proposed in this study is limited to an executable scope that contains three viewpoints: data, operational, and services.

On the other hand, forty studies (85.10%) were discarded since the scope of the reported classification is on specific scenarios (not modelling languages), such as networks, bootnets, and software architecture reconstruction. Other discarded studies focus on specific approaches such as Archimate. However, from these discarded studies, seven works (17.5%) reference the Zachman framework, and six studies (12.76%) make use of this framework (either the classifiers and/or the abstractions).

Despite the search string, the taxonomy proposed in the Zachman framework is still an important reference for the classification of modelling languages and their associated information. Excluding the work found in (Whitman et al., 2001), no other EA frameworks were found for classifying modelling languages. Of the seven primary studies:

- Four works make exclusive use of the Zachman taxonomy for performing classification, and one study references the Zachman framework.
- Four studies use the bi-dimensional taxonomic structure provided by Zachman, and three studies use other taxonomic structures that are derived from Zachman.
- None of the studies define rules for performing classification or any specific criteria to classify modelling elements.

TABLE 5.5: Summary of the primary studies that report EA frameworks with classification of modelling language information (Part I).

Study	Purpose	What is classified?	Relevant findings
(Al-Nasrawi and Ibrahim, 2013)	To propose a comparative evaluation and selection of enterprise architecture frameworks based on Zachman mapping and the development of the lifecycle (SDLC) software, in order to provide guidelines for the selection of an EA.	EA frameworks (and derived artifacts)	The Zachman taxonomy is used to execute the classification
(Nogueira et al., 2013)	To propose a new methodology based on action-research for the implementation of the business, system and technology models of the Zachman framework to assist and facilitate its implementation.	Modelling languages (expressed as models and diagrams)	The classification was done by using the Zachman classifiers
(Kingston and Macintosh, 2000)	To model multiple perspectives at different levels of detail in order to represent knowledge assets of the organization without loss of information.	Modelling techniques	Zachman is used as a simple and logical structure of descriptive representations for the identification of models, which are the basis for the design of the company and for the construction of the company's systems.

TABLE 5.6: Summary of the primary studies that report EA frameworks with classification of modelling language information (Part II).

Study	Purpose	What is classified?	Relevant findings
(Whitman et al., 2001)	To propose a methodology for the classification of enterprise models	EA models (from AE frameworks)	The Zachman taxonomy was taken as a reference architecture to create own classifiers and make a comparison and understanding of enterprise models.
(Shuman, 2010)	To describe a method to explore executable architectures in the context of a well-established architecture (the DoDAF Framework)	IDEF, UML, SysML, and BPMN	
(Lim et al., 1997)	To propose an Enterprise Architecture (EA) taxonomy to support research in the use of EA Frameworks (EAFs) for addressing complexity in enterprise design.	Enterprise models	The classification of information in the Enterprise Integration Modelling context is performed by the classifiers defined in the Zachman taxonomy
(Salih Abraham, 2015)	The Zachman taxonomy was used to guide the development of an Ambient Intelligence Healthcare Monitoring Information Architecture.	UML, ER	

In conclusion, even though a classification mechanism was identified, the applicability of this mechanism was limited regarding the number of works that reported the application of the Zachman taxonomy on modelling languages.

5.3 The previous use of the taxonomy to classify modelling languages

Another key feature of the reference taxonomy used by the MMQEF method is the existence of previous reports about classification of modelling languages. A third review was performed between March-September 2016 to find works that explicitly show the use of the Zachman taxonomy in the classification of elements from modelling languages. Fig. 5.2 summarizes the review protocol applied. For this review, the search string was defined as follows:

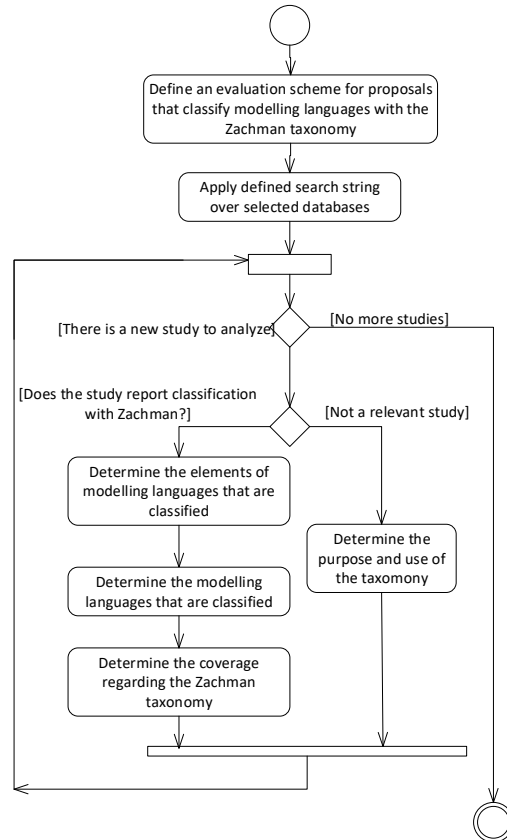
$$(\textit{Zachman} \wedge \textit{modelling languages} \wedge \textit{classification})$$


FIGURE 5.2: Summary of the systematic review protocol performed.

TABLE 5.7: Summary of studies found and useful studies (updated: September 2016).

Database	Found studies	Useful studies
Scopus	153	16
ScienceDirect	80	6
IEEE	21	4
Springer	25	1
ACM	1	0
Total	280	27

TABLE 5.8: Summary of the studies that report the classification of modelling languages with the Zachman taxonomy.

ID	Study	Which languages?	Coverage(which cells are used?)
1	(Salih and Abraham, 2015)	UML (use case, sequence, deployment), ER, DSL for software architecture	All
2	(Mrdalj and Jovanovic, 2005)	UML (deployment, use case, class, package, component, state, communication, sequence)	All
3	(Abu-El Seoud and Klischewski, 2015)	UML (use cases, activity), EPC, ER	Scope, model, System model, and What, How, Who, When, Why columns
4	(Kudryavtsev et al., 2013)	ER, UML (class diagram), BPMN	All
5	(Wang et al., 2013)	Data model, Process model, Network model, Workflow model, Motivation model, Dynamic model, UML, BPMN	All
6	(Liao et al., 2015a)	Class Diagrams, Logical Data Models, process model	All
7	(Kong et al., 2006)	User Sitemap Model, BPM, ER, User Interface Flow Model	What, How, and Where columns; planner and owner rows
8	(Chung et al., 2009)	Modelling languages that meet the phases and views defined by the authors	When, Who, Where, What columns
9	(Florez et al., 2016)	Analysis methods	All
10	(Vargas et al., 2015)	Data Model, Analysis Model, Domain model, Stakeholders, Decision model	All
11	(Kingston and Macintosh, 2000)	UML (class diagram), ER, charts (PERTm GANTT), QOC	All
12	(Baïna et al., 2009)	Holon models, UML (Class diagram), ER, SQL	All
13	(Ouertani et al., 2011)	UML (Class diagram), SQL	All
14	(Utz et al., 2011)	BPMS, EPC, GORE, BSC, OWL, XTM, UML, OWLS,	All
15	(Ostadzadeh et al., 2007)	CWM, UML, BPDm, OCL, EDOC, Timing Diafram, PRR, BSBR, BMM	All
16	(Letsholo et al., 2012)	BPMN, PN, RAD, UML, DFD, EPC	All

As depicted in Table 5.7, the search string produces 280 works, of which the focus was over 27 primary studies that reported the Zachman taxonomy. Sixteen works (59.26%) explicitly report classification using this taxonomy. The remaining works refer to taxonomies with other purposes; one work (3.70%) proposes an extension of the Zachman framework, and four works (14.81%) report the application of the Zachman taxonomy for purposes that differ from classification tasks.

Of the 16 Zachman-classification studies, two works (12.5%) report the classification of modelling languages, 13 works (81.25%) report the classification of models, and 12 works (75%) report the classification of diagrams. Although representations are properties of modelling languages that allow models to be expressed, it was preferable to separate these findings based on the information reported by each proposal. One work (6.25%) reports the classification of complementary elements for modelling languages (in this case, analysis methods for enterprise models). Table 5.8 presents the modelling languages that were classified by the sixteen studies as well as the coverage of the taxonomy that was used for each work.

5.4 Conclusions

From the reported evidence it can be deduced that the taxonomic system of the Zachman framework provides adequate support for classifying modelling languages that support the development of information systems. The combination of the bi-dimensional structure, the visual language, and the rules for performing classification provide a starting point to identify issues in the use, scope, and purpose of modelling languages. In this way, MDE phenomena can be explicitly addressed by the classification of the taxonomy and the posterior activities that MMQEF defines.

Chapter 6

Empirical Evaluation of MMQEF



This chapter evaluates the applicability of the MMQEF method in comparison with other existing methods. We performed an evaluation in which the subjects had to detect quality issues in modelling languages. Two groups of subjects (expert professionals and undergraduate students) and two experimental objects (i.e., two combinations of different modelling languages based on real industrial practices) were used. To analyze the results, we applied quantitative approaches (i.e., statistical tests on the results of the performance measures and the perception of subjects) and qualitative approaches (i.e., the analysis of the comments that subjects freely expressed about the MMQEF method). We used the guidelines reported in (Wohlin et al., 2012c) and (Jedlitschka et al., 2008) to

design and report the validation procedure and its results.

We ran four replications of the procedure in Colombia, with a total of 66 students and 32 professionals over nine months in 2016. The results of the quantitative analysis show a low performance for all of the methods, but a positive perception of MMQEF. The results of the qualitative analysis provide additional evidence that the subjects found MMQEF useful. The application of modelling language quality evaluation methods within MDE settings is indeed tricky, and subjects did not succeed in identifying all quality problems. This empirical validation paves the way for additional investigation on the trade-offs between the methods and potential situational guidelines (i.e., circumstances under which each method is convenient). We encourage further inquiries on industrial applications to incrementally improve the method and tailor it to the needs of professionals working in real industrial environments.

6.1 Introduction

In model-driven engineering (MDE) methods, since *models* are the main artifact, the quality of the modelling languages is important for MDE environments. There are methods to evaluate the quality of modelling languages, but when they are applied in MDE environments some problems appear due to the features

of these environments. This includes multiple modelling languages are used together, model transformations add a level of complexity to the evaluation of quality because the suitability of the source and target languages must be ensured, and the code generation requires quality requirements of modelling languages and models (e.g., the construct deficit in one language or incomplete models may have an effect on the resulting source code).

There is great diversity among researchers about the definition of quality in MDE contexts (Giraldo et al., 2014). Since quality is not a concrete manifestation (or physical component) of modelling artifacts, there are many facets or levels of quality. The different meanings of quality have not been acknowledged, which is an open problem. Some of the identified quality definitions have associated frameworks for evaluating quality issues. These frameworks present specific validation procedures for demonstrating their applicability and utility in accordance with the given quality definitions.

Since 2013, we have performed a systematic review in order to identify the main trends in the definition of quality in MDE contexts (Giraldo et al., 2016b). To date, there are twenty-nine works that provide an explicit definition of quality; of these, only nine works have an associated validation procedure (Challenger et al., 2015; Espinilla et al., 2011; Grobshtein and Dori, 2011; Hindawi et al., 2009; Lange and Chaudron, 2005; Le Pallec and Dupuy-Chessa, 2013; López-Fernández et al., 2014; Maes and Poels, 2007; Merilina, 2005). Using the classification of evaluation techniques that were previously proposed in (Siau and Rossi, 1998) on the nine identified works, we found five works that reported laboratory experiments, two works that used metrics, and other individual works that used survey, metamodelling, and case studies. The laboratory experiments use different evaluation procedures with participants, surveys, and implementation of software tools to demonstrate the applicability of the work that is proposed. The only common aspect in these experiments is their specificity with regard to the scope of each quality definition.

This chapter presents the design and results of a validation procedure that was performed in 2016 (over nine months) to evaluate the applicability and use of the MMQEF method, which is an approach for evaluating quality issues in modelling languages in MDE projects. In accordance with (Siau and Rossi, 1998), which defines some approaches for evaluating information modelling methods, this procedure can be considered to be an *empirical evaluation technique*. The design of the empirical validation was done in accordance with the guidelines described in (Wohlin et al., 2012a) for experimentation in software engineering scenarios.

This chapter makes the following contributions:

- The first application of the MMQEF method by (model-driven) practitioners is reported.
- Since MMQEF is an alternative proposal for evaluating quality issues in MDE projects, its applicability was verified by contrasting it with other quality frameworks for MDE identified by us.

6.2 Background

6.2.1 Problem Statement

The validation of quality methods for MDE contexts, such as the 6C Goals, PoN, and SEQUAL, have challenges. To detect validation procedures for these frameworks, specific publications of the authors about the applicability of the frameworks must be accessed separately. Examples of validation procedures can be found in (Heggset et al., 2015). Some approaches employed in validation are experiments, surveys, interviews, and questionnaires.

Besides the specific modelling scenarios that are required to demonstrate the application of the quality methods for MDE, the quality methods are not directly comparable with each other. Although it is true that the quality frameworks could be similar theories for Information Systems (Gregor, 2006), they differ in their purposes, scope, and procedures for the identification of quality issues. Reported validations present individual applications of the quality frameworks.

6.2.2 Research objective

Using the template defined in (Wohlin et al., 2012c) for the definition of goals in experimentation processes, the main purpose of this validation is described as follows:

Analyze the MMQEF method
for the purpose of characterizing it
with respect to its applicability for finding quality issues for modelling languages
from the point of view of the researcher
in the context of professional experts and undergraduate students analyzing a scenario for the application of multiple modelling languages.

6.3 Design of the empirical validation

6.3.1 Context

The empirical validation was formulated to identify the degree of applicability of the MMQEF method for evaluating quality in MDE contexts, specifically modelling languages as the main artifact of this paradigm. This evaluation also considers other quality methods that are formulated in the MDE literature so that MMQEF could be applied in similar conditions of practice, taking into account that the population of the empirical validation had no previous experience with quality methods for MDE.

A group of participants in Colombia (Spanish-speaking participants) applied the MMQEF method in a model-driven scenario that we had defined beforehand. Two kinds of participants were considered: professional experts, and undergraduate students in software engineering.

For the professionals, the scenario was an organizational application of models until the generation of software for critical business processes. This organizational modelling scenario was previously reported in (Giraldo et al., 2016b),

and it uses UML, BPMN, SPEM, and Flowchart as the modelling languages under analysis.

For the undergraduate students, the scenario was an application of the 4EM method (Sandkuhl et al., 2014b) for enterprise modelling, specifically a variant of the method proposed in the CaaS European project (Stirna, 2013; Bērziša et al., 2015).

During the validation, the participants were asked to find quality evidence for modelling languages that are jointly applied to model an IS project. To do this, the participants used MMQEF and another quality framework for MDE which was freely chosen by each one of them.

Prior to the validation, we identified some quality issues for both scenarios. In the scenario that was proposed for the professionals, we took advantage of a *post-mortem* analysis performed by the researchers who led the project. These researchers were given roles such as the domain expert, the modelling-data leader, and the software engineering leader. For the 4EM scenario, the quality issues were derived from a list that was proposed by the teacher of the software engineering course in which the participant students were enrolled. For the list, the previous knowledge of the students about modelling (i.e., knowledge of UML, MDE, and implementation of model-driven editors in Eclipse EMF) was considered. The didactical goals of the capability lesson for undergraduate students were also considered due to the modelling alternatives that students had for modelling system and software concepts involved in the CDD approach.

Tables 6.1 and 6.2 summarize some of the quality issues for the validation that were expected to be found by the professionals and students, respectively. These lists are not exclusive (i.e., other issues could be reported by the participants).

6.3.2 Experimental units

We used a *convenience sampling* approach to select the participants, who were contacted and invited based on their homogeneous knowledge and condition (academic or professional) for applying quality frameworks.

For the procedure with the professionals, there was a total of thirty-two participants (master's students and professionals) with previous knowledge of MDE and model-driven environments such as Eclipse EMF/GMF. The professional experts came from several software development companies; they had experience in roles such as software architects, software project managers, and senior software developers. They have expertise in software development projects and are currently working in software development companies.

In the group of professionals, we included a group of post-graduate students who are involved in a Software Engineering Master's program. The students were in a Master's course about *Domain-specific languages (DSL) design and implementation*. In addition, those students had previously taken two courses in MDE (introduction to MDE and Applied MDE). The professionals were contacted by email and they voluntarily accepted to participate in the validation.

For the procedure with the undergraduate students, there was a total of sixty-six participants. The validation was performed as part of a lesson of an undergraduate course in Software Engineering. It is an advanced course on this

TABLE 6.1: Example of quality issues expected to be reported by the professionals who participated in the empirical validation.

ID	Quality Issue	Derived from
QIPR01	There was no traceability from models to code.	MMQEF
QIPR02	There was no automatic code generation.	MMQEF
QIPR03	Decoupling between organizational modelling and system modelling.	MMQEF
QIPR04	Misalignment between the modelling languages used and the purposes of modelling.	MMQEF
QIPR05	Excessive stereotyping of the UML modelling language.	MMQEF
QIPR06	Excessive adaptation of languages to model business concerns.	MMQEF, SE-QUAL
QIPR07	Lack of suitability analysis of modelling languages.	MMQEF
QIPR08	Lack of coverage of modelling languages. Some IS issues were not covered by modelling languages (e.g., data, interaction, architectural decisions).	MMQEF
QIPR09	There was no distinction of the purpose of the resulting models (i.e., there were models to communicate ideas, to automate the process, to make systems), but these purposes were not explicitly addressed.	MMQEF, SE-QUAL
QIPR10	There was no integration between modelling languages.	MMQEF
QIPR11	Poor support for deontic level (for organizational and system purposes).	MMQEF, SE-QUAL
QIPR12	Lack of adequate tool support.	MMQEF
QIPR13	Lack of expressiveness of the modelling languages.	6C, MMQEF, SE-QUAL
QIPR14	Lack of communication abilities for the performed modelling.	MMQEF

TABLE 6.2: Example of quality issues expected to be reported by the students who participated in the empirical validation.

ID	Quality Issue
QIST01	Lack of coverage of system and platform concerns.
QIST02	Crossing between business and code levels.
QIST03	Poor support for modelling system concerns.
QIST04	Other modelling language(s) could be appropriately used to model system concerns (e.g., UML).
QIST05	Other modelling languages could be more appropriate for modelling CDD concerns such as (technical) requirements.
QIST06	There is no traceability between high-low levels (including generated code)
QIST07	Poor support for modelling IS concerns such as locations and time.
QIST08	Overloading of symbols and connections to model semantic constructs.
QIST09	The only significant difference in the CDD modelling constructs is the color of the primitive graphics.

subject (the last course for these students). We took advantage of a specific subject of this course: capabilities and maturity models applied in software process improvement. We taught the management of capabilities using the 4EM method (Sandkuhl et al., 2014b), which considers six modelling approaches for designing and managing enterprise capabilities.

We used some artifacts generated in the CaaS project¹ to support the subject of capability. These artifacts included the specific adaptation of the 4EM method for the CaaS project (where the BPMN notation is used to model business processes) and for the Capability-driven Tool (CDT), which is an Eclipse-based editor for modelling capabilities based on the modified 4EM method of the CaaS project.

Previous works such as (Salman et al., 2015) suggest concerns about the suitability of undergraduate students for experimentation in software engineering. Therefore, in the selection of students for the validation, their suitability was justified by their previous knowledge. Thus, we took advantage of the convenience sampling by choosing students who had completed two previous courses in Software Engineering, where they took lessons about conceptual modelling, MDE, and the use of MDE frameworks such as Eclipse EMF for modelling the domain of software solutions and making their software engineering tools (editors). All of the selected students received the same instruction about capability modelling and management before the execution of validation.

For the design of the validation, we used a *Probability - Paired comparison design* to avoid the influence of the quality evaluation methods in MDE during their application by the participants. For both validations (professionals and students), the *paired comparison* was defined as presented in Table 6.3. For

¹<http://caas-project.eu/>

TABLE 6.3: Paired-comparison design for the validation of MMQEF.

<i>i</i> - participant	Treatment 1	Treatment 2
<i>Odd</i> participant	MMQEF	Other method
<i>Even</i> participant	Other method	MMQEF

their participation, the invited professionals and students were rewarded with free seminars and lunches/dinners.

Prior to the use of quality methods, the participants were asked about their previous knowledge about key terms for MMQEF, MDE, modelling languages, and DSLs (Tables 6.7 to 6.11) in order to determine whether or not their previous knowledge might eventually affect the application of the method. The set of terms used by MMQEF is not limited only to the application of the method. These terms are common concepts in the MDE terminology. Thus, the familiarity with these terms facilitates the performance of activities for evaluation of quality that are proposed in MMQEF. Previous knowledge and use of modelling languages and DSLs could also induce key MDE terms.

6.3.3 Experimental material

The objects that were used in the validation were the following:

- The slides of the seminar about *quality in MDE (modelling languages)*.
- A summary of the MMQEF, with the main blocks (components) of the method.
- A description about a modelling scenario where multiple modelling languages (≥ 2) are required for addressing specific IS concerns.
- A questionnaire for characterizing each participant.
- A questionnaire for reporting the quality issues that are identified by the participants (for both the alternative approach and the MMQEF treatment). This instrument contains questions related to the T, DoU, FQI, NQI, QID, and NI dependent variables (described in Table 6.6).
- A survey about the *Perceived ease of use* (PEU), *Perceived usefulness* (PU), and *Intention to use* (IU) for MMQEF. This survey was made using twelve sentences with answers on a Likert scale of 1 to 5 (1: Strongly disagree, 2: Disagree, 3: Neither agree nor disagree, 4: Agree, 5: Strongly agree). In the Likert statements, we used explicit leading phrases about the applicability of MMQEF in order to more easily generate agreement/disagreement opinions from the participants. Table 6.4 presents the Likert statements. These were arranged randomly.

The last three items of the package were grouped into a spreadsheet to facilitate the data collection from the participants.

TABLE 6.4: Likert sentences associated to MEM variables (the *Perceptions* and *Intentions* dimensions).

MEM Variable	ID	Likert question
PEU	L1	MMQEF is easy to understand.
PEU	L2	It is easy to use MMQEF to detect quality issues in modelling languages and models.
PEU	L3	MMQEF is useful for detecting quality issues in modelling languages and models.
IU	L4	You would use MMQEF in later scenarios of quality assessment in models and languages.
PU	L5	The use of MMQEF allows problems in Software Engineering and Information Systems to be addressed using conceptual models.
PU	L6	MMQEF is aligned with the principles of the MDE paradigm (i.e., quality is evaluated from the MDE perspective).
IU	L7	From this experience, the evidence of quality at the level of modelling languages and models will be important in your further Software Engineering and/or Information Systems projects.
PU	L8	MMQEF allows relevant considerations for addressing a project under the model-driven paradigm to be identified.
PEU	L9	MMQEF provides a practical method to identify quality issues in projects developed under the model-driven paradigm.
PU	L10	The taxonomy that is used in MMQEF supports the construction of an Information System using conceptual models, and it also considers the conceptual levels where models can be placed (e.g., from the <i>organization</i> level to the <i>implementation</i> and <i>deployment</i> levels).
PU	L11	The classification of modelling languages and modelling elements is useful in finding quality issues in model-driven projects.
PU	L12	The inferences proposed by MMQEF contribute to identifying quality issues in MDE projects.

participants in accordance with their availability for the validation. The sessions were as follows:

S1: session 1, with professionals (24 participants, April-June 2016).

S2: session 2, with students (46 participants, May 2016).

S3: session 3, with students (20 participants, November 2016).

S4: session 4, with professionals (8 participants, November-December 2016).

Each session was performed in an academic location to facilitate the access of the participants to scientific databases and other resources required in the evaluation of quality for the modelling scenarios proposed in Section 6.3.1. This location also facilitated the face-to-face support between researchers and participants. Risks about situations that could affect the validation were successfully addressed by researchers. All of the participants performed the validation under the same conditions (a computing lab, internet access, modelling tools, and all of the experimental material described in Section 6.3.3).

6.3.5 Hypotheses, parameters, and variables

The following hypotheses were defined for the empirical validation:

H_0 : Compared to alternative methods for evaluating quality in MDE (e.g., SEQUAL, PoN, and 6C Goals), participants do not perceive the applicability of the MMQEF method to find quality issues in MDE projects.

H_a : The applicability of MMQEF is perceived by the users of the method.

TABLE 6.5: Expected treatments for the independent variable.

Factor	Treatments
Method	Application of any other approach
	Application of MMQEF

The independent variable that was defined for the validation was the *method* (its application) to evaluate quality in MDE contexts. Table 6.5 presents the variable with its possible associated values. The other approach could be one of the following options: the 6C Goals, the Physics of Notations (PoN), the SEQUAL framework, or any personal criteria applied by the subject to perform an evaluation procedure. The first three frameworks were taught as part of the seminar that we provided.

Some dependent variables were identified. Table 6.6 describes the variables, where the first six variables were obtained from the application of each method according to the treatments defined in Table 6.3. The fourth and fifth variables refer to *inferences*, i.e., conclusion(s) that participants eventually might deduce when they applied a quality method (e.g., modelling language A is better than modelling language B for addressing a system concern). The last three variables were deduced from the *Perceptions* and *Intentions* dimensions of the Method Evaluation model (MEM) for IS evaluation methods (Moody, 2003).

TABLE 6.6: Dependent variables identified for the validation.

Description	Acronym	Possible Values
Time of application of the method	T	[0 ...)
Degree of understanding of the method	DoU	[0 ... 100]
Were quality issues found?	FQI	YES / NO
Number of quality issues derived by each subject from the proposed scenario	NQI	[0 ...)
Were quality inferences deduced?	QID	YES / NO
Number of <i>inferences</i> deduced from the application of the method	NI	[0 ...)
MEM Perceived Ease Of Use for MMQEF		[1 ... 5]
MEM Perceived Usefulness for MMQEF		[1 ... 5]
MEM Intention To Use for MMQEF		[1 ... 5]

6.3.6 Analysis procedure

The *Sign test* approach was used to analyze H_0 and H_a , (see Section 6.4.2). To analyze the hypotheses, we assume that the behavior of the independent variable (*application of the method*) has a *binomial distribution* because the following four conditions were present:

- The number of observations was fixed (for both validations).
- Each observation was independent.
- Each observation represented one of two possible outcomes (*success* or *failure*).
- The probability of *success* (p) was the same for each outcome.

The distribution is described as $X \sim B(n, P)$ where :

$$\begin{aligned} n &= \text{number of analyzed observations} \\ X &= \text{number of successful (or positive) answers.} \end{aligned}$$

For this analysis, $P = 0.5$, and $p - \text{value} = 0.05$. The null hypothesis is accepted when ($P \leq 0.5$); otherwise ($P > 0.5$), it is rejected. Assuming that quality methods can be comparable through the T, DoU, FQI, NQI, QID, and NI dependent variables, we tested H_0 and H_a with $X \sim B(n, P)$ and $P(X \geq x)$.

Each distribution has its associated dependent variable and the respective values for computing it, including the obtained x value (positive answers). Successful or positive answers (x value) were detected for the cases when the subject assigned a positive response to the MMQEF method in each one of the six related variables:

- $T_{MMQEF} < T_{AnotherMethod}$
- $DoU_{MMQEF} > DoU_{AnotherMethod}$

- $FQI_{MMQEF} = YES \wedge FQI_{AnotherMethod} = NO$
- $NQI_{MMQEF} > NQI_{AnotherMethod}$
- $QID_{MMQEF} = YES \wedge QID_{Another} = NO$
- $NI_{MMQEF} > NI_{AnotherMethod}$

Negative answers were those in which MMQEF received a low or equal score (tie). Supporting distribution outputs were generated with the *Probability Distribution* app from the University of Iowa².

In addition, a direct analysis with the Likert values obtained was used to analyze the MEM variables (see Section 6.4.2)

6.3.7 Execution - deviations

For sessions *S3* and *S4* (Section 6.3.4), we introduced an additional question for the participants in order to determine the reason(s) why they selected the quality method that was employed in the empirical validation as an alternative to MMQEF. We provided some non-exclusive reasons for this question:

- The *ease of use* of the approach.
- The examples found that are formulated for the approach.
- The *suitability* of the approach.
- The documentation of the approach.
- *I think it is the best option.*
- *The approach can be performed in the allotted time (3 hours).*
- *No particular reason. The choice was random.*
- Another unspecified reason.

The participants that were previously in sessions *S1* and *S2* were also contacted by email in order to ask them the same question.

6.4 Analysis

6.4.1 Descriptive analysis

Before the interaction with MMQEF and the other quality evaluation frameworks, the participants were asked whether they knew some key terms that are employed in the MMQEF method. If an affirmative answer about knowledge of these terms was given, the participants were also asked about the application of these concepts in their immediate contexts (professional and academic).

Table 6.7 summarizes the percentages for the knowledge and application of the terms involved. These percentages indicate a positive trend, which could

²Tool available at <http://homepage.stat.uiowa.edu/~mbognar/>

infer an appropriate use of MMQEF due to the importance of those terms in the methodological specification of the MMQEF method. In addition, for the same questions (knowledge and application), if the participants gave an affirmative answer, they were asked to indicate the associated level for both of them using a Likert scale from 1 to 5, with 1 indicating the lowest level of knowledge/application and 5 indicating the highest level. Tables 6.8 and 6.9 present the levels of knowledge/application obtained for professionals and students, respectively.

TABLE 6.7: Percentages of knowledge and application of MMQEF key terms.

MMQEF Key terms	Professionals (n=32)		Students (n=66)	
	Do you know it?	If you do, do you apply it in your professional tasks?	Do you know it?	If you do, do you apply it in your tasks?
MDD	78.13%	60.00%	68.18%	44.44%
MDA	62.50%	45.00%	40.91%	48.15%
MDE	62.50%	65.00%	48.48%	43.75%
Conceptual Modelling	78.13%	80.00%	84.85%	78.57%
Metamodelling	71.88%	52.17%	72.73%	50.00%
Model transformation	59.38%	63.16%	22.73%	73.33%
Transformation languages	59.38%	52.63%	24.24%	56.25%
Traceability	65.63%	80.95%	78.79%	67.31%
Abstraction level	81.25%	80.77%	69.70%	76.09%
Viewpoint	46.88%	73.33%	39.39%	73.08%
View	56.25%	66.67%	78.79%	67.31%
Model-driven technical environments	68.75%	68.18%	71.21%	53.19%
Model-driven tools	84.38%	70.37%	80.30%	75.47%

For both types of participants, there was a dramatic change in their initial perception of familiarity with the MMQEF key terms. While the percentages in Table 6.7 show a positive trend about knowledge and application, the associated levels in Tables 6.8 and 6.9 present moderate (and relatively low) behavior. This indicates a limited comprehension of the terms, which eventually affected the full understanding and use of the MMQEF method (see Section 6.4.2). In both populations, the concept of *Conceptual modelling* can be highlighted for its association with the data representation for relational databases.

TABLE 6.8: Resulting levels of knowledge and application of MMQEF terms for the participant professionals.

MMQEF key terms	Professionals (n=32)					
	Knowledge level			Application level		
	Mean	Median	Mode	Mean	Median	Mode
MDD	2.75	3.00	3.00	2.73	2.00	2.00
MDA	2.42	2.00	2.00	2.44	3.00	3.00
MDE	2.95	3.00	4.00	2.92	3.00	4.00
Conceptual Modelling	3.08	3.00	3.00	3.50	3.00	3.00
Metamodelling	2.82	3.00	3.00	3.33	3.00	3.00
Model transformation	2.50	2.50	1.00	2.58	2.50	2.00
Transformation languages	2.39	2.00	1.00	3.00	4.00	4.00
Traceability	3.20	3.00	4.00	2.94	3.00	3.00
Abstraction level	2.80	3.00	2.00	2.81	3.00	3.00
Viewpoint	2.73	3.00	4.00	3.27	4.00	4.00
View	2.94	3.00	3.00	3.08	3.00	3.00
Model-driven technical environments ³	2.81	3.00	4.00	3.33	4.00	4.00
Model-driven tools ⁴	2.77	3.00	2.00	3.00	3.00	4.00

TABLE 6.9: Resulting levels of knowledge and application of MMQEF terms for the participant students.

MMQEF key terms	Students (n=66)					
	Knowledge level			Application level		
	Mean	Median	Mode	Mean	Median	Mode
MDD	2.29	2.00	3.00	2.15	2.00	2.00
MDA	2.26	2.00	3.00	2.46	3.00	3.00
MDE	2.09	2.00	1.00	2.36	2.00	1.00
Conceptual Modelling	3.20	3.50	4.00	3.16	3.00	3.00
Metamodelling	2.50	2.50	3.00	2.67	3.00	3.00
Model transformation	1.87	1.00	1.00	2.09	2.00	1.00
Transformation languages	1.81	1.50	1.00	1.78	2.00	1.00
Traceability	2.67	3.00	3.00	2.63	3.00	3.00
Abstraction level	2.98	3.00	3.00	3.03	3.00	3.00
Viewpoint	2.85	3.00	2.00	3.05	3.00	3.00
View	3.04	3.00	3.00	3.40	3.00	3.00
Model-driven technical environments	2.64	3.00	3.00	2.64	3.00	3.00
Model-driven tools	2.96	3.00	3.00	3.18	3.00	4.00

After the characterization of the MMQEF terms, the participants were asked about their knowledge and use of modelling languages and domain-specific languages (DSLs). They were also asked about their intention to use these two approaches in their immediate contexts. Tables 6.10 and 6.11 present the results for the professionals and the students, respectively.

The population in both validations reported knowledge of modelling languages (96.88% of the professionals and 100% of the students). UML was the most popular modelling language in both populations (90.32% of the professionals and 98.48% of the students). For the professionals, BPMN was the second modelling alternative (67.74%), and other specific alternatives appeared such as ER (9.68%), i* (6.45%), Communication Analysis (España et al., 2009) (6.45%), SysML, CTT, Flowchart and AADL (each with a percentage of 3.23%). For the students, the ER diagram was the second preferred modelling alternative (28.79% of the population) followed by BPMN (27.27%), the 4EM method associated with the *capabilities management* lesson (21.21%), and SysML (13.64%).

SQL was the DSL that was most known by both populations (58.33% of the professionals and 88.46% of the students). For the professionals, XML was the most known technical DSL (12.50%). Individually, some professionals reported their knowledge of DSLs, such as HTML, R, MPI, and VHDL. The reported technical DSLs for the students were HTML (47.83%), XML (23.08%), and CSS (21.15%).

The above percentages are a consequence of the *convenience sampling* approach that was applied to select the participants. Despite their lack of background in the key terms, the familiarity of the participants with modelling languages made them appropriate participants for discussing quality issues in modelling languages and the posterior application of quality evaluation frameworks.

Although it is true that the participants reported their knowledge and use of DSLs, there was an important trend regarding technical languages that are commonly employed in software development projects. Two professionals reported one non-technical DSL; one used the EERM approach (Do Nascimento Fidalgo et al., 2012), and the other used the DataForm approach (Giraldo et al., 2015c). This clearly indicates a trend in the professional participants for using cognitive tools that allow them to address complexity at the source-code level. No non-technical DSLs were reported by the students despite the fact that they had the knowledge to design and make languages with tools for MDE based on the Eclipse platform.

6.4.2 Testing of hypotheses

First, we detected an important tendency of the participants to choose a quality evaluation method with more prescriptive information and orientation about tasks, steps, procedures, etc., to perform the quality evaluation. A total of 46.875% of the professionals chose the 6C Goals, and 34.375% chose the PoN. Only one professional reported the use of the SEQUAL framework. However, in

³This refers to frameworks such as Eclipse EMF, MetaEdit+, Microsoft Visual DSL Tools, or other frameworks, which allow creating and managing modelling languages.

⁴This refers to specific tools that result from the use of a MDE framework, e.g., an Eclipse-based editor for a specific language that is created from the Eclipse EMF-GMF projects.

TABLE 6.10: Information about knowledge and use of modelling languages and DSLs from the professionals.

Element	Questions	Participants with YES answer	%		
<i>Modelling languages</i>	Do you know modelling languages?	31	96.88%		
	If you do, do you use them?	28	90.32%		
	If you do, what do you use them for?	Options	Total	%	
		For documentation	21	75.00%	
		To generate code	13	46.43%	
		To generate models	5	17.86%	
		to communicate (e.g., to share ideas, to explain some concept)	18	64.29%	
For other purposes	6	21.43%			
DSL	Do you know DSLs?	24	75.00%		
	If you do, do you use them?	22	91.67%		
	If you do, what do you use them for?	Options	Total	%	
		For documentation	7	31.82%	
		To generate code	17	77.27%	
		To generate models	9	40.91%	
		to communicate (e.g., to share ideas, to explain some concept)	8	36.36%	
For other purposes	9	40.91%			

TABLE 6.11: Information about knowledge and use of modelling languages and DSLs from the students.

Element	Questions	Participants with YES answer	%		
<i>Modelling languages</i>	Do you know modelling languages?	66	100%		
	If you do, do you use them?	63	95.45%		
	If you do, what do you use them for?	Options	Total	%	
		For documentation	50	79.37%	
		To generate code	33	52.38%	
		To generate models	19	30.16%	
		to communicate (e.g., to share ideas, to explain some concept)	36	57.14%	
For other purposes	7	11.11%			
DSL	Do you know DSLs?	52	78.79%		
	If you do, do you use them?	44	84.62%		
	If you do, what do you use them for?	Options	Total	%	
		For documentation	21	47.73%	
		To generate code	40	90.91%	
		To generate models	19	43.18%	
		to communicate (e.g., to share ideas, to explain some concept)	12	27.27%	
For other purposes	11	25.00%			

his reported data, there was clearly a conceptual confusion when he applied it. A total of 22.727% of the students chose *personal criteria*. These criteria were widely influenced by *usability* issues derived from the 6C comprehensive goal and some principles of PoN. 6C Goals were reported by 33.333% of the students and 22.727% for the PoN.

In accordance with the analysis that was described in Section 6.3.6, we compared the application of the quality methods for each participant, identifying the cases in which MMQEF demonstrated an advantage over the other selected quality method (i.e., the x parameter of the binomial distribution). This was verified through the sum of the identified favorable cases for MMQEF in each of the values obtained by the T, DoU, FQI, NQI, QID, and NI dependent variables. Figures 6.2 and 6.3 present the results obtained for the professionals and the students, respectively. For each dependent variable, the amount of successful application of MMQEF is indicated by in the x parameter.

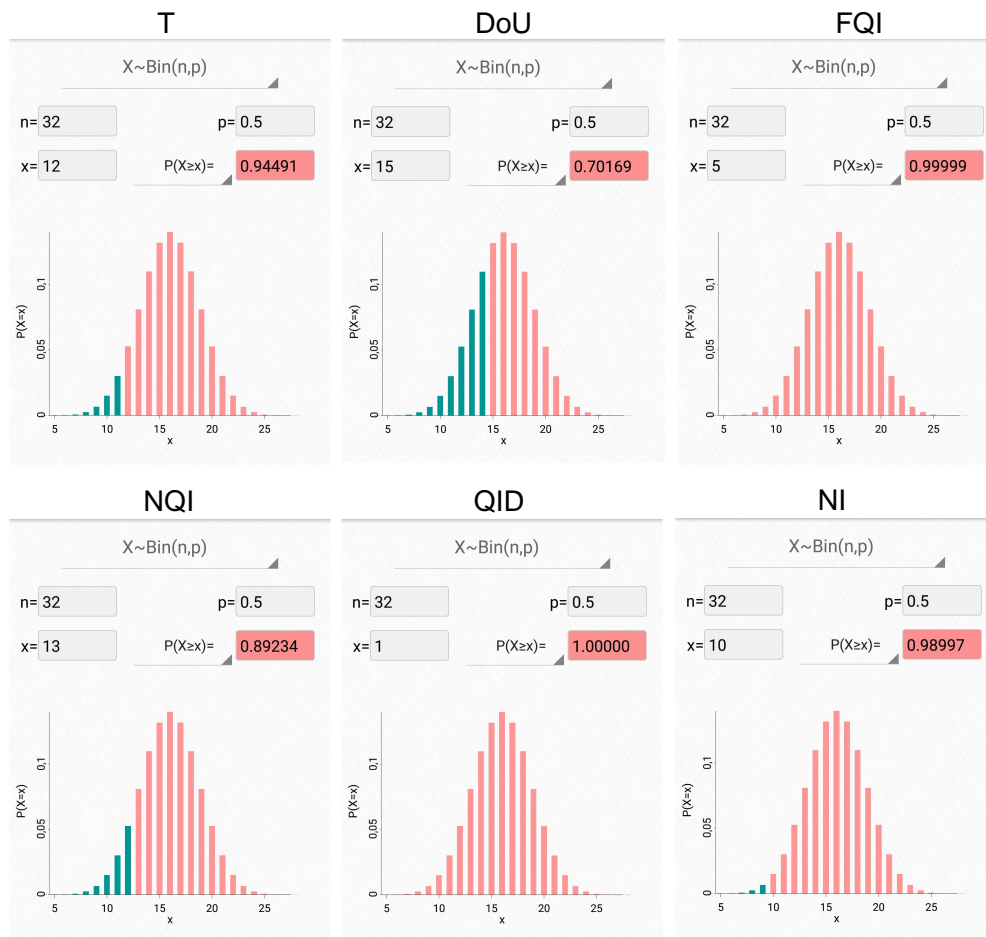


FIGURE 6.2: Resulting binomial distributions for the answers associated to T, DoU, FQI, NQI, QID, and NI by the professionals.

The probability distributions presented in Figures 6.2 and 6.3 oblige us to accept H_0 . From the obtained results, we could make a *Type-I-error* (Wohlin

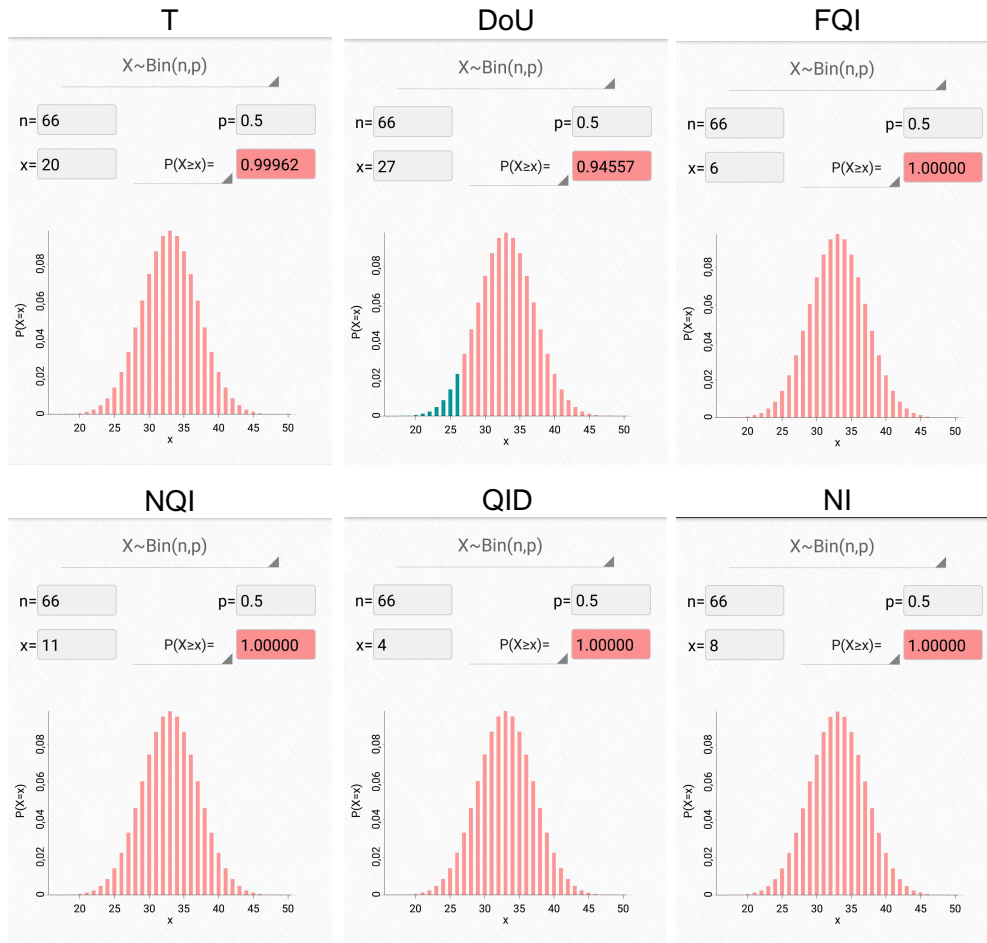


FIGURE 6.3: Resulting binomial distributions for the answers associated to T, DoU, FQI, NQI, QID, and NI by the students.

et al., 2012b), attempting to justify the applicability of the MMQEF method without a pattern of positive distribution over the data.

The above resulting scenarios were an expected output; this a consequence of verifying H_0 and H_a with six questions/metrics. In addition, the resolution of the metrics was based on the subjective criteria of each participant and their first use of the quality evaluation frameworks, including MMQEF. Thus, the overall results require a review of the complementary answers provided by the participants in each validation in order to find evidence that adequately justifies the responses provided.

Initially, the *time used to apply the quality evaluation method* (T) was considered to be a metric to show the practicality of MMQEF when compared to existing methods. However, in the results obtained for this metric, there is no significant difference between the time of application for the MMQEF method and the time of application for the other methods used by the participants (including the *personal criteria* method). For the *significant difference*, we expected a difference of (at least) thirty minutes between the applications of the methods. However, 68.75% of the professionals and 74.24% of the students reported times

without any significant difference (i.e., equal times or times whose difference was less than thirty minutes).

Table 6.12 presents the average value of the dependent variables involved for testing the two hypotheses. It shows that the values are close. The reported average of quality issues that were found and the number of inferences that were reported are relatively low in accordance with the time that was given to the participants for working with each quality method, the supplementary support that was given to the participants, and the additional support that the participants found by themselves. Taking into account these averages, the efficiency of the quality methods (MMQEF and the others selected) is inevitably questioned.

TABLE 6.12: Comparison of dependent variables associated to the hypotheses.

Dependent variables	Professionals		Students	
	MMQEF	Other Method	MMQEF	Other Method
T average (minutes)	61.8387097	52.5	51.7076923	51.2727273
DoU average (%)	57.0967742	56.29032258	63.7076923	67
Number of participants who found quality issues (FQI)	28	24	37	52
NQI Average	2.14285714	2.625	1.81578947	1.94230769
Number of participants who reported inferences (QID)	25	25	39	47
NI Average	1.73076923	1.615384615	1.31707317	1.32653061

However, this finding may be a consequence of the lack of previous knowledge about the model-driven paradigm (as reported in Tables 6.8 and 6.9) and the first interaction of the participants with methods to evaluate quality in MDE. Reported methods for evaluating quality in MDE require great cognitive effort for their understanding and applicability. The previous conception of *quality* of each participant also influenced the performance of the participants during the validation. In addition, quality methods are not directly comparable with each other by using their purposes, scopes, procedures, and conception of quality.

Due to the low values of the NQI averages that are presented in Table 6.12, we took advantage of Tables 6.1 and 6.2 (which present the expected quality issues for professionals and students that we previously considered in Section 6.3.1) in order to determine the applicability of quality methods through a review of the responses from the participants indicating whether or not they found similar issues to those of Tables 6.1 and 6.2. Table 6.13 presents each projected issue with the number of participants that reported similar issues to the ones projected. Table 6.13 also presents other quality issues reported by the professionals, which can be classified according to identified categories for issues. Other quality issues reported by the participants in Table 6.13 are specific quality issues without a common category for grouping them (i.e., a category that differs from the quality

methods) due to their specificity (5 issues reported in other methods by professionals, and 5 issues in MMQEF and 25 issues in the other methods reported by students).

Because of the low differences obtained in the value of the variable, and the number of reported issues that are reported in Table 6.13, the individual justifications delivered by each participant must be reviewed to determine whether or not the MMQEF meets its expected goals for quality evaluation in MDE. In the questionnaire for the application of the methods, we placed text boxes so that the participants could give their free opinion about the quality issues, the deduced inferences, and the Likert sentences.

To do this, we used specific protocols of *qualitative research* (Creswell, 2013) for analyzing data derived from the opinions (Lewins et al., 2010; of Surrey, 2016) through the *content analysis* approach (University, 2017). This analysis allowed us to find evidence about the applicability of MMQEF, and, therefore, to reject H_0 .

After reviewing the opinions and comments that were freely given by the participants, we found fifty-seven sentences of participants with explicit allusion to the quality methods used in the validation. Tables 6.14 to 6.17 present the identified sentences. Each sentence was codified with the data of the session in which it was reported (i.e., the $S1$, $S2$, $S3$, and $S4$ values that were indicated in Section 6.3.4), the data of the participant that reported it (we use the PR abbreviation for the professional participants, and the ST abbreviation for the students), and the source in the filled spreadsheet in which the comment was detected (i.e., the empty text boxes associated to the reporting of quality issues, deduced inferences, and support of responses for the Likert sentences).

Table 6.18 summarizes the identified comments, classifying them by type of comment (positive or negative) and source of the comment (professional or student). Of the fifty-seven statements, we found twenty-seven favorable or positive comments for MMQEF (47.36%) and eighteen negative comments for other methods for quality evaluation in MDE (31.57%). Positive comments for other quality methods (5.263% of the total of the identified comments) are for the PoN method.

Table 6.18 also indicates that the professionals applied the methods with greater judgment, with 64.912% of the obtained comments about quality methods being from the professionals. Their experience clearly influenced the selection and application of the quality frameworks. This finding was posteriorly confirmed by the obtained distributions that are presented in Figures 6.4 and 6.6 in the analysis of the Likert statements (section 6.4.2).

Evidence from Tables 6.14 and 6.18 indicates a positive trend about the application and the relative good performance of MMQEF as a method to identify quality issues in MDE contexts. In addition, most of the expected inferences were taken by participants as positive conclusions of MMQEF instead of deductions over the modelling context under evaluation. Here, we identified an improvement opportunity for MMQEF by providing guidelines to suggest deductions based on the value of the metrics formulated in MMQEF.

The statements in Table 6.17 present reviews and opinions of some of the participants about the methods used. These statements question the frameworks

TABLE 6.13: Number of participants who reported issues similar to those that were projected in Tables 6.1 and 6.2.

ID	Quality Issue	Number of related reported issues	Source
QIPR01	There was no traceability from models until code.	4	MMQEF
QIPR02	There was no automatic code generation.	1	MMQEF
QIPR03	Decoupling between organizational modelling and system modelling.	1	MMQEF
QIPR04	Misalignment between the modelling languages used and the purposes of modelling.	4	MMQEF
QIPR05	Excessive stereotyping of the UML modelling language.	7	MMQEF, PoN, Personal criteria
QIPR06	Excessive adaptation of languages to model business concerns.	4	MMQEF
QIPR07	Lack of suitability analysis of modelling languages.	9	MMQEF, 6C, Personal criteria
QIPR08	Lack of coverage of modelling languages. Some IS issues were not covered by modelling languages (e.g., data, interaction, architectural decisions).	12	MMQEF, 6C
QIPR09	There was no distinction of the purpose of the resulting models (i.e., there were models to communicate ideas, to automate process, to make systems), but these purposes were not explicitly addressed.	2	MMQEF, SEQUAL
QIPR10	There was no integration between modelling languages.	4	MMQEF, 6C
QIPR11	Poor support for deontic (for organizational and system purposes).	1	Personal criteria
QIPR12	Lack of adequate tool support.	0	
QIPR13	Lack of expressiveness of the modelling languages.	6	MMQEF, Personal criteria, PoN
QIPR14	Lack of communication abilities for the performed modelling.	4	6C, PoN
QIST01	Lack of coverage of system and platform concerns	9	MMQEF, Personal criteria
QIST02	Crossing between business and code levels.	9	MMQEF, Personal criteria
QIST03	Poor support for modelling system concerns.	5	MMQEF
QIST04	Other modelling languages could be appropriately used to model system concerns (e.g., UML).	2	MMQEF, PoN
QIST05	Other modelling languages could be more appropriate for modelling CDD concerns such as (technical) requirements.	1	MMQEF
QIST06	There is no traceability between high-low levels (including generated code).	5	MMQEF, 6C, Personal criteria
QIST07	Poor support for modelling IS concerns such as locations and time.	8	MMQEF
QIST08	Overloading of symbols and connections to model semantic constructs.	17	MMQEF, 6C, Personal criteria, PoN
QIST09	The only significant difference in the CDD modelling constructs is the color of the primitive graphics.	8	MMQEF, 6C, Personal criteria, PoN
Other quality issues (from professionals)	There are redundant elements in the modelling languages under evaluation.	1	MMQEF
	Quality issues related to symbols.	1	PoN
	Changeability issues.	2	6C
	Specific quality issues.	5	SEQUAL, 6C, Personal criteria, PoN
Other quality issues (from students)	Specific quality issues.	5	MMQEF
	Specific quality issues.	20	SEQUAL, 6C, Personal criteria, PoN

TABLE 6.14: Statements extracted from participants during the validation with *positive comments for MMQEF* (Part I).

ID	Participant	Source	Sentence
1	S1PR01	Inference	More than one language is needed to cover all aspects. A single language does not have the abstraction levels needed to cover all of the aspects in a system. This can be detected easily by the taxonomy.
2	S1PR03	Inference	MMQEF thoroughly evaluates the proposed models; this allows the users of the framework to have a better perspective of the model under evaluation. MMQEF gives the underlying evaluation points during the design of a modelling language.
3	S1PR04	Issue	When we are filling in the cells of the taxonomy, there is clearly no language that supports all of the viewpoints and all of the abstraction levels. There are domain aspects that cannot be modelled with the language.
4	S1PR04	Likert	Applying MMQEF allows us to detect traceability problems; in addition, it is evident that one modelling language is not enough for modelling a complex context.
5	S1PR09	Inference	In accordance with the evaluation, it is possible to conclude that it is feasible to use MMQEF to identify quality issues.
6	S1PR14	Inference	It is difficult to eliminate subjectivity in the evaluation of a language, but MMQEF provides an approach to show quality issues even with subjectivity.
7	S1PR15	Inference	The analysis can be fine depending on the experience and adaptability that the analyst has with the language. However, the classification may reduce the subjectivity of people that evaluate the languages.
8	S1PR18	Inference	It is possible to evaluate the quality of a modelling language based on several diagrams and elements; MMQEF is a flexible tool.
9	S1PR18	Likert	MMQEF demonstrates that there is not a total modelling language for making a modelling process, and also that there are modelling languages that do not appropriately represent an abstraction level or a viewpoint under analysis..
10	S1PR21	Issue	MMQEF gives us a broad view about the languages in a context; we can easily evaluate which elements are missing in a specific notation. The method lets us determine the optimal language for our process. Quality problems vary according to the scenario.
11	S1PR21	Inference	MMQEF serves to identify integration criteria, and, therefore, we know how useful a language can be.
12	S1PR21	Inference	With the application of MMQEF, we can determine the common elements in the proposed languages, and, therefore, we can determine the most appropriate languages for any scenario.
13	S1PR22	Inference	MMQEF allows us to have a global view of the problem.
14	S1PR23	Inference	It is easy to identify the applicability of each modelling language in the total of the architecture for Information Systems..
15	S4PR04	Likert	MMQEF allows us to detect lacks in the phases of the development of a model.
16	S4PR06	Inference	MMQEF allows us to align the realities of the software development using models with organizational purposes and goals.
17	S4PR08	Likert	MMQEF allows us to assess the quality and appropriateness of the use of several modelling languages to address different situations .
18	S2ST17	Inference	The analysis with MMQEF allows to determine the scope of each modelling languages under evaluation .
19	S2ST25	Likert	MMQEF allow us to perform comparisons with a same standard, which gives results with a same viewpoint. This feature allows us to compare modelling languages through models because the matrix and the detailed information of each language present how a language is structured, how it works, and its relations with other perspectives are.
20	S2ST27	Likert	MMQEF is a practical evaluation method that easily allows us to find and identify quality issues. From the matrix and the detailed study of each modelling language, we can know how a model is structured, how it works internally, what strengths and weakness each modelling language has, and how modelling languages can be applied in further software projects.
21	S2ST36	Inference	The analysis with MMQEF is visually clear and understandable.
22	S2ST38	Inference	The use of multiple models to represent an idea can be confusing for new users; however, with more detail, it is possible to complement concepts. This tool has a lot of potential. More emphasis should be placed on evaluating modelling languages.
23	S2ST44	Likert	MMQEF allowed me to establish a focus in which the abstraction levels must be supported. It results in a completeness level, i.e., the support of each modelling language for the abstraction level that it addresses..
24	S3ST04	Inference	MMQEF becomes attractive and dynamic for its matrix structure and its way each modelling language of contrasting.
25	S3ST05	Inference	From the performed analysis it is possible to conclude that MMQEF is useful in analyzing the quality of modelling languages such as UML, and 4EM.
26	S3ST06	Inference	MMQEF is complete because it covers different modelling aspects.
27	S3ST19	Inference	MMQEF allows us to find quality issues without going into detail about the diagrams or elements that compose them.

TABLE 6.15: Statements extracted from participants during the validation with *negative comments for MMQEF (Part II)*.

ID	Participant	Source	Sentence
28	S1PR02	Inferences	There is not an understanding about the quality model that is used to qualify the modelling languages and indicate possible issues. The provided info is not enough to understand the quality evaluation model and the proposed modelling case.
29	S1PR04	Inferences	Time was too short for understanding the quality model and evaluating the hypothetical case.
30	S1PR06	Inferences	A more thorough analysis is required to detect issues in the language and their possible solutions. More time is needed in order to have solid conclusions.
31	S2ST05	Inferences	MMQEF has problems for understanding graphic primitives.
32	S2ST09	Inferences	MMQEF allows us to classify the scope of each modelling proposal but not to evaluate its quality.
33	S2ST23	Issue	The analysis itself makes the identification of quality issues on models difficult.
34	S2ST43	Inferences	The taxonomic matrix is not the best tool to evaluate quality because a unique conclusion is not obtained about the position of a modelling language over the matrix.
35	S3ST01	Inferences	MMQEF requires more experience in order to more intuitively classify each modelling language, and therefore to make a more complete analysis.
36	S3ST05	Issue	MMQEF does not analyze the graphic primitives of modelling languages, e.g., how the symbols can be arranged, or what the best number of symbols to be used in a modelling project is.

TABLE 6.16: Statements extracted from participants during the validation with *positive comments associated to other quality methods for MDE (Part III)*.

ID	Participant	Source	Sentence
37	S1PR03	Inference	The used framework allows us to make a better analysis of graphical models. It explains the symbols that must be/not be used. It also identifies overload issues for final users of the models.
38	S3ST05	Inference	From the analysis with PoN, I deduce that this framework can be adapted to the modelling language under analysis. The framework determines what principles are applicable in accordance with the syntax of the modelling language.
39	S2ST21	Inference	PoN makes a clear analysis of the components of each modelling language that belong to 4EM.

TABLE 6.17: Statement extracted from participants during the validation with *negative comments for other quality methods* for MDE (Part IV).

ID	Participant	Source	Sentence
40	S1PR04	Inference	The framework is apparently simple for evaluating modelling languages. The concepts of the 6C must be met by the modelling languages under analysis. It is a challenge for the user to find a language that satisfies all the domain context.
41	S1PR06	Inference	The 6C analysis is not enough since it identifies shortcomings of the languages, but it does not guarantee their quality. The 6C analysis is so abstract or generic, and it could generate misinterpretations.
42	S1PR07	Issue	The approach analyzes the quality of languages in a way that is too subjective. It does not detail the relationships of modelling languages with their application and target public.
43	S1PR07	Inference	For me, the used method does not completely cover all the aspects that must taken into account to evaluate a language. Detailed aspects are not considered.
44	S1PR10	Inference	It was not clear for me how to evaluate the model. I do not understand the method. I am not sure if the quality criteria that I applied are the most appropriate. The method should provide a roadmap for its application.
45	S1PR13	Issue	The approach is based on the application of a checklist, so it is subject to the criteria of the evaluator. Even though it is a very subjective approach, it is very easy to apply because there is no specific control to verify the features that are under evaluation.
46	S1PR13	Inference	When criteria as open as 6C are applied together with my previous concepts, we must average the evaluations of some experts in order to make a more precise and adequate judgement of the language under analysis. A set of methodological steps must be formulated to guide the decision making.
47	S1PR14	Inference	The approach is very subjective, precarious, and poor. It does not indicate quality problems, and it is based on the subjectivity of whoever performed the evaluation.
48	S1PR15	Inference	The subjectivity of this method is high. For this reason, the evaluation is reduced to the paradigm of the evaluator. For example, for me, UML meets the completeness goal because it allows all parts of a system to be modelled correctly; however, another evaluator could consider that UML does not meet this goal based on his/her criteria and goals.
49	S1PR18	Inference	This method is too abstract. It indicates only what goals we must evaluate, but it does not indicate how to evaluate them. It produces multiple interpretations.
50	S1PR19	Issue	There is not a comparative guide (or application guideline) to determine if the application of the approach is correct or incorrect. The evaluation is performed subjectively by the involved experts based on their expertise; an instrument, a guide, and an example that serves as baseline for the comparison is necessary.
51	S1PR21	Issue	The method (PoN) only focuses on the concrete syntax; therefore, the result is an analysis about the graphic alternative for modelling elements.
52	S1PR21	Inference	PoN does not clearly justify if a modelling alternative is better than the concrete syntax of UML. Personally, I do not see PoN as a solution to facilitate the cognitive process of adaptation and integration to an environment regarding the initial proposal of UML.
53	S1PR22	Inference	It is too complicated to use all items of the approach. More description it is also required to use it.
54	S1PR24	Inference	The approach focuses on verification rather than validation, which is an important strategy for software quality.
55	S4PR06	Issue	6C demands effort to follow to each C goal. The approach is good, but it can be the most appropriate to evaluate the quality of UML models.
56	S3ST06	Inference	It is a simple approach but it is limited in the specification of important features of quality.
57	S3ST07	Inference	It is too complicated to relate each dimension of the quality framework to the elements of the modelling languages. The framework would generate bigger models, and, therefore, their understanding is difficult.

TABLE 6.18: Summary of the identified comments with positive/negative perceptions of the quality methods in the validation.

Method	Sentences from the Professionals		Sentences from the Students		Total	
	Positive	Negative	Positive	Negative	Positive	Negative
MMQEF	17	3	10	6	27	9
Other methods	1	16	2	2	3	18

used regarding their capacity to evaluate quality issues. Their concerns are related to the scope of the frameworks and the definitions of quality that do not allow specific procedures or steps to be applied. This evidence (statements) also contrasts with the evidence obtained for MMQEF, in which the sentences are in accordance with the goal and the analytic procedure of the method.

Even though the normal distributions are not positive for the MMQEF method and the other alternatives, there is qualitative evidence that demonstrates the applicability of MMQEF for supporting quality analytic procedures in modelling languages (the higher percentage of favorable opinions for MMQEF compared to the favorable opinions and the high percentage of negative comments for the other selected methods).

An analysis was also performed to determine the influence of the quality frameworks on each other as a consequence of the *Paired-comparison design* (defined in Section 6.3.2). To do this, we compared the identified favorable cases for MMQEF in each of the values obtained by the T, DoU, FQI, NQI, QID, and NI dependent variable, regarding the application of methods defined in Table 6.3.

Table 6.19 summarizes the identified favorable cases for MMQEF regarding the treatments of Table 6.3. In both types of participants, there is similar behavior in the value of favorable cases regarding the distribution design. For the T and DoU variables, more cases were favorable for MMQEF when the participants started with other methods; however, for the FQI, NQI, and NI variables, a greater number of cases were reported when the participants started with MMQEF. For the QID variable, there is not a significant difference in the obtained values of favorable cases. Because of the divergence in the behavior of the *Paired-comparison* design, there is not a pattern of influence between methods regarding the treatments of Table 6.3. Therefore, there is no evidence of the influence of the quality frameworks on each other (i.e., MMQEF on the others and vice versa).

Analysis of the MEM variables for MMQEF

One of the main risks of this empirical evaluation is the use of new methods that are absolutely unknown to the participants. In (Wohlin et al., 2012b), Wohlin et al. discuss the risks involved when new methods are tested; they specifically consider issues related to the consistent application of previous methods and their influence on existing methods when new ones are learned. A clear example of such a validation can be found in (Panach et al., 2015a), where an experiment

TABLE 6.19: Comparison of favorable cases for MMQEF regarding the Paired-comparison design.

Expected success answer for MMQEF	Distribution	Number of favorable cases for MMQEF variables	
		Professionals (32)	Students (66)
$T_{MMQEF} < T_{AnotherMethod}$	Starting with MMQEF	4	6
	Starting with other method	8	14
$DoU_{MMQEF} > DoU_{AnotherMethod}$	Starting with MMQEF	6	12
	Starting with other method	9	15
$FQI_{MMQEF} = YES \wedge FQI_{AnotherMethod} = NO$	Starting with MMQEF	5	6
	Starting with other method	0	0
$NQI_{MMQEF} > NQI_{AnotherMethod}$	Starting with MMQEF	10	9
	Starting with other method	3	2
$QID_{MMQEF} = YES \wedge QID_{Another} = NO$	Starting with MMQEF	1	2
	Starting with other method	0	2
$NI_{MMQEF} > NI_{AnotherMethod}$	Starting with MMQEF	7	5
	Starting with other method	3	3

was performed to compare a traditional software process development with a model-driven development process.

However, in the design of this validation, the main challenge is the lack of previous interaction with the quality frameworks for MDE. This was the first time that the participants confronted quality issues in MDE, and, therefore, their first time recognizing and applying the frameworks involved (including MMQEF). Although, each participant applied two quality evaluation methods (freely selecting one of them), the entire population had not considered the presence of quality issues in modelling languages. Therefore, they did not know frameworks or methods for addressing quality evaluation procedures at the model-driven level. This was evident despite the percentage of the participants (both professionals and students) who reported previous knowledge and skills with model-driven technical environments, use of modelling languages, use of domain-specific languages, and metamodeling (see Section 6.4.1).

For this reason, a Likert survey approach was applied to validate the specific MEM dimensions of the MMQEF (i.e., the *Perceived Ease Of Use*, the *Perceived Usefulness*, and the *Intention To Use*) that were described in Table 6.6 of Section 6.3.3. The survey was applied mainly to determine if MMQEF could have been influenced (and affected) by the application of other quality frameworks selected by the participants, complementing the finding of Table 6.19 by the use of the three last dependent variables that were defined in Table 6.6. Tables 6.20 and 6.21 summarize the results obtained in the Likert sentences for the professionals and the students, respectively. Section 6.7 has the URL for accessing the data and the comparative figures associated to each Likert sentence.

Tables 6.20 and 6.21 also present the Cronbach's α that were obtained for the Likert survey of the professionals ($\alpha = 0.883607296$) and the students ($\alpha = 0.7692104$), respectively. Since the obtained values are greater than the expected value for this test (0.7), the reliability of the Likert surveys and their internal consistency are confirmed.

Since we considered *positive responses* to be 4 (Agree) and 5 (Strongly agree) on the Likert scale, there is enough evidence to recognize the positive responses to the MEM dimensions for MMQEF. We identified three Likert statements where 3 (*Neither agree nor disagree*) was assigned by a large number of student participants ($n \geq 20$); however, this number does not exceed the total of responses in the 4 and 5 range. These statements were (in order) Likert 09 (*practicality of MMQEF*), Likert 03 (*usefulness of MMQEF*), and Likert 06 (*alignment of MMQEF with MDE*). The 3-value score that was given to these statements indicates the limited degree of the use of the MDE paradigm by the students and the high cognitive load of the students when they applied the method for the first time.

Of all of the Likert sentences, the professionals gave the greatest number of 3 values (neither agree nor disagree) to the Likert 01 statement (*ease of understanding MMQEF*). This is a consequence of the first interaction of the professionals with the taxonomy proposed in the Zachman framework, which requires an initial cognitive effort for the use of its bi-dimensional structure and rules.

TABLE 6.20: Summary of the Likert responses of the professionals for MMQEF.

Likert Question	Likert scale (Cronbach's $\alpha = 0.883607296$)							
	1	2	3	4	5	Mean	Median	Mode
L1 - <i>Ease of understanding MMQEF</i>	3	0	11	14	4	3.50	4	4
L2 - <i>Ease of using MMQEF</i>	3	0	9	14	6	3.63	4	4
L3 - <i>Usefulness of MMQEF</i>	0	1	7	11	12	3.97	4	5
L4 - <i>MMQEF further intention of use</i>	1	0	5	11	14	4.06	4	5
L5 - <i>Usefulness of MMQEF in SE and IS problems</i>	1	1	3	13	14	4.19	4	5
L6 - <i>Alignment of MMQEF with MDE principles</i>	1	0	9	9	13	4.03	4	5
L7 - <i>Further importance of quality at MDE level</i>	0	0	5	12	14	4.16	4	5
L8 - <i>Considerations of MDE projects addressed with MMQEF</i>	1	0	2	15	14	4.28	4	4
L9 - <i>Practicality of MMQEF</i>	1	3	6	18	4	3.66	4	4
L10 - <i>Usefulness of the taxonomy</i>	1	1	4	10	16	4.22	4.5	5
L11 - <i>Usefulness of classification of modelling languages</i>	0	0	3	11	17	4.31	5	5
L12 - <i>Usefulness of the MMQEF inferences</i>	0	3	8	12	9	3.84	4	4

TABLE 6.21: Summary of the Likert responses of the undergraduate students for MMQEF.

Likert Question	Likert scale (Cronbach's $\alpha = 0.7692104$)							
	1	2	3	4	5	Mean	Median	Mode
L1 - <i>Ease of understanding MMQEF</i>	1	5	19	35	6	3.61	4	4
L2 - <i>Ease of using MMQEF</i>	1	8	21	32	3	3.38	4	4
L3 - <i>Usefulness of MMQEF</i>	2	6	24	28	6	3.45	4	4
L4 - <i>MMQEF further intention of use</i>	1	7	11	35	10	3.66	4	4
L5 - <i>Usefulness of MMQEF in SE and IS problems</i>	0	3	19	27	17	3.88	4	4
L6 - <i>Alignment of MMQEF with MDE principles</i>	0	4	24	27	11	3.68	4	4
L7 - <i>Further importance of quality at MDE level</i>	0	1	9	28	26	4.17	4	4
L8 - <i>Considerations of MDE projects addressed with MMQEF</i>	0	1	17	34	12	3.77	4	4
L9 - <i>Practicality of MMQEF</i>	0	2	25	30	9	3.70	4	4
L10 - <i>Usefulness of the taxonomy</i>	0	2	13	34	17	4.00	4	4
L11 - <i>Usefulness of classification of modelling languages</i>	1	2	19	29	15	3.83	4	4
L12 - <i>Usefulness of the MMQEF inferences</i>	0	4	17	25	19	3.91	4	4

The obtained data demonstrate a trend towards the selection of the 4 and 5 Likert levels in the two groups of participants (professionals and students). To confirm this, we apply a *quartile analysis* for the MEM variables. For each participant, the sum of the values of the Likert sentences associated to each MEM variable was calculated (i.e, the Likert sentences 1, 2, 3, and 9 for the *Perceived Ease Of Use* variable, the Likert sentences 4 and 7 for the *Intention To Use* variable, and the Likert sentences 5, 6, 8, 10, 11, and 12 for the *Perceived Usefulness* variable).

Figures 6.4, 6.5, and 6.6 present the resulting quartile analysis. These figures show the trend of the distribution of the samples to higher values that were expected for each MEM dimension, i.e, 20 for PEU (four Likert sentences), 10 for IU (two Likert sentences), and 30 for PU (six Likert sentences).

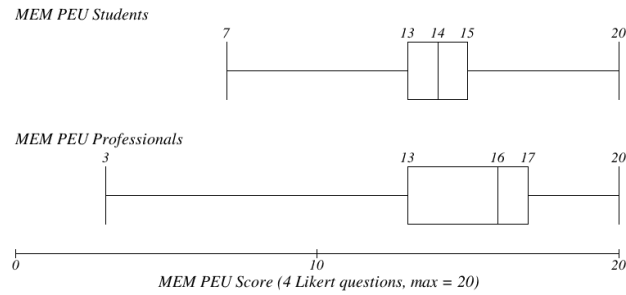


FIGURE 6.4: Quartile analysis for the *MEM Perceived Ease of Use* dimension.

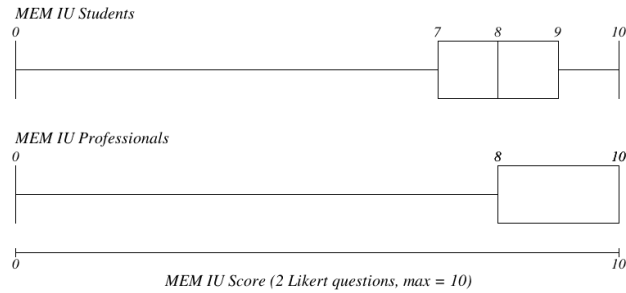


FIGURE 6.5: Quartile analysis for the *MEM Intention to Use* dimension.

In addition, a *Kruskal-Wallis* test procedure was performed in order to confirm the difference between the professionals and the students that were enrolled in the empirical validation. This difference was previously detected in the distributions presented in Figures 6.4 to 6.6. The *Kruskal-Wallis* approach was selected due to its suitability for analyzing data from ordinal variables. Table 6.22 shows the obtained *H-statistic* for the MEM variables. We computed it with the *significance level* $\alpha = 0.1$, the *number of degrees of freedom* $df = 2 - 1 = 1$, and

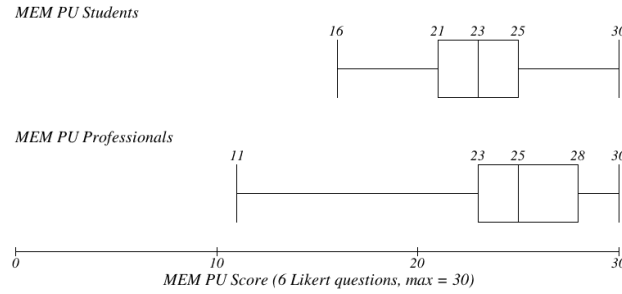


FIGURE 6.6: Quartile analysis for the *MEM Perceived Usefulness* dimension.

$N = 98$. The obtained values are greater than the *Chi-Square* value that is associated for this setting $(2.706)^5$. This evidence confirms the significant difference between the professionals and the students.

TABLE 6.22: H statistic values obtained from the *Kruskal-Wallis* test for the MEM dimensions.

MEM dimension	H-statistic
<i>Perceived Ease Of Use</i>	3.955
<i>Intention To Use</i>	2.905
<i>Perceived Usefulness</i>	7.646

However, for the MEM *IU* variable, the *H statistic* is closer to the *Chi-Square* value. This closeness indicates a similarity of the medians for the professionals and the students in the *IU* sample. When we apply this testing procedure with $\alpha = 0.05$, the hypothesis about difference for the MEM *IU* variable is not rejected since the resulting *H statistic* is less than the *Chi-Square* value for this Kruskal-Wallis test (3.841). Figure 6.5 confirms the similarity in the distribution of the population for this variable. This finding represents a consensus about the importance of identifying and managing the impact of quality issues of modelling languages in model-driven projects and, eventually, the use of MMQEF for quality analysis in similar modelling scenarios. The supporting data for the *Kruskal-Wallis* test can be found in Section 6.7.

In summary, the findings of this section can be described as follows:

- Key terms for the MMQEF method (and the model-driven paradigm in itself) are not properly appropriated by the participants. This influences the application of MMQEF for evaluating quality in model-driven scenarios.
- The quantitative analysis of the application of the quality methods does not indicate an important difference between MMQEF and the other methods. Instead, the obtained results question the efficiency of the performance of the quality methods.

⁵A *Chi-Square* distribution table can be found in <http://sites.stat.psu.edu/~mga/401/tables/Chi-square-table.pdf>

- Qualitative analysis approaches reveal a trend for the applicability of MMQEF regarding the other quality methods. This was deduced from the comments and opinions that were freely reported by the participants.
- The results from the Likert sentences demonstrate that the participants agree and recognize the dimensions of the MEM model (i.e., the *Perceived ease of use*, *Perceived usefulness*, and *Intention to use*) for the MMQEF method.

6.5 Discussion

6.5.1 Evaluation of results and implications

It is widely recognized that quality is an intrinsic property of artifacts in engineering itself. However, there is no common consensus regarding quality beyond its explicit manifestation and management. Instead, multiple definitions and proposals about quality have been identified, each of which is valid based on the specification of quality that is addressed.

The validation processes in frameworks for MDE are highly specific. Concrete scenarios are required to be able to integrally validate quality frameworks in MDE and to demonstrate their feasibility and efficiency. Quality frameworks for MDE could be mutually exclusive depending on the concept of quality being addressed. Due to these conceptual divergences, the methods may not be mutually comparable with each other.

Each framework for quality evaluation in MDE provides specific purposes and advantages based on its specific concept of quality. As a consequence of the broad definition of quality for MDE, several types of quality can be proposed and their purposes can be justified. The challenge here is the systematic integration of these frameworks to promote the adoption of the MDE paradigm by focusing on quality and its associated implications.

6.5.2 Threats to Validity

For this empirical validation, the validity of the results is critical due to the need to review the obtained data and extract evidence about the positive applicability of MMQEF. The participants voluntarily enrolled in the validation, and they were free to leave it at any time. The participants also signed a consent form. The data was integrally managed as it was reported by the participants.

The Method Evaluation model (MEM) for the evaluation methods of Information Systems (Moody, 2003) was also intentionally applied to validate the potential applicability of MMQEF through an approach that is proposed for evaluating Information System methods. Associated items for MEM perception variables (i.e., the *Perceived Ease of Use*, the *Perceived Usefulness*, and the *Intention to Use*) were checked by using approaches to determine the reliability and internal consistency of the obtained responses (Section 6.4.2).

Taking advantage of the checklist of validity threats that was defined in (Wohlin et al., 2012c), Tables 6.23 and 6.24 present a detailed analysis of the threats that were detected for this empirical validation and the corresponding

TABLE 6.23: Analysis of *validity threats* for the empirical validation (Part I).

Validity category	Threat	State	<i>How we addressed it in the empirical validation.</i>
Conclusion	Low statistical power	Addressed	The statistical evidence was respected by following strict protocols for applying the statistical analysis. True patterns from the data are evident. The statistical patterns found obligate us to apply complementary tasks from qualitative research to support the applicability of MMQEF.
	Violated assumptions of statistical tests	Addressed	Statistical assumptions were preserved in the test of hypotheses (binomial distribution) and the analysis of Likert sentences (through a quartile analysis and a Kruskal-Wallis test). For those analyses, we use parameters as they are commonly reported in the literature. In addition, for the Likert analysis, we apply two recognized configurations for discussing a found pattern about the similarity of distributions for a MEM variable.
	Fishing	Partially addressed	The quantitative data provided neutral results; therefore, a qualitative research approach was applied as a complementary method to find patterns over non-quantitative data (evidence of the applicability of MMQEF).
	Error rate	Addressed	Values of the significance level variables that were employed during the analysis (i.e., values for α , p - value, and df) were determined from a literature review about the settings for validations of this kind that are commonly suggested, and also from expert judgment from statistical advisors. No multiple analyses were conducted in the validation.
	Reliability of measures	Addressed	The instruments of the validation employed were double checked. In addition, the validation was applied in four sessions in accordance with the availability of the participants. For all the sessions, the obtained outcome was similar. No differences in the behavior of the results obtained were found.
	Reliability of treatment implementation	Addressed	The treatments in Table 6.5 were carefully applied by all participants in all sessions of the validation.
	Random irrelevancies in experimental setting	Addressed	No elements outside the experimental setting were presented in each session of the validation.
	Random heterogeneity of subjects	Addressed	To guarantee the heterogeneity of the participants in the validation, two kinds of participants were invited: undergraduate students (66) and professionals (32) with verified expertise in real software development projects. The participants were demographically analyzed (Section 6.4.1), no evidence of features in the sample that represent threats for the result was detected.
Internal (single group threat)	History	Addressed	Despite the four sessions that were required in the validation, no effects by the execution of the validation in different times were reported. The participants worked once in the validation. No repeated test was required in the design of the validation.
	Testing		
	Maturation	Partially addressed	The researchers addressed it by the treatments defined in Table 6.3. However, there was an initial cognitive load for interacting the first time with quality methods for MDE. Eventually, this load affected the performance of the participants. Taking into account the number of participants, an alternative for managing the maturation threat was to assign the application of only one method to each participant. However, this decision could have affected the statistical power due to the low results obtained. For validations of this kind, it is complex to involve subjects without affecting the quality of the training phase.
	Instrumentation	Addressed	The form that was designed to collect data from the participants has a simple and practical design to facilitate the interaction of each participant with the instrument. No reports about the cognitive complexity of the form were presented.
	Statistical regression	Addressed	The participants were chosen by a convenience sampling. No additional classification tasks were applied to the participants.
	Selection	Addressed	Due to the previous knowledge about modelling, quality, and MDE topics that were required of the participants, a random selection was not viable.
	Mortality	Not applicable	The participants were free to leave the validation at any time. However, none of the participants dropped out of the validation.
	Ambiguity about direction of causal influence.	Not Applicable	During the evaluation of quality in a modelling project using some proposed methods, the participants must apply subjective criteria to determine the cause/effect relation of the quality issues found. The available information of the methods that was provided by the researchers supports the individual analysis. The ambiguity is implicitly addressed by the application of the quality evaluation methods.
	Diffusion or imitation of treatments	Partially addressed	One case of an imitation in the treatment was detected in two professional participants. However, this did not influence the overall performance of this population. Instead, this finding was carefully managed to analyze the application of the methods that was reported by the participants involved. The imitation of the treatment did not impact the application of the methods of each participant.

TABLE 6.24: Analysis of *validity threats* for the empirical validation (Part II).

Validity category	Threat	State	<i>How we addressed it in the empirical validation.</i>
Construct	Inadequate preoperational explication of constructs	Addressed	To avoid a lack of clarity in the participants with new theories about quality evaluation methods for MDE, the experimental material (Section 6.3.3) was taught in accordance with didactical strategies that facilitate and promote the interaction of the participants with these new (and unknown) theories.
	Mono-operation bias	Partially addressed	For the validation, it was not possible to implement multiple versions of quality modelling scenarios for applying quality methods for MDE due to the complexity of each scenario and the several quality issues that could be reported from the subjective analysis of each participant. Multiple modelling scenarios impact the results about the applicability of methods. Specific values of the independent variable were employed for specific scenarios of applicability. These values and scenarios could limit the demonstration of the quality evaluation for MDE. Therefore, to address this in further validations, we propose validating specific features of the quality methods for MDE with multiple experiences (Section 6.5.4).
	Mono-method bias	Addressed	Multiple (and complementary) measures were used in the validation to evaluate the applicability of the methods and their potential use. Multiple resulting observations from participants are from the subjective interpretation and application of quality methods. Measures have behaved in accordance with the theoretical expectation of the researchers.
	Confounding constructs and levels of constructs	Partially addressed	The relationship between the previous knowledge of the participants and the obtained performance of the methods was explicitly reported (Section 6.4.2). In addition, the training about the quality methods placing emphasis on specific features (including MMQEF) could affect the results of selection and applicability of methods in the participants.
	Hypothesis guessing	Addressed	There was a consensus about the purpose of the validation and the applied treatments. No guesses about the purpose were detected. Prior to the validation, the researchers had identified and removed any possible guessing source for the validation, e.g., the relationship between the performance and any qualification in the case of the students.
	Restricted generalizability across constructs	Addressed	The results obtained can be considered in similar scenarios of experimentation about quality methods for MDE.
	Experimenter expectations	Partially addressed	We avoid any influence of the researchers favorable to MMQEF. This was done by contrasting its applicability with regard to the quality methods that were previously proposed in the MDE literature. Although the researchers had expectations about the potential applicability and potential use of the MMQEF method, during the sessions of the empirical validation, any attempt to influence opinions of participants was avoided. The support for the participants was carefully provided to prevent any opinion (and induction) favorable to MMQEF. A neutral role was assumed by the researchers even though they had positive expectations for MMQEF.
External	Interaction of selection and treatment	Addressed	A suitable population for the empirical validation was convened by applying a convenience by sampling approach (Section 6.3.2). However, the characteristics of the population do not show any signs of a possible influence of their background on the results of any specific treatment (see the demographic analysis in Section 6.4.1).
	Interaction of setting and treatment	Addressed	The researchers were especially careful to provide representative material for the participants, including modelling scenarios for applying quality methods, in accordance with the previous knowledge of the participants (i.e., the CaaS scenario for students, and the UML-BPMN-Flowchart-SPEM scenario for professionals (Section 6.3.1)).
	Interaction of history and treatment	Not applicable	No effects for the days and times of application of the validation were reported.

strategy for addressing them. Overall, we acknowledge having suffered several threats that were difficult to eliminate completely; however, we applied mitigation strategies to minimize the impact of these threats.

6.5.3 Inferences

The following items generalize the findings of the empirical validation:

The selection of practical methods.

Clearly, as we reported at the beginning of Section 6.4.2, the participants searched for quality methods that helped them to make the quality evaluation in the easiest and most practical way. The available supporting material for the quality frameworks influenced the selection of the participants. This was a disadvantage for MMQEF because it is a work-in-progress and it does not have a lot of related publications. The selection of other methods based on their associated supporting material was evident despite the supporting material for MMQEF that we provided. The participants searched for specific examples of applications of the quality methods by reviewing their derived publications.

The selection of more practical frameworks is a consequence of the formulation of the desirable properties that represent quality in modelling languages, despite the lack of the proper description of procedures about how to determine the fulfillment of these properties. For example, with the PoN framework, all of the participants were warned about the lack of a systematic application for this method. The participants were also warned that it was not until 2016 that authors other than the original authors had proposed guidelines to address this application (da Silva Teixeira et al., 2016).

Detected reasons for choosing quality methods

Table 6.25 summarizes the participants' reasons for choosing the alternative quality methods to MMQEF. The tables show that 62.5% of the professionals who participated in the *S4* session and 55% of the students who participated in the *S3* session stated that the *easiness* of the approach was the main reason for choosing it. However, as Table 6.25 reports, there is no conclusive trend for a specific reason, especially from the data obtained from the students.

The participants of the *S1* and *S2* sessions were contacted by email, in which we asked them about the reasons for choosing the alternative quality method that was used in the validation. To date, seventeen professionals have answered the email. Twelve of these professionals also stated the *easiness* of the approach as the main reason for choosing it. Some of these professionals indicated that their selection was also based on a relationship between the easiness of the selected method with the time allotted for working with the alternative method (seven professionals), the examples of application found of the selected approach (six professionals), and the associated documentation of the methods (five professionals).

TABLE 6.25: Reported reasons for choosing the alternative method during the empirical validation.

Proposed reasons	Professionals	Students
The <i>easiness</i> of the approach.	5	11
The examples found that the approach had.	3	8
The <i>suitability</i> of the approach.	0	3
The documentation of the approach.	2	6
I think it is the best option.	1	10
The time required .	1	5
No particular reason. The choice was random.	1	1
Another reason (unspecified).	0	1

Representations as an important source for quality evaluation procedures.

An important finding that has been derived from the analysis of the independent variable (the application of the quality method) was the identification of the sources that were used by the participants to perform the quality assessment of the given models and modelling languages. In the application of each method (MMQEF and the alternative), each subject was queried about which sources of information he/she used to find quality issues. Table 6.26 presents the responses obtained.

TABLE 6.26: Sources of information for detecting quality issues reported by the participants.

Sources of quality issues	Professionals		Students	
	MMQEF	Others	MMQEF	Others
Metamodel (grammar)	12	10	9	12
Representation	18	21	21	34
Complementary info (e.g., the use of a modelling tool)	5	11	11	11
Other sources	5	1	4	5

The participants reported the use of *representations* (i.e., instanced models that are expressed in diagrams and/or textual blocks) associated to the modelling languages as the main sources for applying the quality frameworks and making quality assessments. Representations are the result of a cognitive interpretation of the users of the languages about the possible use and application of the modelling languages. Thus, quality issues are the result of an interaction among the participants with the modelling languages under analysis. Quality issues were detected from the perspective of the participants as the final users of the modelling languages. There is no evidence of quality issues from a *modelling*

language analyst perspective (i.e., the role that creates, designs, or proposes a language for modelling a specific concern).

Although the quality frameworks provide guidelines for the correct use of the modelling languages, the application of languages by their associated representations (i.e., the possible instanced models that could result for a modelling language from an final-user perspective) is an important source of problems perceived by the final users of the languages with respect to *quality in use*. There is a relation between diagrams (instanced models) and the selection of quality methods that work prescriptively with the information extracted from these diagrams.

The need for a modelling context

A key finding that was detected in both validations is the need to consider an explicit modelling scenario upon which the evaluation of the modelling languages could be done. We presented an illustrative scenario to the professionals and the students (Section 6.3.1). It is clear that all of the participants (professionals and students) required a specific context to identify and report quality issues. The context was used as a pivot to detect quality issues. The participants did not compare languages and did not apply a quality method without the modelling context. This acted as a conceptual framework that helped the participants to contrast the scope of the modelling languages under analysis.

Perceived independence of the quality proposals

A clear trend in the obtained results was the application of quality methods as isolated frameworks to perform quality analysis on modelling languages. For the independent variable (*application of the method*), the participants were induced to use MMQEF and any other quality method. This second method was freely selected by participants without any intervention by us.

None of the participants proposed an integration of two or more methods to make quality assessments. All of the participants chose and applied quality methods individually; they were not concerned about any possibility of integrating methods. This indicates that quality methods were used as inductive tools to understand quality issues at the modelling level and to find them based on the quality concept proposed by the selected framework.

6.5.4 Lessons learned

The improvement of the procedure for making inferences in MMQEF

MMQEF provides explicit activities to formulate inferences about the application of the modelling languages and their classification in the taxonomic structure of the method. The taxonomy considers modelling realities from business to technical levels. For this reason, we consider that inferences can be easily detected from the location of the modelling elements and the artifacts regarding the information that can be captured by the cells of the taxonomy.

However, an important result of the validations is that there is a need to improve the MMQEF guidelines in order to make inferences from the application

of the framework. The obtained evidence demonstrates that the inferences are personal conclusions of the participants about the method itself. Thus, more practical and methodological orientation is required so that MMQEF users can detect the consequences of the classification of modelling languages and artifacts in accordance with the perceived application and scope.

6.5.5 The improvement of the documentation for MMQEF

There was an evident disadvantage for the MMQEF method with regard to its associated documentation and supporting material, especially for examples of application. The findings described in Section 6.5.3 demonstrate the preference of the participants for methods that provide explicit examples and documentation about the application of quality methods in specific scenarios. Documentation with prescriptive steps and guidelines for performing evaluation procedures with MMQEF must be developed in order to improve the interaction of the method with its potential target public (i.e., the modelling language analyst and the designer as well as the final user of the modelling languages).

6.5.6 Improving the process of selection and characterization of participants

Previous intermediate/advanced knowledge of software engineering concepts (especially technical knowledge) does not guarantee the suitability of the participants, as was described in Section 6.4.1. The application of quality methods could be affected by previous conceptions from technical levels of software development projects, in which quality is based on the source code of programming languages and the progress of development teams. Further validations must consider the *degree of appropriation* of MDE concepts and supporting technologies by the participants. Several configurations or scenarios for experimentation can be obtained from the identified MDE appropriation of the invited participants.

6.5.7 Validating specific features of quality methods for MDE individually instead of all together

The evaluation of the performance and applicability of quality methods for MDE can be affected by the complexity of their underlying theory. Evaluating all of the features of quality methods in a single validation requires dense material for the experimentation with participants, and, therefore, complex procedures in the experimentation, especially for one session. Validations about specific features of quality methods could be a more practical strategy for identifying and characterizing the effectiveness of these features. In addition, the focus on specific features of quality methods facilitates the eventual formulation of procedures for comparing features from different quality methods with similar principles and intentions.

6.6 Conclusions

6.6.1 Summary

The validation of methods and frameworks for evaluating quality issues in the MDE field is a challenge. The common purpose of these frameworks for the evaluation of quality is not enough to be able to compare these frameworks due to the diversity of concepts about the term quality that is applied in MDE projects. Validation procedures that use individual applications of quality frameworks do not allow the results to be generalized over the wide scope of the model-driven paradigm.

In this chapter, we have reported the design and execution of a validation process for the MMQEF method through an empirical validation with thirty-two professionals in software engineering and sixty-six undergraduate students. The qualitative results from this validation demonstrate the feasibility of the application and the use of the method by potential model-driven practitioners. However, the quantitative and qualitative results also indicate the need to reinforce the current documentation of MMQEF in order to improve the deduction of quality inferences. Approaches from qualitative research were used to analyze the opinions and comments that were delivered by the participants in order to find evidence about the applicability of MMQEF and problems with the other quality methods used.

6.6.2 Impact

There are open challenges for the validation of MMQEF and other quality methods. The most relevant challenge is the application to software and system projects that are developed under the model-driven paradigm. The evidence that was presented in Section 6.4.1 demonstrates a clear influence of technical concerns for using artifacts of MDE. The evaluation of the applicability of quality methods such as MMQEF is highly dependent on the conviction about the central role of models in the development of complex systems and software projects.

MMQEF is not a revolutionary approach for evaluating quality in MDE. Instead, it can complement existing efforts to consolidate quality evaluation procedures by taking advantage of taxonomic analysis with a reference architecture for Information Systems (IS). The results that were obtained in the validation preliminarily reflect the feasibility of the application of MMQEF. Because of the taxonomic structure of the reference architecture that is used in MMQEF, we think it is possible to harmonize the application of existing and new modelling languages and approaches based on their explicit association to the abstraction levels defined in model-driven architecture (MDA) specification and the concerns associated to Information Systems that are generally expressed as viewpoints.

The evaluation of quality issues in model-driven artifacts from an IS perspective could contribute to the adoption of MDE by explicitly managing the scope of the modelling artifacts regarding the IS concerns (which vary from organizational to technical levels) and by identifying the information that satisfies the relevant viewpoints in an IS.

The emphasis on the use of an IS reference architecture and its associated taxonomic structure makes it possible for MMQEF to be used with other quality initiatives for MDE by complementing and supporting specific quality dimensions that are related to IS concerns, such as semantics, pragmatics, and organizational (deontic) dimensions.

Therefore, to correctly address the application of quality methods for MDE such as MMQEF, more MDE scenarios are required, including roles that consider the use of models for critical decisions in a project (e.g., models to support architectural decisions). This requires more availability of technical and personal resources and time. The length of specific sessions such as that used in the validation with MMQEF (3 hours) may be too short to demonstrate the impact of quality initiatives in real scenarios of practice.

6.6.3 Future work

In accordance with the *impact* stated above and taking into account the challenges for MMQEF that are derived from the negative comments described in Table 6.15, we have identified some further empirical evaluations that should be performed:

- Identify and evaluate the applicability, performance, and obtained quality of MDE scenarios in which MMQEF can be applied in order to evaluate and improve their quality.
- Identify the correspondence and potential integration of quality methods for MDE through comparisons of their key concepts and procedures for the identification and evaluation of quality.
- Demonstrate how MMQEF meets the principles of the MDE paradigm in scenarios of Information Systems development and Software Engineering projects.
- Improve the interaction with the taxonomy and activities that are proposed by MMQEF.
- Characterize the variables that allow the performance of methods for quality evaluation of MDE projects to be measured and compared.

Due to the resources that these activities require, we will consider the design and development of these works in the form of *case studies* or *action-research* techniques (Siau and Rossi, 1998).

6.7 Raw data

The supporting material and evidence of the performed validation (including the obtained raw data) can be found at <https://s3.amazonaws.com/mmqef-val/index.htm>. This support includes the following (each item is a link for the specific resource):

- Slides of the seminar (Section 6.3.3).

-
- The questionnaires (forms) given to the participants (Section 6.3.3).
 - The complementary material for the participants, which includes a summary of the MMQEF method (Section 6.3.3).
 - Evidence (images) of the validation procedures.
 - Forms filled out by the professionals.
 - Forms filled out by the students.
 - The characterization of the participants (Section 6.4.1).
 - The results obtained from the professionals.
 - The results obtained from the students.
 - Qualitative (content) analysis (Section 6.4.2).
 - The results for the Likert sentences (Section 6.4.2).
 - Support for the Cronbach analysis (Section 6.4.2).
 - Support for the quartile analysis and the Kruskal-Wallis Test that was performed for the MEM *PEU*, *IU*, and *PU* variables (Section 6.4.2).

Chapter 7

Final conclusions and the forthcoming research roadmap



One of the main challenges of any quality evaluation method for the MDE paradigm is the demonstration of its feasibility for effectively addressing multiple issues that are derived by the central role of the models and their foundational modelling languages. While it is true that a single method does not cover all of the quality issues of model-driven projects, the main contributions of the quality methods can be clearly acknowledged. The methods can complement each other, similarly to the way that quality in traditional software contexts is managed, in which quality has multiple implications and evaluation procedures.

With the specification of the MMQEF method, quality at the MDE level can be inferred from foundational descriptions and representations for Information Systems (IS). This is the starting point of a research program for quality evaluation in MDE that focuses on the systemic application of modelling languages in accordance with the foundational principles of the MDE paradigm and the involved viewpoints of IS projects. Further research initiatives in the short- and mid-term are derived from the formulation of MMQEF. These works are as follows:

7.1 The formulation of application scenarios for MMQEF

This work encompasses the study of the application of the MMQEF method in IS projects in which modelling languages are used to manage viewpoints, such as the project presented in Appendix A. This thesis presents three examples of application for specific cases of multiple modelling languages (the CDD methodology, UML-BPMN, and OO-Method-CA integration) in order to demonstrate the feasibility of the method. Further applications will allow the potential advantages and tradeoffs of MMQEF to be identified, including its complementary use with other frameworks.

MMQEF does not attempt to replace previous theories about quality in MDE nor establish another independent approach for evaluating quality issues.

MMQEF uses the classification theory on IS artifacts that are built with modelling languages. MMQEF promotes the use of taxonomic analysis as an approach that supports the identification and management of quality issues for model-driven projects. This complements the current corpus about approaches for evaluating quality in the model-driven paradigm. Therefore, as part of this work, MMQEF can be used in combination with other quality frameworks in MDE, showing how quality dimensions previously defined (e.g., in the SEQUAL framework) are reached by the application of taxonomic analysis.

Another further work related to the scenarios of application for MMQEF is the demonstration of the potential that the method has to design and manage transformations and mappings of models from previous reasoning about source-target modelling languages. The classification in the reference taxonomy complements decisions about translating concepts of languages, also allowing the underlying implementation that would support the computational achievement of the modelling effort to be identified.

7.2 The support of MMQEF to address the quality issues reported from industrial contexts

This work considers the alignment of MMQEF applicability with the industrial issues reported in Section 2.3.2. The prescriptive procedure defined by MMQEF to evaluate quality issues in the MDE paradigm provides a practical approach to address open questions that are commonly reported in industrial contexts regarding the scope of the MDE itself, the use of modelling languages at different (abstraction) levels, and technical issues that are associated with the application of modelling languages. Beyond the evaluation of quality issues, MMQEF can also be used as a guideline to systematically apply modelling languages in the context of IS development that is supported by conceptual models.

7.3 The consolidation of operative support for the MMQEF

One of the key MMQEF features is the explicit formulation of a technological framework to support the reasoning about modelling languages (Section 4.4). However, this feature was implemented as a proof of concept that demonstrated the technical feasibility for operationalizing the resulting semantic models. This further work considers the implementation of the main components of the architecture presented in Section 4.4.5, including the service for measuring the technical debt for using modelling languages. As part of this thesis, a derived work was proposed in (Giraldo et al., 2015b) which considered the issues for operationalizing quality frameworks (for that case, the PoN framework).

Related works about this topic are emerging (e.g., (Xiao He, 2016)), but these works focus on the technical implementation of technical debt calculation for models at the MOF (EMOF) level. When technical debt is calculated for source code, quality rules are established at the programming language level in order to address good practices in the use of the languages and to facilitate the

maintainability of the generated code. This way, the reported issues are detected in instances of programming languages (i.e., source code). In an analogous way, the rules for technical debt of modelling languages must be defined at the modelling language level and later applied to instances of those languages (i.e., models). MMQEF provides a source to identify and define rules for modelling languages.

7.4 The consolidation of a set of metrics for modelling languages

The MMEQF proposes a set of metrics that are derived from the classification of modelling languages in the bi-dimensional taxonomy (Section 3.3.5). This set of metrics can be viewed as an attempt to measure quality issues of languages from a superior viewpoint, i.e., it is independent of the specific scopes of each modelling language (which are also the source of specific metrics). Despite the usefulness of the metrics, this topic is still as a pending issue (or scientific issue as reported in (Le Pallec and Dupuy-Chessa, 2013)) for quality evaluation in modelling languages.

The consolidation of metrics for modelling languages is a key property for operationalizing evaluation procedures. The technical debt service mentioned above requires the specification about what the rules are and how the values that are associated to those rules must be automatically measured.

7.5 The management of interaction issues in modelling languages (HCI of modelling languages)

This work considers the testing of a hypothesis about the relationship between the use of modelling languages and the semantic information inferred from diagrams. This was inferred from the experiments with users of modelling languages (reported in Section 6), where participants used their previous knowledges of modelling languages to classify them. Here, they reported the *representations* as one of the main inputs to derive the use of modelling languages.

Current research in the diagram field (e.g., (Shimajima, 2015)) shows the presence of semantic elements in diagrammatic representations which could allow the model-driven field to derive additional information about the intention of use and communication of models resulting from modelling languages. This evidence supports the understanding and communication purposes of conceptual modelling (Loucopoulos and Zicari, 1992).

From an interactive perspective, representations are the *main interaction artifact* in a model-driven process. As an interactive artifact, the semantics could be reached by principles and models from *model-based user-interface development* approaches (MBUID (Luyten et al., 2004)). Their associated models (navigation models, interaction models, dialog models) provide mechanisms to manage complementary information in the diagram as the *interactive element* under construction.

These complementary models could provide the *data semantics* of MDA 2.0 for model analytics procedures. MMQEF supports the reasoning about the navigation and organization of modelling languages regarding their perceived use in accordance with a taxonomic structure for information systems.

Another derived work is about determining whether or not MMQEF allows establishing a common mechanism to support communication issues in an IS. It includes the pragmatic dimension of quality that is presented in quality frameworks such as SEQUAL (Krogstie, 2012b). An initial work is derived through analyses about the sufficiency of the MMQEF method to meet the communicative criteria previously defined in (Goldkuhl, 2011) for information systems.

7.6 The promotion of MMQEF as a *Type V Theory* for IS

In accordance with the taxonomy of theory types in IS research that was defined in (Gregor, 2006), the MMQEF method can currently be considered as a *Type IV Theory (explanation and prediction)* due to its support for the formulation of predictions, prepositions, and causal explanations about the use of modelling languages inside an IS development project. However, in the results obtained by the validation of the method with experts and students (Section 6.4), there is clearly a need to provide more prescriptive guidelines for performing the activities defined in the method and improving the inferences from the classification of modelling languages.

Further work is required to detail the specification of the task and activities of MMQEF (in the form of the *Type V theory - design and action says how to do something*) so that users of the method can make more precise inferences in quality evaluation procedures that are supported in the prescriptive descriptions of the method (e.g., by using artifacts and elements for software process deployment approaches such as SPEM).

Chapter 8

Final considerations



Current quality initiatives promote procedures for continuous improvement in the development projects of complex engineering products, such as Software and Information Systems. MDE addresses emergent challenges in engineering projects of this kind through the use of modelling languages to generate and manage models of the required concerns for these projects. Even though quality at the MDE level has been considered, the applicability of the methods for evaluating quality in MDE projects is still an open challenge. As a Software Engineering paradigm, MDE must provide practical procedures for evaluating the quality of artifacts and addressing quality issues.

In this research work, we have proposed a method for evaluating quality issues in MDE projects, the MMQEF method. Following a rigorous research procedure, we have demonstrated how the *quality* conception for MDE is ambiguous and highly dependent on the work of specific model-driven communities. We have also demonstrated how the taxonomical analysis (through the use of the Zachman framework for Information Systems) provides a practical mechanism to evaluate modelling languages by identifying and analyzing their scope, coverage, traceability, suitability, and their support for transformations. The specific IS framework that was used in this work is recognized as a reference architecture for Information Systems.

The method for evaluating quality at the MDE level that is proposed in this thesis is methodological, formal, and technologically supported. In addition, this proposal has been evaluated through a validation procedure with an important number of subjects. A relevant contribution of this work is the derivation of technological tools that support the analytic procedures of the evaluation method. With the MMQEF method that is proposed in this thesis, we are closer for making practical procedures for evaluating and guaranteeing quality in MDE, and, therefore, for supporting projects and practitioners that use the MDE paradigm and its associated environments.

8.0.1 Derived publications

The following works were derived from this thesis:

Journal article

- Fáber D. Giraldo, Sergio España, Óscar Pastor, and William J. Giraldo. Considerations about quality in model-driven engineering. *Software Quality Journal*, pages 1–66, 2016 (<https://doi.org/10.1007/s11219-016-9350-6>).
- Fáber D. Giraldo, Sergio España, Óscar Pastor, and William J. Giraldo. Evaluación de la calidad de lenguajes de modelado a través de análisis taxonómico: una propuesta preliminar. *Revista Ingenierías Universidad de Medellín*, pages 159-172, 2016 (<http://revistas.udem.edu.co/index.php/ingenierias/article/view/1498/1822>), DOI: 10.22395/riium.v15n29a10. This journal has the *A2* category in Colombia, i.e., the highest category for scientific journals that is assigned by COLCIENCIAS through the *Pub-Index* indexing service ¹.

In addition, to date, there are another three paper proposals that were derived from this thesis. These are submissions to journals (currently under review).

Book chapter

- Fáber D. Giraldo, Sergio España, Manuel A. Pineda, William J. Giraldo, and Óscar Pastor. Conciliating model-driven engineering with technical debt using a quality framework. In Selmin Nurcan and Elias Pimenidis, editors, *Information Systems Engineering in Complex Environments*, volume 204 of *Lecture Notes in Business Information Processing*, pages 199–214. Springer International Publishing, 2015 (https://doi.org/10.1007/978-3-319-19270-3_13).

Proceedings in conferences

- F.D. Giraldo, S. España, and Ó. Pastor. Analysing the concept of quality in model-driven engineering literature: A systematic review. In *Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on*, pages 1–12, May 2014 (<https://doi.org/10.1109/RCIS.2014.6861030>).
- Fáber D. Giraldo, Sergio España, Manuel A. Pineda, William J. Giraldo, and Óscar Pastor. Integrating technical debt into MDE. In *Joint Proceedings of the CAiSE 2014 Forum and CAiSE 2014 Doctoral Consortium co-located with the 26th International Conference on Advanced Information Systems Engineering (CAiSE 2014), Thessaloniki, Greece, June 18-20, 2014.*, pages 145–152, 2014 (<http://ceur-ws.org/Vol-1164/>).
- F.D. Giraldo, S. España, W.J. Giraldo, and Ó. Pastor. Modelling language quality evaluation in model-driven information systems engineering: A roadmap. In *Research Challenges in Information Science (RCIS), 2015 IEEE 9th International Conference on*, pages 64–69, May 2015 (<https://doi.org/10.1109/RCIS.2015.7128864>).

¹(<http://scienti.colciencias.gov.co:8084/publindex/>)

Technical reports

- Fáber D. Giraldo, Sergio España, and Óscar Pastor. Evidence of the mismatch between industry and academy in modelling language quality evaluation. *CoRR*, abs/1606.02025, 2016 (<https://arxiv.org/abs/1606.02025>).

8.0.2 Research collaborations

Some collaborations have been established. One of the most important collaboration efforts was with researchers of the Department of Computer and Information Science of the Norwegian University of Science and Technology, with whom the complementary application of the SEQUAL and MMQEF methods was explored. From this collaboration, one journal article proposal was obtained. Some talks about quality challenges in model-driven environments were given, specifically at the EAFIT University Colombia (2013), the University of Santa Cruz do Sul Brazil (2014) and the University of Medellín, Colombia (2015 and 2016). Assistants from areas such as Software, Computing, and Industrial Processes participated in these talks.

This work has also supported the formulation of an emphasis in Software Engineering for the Master in Engineering of the University of Quindío (Colombia). To date, four proposals for master theses have been derived from the topics of this thesis. In addition, this work has contributed to the strengthening of the research capabilities for Software Engineering and MDE topics in the SINFOCI Research Group of the University of Quindío (Colombia).

In August 2015, the Vice-chancellor's Research office of the University of Quindío funded a research project (for 18 months) in which the automation of quality evaluation procedures in modelling languages was addressed. This automation was done by using a platform for the calculus of technical debt in software projects (specifically source code), the SonarQube project². A plugin was added to the SonarQube platform to evaluate models from languages such as UML, BPMN, CTT, and ER. This research was useful in identifying the main challenges that are related to the operationalization of the quality evaluation procedures proposed by the MMQEF method, complementing the analysis performed by the EMAT tool.

In coming collaborations, we are planning the creation and consolidation of a MDE chapter in the Colombian Computing Society in order to integrate the academic and industrial efforts for MDE and model-driven initiatives in Colombia. In addition, some collaborations with Latin American researchers are being formulated to use the topics of this thesis, such as the taxonomic analysis for architectural tactics, the evaluation of specific modelling languages and proposals, and the application of MDE proposals governed by quality principles that are defined in MMQEF. Since October 2016, we have also contacted researchers from the Centre de Recherche en Informatique (CRI) of the Université Paris 1 Panthéon - Sorbonne (France) in order to explore the applicability of formal approaches to derive automatic inferences (and therefore automatic reports and

²Available in <https://www.sonarqube.org/>.

analytics of quality issues) from the concept lattices that are generated by the EMAT tool (Section 4.4.3).

Appendix A

A multiple modelling languages quality scenario



The following scenario is based on a real project from the University of Quindío (Colombia); the implementation of an information system for institutional academic quality management. This system includes all the resources, processes, technology platforms, and legal frameworks required to achieve the *institutional quality accreditation* certification, which is awarded by the Ministry of Education in Colombia to universities that demonstrate excellence in the exercise of their academic and research activities. The accreditation certificate is the result of an internal assessment process that was executed by members interested in the university.

With this modelling scenario we show how current quality proposals do not integrally cover some relevant issues in MDE projects. This modelling scenario helps to identify the applicability of some of the quality works on MDE identified in Sections 2.2.4 and 2.2.5 and emerging quality issues (Section 2.3.4) as a consequence of using modelling languages in the development of an Information System.

A quality evaluation proposal that comes from one of the primary authors identified in Section 2.2.5 was used to analyze this modelling context (the *Physics* of notations proposed in (Moody, 2009)). Even though the quality proposal meets its primary purposes in the analysis of the models and modelling languages involved, other quality issues emerge but they were not covered by the proposal. These issues influence the adoption of a model-driven initiative to manage concerns in information systems.

This information system is characterized by:

- The presence of multiple academic/administrative stakeholders from different areas of knowledge, participating collaboratively in the development of strategies for the generation/management of evidence according to the descriptive models of quality required, and the monitoring of the multiple sub-processes of quality instantiated in the university.

- The alignment with quality descriptive models that define the quality criteria. These include the self-evaluation guides issued by the National Accreditation Council (CNA)¹ under the Ministry of Education of Colombia, as well the ISO 9001 -2015 standard and the Colombian technical standard NTCGP 1000: 2009. The NTCGP 1000: 2009 is a management standard directed towards the evaluation of an institution' performance in terms of quality and social satisfaction during the delivery of services by government entities.
- The development of an organizational culture that is oriented towards the continuous improvement management of the university in the business processes. The support of this goal is the Integrated Management System², which is a web platform where the specification of processes, procedures, and associated institutional formats is published. The related application scenario is framed within the business process called self-assessment for the accreditation and re-accreditation of an undergraduate or graduate program.

A strategy for the collaboration between academic experts and researchers in information systems was developed for the design, construction and deployment of the information system. Its purpose is to formulate conceptual, methodological, and technological tools that support the processes of accreditation and assurance of quality. Each group used modelling languages to represent the phenomena of interest. The panel of experts in quality specified a model for academic quality process³ using a specific variation of the *Flowchart* diagram (a notation selected by those responsible for the integrated management system of the University Quindío to model the processes of the organization). The group of researchers in information systems employed the proper languages of software modelling and data to conceptually support the design and implementation of software platforms for different parts of the accreditation process. The use of different modelling languages for the process of design and construction of the academic quality system favors the process specification through the contributions of the parties involved (views). Three types of models were used in the conceptual modelling of the project:

- *Business process models*: This part of the application design focuses on the modelling of the processes undertaken at the University of Quindío, which are oriented towards business experts and the people who interact with the processes at the university. These models are intended for users of the processes that have no prior knowledge in order to facilitate the understanding of the processes.
- *Business and system models*: These models focus on the design and subsequent implementation of derived software applications to support the

¹<http://www.cna.gov.co/1741/channel.html>

²Available at http://www.uniquindio.edu.co/planeacion/publicaciones/sistema_integrado_de_gestion_1_pub

³Available at <http://www.uniquindio.edu.co/planeacion/descargar.php?idFile=19777>

information system of academic quality, where everything that a software system needs to fulfill customer requirements must be specified. UML models are employed using class, sequence, use case, state, and components diagrams. In addition, a proposal of stereotyped UML formulated by RUP (Kruchten, 2000) is used to model business processes by applying the business modelling discipline defined in this methodological framework. Researchers with different profiles made these models: experts in accreditation and academic quality processes, experts in software engineering, senior/advanced software developers, and data experts. A model-based approach is used to produce the source code of the applications from the models made by the researchers.

- *Data Model*: These models cover the design of the database required for the academic quality system using the core business concepts identified in the domain model made in UML (the class diagram with the most representative concepts of the business according to the business modelling discipline of the RUP). This type of design depends on the expert in data or DBA (Database Administrator) because of the complexity that data modelling can have.

The complexity that is inherent in the development of the academic quality system and the parties involved is the rationale for using multiple modelling languages to help fulfill the interests of each role that is in charge of the implementation of the information system at the University of Quindío. These modelling languages include:

- Flowchart: the language used for making the process flow diagrams.
- UML: the language used for the analysis and design of software.
- E/R: Models used for verifying the design of the database.

A.0.3 Application of multiple models

Fig. A.1 shows a partial view of the self-assessment process for *accreditation and re-accreditation purposes of an undergraduate or graduate program*. Fig. A.2 presents the adaptation of the flow diagram notation used in the specification of business processes for the University of Quindío. This view corresponds to the participation of the experts in the business information system and in the assurance of academic quality processes.

The modelling of business processes is done by using a notation that is particularly suited for experts in institution processes. This notation prioritizes simplicity and a small number of notational constructs to represent the process components accurately. None of the quality standards used for the implementation of quality policies (ISO 9001, NTC GP 1000, CNA) requires a specific graphic language; instead, these standards grant freedom for the modelling processes to be performed autonomously at the discretion of the organization.

Figures A.3 to A.7 present the conceptual models that are formulated by researchers and experts in information systems (mostly in UML) to address the

N°		SYMBOL	FLOW CHART			
			ACTIVITY / DESCRIPTION	RESPONSIBLE (S)	DOCUMENT AND/OR EVIDENCE	OBSERVATIONS
1	START		START			
2	ACTIVITY		Evaluate quality parameters to access the self-assessment process and developing preliminary program documents: historical, legal, or other academic processes.	Curriculum Council	Take into account the new CNA regulations about it.	CURRICULUM COUNCIL MINUTES
3	ACTIVITY		Forward the concept evaluation to the Faculty Council	Chair of Department Code 95 Degree 9		MEMORANDA
4	ACTIVITY		Analyze and discuss the evaluation concept	Faculty Council		FACULTY COUNCIL MINUTES
5	DECISION		If it is necessary, adjust the evaluation concept based on the recommendations of the Faculty Council.	Curriculum Council		CURRICULUM COUNCIL MINUTES
6	ACTIVITY		Refer to the Academic Vice-chancellor to formalize the inscription for the self-assessment accreditation process.	Curriculum and Faculty Councils		MEMORANDA
7	ACTIVITY		Analyze the documents submitted by the Faculty Council and the Curriculum Council.	Central Committee for Academic Quality or in lieu thereof.		
8	ACTIVITY		Submit the analysis results to the Curriculum Council of the respective department.	Central Committee for Academic Quality or in lieu thereof.		MEMORANDA

FIGURE A.1: Current self-assessment process diagram for the University of Quindío (partial view).

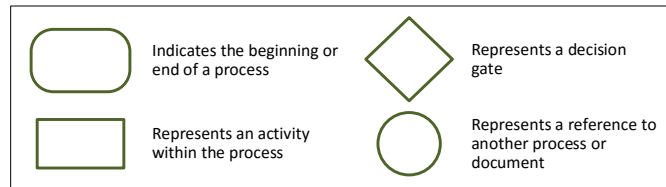


FIGURE A.2: The conventions used for the flowchart adaptation at the University of Quindío.

various considerations associated with academic quality and the derived software platforms (publication of information related to academic quality processes, document management framed in quality contexts, document distribution of quality processes supports, and management of activities).

Due to the methodological alignment with RUP, a UML profile is used for business modelling. Then, the researchers formulate system models. The following models belong to the module of *Memoranda Management System within the Context of the Information System of Institutional Accreditation*⁴.

Business modelling models

In order to understand the organization (i.e., detect current problems, identify improvement potential, identify users, workers, and parties, etc) several stereotyped UML models were employed following the RUP methodological framework (Figures A.3 and A.4). Fig. A.3 -part A - shows the model of *business use cases*. This model illustrates the organization by management process areas of the university. Related business processes are identified as use cases (in light blue). For purposes of readability, they are grouped using standard UML packages. The

⁴Figures A.3 to A.8 show the current application of models in the *Information System of Institutional Accreditation* project. For this reason, these diagrams are presented as they are currently in use. Copyright©SINFOCI Research Group, University of Quindío, 2015.

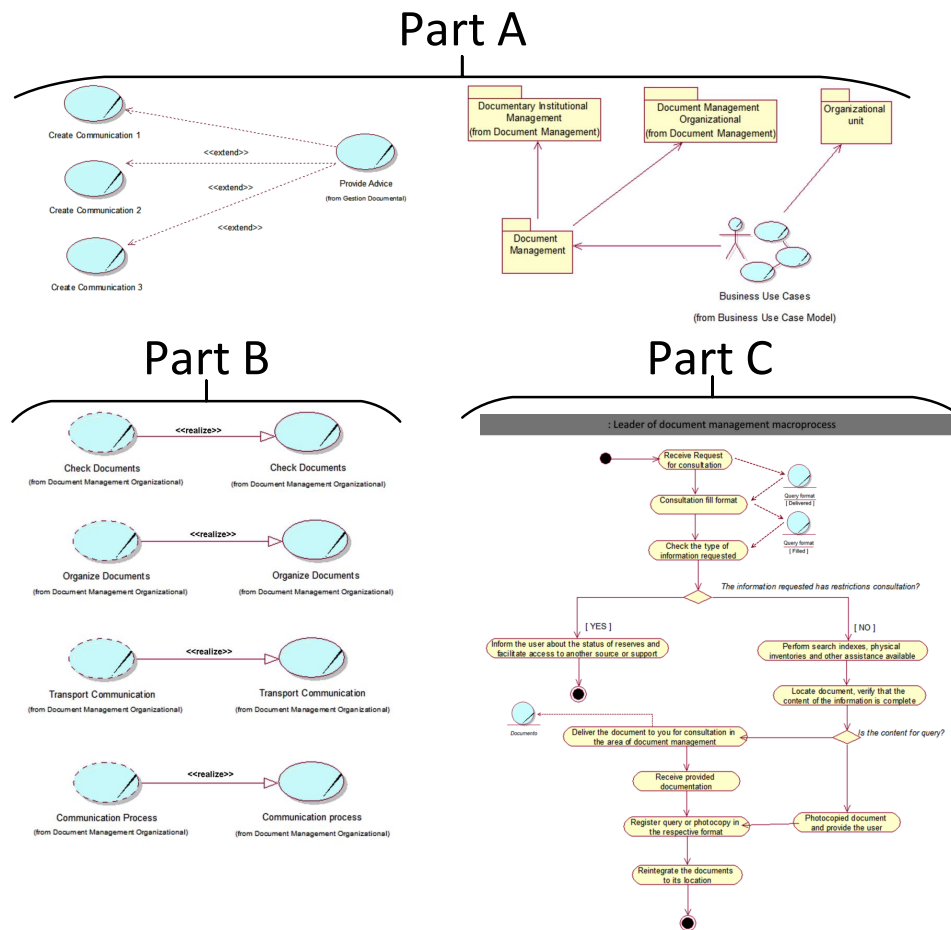


FIGURE A.3: Business modelling models (I)

business use case is a modelling of each business goal and its respective roles. It is used to identify the roles and different deliverables of the works performed.

The model of business use case also contains the business use cases realization (Fig. A.3 -part B) as part of the business analysis model defined in RUP. A realization of a business use case describes how the workflow is in terms of the business objects and their collaboration. A diagram of activities and a diagram of business objects are defined in the realization of a business use case.

The business process model (Fig. A.3 - part C) is a set of logically related tasks that are carried out to generate products and services. A stereotyped UML activity diagram represents this model, where the business entities that are involved in the process tasks are also identified.

The business modelling discipline of RUP considers all the *things* or *something of value* that are observable during the performing of business processes. For this, researchers used the models shown in Fig. A.4. The business entity model (Fig. A.4 - Part A) represents an important part of the information that is handled by business actors and business workers. The business object model

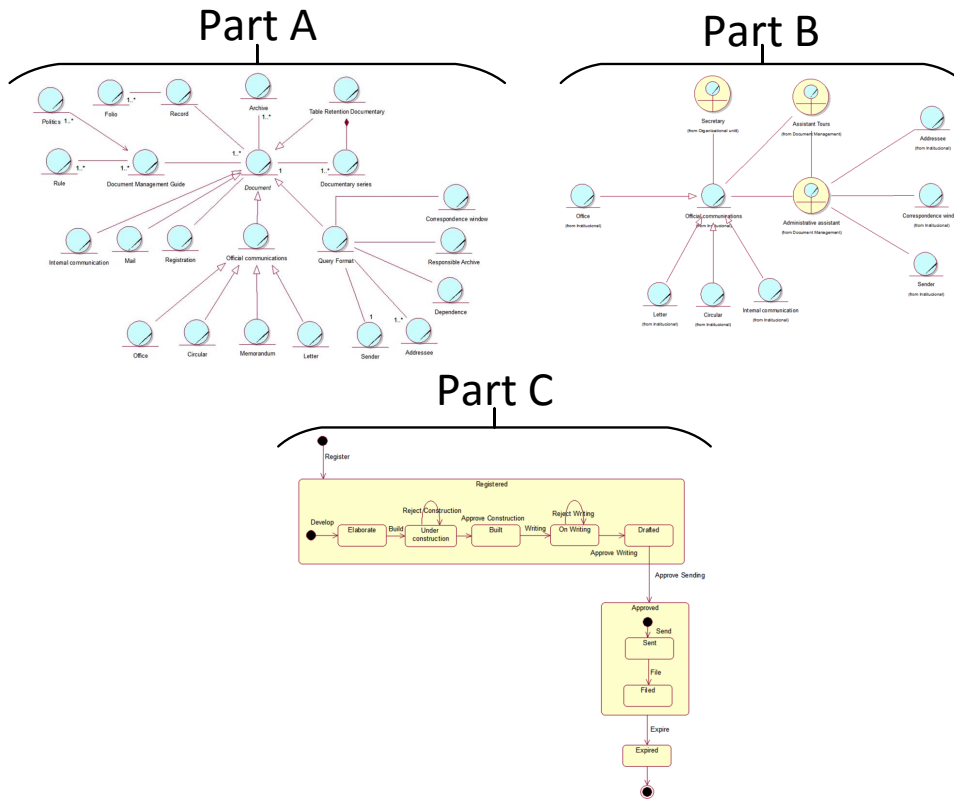


FIGURE A.4: Business modelling models (II)

(Fig. A.4 - Part B) shows the relationship between the business entities associated with different business use cases and the workers associated to those cases. The model serves to show the limits of the business process considered in each business use case.

Finally, a state machine model is used to define the life cycle of the information entities at the University of Quindío. Each state considers a set of specific software features to manage the state associated with an entity at any time during the execution of the process. Fig. A.4 - part C - shows a sample lifecycle for a communication in the context of academic quality.

System models

Once the definition of business processes has been completed, use cases are derived at the software system level by a relationship of traceability whose origin is found in automatable activities of the business process analyzed.

Fig. A.5 - part A - partially shows the features that are implemented for the module of *memoranda management* software of the information system for academic quality. Models of system classes (Fig. A.5 - part B) generate the associated source code (logical view of the application) and sequence diagrams (functional allocation of responsibilities among objects) of Fig. A.5 - Part C.

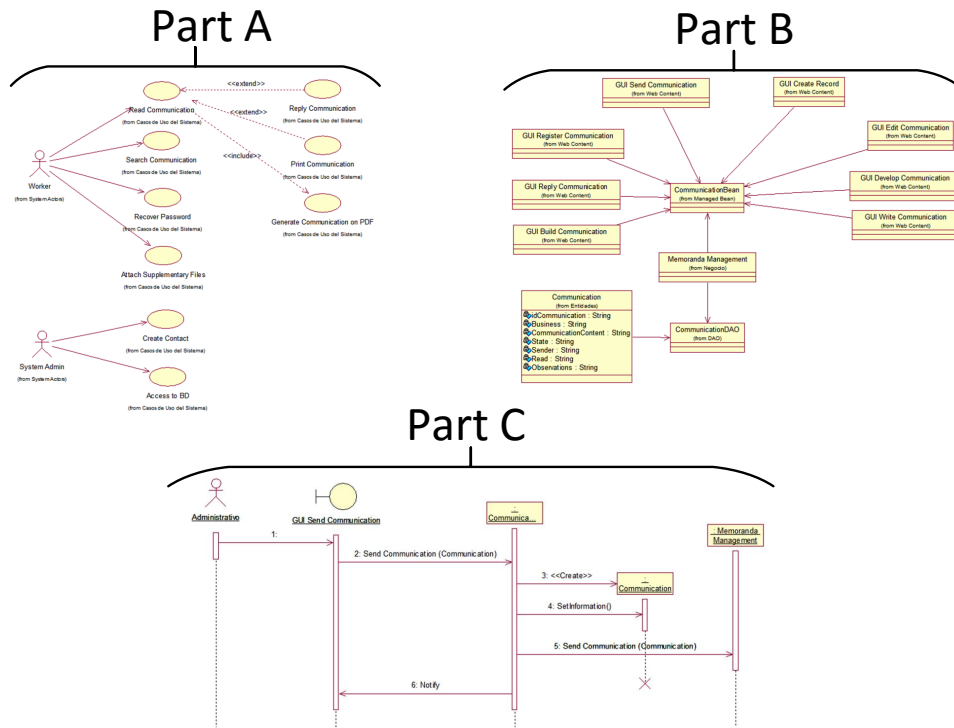


FIGURE A.5: System models (I)

These diagrams (along with their associated specification) are delivered to the project developers who generate the source code in the platforms and development environments that are defined by the technical experts.

Other non-UML systems models were used to conceive and manage specific system views of the *Information System of Institutional Accreditation*. Fig. A.6 shows the *Data model* in the E/R notation. Due to the relational support used in the technological implementation of the modules associated with the quality system, a conceptual representation of the entities associated to the domain addressed by each module is made. This conceptual representation defines the semantics associated with the entities, the consistency constraints at the data level in order to preserve the integrity of the module once it deploys organizationally.

Additionally, as part of the process of architectural decision-making for developing software modules, models elaborated in informal notations are used to address problems associated with specific quality attributes and to facilitate the identification of architectural tactics in the management of these attributes. Fig. A.7 shows an example of a diagram that was developed to discuss the aspects of global integration and the consistency of the information system (taking into account the presence of multiple software modules). The aim of these diagrams is to facilitate the description of architectural alternatives in the consultation and judgment processes so that the consequences and impact of each architectural strategy formulated are easily addressed.

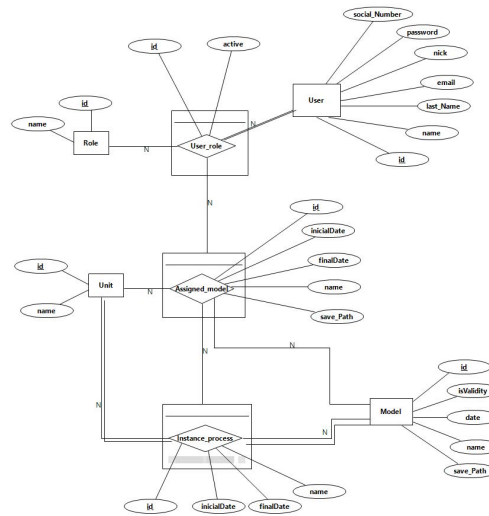


FIGURE A.6: System data model (partial view).

Finally, Fig. A.8 depicts the software products obtained from the conceptual models identified by the researchers to support specific elements of the academic quality system.

A.0.4 The first signs of quality problems

The first signs of quality problems associated with the use of multiple models and different modelling languages can be observed. The first problems can be found by analyzing the visual language used by experts and organizational stakeholders to represent the business processes of the university, since it is the self-assessment process for accreditation and re-accreditation of an undergraduate or graduate program.

The researchers decided to evaluate the graphical notation using the theory of Physics of Notations (PoN) by D.L. Moody (Moody, 2009), which is the most frequently published. The application of this theory provides a scientific basis for comparison, evaluation, improvement, and construction of visual notations used in an organization. The PoN theory proposes nine principles that can be successfully used to assess visual languages of graphic modelling (*Cognitive Integration, Cognitive Fit, Manageable Complexity, Perceptual Discriminability, Semiotic Clarity, Dual Coding, Graphic Economy, Visual Expressiveness, and Semantic Transparency*).

The institution does not use a standard visual language for modelling its business processes. The variant of the flow chart used by the university in the modelling of its processes does not preserve the semantics that is used for this type of notation, which causes the process model to be unclear for the roles that interact with them. Thus, the application of PoN helps validate the flowchart version created in the institution by applying the principles that this theory proposes.

This type of graphic language is not suitable for modelling business processes or complex systems because of its simplicity. In these cases, it is possible to find

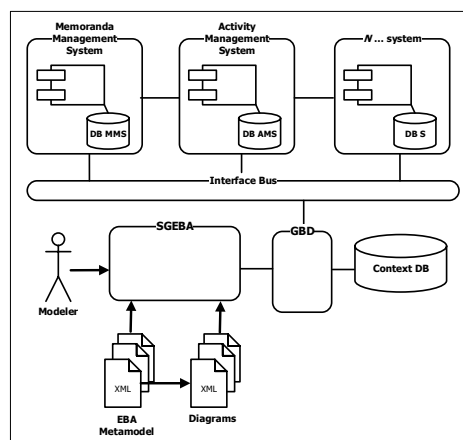


FIGURE A.7: Diagram example for the rationale of an architecture decision.

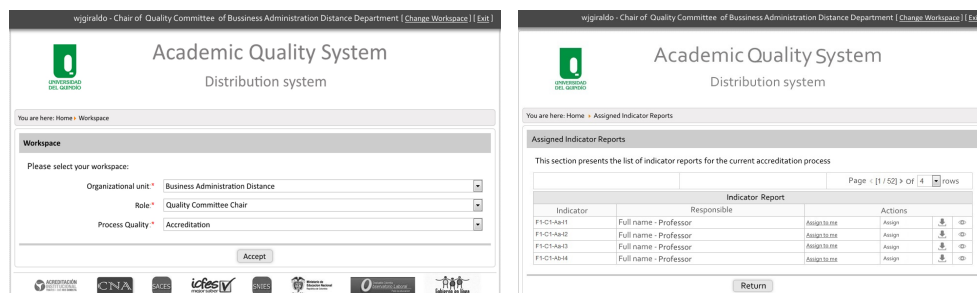


FIGURE A.8: Examples of software products obtained from conceptual models.

many other languages that are also appropriate such as BPMN or UML activity diagram. However due to the lack of knowledge about different alternatives for process modelling, the migration of these processes to other languages has not been done.

The application of the PoN principles in the flowchart diagram variant used in process modelling at the University of Quindío is presented in the following sections.

Semiotic clarity

This principle establishes a one-to-one correspondence between the semantic constructions and the graphic symbols of visual language. When there is not a one-to-one correspondence between the analyzed symbols and their respective semantics, at least one quality problem generated in the notation which is related to *Symbol Deficit*, *Symbol Redundancy*, *Symbol Overload*, or *Symbol Excess*.

Fig. A.9 shows the analysis of notational elements employed in the variant flowchart applied at the University of Quindío compared to the original semantic constructs from the flowchart. The simplicity that is applied at the University

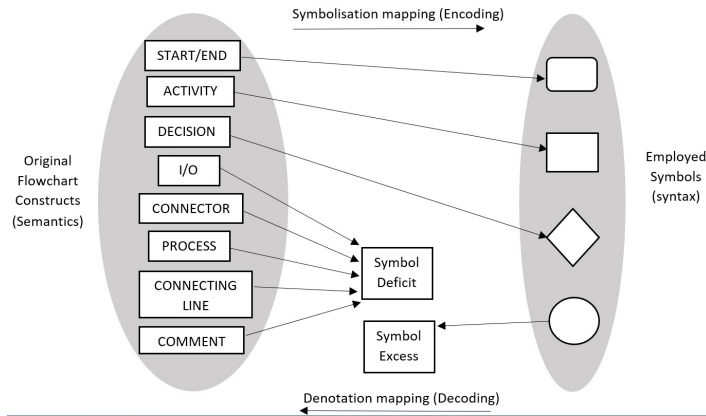


FIGURE A.9: Principle of Semiotic Clarity: there should be a 1:1 correspondence between semantic constructs and graphical symbols

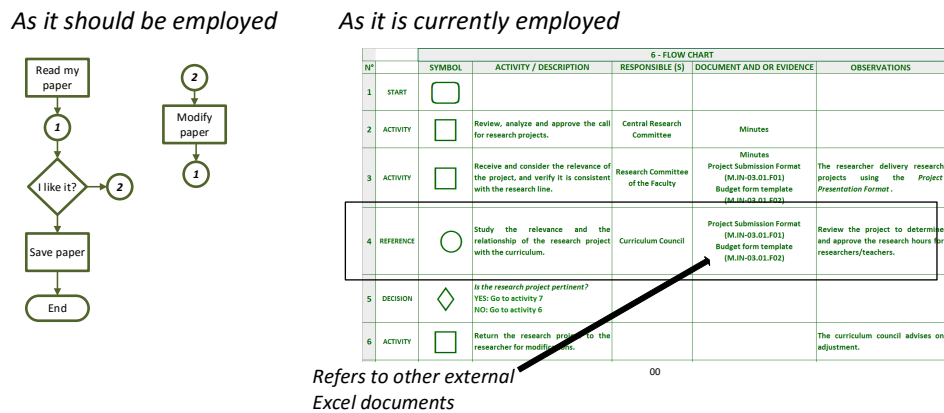


FIGURE A.10: Comparison between the symbols used at the University of Quindío and the symbols used in the semantic construct of the flowcharts

of Quindío for conducting the flowcharts is shown in this analysis because not all the symbols originally formulated by the notation are used. As a result, out of the 16 original notation symbols contained in the university flowchart, only 3 symbols that have the same semantic construct and another construct with a different meaning are used. This analysis found two specific anomalies regarding the principle of semiotic quality, *Symbol Deficit* and *Symbol Excess*.

The *Symbol Deficit* anomaly found represents the lack of 13 symbols by the university in order to meet the standards of a flowchart. For the *Symbol Excess* problem, the use of the visual element **internal connector** is contrasted (Fig. A.10), identifying the meaning given in the description of processes of the University of Quindío and its original semantics according to specifications of the flowcharts (ISO, 1985).

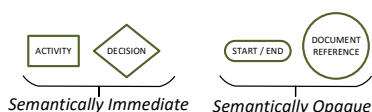


FIGURE A.11: Symbols used in the university that meet the *semantically immediate* / *semantically opaque* categories.

Perceptual Discriminability

This principle is related to the ease and perception with which the symbols used in a graphical notation can be distinguished from each other. Although this principle is supported by the specific adaptation of the flowchart conducted at the university, the main problem found in the analysis of perception is simplicity due to the number of symbols used. This can be seen as something that is relatively handy when making model interpretation of the business process. However, given the complexity of a business process of an organization, it is not feasible to conduct a modelling with so few symbols, since it loses too much of the useful information that provides a better understanding and proper execution of the process.

Semantic Transparency

The principle of semantic transparency refers to the ease of identification of the semantic meaning of a symbol that is used in a graphical notation. This principle considers four possible classifications for the analyzed symbols of the visual language:

- *Semantically Perverse*: When the symbol is observed, it is not easy to identify its meaning.
- *Semantically Opaque*: When the symbol is observed the person arbitrarily relates it to something known in order to identify its meaning.
- *Semantically Translucent*: In order to know the meaning of the symbol, the person requires prior explanation.
- *Semantically Immediate*: The meaning of the analyzed symbol can be identified easily without prior explanation.

The notation used for the modelling of processes at the University of Quindío identifies two semantically transparent symbols (Fig. A.11 - left) since they preserve the semantic construct of the flowcharts. Thanks to this, it is easy to identify their meaning (*semantically immediate*). However the presence of the *semantically opaque* category is also evident (Fig. A.11 - right) because the users of the business process (when noting some of the symbols by intuition and perception) relate what they observe to any known symbol. This gives a meaning that is not generally correct. At the University of Quindío there are symbols for start/end, and there is another symbol for referring to documents or processes.

Visual Expressiveness

The principle of expression evaluates the number of visual variables used and the range of values (capacity) of these variables. It considers the use of space of graphic design and the variation in the whole visual vocabulary. Table A.1 presents the identified values for the variables associated with this principle for the language used in the modelling processes of the University.

TABLE A.1: Visual variables of the flowchart notation used at the University of Quindío

Visual Variable	Usage
<i>Shape</i>	The visual language uses default figures in Excel (simple figures associated with 2D Flowchart Diagrams).
<i>Brightness</i>	The activities are represented by squares in bold. The events and decisions are represented by a box with rounded corners and diamonds, respectively.
<i>Spatial Location (x,y)</i>	Each symbol is located in each cell of the table generated in Excel. Guidance does not use arrows or lines.
<i>Size</i>	The symbols have a predetermined size that cannot be modified.
<i>Colour</i>	Use of symbols in white with green edge.
<i>Texture</i>	Not Used
<i>Orientation</i>	Not Used

The main abnormality is the lack of guidance (arrows, lines, or useful symbols) to denote the process flow of the diagram, which restricts the browsing in the business process modeled. This reduces the diagram to a top-down sequential specification. The sharp demarcation in the application of colors creates identification problems for parts of the process, which affects its cognitive assimilation.

Complexity Management

This principle evaluates the ability of visual languages to present large amounts of data without overloading the human mind. This principle refers to schematic complexity, which is based on the number of elements (instances or symbols) used in the diagrams. When analyzing this principle on the models of the business processes of the university, a high level of complexity due to the high number of activities (see Fig. A.1) is presented. This hinders the understanding and implementation of the process. To reduce the levels of schematic complexity in models of business processes, subprocesses are generally used to group activities. This minimizes the number of symbols used in the modelling of the process and achieves a better understanding of the workflow.

Dual Coding

This principle measures the use of text and graphics that are used together to transmit information. Specifically, the use of labels (text) plays a critical role in

the interpretation of business diagrams since it defines and clarifies the semantics of the processes directly on the diagrams (i.e., the correspondence with the real-world domain).

The symbols used in the modelling of business processes at the University of Quindío have text labels to help interpret the flowcharts. The graphics used are inside Excel cells, which have several adjoining cells with associated text that provide information for the people who interact with these diagrams. The main drawback of these diagrams used is their excessive emphasis on the textual representation (Fig. A.1). The visual elements fulfill a decorative function instead of a reasoning and communication function about the business process itself. The interpretation and expressiveness of the process models are directly affected by the excessive simplicity of the notation. The text itself becomes the central element of each diagram.

Graphic Economy

This principle states that the graphical complexity of a notation must be cognitively manageable. The number of visually distinct symbols of the notation indicates the complexity of a chart. This principle is critical to help the understanding and expressiveness of process models. The graphical notation used at the University of Quindío is too minimalist (there are only 4 symbols out of the 16 originally specified in the flowcharts). This makes it less useful for the modelling of systems or complex processes given their lack of semantic support from the specific syntax employed.

A preliminary application of the PoN method identifies the shortcomings of the modelling language that is currently used at the University of Quindío. This application highlighting its simplicity for the specification of the process models since the flowcharts do not meet the requirements for the modelling of processes and complex systems.

A.0.5 Limitations of the selected approach to evaluate the quality of the models of the modelling scenario

The processes for the management of academic quality are highly changing and dynamic, mainly because of regulatory updates from the authorities that govern academic quality in Colombia (the Ministry of Education and CNA). These changes affect organizations that voluntarily apply for accreditation processes, as is the case of the University of Quindío. Additionally, there are specific organizational conditions (administrative restructuring, updating of procedures, involvement of experts from different areas of knowledge, etc.) within the institution that affect the quality process models, which in turn affect the models that conceptually support the information systems generated.

The office of Planning and Development of the University of Quindío starts the exploration of a strategy of business process management using the BPM discipline with its associated notation (BPMN). To do this, in conjunction with the researchers involved in the project, a systemic approach for the selection of BPM tools (commonly known as BPM Suites) applied. This assessment was reported in (Gallego et al., 2015). Once the most suitable BPM Suite for the

institution was selected, the researchers formulate an initial proposal in BPMN for the business process of self-evaluation from the specification presented in Fig. A.1. A model containing 14 roles, 67 activities, and 67 attachments was obtained.

The proposed model was presented to them. Both the experts and the people from the planning department had difficulty understanding the model due to the high cognitive load and information present in the diagram generated. As reported in (Gallego et al., 2015), the researchers formulated an intervention to the original specification of the model to facilitate understanding by the business experts. This clearly shows the emergence of quality issues such as expressiveness, understandability, completeness, and appropriateness of the models.

From the perspective of researchers in information systems, the system models in UML and other languages (with their conceptual support) contribute to the creation of communication scenarios and documentation on which they make decisions that are related to a specific technological implementation. The modelling tools that are used support the automatic generation of source code (MDD). However, the emphasis on conceptual modelling of the different components of the information system require an extra effort for their subsequent translation into a specific platform of implementation. This is due to the particularities that must be developed in order to support the essential features of any model that formulated in the project on that platform.

Despite the considerable number of system conceptual models generated by the research group (especially the use of the UML profile for business modelling), their importance was perceived with relative apathy by the business experts at the University of Quindío. This was mainly due to the lack of alignment between the models of the information system and the specification of the models of organizational processes. Although the generation of information system platforms was delegated to the researchers because of the innovative nature of the conceptual models used to develop an information system for academic quality, the system models are limited exclusively to the use of roles for analysts, designers, and software developers. Therefore, in order to avoid suspicion and loss of confidence in the system models by the business experts and the users of the self-assessment process, the development team had to generate incremental versions of the components of the information system modelling. This produced software solutions that allowed the users and people involved in the self-evaluation process to appreciate the feasibility of the innovative proposals made by the researchers. In this case, the models contained reference information to support implementation decisions, but they were not used to automatically generate the underlying infrastructure of code (a model-based approach instead of model-driven one was used).

While models in this project played a strategic role at the organizational and conceptual support level of an information system with computational implementation, there was a decoupling between the organizational modelling and the system modelling. This caused duplication of the modelling effort and lack of mechanisms for traceability that covered the evolution of business aspects for their respective technological implementation.

In the implementation at the University of Quindío the understandability of

models was important, but there are still other questions that remain open. For example, the suitability of UML models to address organizational concerns are not covered by current modelling efforts at the University of Quindío.

Appendix B

The process-delivery diagram (PDD) specification for the MMQEF method



Continuing with the specification started in Section 3.3, in this appendix, we show the associated activities and concepts required for the MMEQF method in more detail. Tables B.1 to B.6 provide a more thorough description of the activities associated to the evaluation of quality of modelling languages through the reference taxonomy. Following the PDD approach, each activity block is mapped to a *concept diagram*, which contains the key elements involved in the taxonomic analysis. Concepts depicted in gray describe terms that are extracted from previous foundational models, such as the MDA foundation model (Object and Reference Model Subcommittee of the Architecture Board, 2010) and the ISO 42010 (612, 2011) conceptual models.

In addition, Tables B.7 and B.8 describe the concepts involved in the taxonomic evaluation procedure. The main purpose of these tables is to show that the main concepts are from relevant referents for the MDE conceptualization. Few concepts are required to understand the full applicability of the taxonomic analysis.

TABLE B.1: Description of activities related to the *Determine the organization of the modelling language* block.

Activity	Sub-activity	Description
Determine the organization of the modelling language	Associate the elements of the modelling language under analysis to taxonomy cells	Each one of the input elements of a modelling language, either REPRESENTATIONS or ABSTRACT SYNTAX, are located in a cell or a set of CELLS based on the closeness of the element with the purpose of the CELLS (i.e., the FOUNDATIONAL CONSTRUCT of the CELLS with the associated ABSTRACTION LEVEL.)
	Relate the information extracted from diagrams to the essential model of the analyzed cell	Key concepts associated to the INFORMATION depicted by REPRESENTATIONS that belong to modelling languages are contrasted regarding the scope of the essential model that governs the unit taxonomic (CELL or set of CELLS). Key concepts could be either conceptual entities or operations.
	Resolve the compliance of the modelling language elements with DSL cell constructs	A rationale about why the information of the element (REPRESENTATION, ABSTRACT SYNTAX) meets the essential model that governs the specific CELL(s) involved.

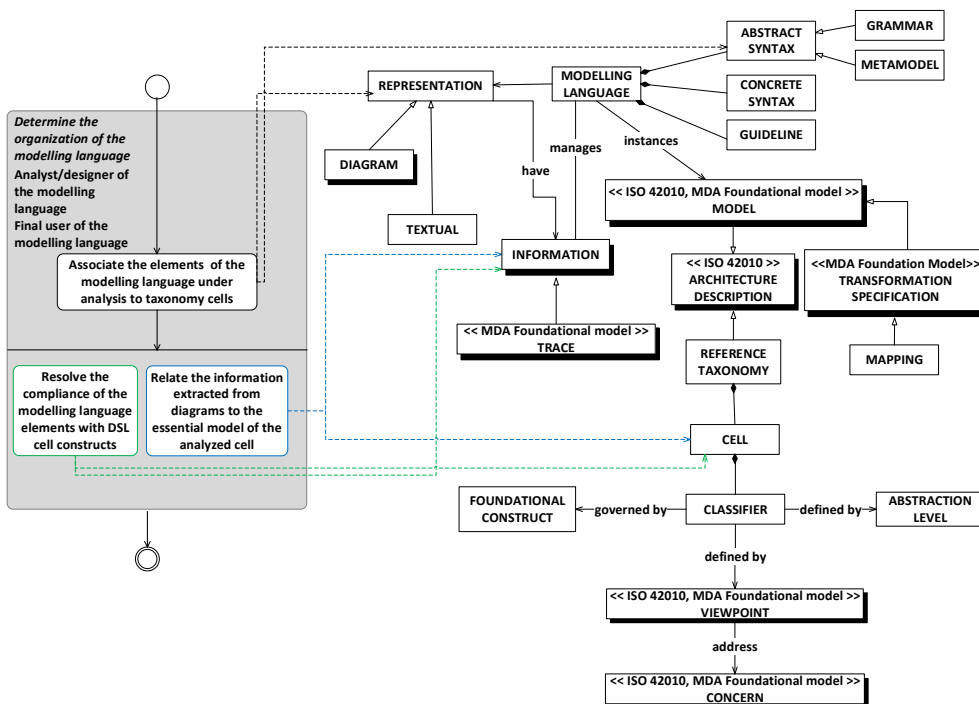


FIGURE B.1: Association of activities of the *Determine the organization of the modelling language* block with concepts of the MMQEF metamodel.

TABLE B.2: Description of activities related to the *Identify the explicit traces that support the navigation between abstraction levels / viewpoints* block (Part I).

Activity	Sub-activity	Description
Identify the explicit traces that support the navigation between abstraction levels / viewpoints	Find the traceability mechanism	These are the capacities of the language for supporting the incremental evolution of FOUNDATIONAL CONSTRUCTS inside a specific column (VIEWPOINT), checking how the FOUNDATIONAL CONSTRUCT of the VIEWPOINT is preserved when it passes through all ABSTRACTION LEVELS that the LANGUAGE supports. For example, in the <i>What</i> VIEWPOINT (data FOUNDATIONAL CONSTRUCT), a traceability link is from the <i>Domain model – conceptual model – ER entities – Tables</i> in a SQL engine. Thus, TRACES are from MODELS belonging to the same VIEWPOINT and some ABSTRACTION LEVELS. If the TRACES are from PIM-PSM levels these are a specification of a MAPPING.
	Identify the support for each abstraction level supported	Key concepts associated to the INFORMATION depicted by REPRESENTATIONS that belong to modelling languages are contrasted regarding the scope of the essential model that governs the unit. For each of the modelling artifacts under analysis the TRACES and MAPPINGS relations are checked in order to determine the support that the modelling artifacts provides to the MDA levels (CIM, PIM, PSM). This activity checks the relationships between elements of a MODELLING LANGUAGE that support multiple ABSTRACTION LEVELS in order to determine the explicit MAPPING between MODELS of different levels.

TABLE B.3: Description of activities related to the *Identify the explicit traces that support the navigation between abstraction levels / viewpoints* block (Part II).

Activity	Sub-activity	Description
Identify the explicit traces that support the navigation between abstraction levels / viewpoints	Identify explicit mechanisms for taxonomic meta-model preservation	This verifies how organizational-domain and system concerns are traced until specific technical implementations. In this way, the execution of model transformations are in accordance with the semantic domain where the modelling act occurs. It helps detect whether the reasoning about the relations are a consequence of progressive preservation of semantic constructs (that is added to incremental INFORMATION of the respective level in a top-down path). In addition, the relation could be the result of semantic changes introduced by considering at least two different viewpoints (e.g., looking for the support of a modelling artifact in the same row or ABSTRACTION LEVEL of the REFERENCE TAXONOMY).
	Reason about the justification of multi-viewpoint supported relationships	When the resulting organization of model elements of the REFERENCE TAXONOMY indicates some relationships between INFORMATION in different viewpoints (two or more), supported by a derivation rationale, a justification about why this derivation is compliant with the FOUNDATIONAL CONSTRUCTS of the involved CELLS is required. This analysis makes it possible to identify what the contribution is for making a single model of each ABSTRACTION LEVEL from the classified MODEL ELEMENT.

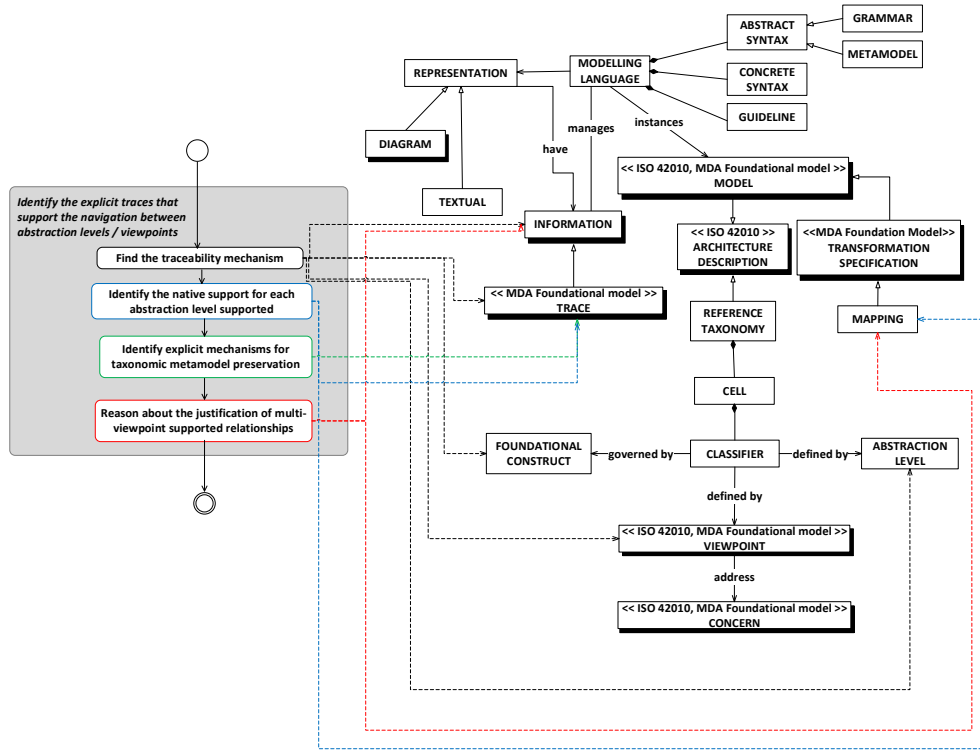


FIGURE B.2: Association of activities of the *Identify the explicit traces that support the navigation between abstraction levels / viewpoints* block with concepts of the MMQEF metamodel.

TABLE B.4: Description of activities related to the *Identify the capacities for model transformations* block.

Activity	Sub-activity	Description
Identify the capacities for model transformations	Find the support for transformation specifications	This activity promotes the specification of TRANSFORMATION SPECIFICATION MODELS from the semantic closeness of concepts belonging to the ABSTRACT SYNTAX of the involved MODELS or modelling artifacts. The semantic closeness is the explicit association of the involved concepts with the associated FOUNDATIONAL CONCEPTS of the CLASSIFIERS from CELLS.
	Determine the support for mappings	It verifies if MODEL ELEMENTS classified at the System level of the REFERENCE TAXONOMY can generate MODEL ELEMENTS at the Technology level of the same VIEWPOINT of the taxonomy.
	Define the explicit computational support of the modelling language	According to the TRACES mechanisms provided the MODELLING LANGUAGE; this activity checks them in order to identify the respective computational support (code, logical or physical computational artifact) for conceptual MODELS from higher levels.

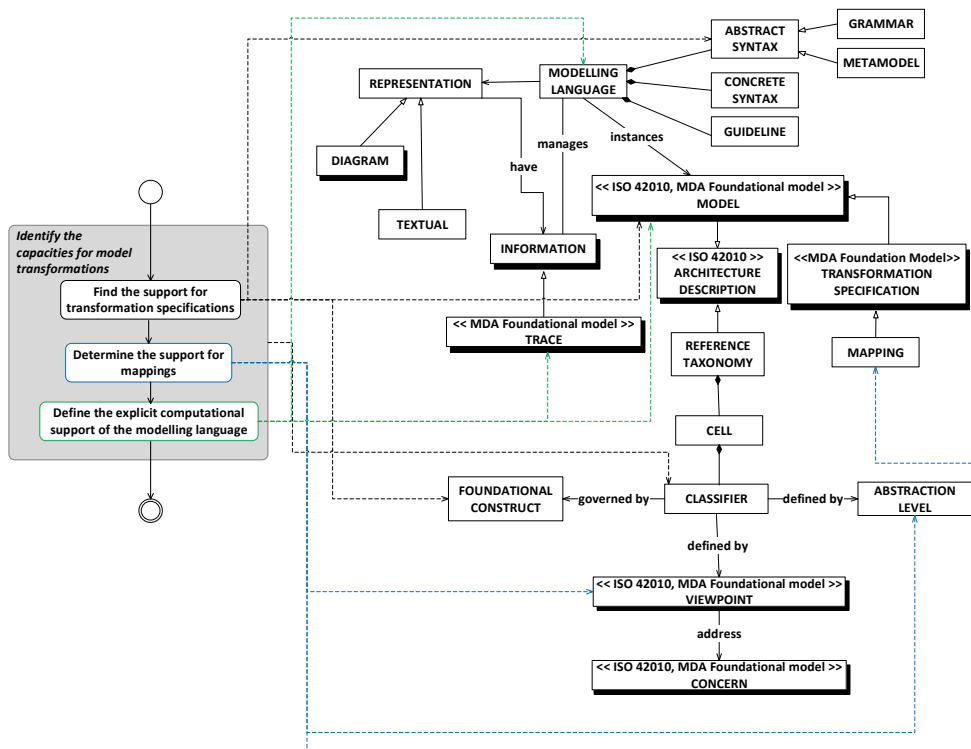


FIGURE B.3: Association of activities of the *Identify the capacities for model transformations* block with concepts of the MMQEF metamodel.

TABLE B.5: Description of activities related to the *Find the mechanism for integration* block.

Activity	Sub-activity	Description
Find the mechanism for integration	Identify explicit viewpoint integration support	This reviews the capabilities offered by the MODELLING LANGUAGE for integrating it with other languages and verifies that a modelling artifact focuses only on an IS CONCERN so that new related concerns will be managed by other more suitable modelling mechanisms.
	Establish the degree of taxonomic independence	This verifies that the proposed classification for a MODELLING LANGUAGE, using the REFERENCE TAXONOMY, is independent of previous classifications formulated for the same language through its analysis on its CONSTRUCTs or INFORMATION extracted from REPRESENTATION. It establishes the clearly differentiating features for the language which serves to justify its applicability over a set of IS CONCERNS.
	Define the coverage for each abstraction level	This determines how much information can be captured by the MODELLING LANGUAGE for an IS project. The amount of information is calculated from the number of CELLS covered by the MODELLING LANGUAGE through its classified elements.

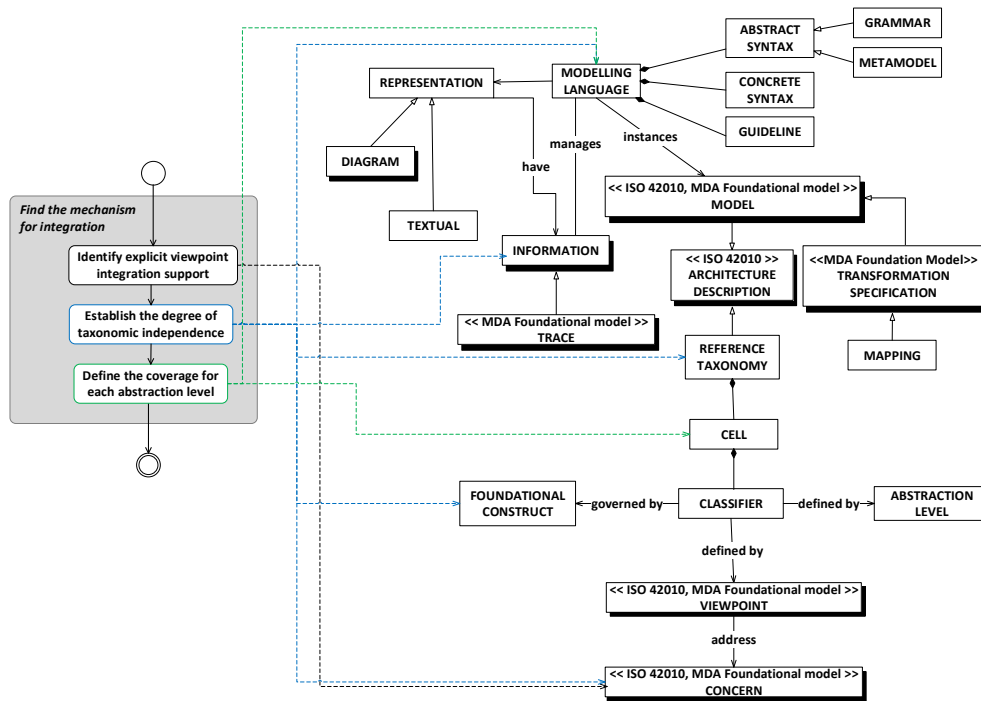


FIGURE B.4: Association of activities of the *Find the mechanism for integration* block with concepts of the MMQEF metamodel.

TABLE B.6: Description of activities related to the *Define suitability issues* block.

Activity	Sub-activity	Description
Define suitability issues	Define semantic commonality between languages	For all the MODELLING LANGUAGES that share a CELL in the taxonomic analysis, the specific elements that produce the classification in the involved CELLS are identified.
	Select modelling alternatives	This is a decision about the best alternatives for modelling an IS CONCERN based on the coverage supported by the MODELLING LANGUAGES. Prioritization of decisions are based on the total support for a CONCERN managed by a CELL or the coverage of CELLS.

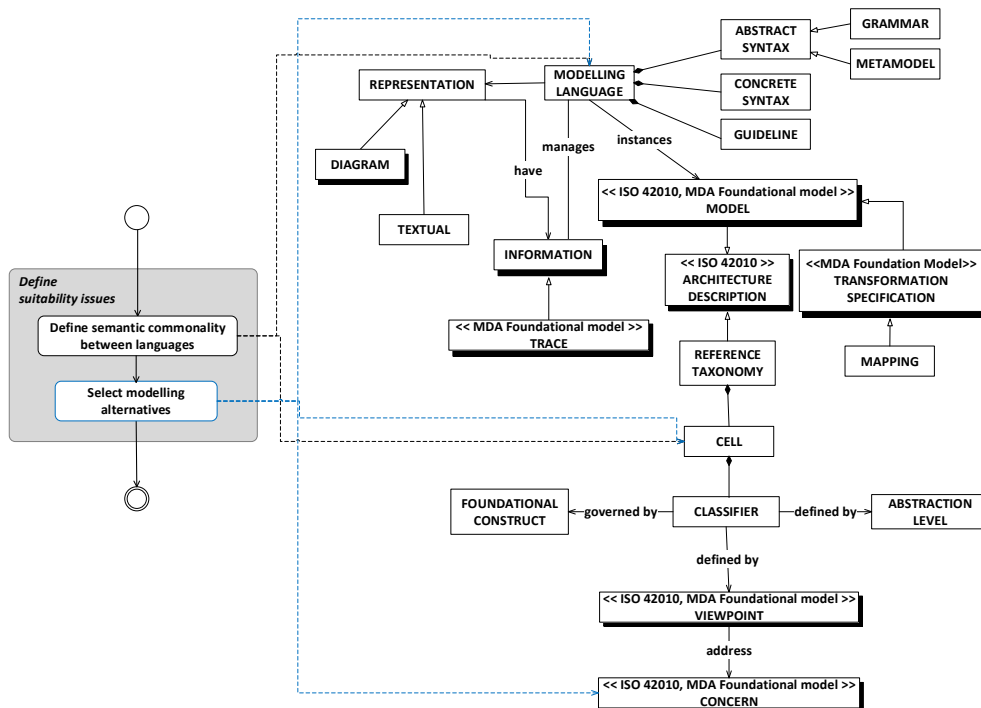


FIGURE B.5: Association of activities of the *Define suitability issues* block with concepts of the MMQEF metamodel.

TABLE B.7: Concept table for the MMQEF method in PDD convention (I).

Concept	Description
VIEWPOINT	From (612, 2011; Object and Reference Model Subcommittee of the Architecture Board, 2010), it represents the criteria and the set of conventions used for formulating views and especially for framing CONCERNS.
CONCERN	From (612, 2011; Object and Reference Model Subcommittee of the Architecture Board, 2010), this is the interest in a system of a stakeholder.
ABSTRACTION LEVEL	It refers to a restriction proposed to model and manage specific phenomena on an IS. The MDA architecture specification defines some abstract hierarchical level (Computational Independent Model CIM- Platform Independent Model PIM- Platform Specific Model PSM- Technical Implementation), which can frame the models used in a model-driven project.
FOUNDATIONAL CONSTRUCT	Enumeration that represents each of the single models associated to each column of the taxonomy: What (thing), How (process), Why (purpose), Where (node), When (event), Who (people).
CLASSIFIER	It refers to the logical combination (crossing) of the FOUNDATIONAL CONSTRUCTS of each column of the taxonomy, with the ABSTRACTION LEVELS defined in the MDA reference architecture applied in the taxonomy.
CELL	Each one of the graphical elements that compose the REFERENCE TAXONOMY, which allows elements to be classified according to the CLASSIFIERS that these contain.
REFERENCE TAXONOMY	An ARCHITECTURE DESCRIPTION of an IS, which considers all essential elements that conform this system, considering them from organizational to technical implementation ABSTRACTION LEVELS.
ARCHITECTURE DESCRIPTION	According to (612, 2011), it is a word product used to express (depict) an architecture.
TRANSFORMATION SPECIFICATION	According to (Object and Reference Model Subcommittee of the Architecture Board, 2010), it is a model that defines how different elements will relate to each other. These elements belong to models or artifacts of the same system at different levels of refinement.
MAPPING	According to (Object and Reference Model Subcommittee of the Architecture Board, 2010), it is a model that provides specifications for transformation of a PIM ABSTRACTION LEVEL into a PSM ABSTRACTION LEVEL for a specific platform. The platform model will determine the nature of the mapping.
TRACE	According to (Object and Reference Model Subcommittee of the Architecture Board, 2010), it is the set of INFORMATION that defines how a CONCERN is preserved throughout its crossing of all the ABSTRACTION LEVELS involved in an IS project (from a CIM to Technical Implementation levels).

TABLE B.8: Concept table for the MMQEF method in PDD convention (II).

Concept	Description
MODEL	From (612, 2011; Object and Reference Model Subcommittee of the Architecture Board, 2010), a model can be anything: a concept or a work product. A model is valid if it helps to answer questions about a system under consideration.
INFORMATION	According to (Harel and Rumpe, 2000), INFORMATION is the result of an <i>interpretation</i> process that assigns a meaning to each piece of <i>data</i> . Thus, the data is a syntactic representation of INFORMATION. The relation between INFORMATION and data is implicit. (Adriaans, 2013) defines the INFORMATION as an abstract massnoun used to denote any amount of data, code, or text that is stored, sent, received, or manipulated in any medium.
MODELLING LANGUAGE	We coincide with the definition presented in (da Silva, 2015), where the modelling language is a set of all possible MODELS that are conformant with an ABSTRACT SYNTAX, represented by one or more CONCRETE SYNTAX, and that satisfy a given semantic.
ABSTRACT SYNTAX	A <i>metamodel</i> with all the concepts identified for a MODELLING LANGUAGE at the meta-domain level (da Silva, 2015). It identifies the concepts, abstractions and relations underlying the application domain.
GRAMMAR	A specific technique for the definition of an ABSTRACT SYNTAX for textual languages (textual DSLs) and even natural languages.
METAMODEL	A specific technique for the definition of an ABSTRACT SYNTAX for modelling languages, using an UML profile mechanism of the class diagram that is compliant with the MOF language (omg, 2015) (or its variations).
CONCRETE SYNTAX	According to (da Silva, 2015), it is the notation, or the way users of a modelling language will learn and will use it, either by reading or by writing and designing the models.
GUIDELINE	Continuing with the definition presented in (da Silva, 2015), it is a consequence of the <i>pragmatics</i> of a MODELLING LANGUAGE, which helps and guides how to use it in the most appropriate way.
REPRESENTATION	It refers to the depiction employed by a modelling language for representing a (or a set of) MODEL(s) that express some concern in an IS.
DIAGRAM	A REPRESENTATION that uses graphical symbols, generally based on nodes (in a connected lattice). These have lines that represent relationships with each other.
TEXTUAL	A REPRESENTATION that uses textual symbols for modelling specific concerns (generally textual DSLs).

Bibliography

- (1985). Iso information processing – documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts. *ISO 5807:1985(E)*, pages 1–25.
- (1998). Iso/iec 10746-1:1998 information technology – open distributed processing - reference model: Overview. *ISO*, pages 1–76.
- (2000). Iso 15704:2000 industrial automation systems – requirements for enterprise-reference architectures and methodologies. *ISO*, pages 1–43.
- (2011). Iso/iec/ieee systems and software engineering – architecture description. *ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000)*, pages 1–46.
- Abid, A., Messai, N., Rouached, M., Devogele, T., and Abid, M. (2015). A semantic-aware framework for composite services engineering based on semantic similarity and concept lattices. In Nurcan, S. and Pimenidis, E., editors, *Information Systems Engineering in Complex Environments*, volume 204 of *Lecture Notes in Business Information Processing*, pages 148–164. Springer International Publishing.
- Abran, A., Moore, J. W., Bourque, P., Dupuis, R., and Tripp, L. L. (2013). *Guide to the Software Engineering Body of Knowledge (SWEBOK) version 3 public review*. IEEE. ISO Technical Report ISO/IEC TR 19759.
- Abu-El Seoud, M. and Klischewski, R. (2015). Mediating citizen-sourcing of open government applications—a design science approach. In *International Conference on Electronic Government*, pages 118–129. Springer.
- Adriaans, P. (2013). Information. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Fall 2013 edition.
- Agner, L. T. W., Soares, I. W., Stadzisz, P. C., and Simão, J. M. (2013). A brazilian survey on {UML} and model-driven practices for embedded software development. *Journal of Systems and Software*, 86(4):997 – 1005. {SI} : Software Engineering in Brazil: Retrospective and Prospective Views.
- Al-Nasrawi, S. and Ibrahim, M. (2013). An enterprise architecture mapping approach for realizing e-government. In *Communications and Information Technology (ICCIT), 2013 Third International Conference on*, pages 17–21.
- Amstel, M. F. V. (2010). The right tool for the right job: Assessing model transformation quality. pages 69–74. Affiliation: Eindhoven University of Technology, P.O. Box 513, 5600 MB, Eindhoven, Netherlands. Cited By (since 1996):1.
- Aranda, J., Damian, D., and Borici, A. (2012). Transition to model-driven engineering: What is revolutionary, what remains the same? In *Proceedings of the 15th International Conference on Model Driven Engineering Languages and Systems, MODELS'12*, pages 692–708, Berlin, Heidelberg. Springer-Verlag.
- Arendt, T. and Taentzer, G. (2013). A tool environment for quality assurance based on the eclipse modeling framework. *Automated Software Engg.*, 20(2):141–184.
- Atkinson, C., Bunse, C., and Wüst, J. (2003). *Driving component-based software development through quality modelling*, volume 2693. Cited By (since 1996):3.
- Bařna, S., Panetto, H., and Morel, G. (2009). New paradigms for a product oriented modelling: Case study for traceability. *Computers in Industry*, 60(3):172 – 183.
- Baker, P., Loh, S., and Weil, F. (2005). Model-driven engineering in a large industrial context — motorola case study. In Briand, L. and Williams, C., editors, *Model Driven Engineering Languages and Systems*, volume 3713 of *Lecture Notes in Computer Science*, pages 476–491. Springer Berlin Heidelberg.
- Barišić, A., Amaral, V., Goulão, M., and Barroca, B. (2011). Quality in use of domain-specific languages: A case study. In *Proceedings of the 3rd ACM SIGPLAN Workshop on Evaluation and Usability of Programming Languages and Tools, PLATEAU '11*, pages 65–72, New York, NY, USA. ACM.
- Basili, V. R., Caldiera, G., and Rombach, H. D. (1994). The goal question metric approach. In *Encyclopedia of Software Engineering*. Wiley.
- Becker, J., Bergener, P., Breuker, D., and Rackers, M. (2010). Evaluating the expressiveness of domain specific modeling languages using the bunge-wand-weber ontology. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10.
- Bendaoud, R., Napoli, A., and Toussaint, Y. (2008). Formal concept analysis: A unified framework for building and refining ontologies. In Gangemi, A. and Euzenat, J., editors, *Knowledge Engineering: Practice and Patterns*, volume 5268 of *Lecture Notes in Computer Science*, pages 156–171. Springer Berlin Heidelberg.
- Bertrand Portier, L. A. (2009). Model driven development misperceptions and challenges.

- Bērziša, S., España, S., Grabis, J., Henkel, M., Jokste, L., Kampars, J., Koç, H., Sandkuhl, K., Stirna, J., Valverde, F., and Zdravkovic, J. (2015). Capability as a service in digital enterprises collaborative project number 611351. deliverable 5.2: The initial version of capability driven development methodology. Technical report.
- Bézivin, J. and Kurtev, I. (2005). Model-based technology integration with the technical space concept. In *In: Proceedings of the Metainformatics Symposium, Springer-Verlag*. Springer-Verlag.
- Bjeković, M., Proper, H., and Sottet, J.-S. (2014). Enterprise modelling languages. In Shishkov, B., editor, *Business Modeling and Software Design*, volume 173 of *Lecture Notes in Business Information Processing*, pages 1–23. Springer International Publishing.
- Borchmann, D., Peñaloza, R., and Wang, W. (2014). Classifying software bug reports using methods from formal concept analysis. *Studia Universitatis Babeş-Bolyai Informatica*, 59:10–27. Supplemental proceedings of the 12th International Conference on Formal Concept Analysis (ICFCA'14).
- Boronat, A. and Meseguer, J. (2008). An algebraic semantics for mof. In Fiadeiro, J. L. and Inverardi, P., editors, *Proc. FASE 2008*, volume 4961 of *Lecture Notes in Computer Science*, pages 377–391. Springer.
- Botchkarev, A. and Andru, P. (2011). A return on investment as a metric for evaluating information systems: Taxonomy and application. *Interdisciplinary Journal of Information, Knowledge, and Management (IJKM)*, 6:245–269.
- Brambilla, M. (2016). How mature is of model-driven engineering as an engineering discipline @ONLINE.
- Brambilla, M. and Fraternali, P. (2014). Large-scale model-driven engineering of web user interaction: The webml and webratio experience. *Science of Computer Programming*, 89, Part B(0):71 – 87. Special issue on Success Stories in Model Driven Engineering.
- Brinkkemper, S., Saeki, M., and Harmsen, F. (1999). Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, 24(3):209 – 228. 10th International Conference on Advanced Information Systems Engineering.
- Brossard, A., Abed, M., and Kolski, C. (2011). Taking context into account in conceptual models using a model driven engineering approach. *Information and Software Technology*, 53(12):1349 – 1369.
- Brown, A. (2009). Simple and practical model driven architecture (mda) @ONLINE.
- Bruel, J.-M., Combemale, B., Ober, I., and Raynal, H. (2015). Mde in practice for computational science. *Procedia Computer Science*, 51:660–669.
- Bubenko, J. A., P. A. and Stirna, J. (2001). User guide of the knowledge management approach using enterprise knowledge patterns, deliverable d3. ist programme project hypermedia and pattern based knowledge management for smart organisations, project no. ist- 2000-28401. Technical report.
- Budgen, D., Burn, A. J., Brereton, O. P., Kitchenham, B. A., and Pretorius, R. (2011). Empirical evidence about the uml: a systematic literature review. *Software: Practice and Experience*, 41(4):363–392.
- Burden, H., Heldal, R., and Whittle, J. (2014). Comparing and contrasting model-driven engineering at three large companies. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '14*, pages 14:1–14:10, New York, NY, USA. ACM.
- Burge, J. E., Carroll, J. M., McCall, R., and Mistrk, I. (2008). *Rationale-Based Software Engineering*. Springer Publishing Company, Incorporated, 1 edition.
- Cabot, J. Has mda been abandoned (by the omg)?
- Cabot, J. (2009). Modeling will be commonplace in three years time @ONLINE.
- Cabot, J. (2013). Is transitioning to MDE revolutionary (for companies adopting it)?
- Cachero, C., Poels, G., Calero, C., and Marhuenda, Y. (2007). Towards a Quality-Aware Engineering Process for the Development of Web Applications. Working Papers of Faculty of Economics and Business Administration, Ghent University, Belgium 07/462, Ghent University, Faculty of Economics and Business Administration.
- Carter, Jr., J. A. (1986). A taxonomy of user-oriented functions. *Int. J. Man-Mach. Stud.*, 24(3):195–292.
- Centre, P. R. (2015). Capability delivery in action: Showcasing the cdd approach.
- Challenger, M., Kardas, G., and Tekinerdogan, B. (2015). A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems. *Software Quality Journal*, pages 1–41.
- Chaudron, M. V., Heijstek, W., and Nugroho, A. (2012). How effective is uml modeling ? *Software & Systems Modeling*, 11(4):571–580. J2: Softw Syst Model.
- Chenouard, R., Granvilliers, L., and Soto, R. (2008). Model-driven constraint programming. pages 236–246. Affiliation: CNRS, LINA, Université de Nantes, France; Affiliation: Pontificia Universidad Católica de Valparaíso, Chile. Cited By (since 1996):8.
- Chung, S., Won, D., Baeg, S.-H., and Park, S. (2009). Service-oriented reverse reengineering: 5w1h model-driven re-documentation and candidate services identification. In *2009 IEEE International Conference on Service-Oriented Computing and Applications (SOCA)*, pages 1–6. IEEE.
- Clark, T. and Muller, P.-A. (2012). Exploiting model driven technology: a tale of two startups. *Software & Systems Modeling*, 11(4):481–493.

- Combemale, B., Crégut, X., Garoche, P.-L., and Thirioux, X. (2009). Essay on Semantics Definition in MDE. An Instrumented Approach for Model Verification. *Journal of Software (JSW)*, 4(9):943–958.
- Corneliusson, L. (2008). What do you think of model-driven software development?
- Costagliola, G., Delucia, A., Orefice, S., and Polese, G. (2002). A classification framework to support the design of visual languages. *Journal of Visual Languages & Computing*, 13(6):573 – 600.
- Costal, D., Gómez, C., and Guizzardi, G. (2011). Formal semantics and ontological analysis for understanding sub-setting, specialization and redefinition of associations in uml. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6998 LNCS:189–203. cited By (since 1996)3.
- Creswell, J. (2013). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications.
- Crowder, J. A., Carbone, J. N., and Demijohn, R. (2016). *Systems Engineering Tools and Practices*, pages 89–103. Springer International Publishing, Cham.
- Cruz-Lemus, J. A., Maes, A., Género, M., Poels, G., and Piattini, M. (2010). The impact of structural complexity on the understandability of uml statechart diagrams. *Information Sciences*, 180(11):2209–2220. Cited By (since 1996):14.
- Cuadrado, J. S., Izquierdo, J. L. C., and Molina, J. G. (2014). Applying model-driven engineering in small software enterprises. *Science of Computer Programming*, 89, Part B(0):176 – 198. Special issue on Success Stories in Model Driven Engineering.
- da Silva, A. R. (2015). Model-driven engineering: A survey supported by the unified conceptual model. *Computer Languages, Systems & Structures*, 43:139 – 155.
- da Silva Teixeira, d. G. M., Quirino, G. K., Gailly, F., de Almeida Falbo, R., Guizzardi, G., and Perini Barcellos, M. (2016). *PoN-S: A Systematic Approach for Applying the Physics of Notation (PoN)*, pages 432–447. Springer International Publishing, Cham.
- Davies, I., Green, P., Rosemann, M., Indulska, M., and Gallo, S. (2006). How do practitioners use conceptual modeling in practice? *Data & Knowledge Engineering*, 58(3):358 – 380. Including the special issue : {ER} 2004ER 2004.
- Davies, J., Milward, D., Wang, C.-W., and Welch, J. (2015). Formal model-driven engineering of critical information systems. *Science of Computer Programming*, 103(0):88 – 113. Selected papers from the First International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS 2012).
- de Graaf, K., Liang, P., Tang, A., van Hage, W., and van Vliet, H. (2014). An exploratory study on ontology engineering for software architecture documentation. *Computers in Industry*, 65(7):1053 – 1064.
- de la Vara, J. L., Díaz, J. S., and Pastor, O. (2007). Integración de un entorno de producción automática de software en un marco de alineamiento estratégico (in spanish). In *Anais do WER07 - Workshop em Engenharia de Requisitos, Toronto, Canada, May 17-18, 2007*, pages 68–79.
- de Oca, I. M.-M., Snoeck, M., Reijers, H. A., and Rodríguez-Morffi, A. (2015). A systematic literature review of studies on business process modeling quality. *Information and Software Technology*, 58:187 – 205.
- DeLone, W. H. and McLean, E. R. (1992). Information systems success: The quest for the dependent variable. *Information Systems Research*, 3(1):60–95.
- DenHaan, J. (2009). 8 reasons why model driven development is dangerous @ONLINE.
- DenHaan, J. (2010). Model driven engineering vs the commando pattern @ONLINE.
- DenHaan, J. (2011a). Why aren't we all doing model driven development yet @ONLINE.
- DenHaan, J. (2011b). Why there is no future model driven development @ONLINE.
- Di Ruscio, D., Iovino, L., and Pierantonio, A. (2013). Managing the coupled evolution of metamodels and textual concrete syntax specifications. pages 114–121. cited By (since 1996)0.
- Dijkman, R. M., Dumas, M., and Ouyang, C. (2008). Semantics and analysis of business process models in {BPMN}. *Information and Software Technology*, 50(12):1281 – 1294.
- Do Nascimento Fidalgo, R., De Souza, E. M., España, S., De Castro, J. B., and Pastor, O. (2012). *EERMM: A Metamodel for the Enhanced Entity-Relationship Model*, pages 515–524. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Domínguez-Mayo, F. J., Escalona, M. J., Mejías, M., Ramos, I., and Fernández, L. (2011). A framework for the quality evaluation of mdwe methodologies and information technology infrastructures. *International Journal of Human Capital and Information Technology Professionals*, 2(4):11–22.
- Domínguez-Mayo, F. J., Escalona, M. J., Mejías, M., and Torres, A. H. (2010). A quality model in a quality evaluation framework for mdwe methodologies. pages 495–506. Affiliation: Departamento de Lenguajes y Sistemas Informáticos, University of Seville, Seville, Spain. Cited By (since 1996):1.
- Dubray, J.-J. (2011). Why did mde miss the boat?
- Egido, J. C., González, T., Juanes, R., Llorca, C., Grabis, J., Stirna, J., and Zdravkovic, J. (2014). Capability as a service in digital enterprises collaborative project number 611351. deliverable 1.3: Vision of the cdd methodology. Technical report.

- Escalona, M. J., Gutiérrez, J. J., Pérez-Pérez, M., Molina, A., Domínguez-Mayo, E., and Domínguez-Mayo, F. J. (2011). *Measuring the Quality of Model-Driven Projects with NDT-Quality*, pages 307–317. Springer New York.
- España, S., Ruiz, M., and González, A. (2012). Systematic derivation of conceptual models from requirements models: A controlled experiment. In *2012 Sixth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12.
- España, S., González, A., and Pastor, Ó. (2009). *Communication Analysis: A Requirements Engineering Method for Information Systems*, pages 530–545. Springer Berlin Heidelberg, Berlin, Heidelberg.
- España, S., González, T., Grabis, J., Jokste, L., Juanes, R., and Valverde, F. (2014). *Advanced Information Systems Engineering Workshops: CAiSE 2014 International Workshops, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, chapter Capability-Driven Development of a SOA Platform: A Case Study, pages 100–111. Springer International Publishing, Cham.
- España Cubillo, S. (2012). *Methodological integration of Communication Analysis into a model-driven software development framework*. PhD thesis.
- Espinilla, M., Domínguez-Mayo, F. J., Escalona, M. J., Mejías, M., Ross, M., and Staples, G. (2011). *A Method Based on AHP to Define the Quality Model of QuEF*, volume 123, pages 685–694. Springer Berlin Heidelberg.
- Essien, J. (2015). *Model driven validation approach for enterprise architecture and motivation extensions*. PhD thesis, University of West London.
- Fabra, J., Castro, V. D., Álvarez, P., and Marcos, E. (2012). Automatic execution of business process models: Exploiting the benefits of model-driven engineering approaches. *Journal of Systems and Software*, 85(3):607 – 625. Novel approaches in the design and implementation of systems/software architecture.
- Falkenberg, E. D., Hesse, W., Lindgreen, P., Nilsson, B. E., Oei, J. L. H., Rolland, C., Stamper, R. K., Assche, F. J. M. V., Verrijn-Stuart, A. A., and Voss, K. (1996). Frisco: A framework of information system concepts. Technical report, The IFIP WG 8. 1 Task Group FRISCO.
- Falleri, J.-R., Huchard, M., and Nebut, C. (2008). Empirical comparison of two class model normalization techniques: Obstacles and questions. In Arisholm, E., Briand, L., and Anda, B., editors, *ESMDE'08: Workshop on Empirical Studies of Model-Driven Engineering*, volume 392, pages 21–30, Toulouse, France. CEUR-WS.org.
- Fettke, P., Houy, C., Vella, A.-L., and Loos, P. (2012). *Towards the Reconstruction and Evaluation of Conceptual Model Quality Discourses – Methodical Framework and Application in the Context of Model Understandability*, volume 113 of *Lecture Notes in Business Information Processing*, chapter 28, pages 406–421. Springer Berlin Heidelberg.
- Finnie, S. (2015). Modeling community: Are we missing something?
- Florez, H., Sánchez, M., and Villalobos, J. (2016). A catalog of automated analysis methods for enterprise models. *SpringerPlus*, 5(1):1–24.
- for Standardization/International Electrotechnical Commission, I. O. et al. (2014). Iso/iec 24744:2014. *Software Engineering-Metamodel for Development Methodologies*.
- Fournier, C. (2008). Is uml practical? @ONLINE.
- France, R. and Rumpe, B. (2007a). Model-driven development of complex software: A research roadmap. In *2007 Future of Software Engineering, FOSE '07*, pages 37–54, Washington, DC, USA. IEEE Computer Society.
- France, R. and Rumpe, B. (2007b). Model-driven development of complex software: A research roadmap. In *Future of Software Engineering, 2007. FOSE '07*, pages 37–54.
- Frank, U. (2012). Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. *Software & Systems Modeling*, 13(3):941–962.
- Frankel, D. S., Harmon, P., Mukerji, J., Odell, J., Owen, M., Rivitt, P., Rosen, M., and Soley, R. M. (2003). The Zachman Framework and the OMG's Model Driven Architecture.
- Freeman, L. C. and White, D. R. (1993). Using galois lattices to represent network data. In *Sociological methodology*, pages 127–146.
- Gallego, M., Giraldo, F. D., and Hitpass, B. (2015). Adapting the pbec-otss software selection approach for bpm suites: an application case. In *2015 34th International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–10.
- Galvão, I. and Goknil, A. (2007). Survey of traceability approaches in model-driven engineering. pages 313–324. cited By (since 1996)22.
- Gargantini, A., Riccobene, E., and Scandurra, P. (2009). A semantic framework for metamodel-based languages. *Autom. Softw. Eng.*, 16(3-4):415–454.
- Garner, B. and Raban, R. (1999). Context management in modeling information systems (is). *Information and Software Technology*, 41(14):957 – 961.
- Gemino, A. and Wand, Y. (2005). Complexity and clarity in conceptual modeling: Comparison of mandatory and optional properties. *Data & Knowledge Engineering*, 55(3):301 – 326. Quality in conceptual modelingFive examples of the state of artThe International Workshop on Conceptual Modeling Quality 2002 and 2003.

- Giraldo, F., España, S., Giraldo, W., and Pastor, O. (2015a). Modelling language quality evaluation in model-driven information systems engineering: A roadmap. In *Research Challenges in Information Science (RCIS), 2015 IEEE 9th International Conference on*, pages 64–69.
- Giraldo, F., España, S., and Pastor, O. (2014). Analysing the concept of quality in model-driven engineering literature: A systematic review. In *Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on*, pages 1–12.
- Giraldo, F. D., España, S., Pineda, M. A., Giraldo, W., and Pastor, O. (2015b). Conciliating model-driven engineering with technical debt using a quality framework. In Nurcan, S. and Pimenidis, E., editors, *Information Systems Engineering in Complex Environments*, volume 204 of *Lecture Notes in Business Information Processing*, pages 199–214. Springer International Publishing.
- Giraldo, F. D., España, S., and Pastor, O. (2016a). Evidences of the mismatch between industry and academy on modelling language quality evaluation. *CoRR*, abs/1606.02025.
- Giraldo, F. D., España, S., Pastor, Ó., and Giraldo, W. J. (2016b). Considerations about quality in model-driven engineering. *Software Quality Journal*, pages 1–66.
- Giraldo, O. W. J., Arias, M. R., Collazos, O. C. A., Molina, A. I., Ortega, C. M., and Redondo, M. A. (2015c). Fast functional prototyping of user interfaces based on dataform models, a tool (tooldataform). In *2015 10th Computing Colombian Conference (10CCC)*, pages 172–179.
- Goldkuhl, G. (2011). Actability criteria for design and evaluation: Pragmatic qualities of information systems. *Int. J. Inf. Syst. Soc. Chang.*, 2(3):1–15.
- Goldkuhl, G., Lind, M., and Seigerroth, U. (1998). Method integration: the need for a learning perspective. *Software, IEE Proceedings -*, 145(4):113–118.
- González, A., España, S., Ruiz, M., and Pastor, Ó. (2011). *Systematic Derivation of Class Diagrams from Communication-Oriented Business Process Models*, pages 246–260. Springer Berlin Heidelberg, Berlin, Heidelberg.
- González, C. and Cabot, J. (2014). Formal verification of static software models in mde: A systematic review. *Information and Software Technology*, 56(8):821–838. cited By (since 1996)0.
- González, C. A., Büttner, F., Clarisó, R., and Cabot, J. (2012). Emftocsp: A tool for the lightweight verification of emf models. pages 44–50. Affiliation: À%ocole des Mines de Nantes, INRIA, LINA, Nantes, France; Affiliation: Universitat Oberta de Catalunya, Barcelona, Spain. Cited By (since 1996):1.
- Gorschek, T., Tempero, E., and Angelis, L. (2014). On the use of software design models in software development practice: An empirical investigation. *Journal of Systems and Software*, 95(0):176 – 193.
- Goulão, M., Amaral, V., and Mernik, M. (2016). Quality in model-driven engineering: a tertiary study. *Software Quality Journal*, pages 1–33.
- Green, T. and Petre, M. (1996). Usability analysis of visual programming environments: A ‘cognitive dimensions’ framework. *Journal of Visual Languages & Computing*, 7(2):131 – 174.
- Gregor, S. (2006). The nature of theory in information systems. *MIS Q.*, 30(3):611–642.
- Grobshtein, Y. and Dori, D. (2011). Generating sysml views from an opm model: Design and evaluation. *Systems Engineering*, 14(3):327–340.
- Guarino, N. and Welty, C. A. (2000). A formal ontology of properties. In *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management, EKAW ’00*, pages 97–112, London, UK, UK. Springer-Verlag.
- Gurr, C. (2002). Combining semantic and cognitive accounts of diagrams. In Anderson, M., Meyer, B., and Olivier, P., editors, *Diagrammatic Representation and Reasoning*, pages 125–140. Springer London.
- Haan, J. d. (2008). 8 reasons why model-driven approaches (will) fail.
- Harel, D. and Rumpe, B. (2000). Modeling languages: Syntax, semantics and all that stuff, part i: The basic stuff. Technical report, Jerusalem, Israel, Israel.
- Harel, D. and Rumpe, B. (2004). Meaningful modeling: What’s the semantics of "semantics"? *Computer*, 37(10):64–72.
- Hebig, R. and Bendraou, R. (2014). On the need to study the impact of model driven engineering on software processes. In *Proceedings of the 2014 International Conference on Software and System Process, ICSSP 2014*, pages 164–168, New York, NY, USA. ACM.
- Heggset, M., Krogstie, J., and Wesenberg, H. (2015). The influence of syntactic quality on pragmatic quality of enterprise process models. *CSIMQ*, 5:1–13.
- Heidari, F. and Loucopoulos, P. (2014). Quality evaluation framework (qef): Modeling and evaluating quality of business processes. *International Journal of Accounting Information Systems*, 15(3):193 – 223. Business Process Modeling.
- Henderson-Sellers, B. and Gonzalez-Perez, C. (2010). Granularity in conceptual modelling: Application to metamodels. In Parsons, J., Saeki, M., Shoval, P., Woo, C., and Wand, Y., editors, *Conceptual Modeling – ER 2010*, volume 6412 of *Lecture Notes in Computer Science*, pages 219–232. Springer Berlin Heidelberg.

- Heymans, P., Schobbens, P. Y., Trigaux, J. C., Bontemps, Y., Matulevicius, R., and Classen, A. (2008). Evaluating formal properties of feature diagram languages. *Software, IET*, 2(3):281–302. ID: 2.
- Hindawi, M., Morel, L., Aubry, R., and Sourrouille, J.-L. (2009). *Description and Implementation of a UML Style Guide*, volume 5421, pages 291–302. Springer Berlin Heidelberg.
- Hoang, D. (2012). Current limitations of mdd and its implications @ONLINE.
- Hodges, W. (2013). Model theory. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Fall 2013 edition.
- Hutchinson, J., Rouncefield, M., and Whittle, J. (2011a). Model-driven engineering practices in industry. In *Proceedings of the 33rd International Conference on Software Engineering, ICSE '11*, pages 633–642, New York, NY, USA. ACM.
- Hutchinson, J., Whittle, J., and Rouncefield, M. (2014). Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure. *Science of Computer Programming*, 89, Part B(0):144 – 161. Special issue on Success Stories in Model Driven Engineering.
- Hutchinson, J., Whittle, J., Rouncefield, M., and Kristoffersen, S. (2011b). Empirical assessment of mde in industry. In *Proceedings of the 33rd International Conference on Software Engineering, ICSE '11*, pages 471–480, New York, NY, USA. ACM.
- Igarza, I. M. H., Boada, D. H. G., and Valdés, A. P. (2012). Una introducción al desarrollo de software dirigido por modelos. *Serie Científica*, 5(3).
- ISO/IEC (2001). *ISO/IEC 9126. Software engineering – Product quality*. ISO/IEC.
- Izurrieta, C., Rojas, G., and Griffith, I. (2015). Preemptive management of model driven technical debt for improving software quality. In *Proceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures, QoSA '15*, pages 31–36, New York, NY, USA. ACM.
- Jalali, S. and Wohlin, C. (2012). Systematic literature studies: Database searches vs. backward snowballing. In *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '12*, pages 29–38, New York, NY, USA. ACM.
- Jedlitschka, A., Ciolkowski, M., and Pfahl, D. (2008). *Reporting Experiments in Software Engineering*, pages 201–228. Springer London, London.
- Kahraman, G. and Bilgen, S. (2013). A framework for qualitative assessment of domain-specific languages. *Software & Systems Modeling*, pages 1–22.
- Kautz, K. and Nagm, F. The advancement of is evaluation: a literature review. In *PACIS 2008 Proceedings. Paper 66*.
- Kautz, K. and Nagm, F. (2008). The advancement of is evaluation: a literature review. In *PACIS 2008 Proceedings.*, page 66, <http://aisel.aisnet.org/pacis2008/66>. AIS Electronic Library (AISeL).
- Kessentini, M., Langer, P., and Wimmer, M. (2013). Searching models, modeling search: On the synergies of sbse and mde. In *Combining Modelling and Search-Based Software Engineering (CMSBSE), 2013 1st International Workshop on*, pages 51–54.
- Kingston, J. (2008). Multi-perspective ontologies: Resolving common ontology development problems. *Expert Systems with Applications*, 34(1):541 – 550.
- Kingston, J. and Macintosh, A. (2000). Knowledge management through multi-perspective modelling: representing and distributing organizational memory. *Knowledge-Based Systems*, 13(2-3):121 – 131.
- Kitchenham, B. and Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical Report EBSE 2007-001, Keele University and Durham University Joint Report.
- Kitchenham, B., Pflieger, S., Pickard, L., Jones, P., Hoaglin, D., El Emam, K., and Rosenberg, J. (2002). Preliminary guidelines for empirical research in software engineering. *Software Engineering, IEEE Transactions on*, 28(8):721–734.
- Kleppe, A. (2008). *Software Language Engineering: Creating Domain-Specific Languages Using Metamodels*. Addison-Wesley Professional, 1 edition.
- Klinke, M. (2008). Do you use mda/mdd/mdsd, any kind of model-driven approach? will it be the future?
- Köhnlein, J. (2013). Eclipse diagram editors from a user's perspective.
- Kolovos, D. S., Paige, R. F., and Polack, F. A. (2008). The grand challenge of scalability for model driven engineering. In *Models in Software Engineering*, pages 48–53. Springer.
- Kolovos, D. S., Rose, L. M., Matragkas, N., Paige, R. F., Guerra, E., Cuadrado, J. S., De Lara, J., Ráth, I., Varró, D., Tisi, M., and Cabot, J. (2013). A research roadmap towards achieving scalability in model driven engineering. In *Proceedings of the Workshop on Scalability in Model Driven Engineering, BigMDE '13*, pages 2:1–2:10, New York, NY, USA. ACM.
- Kong, X., Liu, L., and Lowe, D. (2006). Separation of concerns: a web application architecture framework. *Journal of digital information*, 6(2).
- Krill, P. (2016). Uml to be ejected from microsoft visual studio (infoworld).

- Krogstie, J. (2012a). *Methodologies for Computerised Information Systems Support in Organisations*, pages 19–87. Springer London, London.
- Krogstie, J. (2012b). *Model-Based Development and Evolution of Information Systems: A Quality Approach*, chapter Quality of Models, pages 205–247. Springer London, London.
- Krogstie, J. (2012c). *Model-Based Development and Evolution of Information Systems: A Quality Approach*, chapter Quality of Modelling Languages, pages 249–280. Springer London, London.
- Krogstie, J. (2012d). *Model-Based Development and Evolution of Information Systems: A Quality Approach*. Springer Publishing Company, Incorporated.
- Krogstie, J. (2012e). *Quality of Models*, pages 205–247. Springer London.
- Krogstie, J. (2012f). *Specialisations of SEQUAL*, pages 281–326. Springer London.
- Krogstie, J., Lindland, O. I., and Sindre, G. (1995). Defining quality aspects for conceptual models. In *Proceedings of the IFIP International Working Conference on Information System Concepts: Towards a Consolidation of Views*, pages 216–231, London, UK, UK. Chapman & Hall, Ltd.
- Kruchten, P. (2000). *The Rational Unified Process: An Introduction, Second Edition*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition.
- Kruchten, P., Nord, R., and Ozkaya, I. (2012). Technical debt: From metaphor to theory and practice. *Software, IEEE*, 29(6):18–21.
- Kudryavtsev, D., Gavrilova, T., and Leshcheva, I. (2013). One approach to the classification of business knowledge diagrams: practical view. In *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on*, pages 1259–1265. IEEE.
- Kulkarni, V., Reddy, S., and Rajbhoj, A. (2010). Scaling up model driven engineering – experience and lessons learnt. In Petriu, D., Rouquette, N., and Haugen, y., editors, *Model Driven Engineering Languages and Systems*, volume 6395 of *Lecture Notes in Computer Science*, pages 331–345. Springer Berlin Heidelberg.
- Laguna, M. A. and Marqués, J. M. (2010). Uml support for designing software product lines: The package merge mechanism. 16(17):2313–2332.
- Lange, C. (2007a). Model size matters. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4364 LNCS:211–216. cited By (since 1996)1.
- Lange, C. and Chaudron, M. (2005). Managing Model Quality in UML-Based Software Development. In *Software Technology and Engineering Practice, 2005. 13th IEEE International Workshop on*, pages 7–16.
- Lange, C., Chaudron, M. R. V., Muskens, J., Somers, L. J., and Dortmans, H. M. (2003). An empirical investigation in quantifying inconsistency and incompleteness of uml designs. In *Incompleteness of UML Designs”, Proc. Workshop on Consistency Problems in UML-based Software Development, 6 th International Conference on Unified Modeling Language, UML 2003*.
- Lange, C., DuBois, B., Chaudron, M., and Demeyer, S. (2006). An experimental investigation of uml modeling conventions. In Nierstrasz, O., Whittle, J., Harel, D., and Reggio, G., editors, *Model Driven Engineering Languages and Systems*, volume 4199 of *Lecture Notes in Computer Science*, pages 27–41. Springer Berlin Heidelberg.
- Lange, C. F. J. and Chaudron, M. R. V. (2006). Effects of defects in uml models: an experimental investigation. In *Proceedings of the 28th international conference on Software engineering, ICSE ’06*, pages 401–411, New York, NY, USA. ACM.
- Lange, C. J. (2007b). *Model Size Matters*, volume 4364, pages 211–216. Springer Berlin Heidelberg.
- Laurent, Y., Bendraou, R., and Gervais, M. P. (2013). Executing and debugging uml models: An fuml extension. pages 1095–1102. Affiliation: LIP6, UPMC Paris Universitias, France; Affiliation: LIP6, University of Paris Ouest, Nanterre, France.
- Laware, G. and Kowalkowski, F. (2005). The business value of taxonomies and ontologies for web & knowledge management practices. pages 41–47. cited By 0.
- Le Pallec, X. and Dupuy-Chessa, S. (2013). Support for quality metrics in metamodelling. In *Proceedings of the Second Workshop on Graphical Modeling Language Development, GMLD ’13*, pages 23–31. ACM.
- Leist, S. and Zellner, G. (2006). Evaluation of current architecture frameworks. In *Proceedings of the 2006 ACM Symposium on Applied Computing, SAC ’06*, pages 1546–1553, New York, NY, USA. ACM.
- Letsholo, K., Chioasca, E. V., and Zhao, L. (2012). An integration framework for multi-perspective business process modeling. In *Services Computing (SCC), 2012 IEEE Ninth International Conference on*, pages 33–40.
- Lewins, A., Taylor, C., and Gibbs, G. R. (2010). What is qualitative data analysis (qda) ? (online).
- Liao, Y., Lezoche, M., Panetto, H., Boudjlida, N., and Loures, E. R. (2015a). Semantic annotation for knowledge explicitation in a product lifecycle management context: a survey. *Computers in Industry*, 71:24–34.
- Liao, Y., Lezoche, M., Panetto, H., Boudjlida, N., and Rocha Loures, E. (2015b). Semantic Annotation for Knowledge Explicitation in a Product Lifecycle Management Context: a Survey. *Computers in Industry*, 71:24–34.
- Lim, S. H., Juster, N., and de Pennington, A. (1997). Enterprise modelling and integration: a taxonomy of seven key aspects. *Computers in Industry*, 34(3):339 – 359.

- Linders, B. (2015). New developments in model driven software engineering.
- Lindland, O. I., Sindre, G., and Sølvberg, A. (1994). Understanding quality in conceptual modeling. *IEEE Softw.*, 11(2):42–49.
- Liu, Y. and Li, X. (2014). A fca-based approach to data integration in the university information system. In Li, S., Jin, Q., Jiang, X., and Park, J. J. H., editors, *Frontier and Future Development of Information Technology in Medicine and Education*, volume 269 of *Lecture Notes in Electrical Engineering*, pages 763–771. Springer Netherlands.
- López-Fernández, J. J., Guerra, E., and de Lara, J. (2014). Assessing the quality of meta-models. *11th Workshop on Model Driven Engineering, Verification and Validation MoDeVVa 2014*, page 10.
- Loucopoulos, P. and Zicari, R. (1992). *Conceptual Modeling, Databases, and Case: An Integrated View of Information Systems Development*. John Wiley & Sons, Inc., New York, NY, USA.
- Lukman, T., Godena, G., Gray, J., Heričko, M., and Strmčnik, S. (2013). Model-driven engineering of process control software – beyond device-centric abstractions. *Control Engineering Practice*, 21(8):1078 – 1096.
- Luyten, K., Prof, P., and Coninx, K. (2004). Dynamic user interface generation for mobile and embedded systems with model-based user interface development.
- Lyytinen, K. (1987). Critical issues in information systems research. chapter A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations, pages 3–41. John Wiley & Sons, Inc., New York, NY, USA.
- Maes, A. and Poels, G. (2007). Evaluating quality of conceptual modelling scripts based on user perceptions. *Data Knowl. Eng.*, 63(3):701–724.
- Malik, N. (2009). Why the zachman framework is not an ontology @ONLINE.
- Marín, B., Giachetti, G., Pastor, O., and Abran, A. (2010). A quality model for conceptual models of mdd environments. *Adv. Soft. Eng.*, 2010:1:1–1:17.
- Marín, B., Giachetti, G., Pastor, O., Vos, T. E. J., and Abran, A. (2013). Using a functional size measurement procedure to evaluate the quality of models in mdd environments. *ACM Trans. Softw. Eng. Methodol.*, 22(3):26:1–26:31.
- Marín, B., Salinas, A., Morandé, J., Giachetti, G., and de la Vara, J. (2014). Key features for a successful model-driven development tool. In *Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on*, pages 541–548. IEEE.
- Martin, R. and Robertson, E. L. (1999). Formalization of multi-level zachman frameworks (technical report no. 522). Technical report, Computer Science Department, Indiana University.
- Matinlassi, M. (2005). Quality-driven software architecture model transformation. In *Software Architecture, 2005. WICSA 2005. 5th Working IEEE/IFIP Conference on*, pages 199–200.
- Mayerhofer, T. (2012). Testing and debugging uml models based on fuml. In *Software Engineering (ICSE), 2012 34th International Conference on*, pages 1579–1582. ID: 7.
- Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. (2010). Seven process modeling guidelines (7pmg). *Inf. Softw. Technol.*, 52(2):127–136.
- Mens, T. and Gorp, P. V. (2006). A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, 152(0):125 – 142. Proceedings of the International Workshop on Graph and Model Transformation (GraMoT 2005) Graph and Model Transformation 2005.
- Merilinnä, J. (2005). *A Tool for Quality-Driven Architecture Model Transformation*. PhD thesis, VTT Technical Research Centre of Finland.
- Mijatov, S., Langer, P., and Mayerhofer, T. (2013). A framework for testing uml activities based on fuml. In *Workshop on Model Driven Engineering, Verification and Validation - MoDeVVa 2013*, volume 1069/, pages 11–20. CEUR.
- Mohagheghi, P. and Aagedal, J. (2007). Evaluating quality in model-driven engineering. In *Proceedings of the International Workshop on Modeling in Software Engineering, MISE '07*, pages 6–, Washington, DC, USA. IEEE Computer Society.
- Mohagheghi, P. and Dehlen, V. (2008a). *Developing a quality framework for model-driven engineering*, volume 5002 LNCS. Cited By (since 1996):4.
- Mohagheghi, P. and Dehlen, V. (2008b). Where is the proof? - a review of experiences from applying mde in industry. In Schieferdecker, I. and Hartman, A., editors, *Model Driven Architecture – Foundations and Applications*, volume 5095 of *Lecture Notes in Computer Science*, pages 432–443. Springer Berlin Heidelberg.
- Mohagheghi, P., Dehlen, V., and Neple, T. (2009a). Definitions and approaches to model quality in model-based software development - a review of literature. *Inf. Softw. Technol.*, 51(12):1646–1669.
- Mohagheghi, P., Dehlen, V., and Neple, T. (2009b). Definitions and approaches to model quality in model-based software development – a review of literature. *Information and Software Technology*, 51(12):1646 – 1669. Quality of {UML} Models.

- Mohagheghi, P., Fernandez, M., Martell, J., Fritzsche, M., and Gilani, W. (2009c). Mde adoption in industry: Challenges and success criteria. In Chaudron, M., editor, *Models in Software Engineering*, volume 5421 of *Lecture Notes in Computer Science*, pages 54–59. Springer Berlin Heidelberg.
- Mohagheghi, P., Gilani, W., Stefanescu, A., and Fernandez, M. (2013a). An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases. *Empirical Software Engineering*, 18(1):89–116.
- Mohagheghi, P., Gilani, W., Stefanescu, A., Fernandez, M., Nordmoen, B., and Fritzsche, M. (2013b). Where does model-driven engineering help? experiences from three industrial cases. *Software and Systems Modeling*, 12(3):619–639. cited By (since 1996)0.
- Molina, A. I., Giraldo, W. J., Gallardo, J., Redondo, M. A., Ortega, M., and García, G. (2012). Ciat-gui: A mde-compliant environment for developing graphical user interfaces of information systems. *Advances in Engineering Software*, 52(0):10 – 29.
- Molina, A. I., Giraldo, W. J., Ortega, M., Redondo, M. A., and Collazos, C. A. (2014). Model-driven development of interactive groupware systems: Integration into the software development process. *Science of Computer Programming*, 89, Part C(0):320 – 349.
- Molina, F. and Toval, A. (2009). Integrating usability requirements that can be evaluated in design time into model driven engineering of web information systems. *Advances in Engineering Software*, 40(12):1306 – 1317. Designing, modelling and implementing interactive systems.
- Monperrus, M., Jézéquel, J.-M., Champeau, J., and Hoeltzener, B. (2008a). *Measuring models*. cited By (since 1996)4.
- Monperrus, M., Jézéquel, J.-M., Champeau, J., and Hoeltzener, B. (2008b). A model-driven measurement approach. In *Proceedings of the ACM/IEEE 11th International Conference on Model Driven Engineering Languages and Systems (MODELS'2008)*.
- Moody, D. (2006). Dealing with "map shock": A systematic approach for managing complexity in requirements modelling. Luxembourg.
- Moody, D. (2009). The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Trans. Softw. Eng.*, 35(6):756–779.
- Moody, D. L. (2003). The method evaluation model: a theoretical model for validating information systems design methods. In *Proceedings of the 11th European Conference on Information Systems, ECIS 2003, Naples, Italy 16-21 June 2003*, pages 1327–1336.
- Moody, D. L. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data & Knowledge Engineering*, 55(3):243–276.
- Moody, D. L. and Shanks, G. G. (2003). Improving the quality of data models: empirical validation of a quality management framework. *Inf. Syst.*, 28(6):619–650.
- Moody, D. L., Sindre, G., Brasethvik, T., and Sølberg, A. (2002). Evaluating the quality of process models: Empirical testing of a quality framework. In *Proceedings of the 21st International Conference on Conceptual Modeling, ER '02*, pages 380–396, London, UK, UK. Springer-Verlag.
- Mora, B., Ruiz, F., García, F., and Piattini, M. (2006). Definición de lenguajes de modelos mda vs dsl.
- Morais, F. and da Silva, A. R. (2015). Assessing the quality of user-interface modeling languages. In *Proceedings of the 17th International Conference on Enterprise Information Systems*, pages 311–319.
- Mrdalj, S. and Jovanovic, V. M. (2005). Mapping the uml to the zachman framework. In *AMCIS*.
- Muntermann, J., Nickerson, R., and Varshney, U. (2015). Towards the development of a taxonomic theory. In *Proceedings of the Twenty-first Americas Conference on Information Systems (AMCIS), Puerto Rico, 2015, AMCIS'15*. AIS Electronic Library (AISeL).
- Mussbacher, G., Amyot, D., Breu, R., Bruel, J.-M., Cheng, B., Collet, P., Combemale, B., France, R., Heldal, R., Hill, J., Kienzle, J., Schöttle, M., Steimann, F., Stikkolorum, D., and Whittle, J. (2014). The relevance of model-driven engineering thirty years from now. In Dingel, J., Schulte, W., Ramos, I., Abrahão, S., and Insfran, E., editors, *Model-Driven Engineering Languages and Systems*, volume 8767 of *Lecture Notes in Computer Science*, pages 183–200. Springer International Publishing.
- Nelson, H. J., Poels, G., Genero, M., and Piattini, M. (2005). Quality in conceptual modeling: Five examples of the state of the art. *Data Knowl. Eng.*, 55(3):237–242.
- Nelson, H. J., Poels, G., Genero, M., and Piattini, M. (2012). A conceptual modeling quality framework. *Software Quality Journal*, 20(1):201–228.
- Nogueira, J. M., Romero, D., Espadas, J., and Molina, A. (2013). Leveraging the zachman framework implementation using action – research methodology – a case study: Aligning the enterprise architecture and the business goals. *Enterp. Inf. Syst.*, 7(1):100–132.
- Noran, O. (2003). An analysis of the zachman framework for enterprise architecture from the {GERAM} perspective. *Annual Reviews in Control*, 27(2):163 – 183.
- Nugroho, A. (2009). Level of detail in {UML} models and its impact on model comprehension: A controlled experiment. *Information and Software Technology*, 51(12):1670 – 1685. Quality of {UML} Models.

- Object and Reference Model Subcommittee of the Architecture Board (2005). A Proposal for an MDA Foundation Model, ormsc/05-04-01. Technical report, Object Management Group.
- Object and Reference Model Subcommittee of the Architecture Board (2010). A Proposal for an MDA Foundation Model, ormsc/10-09-06. Technical report, Object Management Group.
- of Surrey, U. (2016). Analysing qualitative research data (online).
- Olivé, A. (2001). Taxonomies and derivation rules in conceptual modeling. In *Proceedings of the 13th International Conference on Advanced Information Systems Engineering, CAiSE '01*, pages 417–432, London, UK, UK. Springer-Verlag.
- OMG (2003). MDA Guide Version 1.0.1.
- OMG (2014a). Diagram Definition (DD) Version 1.1 Document Number: ptc/2014-03-02 .
- OMG (2014b). MDA Guide revision 2.0.
- omg (2015). *Meta Object Facility (MOF) Core Specification Version 2.5*.
- OMG (2016). Ea-mde: What affects the success of mde in industry @ONLINE.
- (OMG), O. M. G. (2011). Business process model and notation (bpmn) version 2.0. Technical report.
- Opdahl, A. L. and Henderson-Sellers, B. (2002). Ontological evaluation of the uml using the bunge-wand-weber model. *Software and Systems Modeling*, 1(1):43–67.
- Ortiz, J. C., Quinteros, E., Abuawad, O., Torricio, F., and Ojeda, J. D. (2013). Primer parcial-mda (model driven architecture) @ONLINE.
- Ostadzadeh, S. S., Aliche, F. S., and Ostadzadeh, S. A. (2007). A method for consistent modeling of zachman framework cells. In *Advances and Innovations in Systems, Computing Sciences and Software Engineering*, pages 375–380. Springer.
- Ouertani, M., Baïna, S., Gzara, L., and Morel, G. (2011). Traceability and management of dispersed product knowledge during design and manufacturing. *Computer-Aided Design*, 43(5):546 – 562. Emerging Industry Needs for Frameworks and Technologies for Exchanging and Sharing Product Lifecycle Knowledge.
- Pallec, X. and Dupuy-Chessa, S. (2013). Support for quality metrics in metamodeling. pages 23–31. cited By (since 1996)0.
- Panach, J. I., España, S., Dieste, Ó., Pastor, Ó., and Juristo, N. (2015a). In search of evidence for model-driven development claims: An experiment on quality, effort, productivity and satisfaction. *Information and Software Technology*, 62:164 – 186.
- Panach, J. I., Juristo, N., Valverde, F., and Pastor, Ó. (2015b). A framework to identify primitives that represent usability within model-driven development methods. *Information and Software Technology*, 58(0):338 – 354.
- Parsons, J. and Wand, Y. (2008). Using cognitive principles to guide classification in information systems modeling. *MIS Quarterly*, 32(4):pp. 839–868.
- Pastor, Ó. and España, S. (2012). *Full Model-Driven Practice: From Requirements to Code Generation*, pages 701–702. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Pastor, O., Insfrán, E., Pelechano, V., Romero, J., and Merseguer, J. (2013). *OO-METHOD: An OO Software Production Environment Combining Conventional and Formal Methods*, pages 139–152. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Pastor, O. and Molina, J. C. (2007). *Model-Driven Architecture in Practice: A Software Production Environment Based on Conceptual Modeling*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- Petre, M. (2013). Uml in practice. In *Proceedings of the 2013 International Conference on Software Engineering, ICSE '13*, pages 722–731, Piscataway, NJ, USA. IEEE Press.
- Piattini, M., Poels, G., Genero, M., Fernández-Saez, A. M., and Nelson, H. J. (2011). Research review: A systematic literature review on the quality of uml models. *J. Database Manage.*, 22(3):46–70.
- Picek, R. and Strahonja, V. (2007). Model driven development-future or failure of software development. In *IIS*, volume 7, pages 407–413.
- Pierson, H. (2007). Model-driven development (part 2) @ONLINE.
- Platania, G. (2016). Model driven architecture don't work! @ONLINE.
- Poruban, J., Bacikova, M., Chodarev, S., and Nosal, M. (2014). Pragmatic model-driven software development from the viewpoint of a programmer: Teaching experience. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*, pages 1647–1656. IEEE.
- Prat, N., Comyn-Wattiau, I., and Akoka, J. (2015). A taxonomy of evaluation methods for information systems artifacts. *Journal of Management Information Systems*, 32(3):229–267.
- Priss, U. (2006). Formal concept analysis in information science. *Annual Rev. Info. Sci. & Technol.*, 40(1):521–543.

- Quevedo, C. L., España, S., García, M. E., Hita, A. G., Cardona, T. G., Grabis, J., Henkel, M., Pascual, R. J., Jokste, L., and Giromé, F. V. (2015). Capability as a service in digital enterprises collaborative project number 611351 deliverable 4.2 capability models for soa technological platforms. Technical report.
- Quintero, J., Rucínque, P., Anaya, R., and Piedrahita, G. (2012). How face the top mde adoption problems. In *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*, pages 1–10. IEEE.
- Quintero, J. B. and Muñoz, J. F. D. (2011). Reflexiones acerca de la adopción de enfoques centrados en modelos en el desarrollo de software. *Ingeniería y Universidad*, 15(1):219–243.
- Quora (2014). Is uml trivial? @ONLINE.
- Quora (2015a). Is the uml still widely used? is it still an important tool in today's industry?@ONLINE.
- Quora (2015b). Why has uml usage declined in industry? @ONLINE.
- Reijers, H. A., Mendling, J., and Recker, J. (2015). *Business Process Quality Management*, pages 167–185. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Richards, D. (2000). A situated cognition approach to conceptual modelling. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, pages 10 pp. vol.2–.
- Rios, E., Bozheva, T., Bediaga, A., and Guilloreau, N. (2006). Mdd maturity model: A roadmap for introducing model-driven development. In Rensink, A. and Warmer, J., editors, *Model Driven Architecture – Foundations and Applications*, volume 4066 of *Lecture Notes in Computer Science*, pages 78–89. Springer Berlin Heidelberg.
- Romero, J., Jaen, J., and Vallecillo, A. (2009). Realizing correspondences in multi-viewpoint specifications. In *Enterprise Distributed Object Computing Conference, 2009. EDOC '09. IEEE International*, pages 163–172.
- Rosemann, M. and Green, P. (2000). Integrating multi-perspective views into ontological analysis. In *Proceedings of the Twenty First International Conference on Information Systems, ICIS '00*, pages 618–627, Atlanta, GA, USA. Association for Information Systems.
- Rueda, U., España, S., and Ruiz, M. (2015). GREAT process modeller user manual. *CoRR*, abs/1502.07693.
- Ruiz, M., Costal, D., España, S., Franch, X., and Pastor, Ó. (2014). Integrating the goal and business process perspectives in information system analysis. In Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., and Horkoff, J., editors, *Advanced Information Systems Engineering*, volume 8484 of *Lecture Notes in Computer Science*, pages 332–346. Springer International Publishing.
- Saada, H., Dolques, X., Huchard, M., Nebut, C., and Sahraoui, H. (2012). Learning Model Transformations from Examples using FCA: One for All or All for One? In Szathmary, L. and Priss, U., editors, *CLA'2012: 9th International Conference on Concept Lattices and Applications*, pages 45–56, Fuengirola (Málaga), Spain. Universidad de Malaga.
- SăCărea, C. and Varga, V. (2014). Triadic approach to conceptual design of xml data. *Studia Universitatis Babeş-Bolyai, Informatica*, 59.
- Saghafi, A. and Wand, Y. (2014). Do ontological guidelines improve understandability of conceptual models? a meta-analysis of empirical work. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*, pages 4609–4618.
- Salih, A. S. M. and Abraham, A. (2015). Ambient intelligence healthcare monitoring information architecture (aihmia). *International Journal of Computer Information Systems and Industrial Management Applications*, pages 41–52.
- Salman, I., Misirli, A. T., and Juristo, N. (2015). Are students representatives of professionals in software engineering experiments? In *Proceedings of the 37th International Conference on Software Engineering - Volume 1, ICSE '15*, pages 666–676, Piscataway, NJ, USA. IEEE Press.
- Sandkuhl, K. and Stirna, J. Template for the documentation of method components in caas. Technical report, Capability as a Service in digital enterprises - Collaborative Project Number 611351.
- Sandkuhl, K., Stirna, J., Persson, A., and Wiotzki, M. (2014a). *Enterprise Modeling: Tackling Business Challenges with the 4EM Method*. Springer Publishing Company, Incorporated.
- Sandkuhl, K., Stirna, J., Persson, A., and Wiłotzki, M. (2014b). *Overview of the 4EM Method*, pages 75–86. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Sayeb, K., RIEU, D., Mandran, N., and Dupuy-Chessa, S. (2012). Qualité des langages de modélisation et des modèles : vers un catalogue des patrons collaboratifs. pages 429–446, Montpellier, France.
- Schalles, C. (2013). *A Framework for Usability Evaluation of Modeling Languages (FUEML)*, chapter 4, pages 43–68. Springer Fachmedien Wiesbaden.
- Schmidt, D. C. Strategic management of architectural technical debt (on-line).
- Seddon, P. B. (1997). A respecification and extension of the delone and mclean model of is success. *Information Systems Research*, 8(3):240–253.
- Sessions, R. (2007). Comparison of the Top Four Enterprise Architecture Methodologies. Technical report.
- Shanks, G. G. and Darke, P. (1997). Quality in conceptual modelling: Linking theory and practice. In *The Third Pacific Asia Conference on Information Systems, PACIS 1997, Brisbane, Australia, April, 1-5, 1997*, page 76.

- She, S., Ryssel, U., Andersen, N., Wąsowski, A., and Czarnecki, K. (2014). Efficient synthesis of feature models. *Information and Software Technology*, 56(9):1122 – 1143. Special Sections from “Asia-Pacific Software Engineering Conference (APSEC), 2012” and “Software Product Line conference (SPLC), 2012”.
- Shekhovtsov, V. A., Mayr, H. C., and Kop, C. (2014). Chapter 3 - harmonizing the quality view of stakeholders. In Stal, I. M. B. E. R., editor, *Relating System Quality and Software Architecture*, pages 41 – 73. Morgan Kaufmann, Boston.
- Shimajima, A. (2015). Semantic properties of diagrams and their cognitive potentials.
- Shuman, E. A. (2010). Understanding executable architectures through an examination of language model elements. In *Proceedings of the 2010 Summer Computer Simulation Conference, SCSC '10*, pages 483–497, San Diego, CA, USA. Society for Computer Simulation International.
- Siau, K. (2010). An analysis of unified modeling language (uml) graphical constructs based on bwv ontology. *Journal of Database Management*, 21(1):i–viii. cited By (since 1996)2.
- Siau, K. and Rossi, M. (1998). Evaluation of information modeling methods—a review. In *System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on*, volume 5, pages 314–322 vol.5.
- Silva, F. S., Soares, F. S. F., Peres, A. L., de Azevedo, I. M., Vasconcelos, A. P. L., Kamei, F. K., and de Lemos Meira, S. R. (2015). Using {CMMI} together with agile software development: A systematic review. *Information and Software Technology*, 58(0):20 – 43.
- Singh, Y. and Sood, M. (2009). Model driven architecture: A perspective. In *Advance Computing Conference, 2009. IACC 2009. IEEE International*, pages 1644–1652. IEEE.
- Smith, R. (2013). On the value of a taxonomy in modeling. In Tolk, A., editor, *Ontology, Epistemology, and Teleology for Modeling and Simulation*, volume 44 of *Intelligent Systems Reference Library*, pages 241–254. Springer Berlin Heidelberg.
- Sokal, R. R. (1974). Classification: Purposes, principles, progress, prospects. *Science*, 185(4157):1115–1123.
- Sowa, J. F. and Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Syst. J.*, 31(3):590–616.
- Staron, M. (2006). Adopting model driven software development in industry – a case study at two companies. In Nierstrasz, O., Whittle, J., Harel, D., and Reggio, G., editors, *Model Driven Engineering Languages and Systems*, volume 4199 of *Lecture Notes in Computer Science*, pages 57–72. Springer Berlin Heidelberg.
- Stirna, J. (2013). Caas – capability as a service for digital enterprises. fp 7 ict programme collaborative project no: 611351.
- Stockdale, R. and Standing, C. (2006). An interpretive approach to evaluating information systems: A content, context, process framework. *European Journal of Operational Research*, 173(3):1090 – 1102.
- Störrle, H. and Fish, A. (2013). Towards an operationalization of the physics of notations for the analysis of visual languages. In Moreira, A., Schätz, B., Gray, J., Vallecillo, A., and Clarke, P., editors, *Model-Driven Engineering Languages and Systems*, volume 8107 of *Lecture Notes in Computer Science*, pages 104–120. Springer Berlin Heidelberg.
- Tairas, R. and Cabot, J. (2013). Corpus-based analysis of domain-specific languages. *Software & Systems Modeling*, pages 1–16.
- Tang, A., Han, J., and Chen, P. (2004). A comparative analysis of architecture frameworks. In *Software Engineering Conference, 2004. 11th Asia-Pacific*, pages 640–647.
- Teppola, S., Parviainen, P., and Takalo, J. (2009). Challenges in deployment of model driven development. In *Software Engineering Advances, 2009. ICSEA '09. Fourth International Conference on*, pages 15–20.
- Thalheim, B. (2011). The theory of conceptual models, the theory of conceptual modelling and foundations of conceptual modelling. In Embley, D. W. and Thalheim, B., editors, *Handbook of Conceptual Modeling*, pages 543–577. Springer Berlin Heidelberg.
- Thalheim, B. (2012). Syntax, semantics and pragmatics of conceptual modelling. In Bouma, G., Ittoo, A., Métails, E., and Wortmann, H., editors, *Natural Language Processing and Information Systems*, volume 7337 of *Lecture Notes in Computer Science*, pages 1–10. Springer Berlin Heidelberg.
- Thompson, S. K. and Seber, G. A. F. (1996). *Adaptive Sampling*. John Wiley & Sons, New York, USA, 1st edition.
- Tom, E., Aurum, A., and Vidgen, R. (2013). An exploration of technical debt. *Journal of Systems and Software*, 86(6):1498 – 1516.
- Tomassetti, F., Torchiano, M., Tiso, A., Ricca, F., and Reggio, G. (2012). Maturity of software modelling and model driven engineering: A survey in the italian industry. In *Evaluation Assessment in Software Engineering (EASE 2012), 16th International Conference on*, pages 91–100.
- Tone (2010). What are the benefits and risks of moving to a model driven architecture approach?
- Torchiano, M., Tomassetti, F., Ricca, F., Tiso, A., and Reggio, G. (2013). Relevance, benefits, and problems of software modelling and model driven techniques—a survey in the italian industry. *Journal of Systems and Software*, 86(8):2110 – 2126.
- Tupper, C. D. (2011). 2 - enterprise architecture frameworks and methodologies. In Tupper, C. D., editor, *Data Architecture*, pages 23 – 55. Morgan Kaufmann, Boston.

- University, C. S. (2017). An introduction to content analysis (online).
- Utz, W., Woitsch, R., and Karagiannis, D. (2011). Conceptualisation of hybrid service models: An open models approach. In *Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual*, pages 494–499.
- Vallecillo, A. (2010). *On the Combination of Domain Specific Modeling Languages Modelling Foundations and Applications*, volume 6138 of *Lecture Notes in Computer Science*, pages 305–320. Springer Berlin / Heidelberg.
- Vallecillo, A. (2014). On the industrial adoption of model driven engineering. is your company ready for mde? *International Journal of Information Systems and Software Engineering for Big Companies*.
- Van Amstel, M. (2010). The right tool for the right job: Assessing model transformation quality. pages 69–74. cited By (since 1996)3.
- van Amstel, M., Lange, C., and van den Brand, M. (2009). Using metrics for assessing the quality of asf+sdf model transformations. In Paige, R., editor, *Theory and Practice of Model Transformations*, volume 5563 of *Lecture Notes in Computer Science*, pages 239–248. Springer Berlin Heidelberg.
- Van Der Straeten, R., Mens, T., and Van Baelen, S. (2009). Challenges in model-driven software engineering. In Chaudron, M., editor, *Models in Software Engineering*, volume 5421 of *Lecture Notes in Computer Science*, pages 35–47. Springer Berlin Heidelberg.
- Vara, J. M. and Marcos, E. (2012). A framework for model-driven development of information systems: Technical decisions and lessons learned. *Journal of Systems and Software*, 85(10):2368 – 2384. Automated Software Evolution.
- Vargas, A., Boza, A., Patel, S., Patel, D., Cuenca, L., and Ortiz, A. (2015). Inter-enterprise architecture as a tool to empower decision-making in hierarchical collaborative production planning. *Data & Knowledge Engineering*, pages –.
- Visser, Eelco, V. D. A. and Jos, W. (2007). Model-driven software evolution: A research agenda. In *In Proc. Int. Ws on Model-Driven Software Evolution held with the ECSMR'07*.
- Wand, Y., Monarchi, D. E., Parsons, J., and Woo, C. C. (1995). Theoretical foundations for conceptual modelling in information systems development. *Decision Support Systems*, 15(4):285 – 304.
- Wang, Y., Zhao, L., Wang, X., Yang, X., and Supakkul, S. (2013). Plant: A pattern language for transforming scenarios into requirements models. *International Journal of Human-Computer Studies*, 71(11):1026–1043.
- Weerd, I. v. d. and Brinkkemper, S. (2009). Meta-modeling for situational analysis and design methods. In *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications*, pages 38–58. IGI Global.
- Wegmann, A., Kotsalainen, A., Matthey, L., Regev, G., and Giannattasio, A. (2008). Augmenting the zachman enterprise architecture framework with a systemic conceptualization. In *Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference, EDOC '08*, pages 3–13, Washington, DC, USA. IEEE Computer Society.
- Wehrmeister, M. A., de Freitas, E. P., Binotto, A. P. D., and Pereira, C. E. (2014). Combining aspects and object-orientation in model-driven engineering for distributed industrial mechatronics systems. *Mechatronics*, 24(7):844 – 865. 1. Model-Based Mechatronic System Design 2. Model Based Engineering.
- Welty, C. and Guarino, N. (2001). Supporting ontological analysis of taxonomic relationships. *Data Knowl. Eng.*, 39(1):51–74.
- Whitman, L., Ramachandran, K., and Ketkar, V. (2001). A taxonomy of a living model of the enterprise. In *Proceedings of the 33rd conference on Winter simulation*, pages 848–855. IEEE Computer Society.
- Whittle, J., Hutchinson, J., and Rouncefield, M. (2014). The state of practice in model-driven engineering. *Software, IEEE*, 31(3):79–85.
- Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., and Heldal, R. (2013). Industrial adoption of model-driven engineering: Are the tools really the problem? In Moreira, A., Schätz, B., Gray, J., Vallecillo, A., and Clarke, P., editors, *Model-Driven Engineering Languages and Systems*, volume 8107 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg.
- Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., and Heldal, R. (2015). A taxonomy of tool-related issues affecting the adoption of model-driven engineering. *Software & Systems Modeling*, pages 1–19.
- Wieringa, R. (2009). Design science as nested problem solving. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, DESRIST '09*, pages 8:1–8:12, New York, NY, USA. ACM.
- Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14*, pages 38:1–38:10, New York, NY, USA. ACM.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., and Regnell, B. (2012a). *Experimentation in Software Engineering*. Springer.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012b). *Operation*, pages 117–122. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012c). *Planning*, pages 89–116. Springer Berlin Heidelberg, Berlin, Heidelberg.

- Wolff, K. E. (1993). A first course in formal concept analysis. *StatSoft*, 93:429–438.
- Wolterink, T. (2009). Operational semantics applied to model driven engineering.
- Xiao He, Paris Avgeriou, P. L. Z. L. (2016). Technical debt in mde: A case study on gmf/emf-based projects. In *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016)*, Saint-Malo France.
- Yue, T., Ali, S., and Elaasar, M. (2010). A framework for measuring quality of models: Experiences from a series of controlled experiments. Technical Report 2010-17 (v2), Simula Research Laboratory.
- Zachman, J. A. (1987). A framework for information systems architecture. *IBM Syst. J.*, 26(3):276–292.
- Zachman, J. A. (2003). Excerpted from the zachman framework: A primer for enterprise engineering and manufacturing (omg-brwg rfi response assessment).
- Zhao, L., Letsholo, K., Chioasca, E.-V., Sampaio, S., and Sampaio, P. (2012). Can business process modeling bridge the gap between business and information systems? In *Proceedings of the 27th Annual ACM Symposium on Applied Computing, SAC '12*, pages 1723–1724, New York, NY, USA. ACM.

Index

- 6C, 19, 30, 88, 121
- Abstract syntax, 206, 209, 214
- Abstraction, 12
 - Level of, 213
 - Levels, 51, 65, 93
- Abstraction level, 207, 208, 211
- Architecture description, 66, 69, 84, 213

- BPMN, 110, 112–114, 181, 187

- CDD methodology, 77
- Cell, 206, 208, 211–213
- CIM, 8, 68, 95, 113, 115
- Classification, 84, 99
 - Clarity of, 99
- Classifier, 213
- Communication, 66
- Communicational Analysis, 114
- Completeness, 99
- Concern, 208, 211
- Concerns, 1, 62, 212, 213
 - Separation of, 66
- Concrete syntax, 214
- Construct, 211
- CTT, 187

- Design Science, 4

- EMAT, 76, 103, 108–110, 112, 120, 187
- ER, 187
- Exhaustiveness, 99
- Expandability, 99

- FCA, 92, 100–103, 108, 109, 112
- Foundational construct, 206, 208, 213
- Framework, 3
 - Conceptual, 61
 - Methodological, 61
 - Technological, 91
- FRISCO, 10, 13, 88

- Grammar, 214
- Guideline, 214

- Information, 206, 208, 211, 214
- Information systems, 1, 8, 61
 - And taxonomies, 126
 - Description of, 64
 - Theory types for, 184
- Integration, 211

- Lattice, 105

- Mapping, 207, 209, 213
- MDA, 7, 8, 10, 26, 37, 47, 53, 58, 65, 95, 99
- MDE, 1, 7, 8, 19, 37, 47, 56, 58, 59, 66, 71, 73, 88, 92, 177, 178, 185, 187
 - Compliant, 119
 - Environment, 3
 - Metrics, 74
 - Organizational support of, 38
 - Principles, 50
 - Quality issues in, 10
- Meta-property of Taxonomies, 96
 - Dependency, 97
 - Essential, 97
 - Identity, 96
 - Unity, 96
- Metamodel, 214
- Method Evaluation model, 146
- Metrics, 183
- MMQEF, 61, 67, 76, 88, 89, 94, 99, 110, 113, 115, 118–120, 126, 134, 175, 177, 178, 181, 182, 185, 205
 - Components, 69
 - Empirical validation, 137
- Model, 11, 47, 48, 209, 214
 - Element, 208, 209
- Modelling language, 206, 209, 211
 - Organization, 206
- Modelling languages, 1, 11, 55, 61, 183, 212, 214
 - HCI of, 183
 - in combination, 49

- Models, 62
 - Conceptual, 8
 - Dynamics of, 54
 - Integration, 67
- MOF, 7, 26
- OCL, 7, 26
- OMG, 7, 8, 26, 47, 58, 72, 112
- Ontological analysis, 63
 - Agile, 120
- Ontology, 84
- PDD, 69, 71, 205
- Physics of Notations, 121, 189
- PIM, 8, 68, 115
- Platform, 12
- PSM, 8, 68, 115
- QEF, 23
- Quality, 7
 - Academic/research issues, 36
 - Analytics, 71
 - Categories of, 19
 - Definition of, 9
 - Evaluation of, 69
 - for MDE, 94
 - Frameworks, 38
 - Industrial issues, 36
 - Issues, 66
 - Metrics, 51
 - Open challenges, 47
- Representation, 206, 214
 - Diagram, 214
 - Textual, 214
- Semantic, 120
- Semantic domain, 208
- SEQUAL, 19, 30, 88, 120, 182
- Suitability, 13, 66, 92, 212
- SysML, 7, 26
- System, 11
- Taxonomic
 - Metamodel preservation, 208
- Taxonomy, 66, 84, 92, 93, 95, 209
 - Analysis, 64, 83, 101
 - Analytic, 93
 - Cells, 206
 - Definition of, 126
 - Formal support, 96
 - Framework, 66
 - Reference, 64, 70, 208, 213
 - Structure, 67, 85
 - The Backbone taxonomy, 97
 - Theory, 92, 98
 - Zachman, 72, 128, 134, 136
- Technical debt, 56, 183
- The OO-Method, 114, 116
- Tool, 103
- Tools, 37, 54, 149, 151
 - Support, 54, 76
- Trace, 207, 208, 213
 - Mechanism, 209
- Traceability, 92, 207
- Trade-off, 83
- Trade-off analysis, 109
- Transformation, 12, 213
- Transformations, 70, 73, 120
 - Capacities of, 209
 - Models, 52, 67, 95
 - Specifications, 209
- UML, 7, 26, 110, 113, 114, 181, 187
- Usefulness, 98
- View, 11
- Viewpoint, 1, 11, 207–209, 213
- Views, 62
- Zachman framework, 62, 64, 85, 93, 98, 109, 125, 128, 131, 136



Coffee is one of the most famous drinks in the world and the second most traded commodity¹. Its use on the cover was intentional because quality of coffee is a challenge in its planting, harvesting, threshing, roasting, and preparation. Some quality properties are from the bean, others are from the context of the plant and its surroundings, others are from the coffee brewing techniques, and others are from the people who make the coffee.

Whatever the methods used for addressing quality issues, the final product meets the noblest of purposes.

Quality at the MDE level is only comparable with the satisfaction of a good coffee . . .

La maîtresse de la maison doit toujours s'assurer que le café est excellent; et le maître, que les liqueurs sont de premier choix.

The hostess should always ensure that the coffee is optimal, and the host that of the wines are of top quality.

Jean Anthelme Brillat-Savarin

¹ <http://www.globalexchange.org/fairtrade/coffee/faq>

A framework for evaluating the quality of modelling languages in MDE environments

Information System development methods that follow the model-driven engineering (MDE) paradigm commonly prescribe the use of multiple viewpoints for addressing concerns. Viewpoints have associated modelling languages in accordance with the MDE paradigm. They often have different abstraction and granularity levels. The enactment of such methods in projects involves risks that threaten their success.

This work proposes a method to evaluate the quality of modelling languages, that is, the assessment of quality issues that appear when modelling languages are used within an MDE project. These issues include their suitability, the explicit treatment of the semantics, the model transformations as a controlled process, and the computational support of the modelling artifacts being analyzed. The aim is to aid MDE method engineers in the task of systematically applying modelling languages in the development of Information Systems.

The evaluation of quality issues in MDE projects is based on an analytics process that is performed with a taxonomy. This taxonomy is also a reference architecture for Information Systems. We use the classification language, the taxonomic structure, and the rules proposed by the taxonomy to evaluate modelling languages and to derive inferences of quality issues in modelling projects.