Summer 6-15-2016

A SEMANTIC META MODEL OF SPREADSHEETS

Thomas Reschenhofer
Technical University of Munich, reschenh@in.tum.de

Bernhard Waltl
Technical University of Munich, b.waltl@tum.de

Sirma Gjorgievska
Technical University of Munich, sirma.gjorgievska@tum.de

Florian Matthes
Technical University of Munich, matthes@in.tum.de

Recommended Citation
Reschenhofer, Thomas; Waltl, Bernhard; Gjorgievska, Sirma; and Matthes, Florian, "A SEMANTIC META MODEL OF SPREADSHEETS" (2016). *Research Papers*. 90.
http://aisel.aisnet.org/ecis2016_rp/90

A SEMANTIC META MODEL OF SPREADSHEETS

Research Paper

Reschenhofer, Thomas, Technical University of Munich, Germany, reschenh@in.tum.de

Waltl, Bernhard, Technical University of Munich, Germany, b.waltl@tum.de

Gjorgievska, Sirma, Technical University of Munich, Germany, sirma.gjorgievska@tum.de

Matthes, Florian, Technical University of Munich, Germany, matthes@in.tum.de

Abstract

Spreadsheets are widely used as decision support systems. However, as numerous studies have shown, many of them contain errors. A major reason for this is the lack of an explicit design of the data model, which at the same time ensures flexibility and is one of the defining features of the spreadsheet paradigm. Therefore, in recent years many approaches to model-driven spreadsheet development emerged addressing this issue.

In this paper, we propose a meta model which defines how the semantic models of today's spreadsheets can be described. Thereby, it builds on the foundations of related approaches to model-driven spreadsheet development, and also captures semantic patterns of spreadsheets typically occurring in practice. Based on this semantic meta model, novel spreadsheet solutions can address concrete semantic patterns and thus reduce the risk of errors on the one hand, and improve the usability of spreadsheets on the other hand.

Keywords: Spreadsheet, Model-driven design, Semantic patterns, Design science research.

1 Introduction

Spreadsheets are widely used in industry, in particular as business applications and decision support systems (Pemberton and Robson, 2000; Grossman *et al.*, 2005). They are used in nearly all companies in the US and Europe (Bradley and McDaid, 2009; Panko, 2006) and in a plethora of business domains, e.g., financial reporting, workload planning, and general administration (Pemberton and Robson, 2000; Reschenhofer and Matthes, 2015a). In many cases, spreadsheets are also critical for the support of business processes (Gable *et al.*, 1991). As a consequence, errors in spreadsheets can have a significant impact on an organization's business operation (Caulkins *et al.*, 2007; Powell *et al.*, 2009). And as numerous studies have shown (Panko, 2006, 2000; Panko and Halverson Jr., 1996; Reschenhofer and Matthes, 2015b), the frequency of errors in spreadsheets in practice is indeed very high. Many researchers addressed this issue by proposing approaches for avoiding, identifying, classifying and fixing spreadsheet errors (Powell *et al.*, 2008). Furthermore, Brown and Gould (1987) also studied causes of spreadsheet errors, including typing and copying mistakes, errors in logic and formulas, erroneous cell references, and misplacement of data. Most of those causes are reducible to the implicit design approach of spreadsheets (Erwig *et al.*, 2005), i.e., the actual design of a spreadsheet is implicitly defined by the placement of data within the spreadsheet grid and its formulas (Hermans *et al.*, 2012). Therefore, there is no explicit definition of the spreadsheet's semantic model and thus no separation between its data and its structure. In this context, the spreadsheet's semantic model captures the type, structure, and logic of the spreadsheet and its data.

The lack of separation of data and structure in spreadsheets induces several considerable drawbacks. On the one hand, the usability of spreadsheets suffers from this issue due to the fact that the spreadsheet tool is not able to provide any guidance to end-users based on a predefined semantic model. By such a semantic model (e.g., a list of semantic entities consisting of certain attributes, or financial time series), the spreadsheet would be able to support the user in managing, analyzing, and visualizing the data by proposing proper methods or visualization techniques tailored to the actual semantic model (e.g., line charts for time series). On the other hand, the lack of an explicit spreadsheet design significantly impedes the maintenance of spreadsheets, since it is not possible to define constraints regarding the change of spreadsheet data and structure in order to control the evolution of a spreadsheet (Hermans *et al.*, 2012; Cunha *et al.*, 2012c). This means that the implicit structure of a spreadsheet can be changed by end-users by simply changing the respective data. Apart from those usability and maintenance issues, the missing separation of data and structure also imposes legal issues, especially in domains like the financial sector. Spreadsheets have become predominant in modeling, assessing, and evaluating products and risks. Consequently, several withdraws exist regarding risk management, validity, and reliability (Bretz, 2012, 2009), which caused the legislator to react by promulgating regulations, e.g., Basel II (Basel Committee on Banking Supervision, 2004) in the EU, or the Sarbanes-Oxley Act (United States Congress, 2002) in the US. Those regulations now prompt supervising authorities to focus in particular on spreadsheets, their development, usage, and life-cycle (Bretz, 2012). It is exactly the lack of separation between structure and data which is highlighted by supervising authorities as a major concern when using spreadsheets (Panko and Ordway, 2005).

Although the implicit design approach is a major cause of spreadsheet errors, it induces great flexibility and is a defining feature of the spreadsheet paradigm. Therefore, a novel spreadsheet addressing the aforementioned usability, maintenance, and legal issues has to provide respective facilities for explicitly defining the spreadsheet's semantic model while preserving a certain degree of flexibility of the spreadsheet design and a spreadsheet-like user experience. As a first step towards such a novel spreadsheet approach, this paper aims to reveal how to describe the semantic structure of today's spreadsheets. Capturing the semantic structure of spreadsheets with models can be done by respective meta models. For example, the Meta Object Facility (MOF) does the very same for the Unified Model Language (UML) by describing rules and constraints for designing UML models (Object Management Group, 2014a). Analogous to MOF, this paper proposes a semantic meta model which is able to describe the semantic models of prevalent spreadsheets, including all semantic patterns which occur in today’s spreadsheets. Therefore, the present work answers the following research questions:

1) What are common semantic models and patterns of real-world spreadsheets?
2) What is a suitable meta model to describe those semantic models?

The answers to those questions serve as foundation for future research activities, namely the design of a novel spreadsheet solution capturing the semantic structure of its data and thus addressing usability, maintenance, and legal issues.

The remainder of this paper is organized as follows: Section 2 summarizes related work in the field of spreadsheet research and in particular in the field of model-based spreadsheet approaches. Thereafter, Section 3 describes the applied research method for deriving a semantic meta model for spreadsheets based on existing related literature as well as existing real-world spreadsheets. Section 4 covers this paper’s main contribution, namely the description of a semantic meta model of spreadsheets capable of describing the semantics of prevalent real-world spreadsheets, while Section 5 shows the results of the evaluation of the proposed meta model. Finally, Section 6 concludes the present work, critically reflects on its results, and proposes future research opportunities.

2 Related Work

Due to the dissemination and popularity of spreadsheets in practice, they have been subject of research for over decades (Panko, 2006), including research about spreadsheet models, inference of types, and other areas which are related to the present work, and which are described in the following section.

Mittermeir and Clermont (2002) introduced an approach for automatically detecting the semantic structure of spreadsheets. In this sense, they define equivalence classes of spreadsheet cells, e.g., logical equivalence for cells with the structurally same formulas, but different constant values and cell references, or structural equivalence for cells containing the same operations in the same order. Based on those equivalence classes as well as the spatial situation of cells, semantic classes can be derived, which in turn can be visualized in a dependency graph whose edges represent cell references across the semantic classes. Although the approach of Mittermeir and Clermont already supports the identification of individual semantic classes within spreadsheets, it lacks the facilities to describe how those individual semantic classes belong together, and how to compose them to an integrated semantic model of the whole spreadsheet as aimed by the present work.

Partially based on the work by Mittermeir and Clermont, Hermans *et al.* (2010) developed a tool for automatically extracting class diagrams from spreadsheets. The foundations for the class diagram extraction are a set of simple cell types – namely label, data, formula, and empty cells – and a library of common structural and semantic patterns which can be built to classes. The simplest semantic pattern of this library is the *Simple class* pattern representing a list of static instances. Based on this, Hermans et al. suggest more sophisticated patterns including methods (columns containing row-based formulas), aggregations (single cells containing a column-based formula), and associations (columns containing references to elements of another class). However, since their goal is the generation of UML class diagrams, the respective meta model determining the actual set of rules for designing spreadsheet models is limited to the constraints as defined by the MOF (Object Management Group, 2014a). And as their evaluation shows, not even 50 % of the spreadsheets of the EUSES Spreadsheet Corpus (Fisher and Rothermel, 2005) can be processed with their tool, which implies that there is a huge share of semantic structures which cannot be captured by the approach of Hermans *et al.* (2010). The present work proposes an alternative meta model which is able to describe more complex semantic spreadsheet models.

A very famous approach to model-based spreadsheet development is the ClassSheet approach (Erwig and Engels, 2005). Thereby, the semantics of a spreadsheet is defined by a template capturing structural information, layout information, and data dependencies. For defining those templates, the Visual Template Specification Language (ViTSL) is used by which the spreadsheet designer can specify, i.a., column and row types as well as how those structure elements (or whole blocks consisting of multiple structure elements) can be repeated horizontally and/or vertically (Abraham *et al.*, 2005). Cunha *et al.* (2012c) took the ClassSheet approach even further by embedding it in a prevalent spreadsheet system in an user-friendly way, i.e., end-users can interact with the spreadsheet model as well as its data in the same way. In this sense, the ClassSheet is seamlessly integrated into the familiar and proven spreadsheet environment, which makes it easy for the actual end-user to become familiar with the ClassSheet and the model-driven design process in general.

Around the ClassSheet approach, many related model-driven spreadsheet development approaches emerged over time. For example, Abraham and Erwig (2004) and Erwig and Burnett (2002) investigated the inference of types and units of spreadsheet cells based on spatial and content analysis. On another note, Burnett *et al.* (2003) propose the use assertions for defining constraints in spreadsheets in order to be able to better control the evolution of spreadsheets. Particularly the research group of Cunha et al. is very active in the area of model-driven spreadsheet development, e.g., by extending ClassSheets with the capability of having relations to other ClassSheets, deriving UML class diagrams from ClassSheets (Cunha *et al.*, 2012d), inferring ClassSheet models from spreadsheet instances (Cunha *et al.*, 2014a), proposing a bidirectional model-driven spreadsheet environment based on ClassSheets (Cunha *et al.*, 2012a; Cunha *et al.*, 2012b), and embedding model-driven spreadsheet queries into the ClassSheet environment (Cunha *et al.*, 2014b). Thereby, the ClassSheet approach and its ecosystem already cover a huge set of facilities to describe and use semantic models of spreadsheets. Nevertheless, ClassSheets do not distinguish semantic patterns within spreadsheets, e.g., list of entities, time series, matrices, etc. Although ClassSheets are basically capable of describing the semantic structure of spreadsheets, there is still no research about what common semantic spreadsheet patterns are, and how they can be described properly.

As a consequence, the present work targets this research gap and proposes a meta model which captures different semantic patterns and which is based on the works of Hermans *et al.* (2010), Erwig and Engels (2005), and Cunha *et al.* (2012c). Hence, the meta model tries to serve as an integrated framework for describing all semantic patterns which occur in real-world spreadsheets.

3 Research Method

The present work is organized along the design-science research method by Hevner *et al.* (2004). Figure 1 depicts the applied and adapted Information Systems Research Framework, showing the actual environment, knowledge base, design artifact, and evaluation strategy as adopted to the present work.

On the left-hand side of the Information Systems Research Framework, the environment imposes certain requirements and business needs to the design artifact. In the context of the present work, the actual research environment are the fields of spreadsheet research in particular and end-user development in general. Problems and issues of spreadsheets which are applied as business applications for supporting business processes were already motivated and described in Section 1 of the present work, and include aspects like usability, maintenance, and legal issues. The derived business need is therefore a facility to describe prevalent semantic spreadsheet patterns. This facility – the meta model as described in this paper – then forms the foundation for the design of a novel spreadsheet solution which is able to address those issues. On the right-hand side of the framework, the knowledge base defines the theoretical foundations and methodologies which are used for the design of the artifact. The actual knowledge base for this work was already discussed in Section 2.

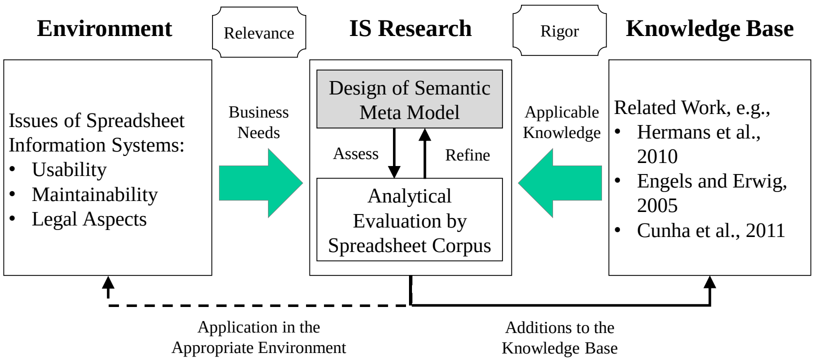


Figure 1. Overview over this work's research methodology based on the Information Systems Research Framework by Hevner et al. (2004)

The research contribution of this work is the design of a meta model for describing semantic structures of spreadsheets. In this sense, it captures not only structural aspects, but also the semantic type of the spreadsheet data and its components. The work by Hermans *et al.* (2010) about semantic spreadsheet patterns as well as the ClassSheet approach by Erwig and Engels (2005) served as a starting point. Based on this, the meta model underwent multiple iterations of evaluating the meta model and refining it according to the findings of the evaluation. The iteration was repeated as long as the evaluation revealed shortcomings of the meta model.

The evaluation as part of the design cycle was done by applying the designed meta model to the Enron spreadsheet corpus (Hermans and Murphy-Hill, 2014). This corpus contains over 15,000 spreadsheets from the Enron Corporation. In contrast to the EUSES spreadsheet corpus (Fisher and Rothermel, 2005) – which mainly contains spreadsheets crawled with search engines – the Enron corpus consists of concrete applications of spreadsheets in an enterprise. Furthermore, the study by Hermans and Murphy-Hill (2014) indicates that the spreadsheets of the Enron corpus are more complex (e.g., a relatively higher number of formulas) than those of the EUSES corpus. Consequently, it is plausible that a meta model which is able to describe the spreadsheets of the Enron corpus, can also be applied to the EUSES corpus.

4 A Semantic Meta Model of Spreadsheets

By applying the design-science research method as described in Section 3, we developed an integrated conceptual meta model which is capable of describing the semantic models of typical spreadsheets in industry. The basic structure is depicted in Figure 2 and defines that a spreadsheet contains potentially multiple worksheets, which in turn can contain multiple semantic components. A semantic component describes a block of cells which semantically belongs together. In the context of this work, we identified five different types of semantic components, namely entity lists, time series, matrices, singletons, and charts. Those types form the semantic patterns as outlined in Section 1, and are described in detail in the following subsections.

Figure 3 shows three enumeration types which are used throughout the semantic meta model within different semantic patterns. Thereby, the *DataType* enumeration type captures not only basic spreadsheet cell-types (e.g., number, date-time, string), but also types with richer semantics (e.g., currency). Furthermore, the enumeration type also includes *Reference* describing an association to another element of a semantic model (induced by foreign key concept as described by Cunha *et al.* (2012d). When specifying the attribute type of an entity list, attribute constraints can add further information to those relations, e.g., to which semantic component the attribute might refer. The *Resolution* enumeration type contains different time resolutions for describing temporal aspects within semantic patterns, while *ChartType* captures different forms of presentations of charts. Those enumeration types are understood to be blueprints: For specific cases they might have to be adapted by adding, changing or removing certain enumeration elements.

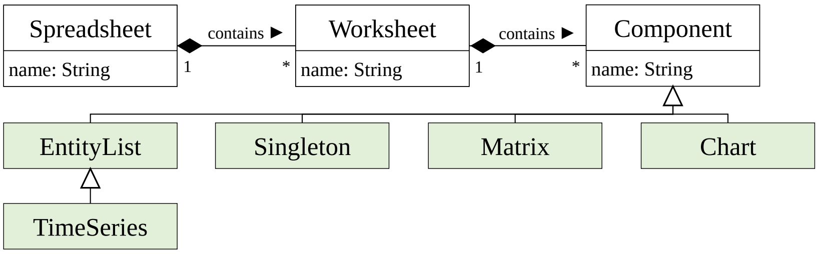


Figure 2. Excerpt of the semantic meta model showing its basic structure and capturing the semantic patterns (green) as identified in the present work.

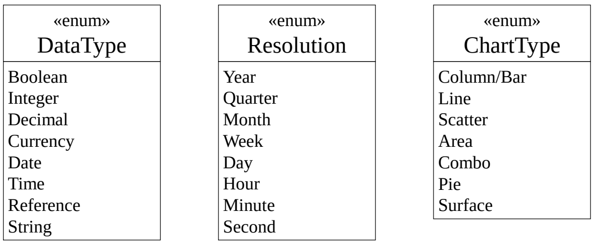


Figure 3. Enumeration types used throughout the semantic meta model.

4.1 Semantic Pattern: Entity List

The most discussed and best known semantic pattern in spreadsheet research is the entity list as described in Figure 4. The works of Hermans *et al.* (2010), Erwig and Engels (2005), and Cunha *et al.* (2012c) already focused on spreadsheets of this semantic pattern.

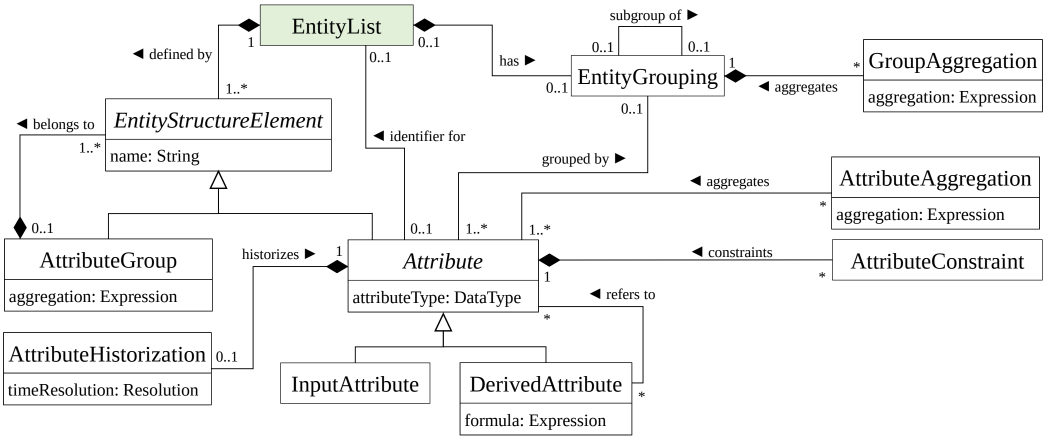
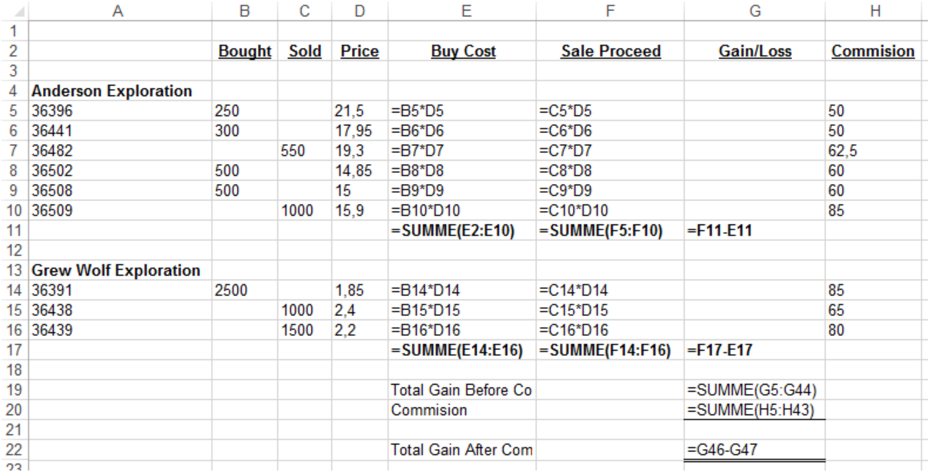


Figure 4. Excerpt of the semantic meta model describing the entity list pattern.

An entity list describes a semantic model consisting of a set of structured records, similar to tables in relational databases. The structure of entity lists is defined by *Attribute*s, which might be hierarchically organized in – potentially nested – *AttributeGroup*s. The *aggregationFormula* of an *AttributeGroup* describes how the values of the group's elements are aggregated to one single group value (e.g., when visually collapsing a group). In the meta model, the hierarchical organization of attributes is constituted by the composite design pattern (Gamma *et al.*, 1994). The concept of attribute groups was induced by the horizontally expandable cell groups of ClassSheets (Erwig and Engels, 2005). Attributes are either input attributes (cf., *Simple class* pattern by Hermans *et al.* (2010)) or derived attributes (cf., methods as defined by Hermans *et al.* (2010)). Input attributes describe fields which have to be entered by the spreadsheet users, while derived attributes are defined by formulas and compute their values automatically, primarily based on the values of attributes of the same entity, but potentially also based on a more complex query. For example, the attributes *Bought*, *Sold*, and *Price* in the spreadsheet in Figure 5 are input attributes, while *Buy Cost* and *Sale Proceed* are derived attributes (as indicated by the spreadsheet formulas). In most cases, one of the attributes defines an identifier for the entities (induced by the primary key concept as described by Cunha *et al.* (2012d)).

While *AttributeGroup*s address the horizontally expandable cell groups of ClassSheets (Erwig and Engels, 2005), *EntityGrouping*s analogously address the vertically expandable cell groups. Those groupings apply a *group-by* clause to the entities based on certain grouping key attributes. Similar to SQL, aggregations can be applied to groups, which is represented in the meta model by the *GroupAggregation* class. Therefore, a proper spreadsheet software could allow to visually collapse groups of entities and use the aggregations to compute representing values for the group. In the exemplary spreadsheet in Figure 5, the entities are grouped by an implicit category attribute (e.g., *Anderson Exploration*), whereas the grouping also defines aggregations (e.g., sum of *Buy Cost* per group). *EntityGrouping*s are potentially nested, wherefore the meta model in Figure 4 also defines a subgroup relation.



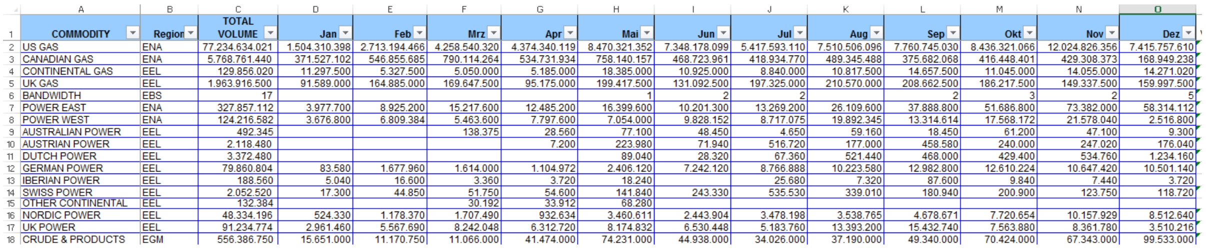
	A	B	C	D	E	F	G	H
1								
2		**Bought**	**Sold**	**Price**	**Buy Cost**	**Sale Proceed**	**Gain/Loss**	**Commision**
3								
4	**Anderson Exploration**							
5	36396	250		21,5	=B5*D5	=C5*D5		50
6	36441	300		17,95	=B6*D6	=C6*D6		50
7	36482		550	19,3	=B7*D7	=C7*D7		62,5
8	36502	500		14,85	=B8*D8	=C8*D8		60
9	36508	500		15	=B9*D9	=C9*D9		60
10	36509		1000	15,9	=B10*D10	=C10*D10		85
11					**=SUMME(E2:E10)**	**=SUMME(F5:F10)**	**=F11-E11**	
12								
13	**Grew Wolf Exploration**							
14	36391	2500		1,85	=B14*D14	=C14*D14		85
15	36438		1000	2,4	=B15*D15	=C15*D15		65
16	36439		1500	2,2	=B16*D16	=C16*D16		80
17					**=SUMME(E14:E16)**	**=SUMME(F14:F16)**	**=F17-E17**	
18								
19					Total Gain Before Co		=SUMME(G5:G44)	
20					Commision		=SUMME(H5:H43)	
21								
22					Total Gain After Com		=G46-G47	

Figure 5. *The spreadsheet "chris_dorland__1630__1999_equities" of the Enron spreadsheet corpus as an example for an entity list having InputAttributes, DerivedAttributes, EntityGroupings, GroupAggregations, and AttributeAggregations.*

*Attribute*s not only define the structure of entity lists, but they also have additional semantics. As defined by Hermans *et al.* (2010) and Erwig and Engels (2005), *AttributeAggregation*s are semantically associated with the *Attribute* and specify how the corresponding values of each entity are aggregated to one single value (e.g., by summing them up). For example, the spreadsheet in Figure 5 defines an aggregation for summing up the values of the *Gain/Loss* attribute.

Furthermore, constraints (induced by assertions as defined by Burnett *et al.* (2003)) can be defined for *Attribute*s restricting their actual value domain, e.g., by specifying a range for numerical attributes, a regular expression for string attributes, or a target entity list for reference attributes. Therefore, constraints allow the spreadsheet to perform not only type-based validation, but also advanced validation based on the restrictions as defined by the constraints of the corresponding attribute. In this sense, the class *AttributeConstraint* is an extension point of the meta model, which means that it can be extended by specific constraints classes depending on concrete cases of spreadsheet information systems.

Many spreadsheet applications explicitly capture the temporal evolution of certain attributes. For example, the spreadsheet in Figure 6 defines an implicit attribute *Volume* which has a value for each month of the year. This can be represented with the meta model by an *AttributeHistorization* with the respective *timeResolution*. Therefore, an *AttributeHistorization* describes a time series within each individual entity of a corresponding entity list.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	COMMODITY	Region	TOTAL VOLUME	Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez
2	US GAS	ENA	77.234.634.021	1.504.310.398	2.713.194.466	4.258.540.320	4.374.340.119	8.470.321.352	7.348.178.099	5.417.593.110	7.510.506.096	7.760.745.030	8.436.321.066	12.024.826.356	7.415.757.610
3	CANADIAN GAS	ENA	5.768.761.440	371.527.102	546.855.685	790.114.264	534.731.934	758.140.157	468.723.961	418.934.770	489.345.488	375.682.068	416.448.401	429.308.373	168.949.238
4	CONTINENTAL GAS	EEL	129.856.020	11.297.500	5.327.500	5.050.000	5.185.000	18.385.000	10.925.000	8.840.000	10.817.500	14.657.500	11.045.000	14.055.000	14.271.020
5	UK GAS	EEL	1.963.916.500	91.589.000	164.885.000	169.647.500	95.175.000	199.417.500	131.092.500	197.325.000	210.570.000	208.662.500	186.217.500	149.337.500	159.997.500
6	BANDWIDTH	EBS	17					1	2	2		2	3	2	5
7	POWER EAST	ENA	327.857.112	3.977.700	8.925.200	15.217.600	12.485.200	16.399.600	10.201.300	13.269.200	26.109.600	37.888.800	51.686.800	73.382.000	58.314.112
8	POWER WEST	ENA	124.216.582	3.676.800	6.809.384	5.463.600	7.797.600	7.054.000	9.828.152	8.717.075	19.892.345	13.314.614	17.568.172	21.578.040	2.516.800
9	AUSTRALIAN POWER	EEL	492.345			138.375	28.560	77.100	48.450	4.650	59.160	18.450	61.200	47.100	9.300
10	AUSTRIAN POWER	EEL	2.118.480				7.200	223.980	71.940	516.720	177.000	458.580	240.000	247.020	176.040
11	DUTCH POWER	EEL	3.372.480					89.040	28.320	67.360	521.440	468.000	429.400	534.760	1.234.160
12	GERMAN POWER	EEL	79.860.804	83.580	1.677.960	1.614.000	1.104.972	2.406.120	7.242.120	8.766.888	10.223.580	12.982.800	12.610.224	10.647.420	10.501.140
13	IBERIAN POWER	EEL	188.560	5.040	16.600	3.360	3.720	18.240		25.680	7.320	87.600	9.840	7.440	3.720
14	SWISS POWER	EEL	2.052.520	17.300	44.850	51.750	54.600	141.840	243.330	535.530	339.010	180.940	200.900	123.750	118.720
15	OTHER CONTINENTAL	EEL	132.384			30.192	33.912	68.280							
16	NORDIC POWER	EEL	48.334.196	524.330	1.178.370	1.707.490	932.634	3.460.611	2.443.904	3.478.198	3.538.765	4.678.671	7.720.654	10.157.929	8.512.640
17	UK POWER	EEL	91.234.774	2.961.460	5.567.690	8.242.048	6.312.720	8.174.832	6.530.448	5.183.760	13.393.200	15.432.740	7.563.880	8.361.780	3.510.216
18	CRUDE & PRODUCTS	EGM	556.386.750	15.651.000	11.170.750	11.066.000	41.474.000	74.231.000	44.938.000	34.026.000	37.190.000	49.340.000	70.424.000	67.343.000	99.533.000

Figure 6. The spreadsheet "andy_zipper__111__Brokerage fees based on Volumes yr 2000_4-16-011" of the Enron spreadsheet corpus as an example for an entity list having an AttributeHistorization.

4.2 Semantic Pattern: Time series

Based on the entity list pattern, we identified time series as an additional semantic pattern which commonly occurs in spreadsheets. A time series is a sequence of potentially multiple data points, typically consisting of successive measurements or observations on one or more variables (Brockwell and Davis, 2009). Usually the observations are chronological and taken at regular intervals (cf. *Resolution* enumeration type in Figure 3). In practice, time series typically capture the temporal evolution of numerical variables, e.g. sales, inventory, customer counts, interest rates, costs, etc.

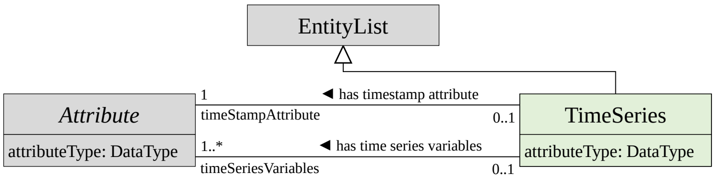


Figure 7. Excerpt of the semantic meta model describing the time series pattern, which is based on the entity list pattern. The grey elements are already defined by the entity list pattern in Figure 4, wherefore the TimeSeries class and its associations are the only elements added to the meta model.

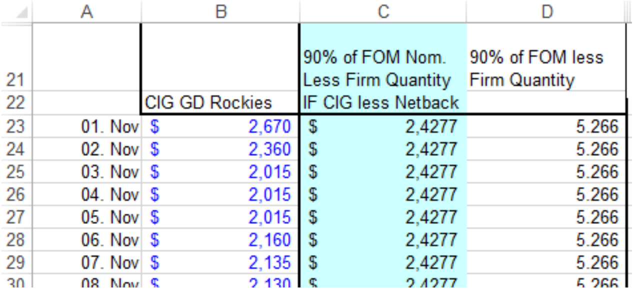
The time series pattern extends the entity lists as shown in Figure 7. The *TimeSeries* class derives from the *EntityList*, and extends it by two new associations. The first one is an explicit relation to the attribute which captures the actual time-stamp of the time series elements, while the second association defines the time series variables. Since the *timeStampAttribute* has to represent a date or time, we add the following OCL (Object Management Group, 2014b) constraint to the semantic meta model:

```
context TimeSeries
inv:
let at : DataType =  self.timeStampAttribute.attributeType in
  at = DataType::Date or
  at = DataType::Time or
  at = DataType::Integer
```

This OCL expression constraints the *timeStampAttribute* to be either of type *Date*, *Time*, or *Integer* (e.g., in case of a yearly time resolution), and thus to represent a valid time-stamp. Furthermore, the fact that time series variables have to be numerical (Brockwell and Davis, 2009) can be expressed by the following OCL constraint:

```
context TimeSeries
inv:
self.timeSeriesVariables->forAll(v |
  v.attributeType = DataType::Integer or
  v.attributeType = DataType::Decimal or
  v.attributeType = DataType::Currency)
```

Thereby, this OCL constraint ensures that all attributes of the time series which are defined as the time series' variables are either of type *Integer*, *Decimal*, or *Currency*. Figure 8 shows an example of a time series with the first column as its time-stamp (with time resolution *Day*), two time series variables of type *Currency*, and one time series variable of type *Integer*.



	A	B	C	D
21			90% of FOM Nom. Less Firm Quantity IF CIG less Netback	90% of FOM less Firm Quantity
22		CIG GD Rockies		
23	01. Nov	$ 2,670	$ 2,4277	5.266
24	02. Nov	$ 2,360	$ 2,4277	5.266
25	03. Nov	$ 2,015	$ 2,4277	5.266
26	04. Nov	$ 2,015	$ 2,4277	5.266
27	05. Nov	$ 2,015	$ 2,4277	5.266
28	06. Nov	$ 2,160	$ 2,4277	5.266
29	07. Nov	$ 2,135	$ 2,4277	5.266
30	08. Nov	$ 2,130	$ 2,4277	5.266

Figure 8. *The spreadsheet "paul_lucci__28348__north central email" of the Enron spreadsheet corpus as an example for time series.*

4.3 Semantic Pattern: Singleton

Contrary to entity lists and time series, the singleton pattern describes semantic components which only capture a single structured data item instead of a potentially expandable list of data records. In this context, structured data item means that a singleton can consist of multiple and – as described by Knight *et al.* (2000) – nested (singleton-) attributes of diverse types. The structure of a singleton can be described by the meta model as shown in Figure 9. Thereby, the composite pattern (Gamma *et al.*, 1994) is implemented by the *SingletonStructureElement* (*Component*), the *SingletonAttributeGroup* (*Composite*), and the *SingletonAttribute* (*Leaf*), enabling the definition of deeply nested and thus hierarchical attributes. Analogous to the *AttributeGroup* of the entity list, the *aggregation* attribute of the *SingletonAttributeList* allows to aggregate the values of all sub-attributes in order to compute a value representing the group. This is in particular helpful if the spreadsheet system allows to collapse groups and thus to hide the sub structure of an attribute group.

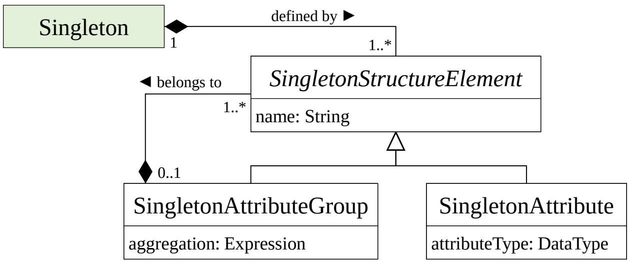
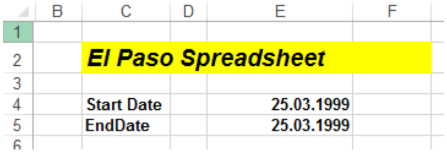


Figure 9. *Excerpt of the semantic meta model describing the singleton pattern.*

Figure 10 shows an exemplary singleton with two attributes, both of type date. Syntactically, most singletons could also be interpreted as entity lists when considering the singleton attributes (e.g., *Start*

Date and *Enddate*) and the respective values as entities. However, singletons have a different semantics: While in entity lists, the users are supposed to add new entities (which does not change the design of the entity list), the set of rows of singletons are fixed and captured by the singleton's structure, which means that vertically expanding a singleton changes its design. This semantic detail is the difference between entity lists and singletons.



	B	C	D	E	F
1					
2		El Paso Spreadsheet			
3					
4		Start Date		25.03.1999	
5		EndDate		25.03.1999	

Figure 10. The spreadsheet "albert_meyers__1__1-25act" of the Enron spreadsheet corpus as an example for the singleton pattern.

4.4 Semantic Pattern: Matrix

The matrix pattern describes semantic spreadsheet components consisting of not only one dimension of structure elements (e.g., attributes and attribute groups in the entity list), but of two. In this context, those dimensions are represented by the two axes of a matrix, which in turn consist of multiple *MatrixAxisElements*. The respective excerpt of the semantic meta model describing the matrix pattern is depicted in Figure 11, and defines the type of the cells of the matrix as an attribute of the *Matrix* class. Therefore, all values of the matrix (i.e., the values of its cells) have to be of the same type.

Although the semantic meta model could be extended to enable more complex matrix axes, e.g., by applying the composite pattern (Gamma *et al.*, 1994) to allow a hierarchical axis structure, there is no evidence in related literature or in spreadsheets of the Enron spreadsheet corpus (Hermans and Murphy-Hill, 2014) to do so.

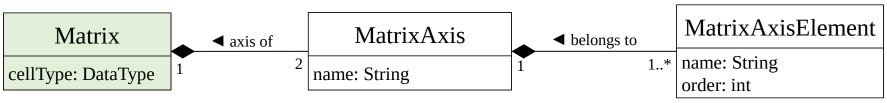
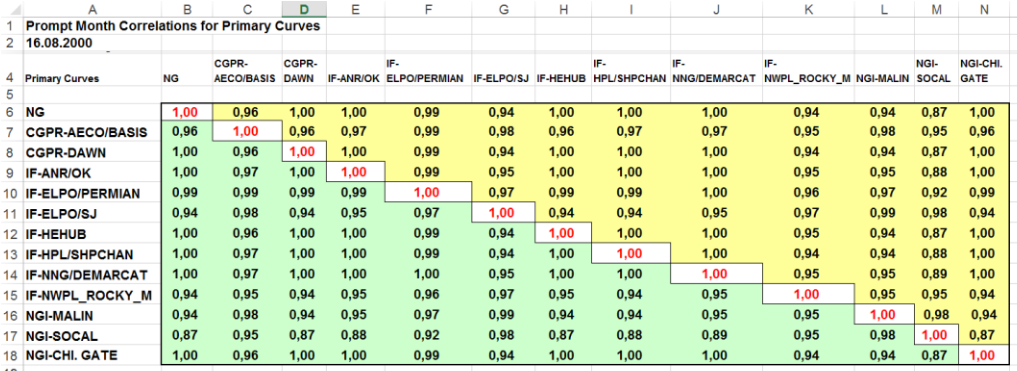


Figure 11. Excerpt of the semantic meta model describing the matrix pattern.

An exemplary application of the matrix pattern are correlation matrices, as also shown in Figure 12. In general, the values of the cells in a correlation matrix determine the correlation between the respective axis elements in the x-axis and y-axis. Therefore, those correlation matrices can be described by the matrix pattern with a cell type *Decimal*.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Prompt Month Correlations for Primary Curves													
2	16.08.2000													
4	Primary Curves	NG	CGPR-AECO/BASIS	CGPR-DAWN	IF-ANR/OK	IF-ELPO/PERMIAN	IF-ELPO/SJ	IF-HEHUB	IF-HPL/SHPCHAN	IF-NNG/DEMARCAT	IF-NWPL_ROCKY_M	NGI-MALIN	NGI-SOCAL	NGI-CHI. GATE
5														
6	NG	1,00	0,96	1,00	1,00	0,99	0,94	1,00	1,00	1,00	0,94	0,94	0,87	1,00
7	CGPR-AECO/BASIS	0,96	1,00	0,96	0,97	0,99	0,98	0,96	0,97	0,97	0,95	0,98	0,95	0,96
8	CGPR-DAWN	1,00	0,96	1,00	1,00	0,99	0,94	1,00	1,00	1,00	0,94	0,94	0,87	1,00
9	IF-ANR/OK	1,00	0,97	1,00	1,00	0,99	0,95	1,00	1,00	1,00	0,95	0,95	0,88	1,00
10	IF-ELPO/PERMIAN	0,99	0,99	0,99	0,99	1,00	0,97	0,99	0,99	1,00	0,96	0,97	0,92	0,99
11	IF-ELPO/SJ	0,94	0,98	0,94	0,95	0,97	1,00	0,94	0,94	0,95	0,97	0,99	0,98	0,94
12	IF-HEHUB	1,00	0,96	1,00	1,00	0,99	0,94	1,00	1,00	1,00	0,95	0,94	0,87	1,00
13	IF-HPL/SHPCHAN	1,00	0,97	1,00	1,00	0,99	0,94	1,00	1,00	1,00	0,94	0,94	0,88	1,00
14	IF-NNG/DEMARCAT	1,00	0,97	1,00	1,00	1,00	0,95	1,00	1,00	1,00	0,95	0,95	0,89	1,00
15	IF-NWPL_ROCKY_M	0,94	0,95	0,94	0,95	0,96	0,97	0,95	0,94	0,95	1,00	0,95	0,95	0,94
16	NGI-MALIN	0,94	0,98	0,94	0,95	0,97	0,99	0,94	0,94	0,95	0,95	1,00	0,98	0,94
17	NGI-SOCAL	0,87	0,95	0,87	0,88	0,92	0,98	0,87	0,88	0,89	0,95	0,98	1,00	0,87
18	NGI-CHI. GATE	1,00	0,96	1,00	1,00	0,99	0,94	1,00	1,00	1,00	0,94	0,94	0,87	1,00

Figure 12. The spreadsheet "fletcher__sturm__10910_ng gas correlation matrix" of the Enron spreadsheet corpus as an example for the matrix pattern.

4.5 Semantic Pattern: Charts

In addition to the user-friendly management and analysis of domain-specific data, spreadsheet software typically also provides facilities to define and configure different types of visualizations, e.g., a line chart depicting the temporal evolution of a time series. Although the present work does not discuss semantics of those visualizations, the class *Chart* is still considered to be a semantic component as shown in Figure 2. However, as defined in Figure 13, the semantic meta model only captures the type of the chart as an attribute of the enumeration type *ChartType* (as depicted in Figure 3) as well as an association to other semantic components which serve as data sources for the generation of the chart. Again, certain spreadsheet tools might provide additional chart types, wherefore the *ChartType* enumeration type of the semantic meta model can be extended by additional elements.

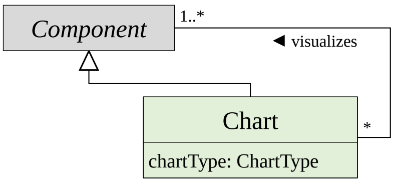


Figure 13. Excerpt of the semantic meta model describing the chart pattern.

5 Evaluation

Based on the applied research method as described in Section 3, we iteratively evaluated the appropriateness and expressiveness of the meta model by applying it to spreadsheets of the Enron spreadsheet corpus (Hermans and Murphy-Hill, 2014). However, due to the large expenditure of the manual analysis of the spreadsheet semantics, we did not analyze all 15,929 spreadsheets of the Enron corpus, but continued the iterative design approach until we reached a stable version of the meta model. Therefore, we observed 200 randomly selected spreadsheets containing potentially multiple worksheets (average: 2.7 worksheets per spreadsheet), which in turn consist of multiple semantic components (average: 2.0 semantic components per worksheet, 5.2 semantic components per spreadsheet).

Component	Number of components	Number of spreadsheets with component
Entity list	546 (53 %)	154 (77 %)
Time series	233 (22 %)	74 (37 %)
Singleton	107 (10 %)	31 (16 %)
Matrix	2 (~ 0 %)	1 (~ 0 %)
Chart	152 (15 %)	31 (16 %)

Table 1. General statistics about observed spreadsheets and their semantic components.

Table 1 shows an overview of which semantic patterns were identified how often in the observed spreadsheets of the Enron corpus, i.e., the number of identified semantic components for each of the patterns as well as the number of spreadsheets in which each pattern occurs at least once. It is noteworthy that the number of spreadsheets with entity lists is significantly higher than the numbers which were identified in the evaluation of the class diagram extraction by Hermans *et al.* (2010). One possible reason for this mismatch is that we also captured more complex structured entity lists which are not processible by the tool of Hermans et al. Another cause could be that they used the EUSES spreadsheet corpus (Fisher and Rothermel, 2005), while the present work's evaluation is based on the Enron corpus. Time series are the second most common semantic pattern in spreadsheets, followed by singletons and charts. The matrix pattern was identified only once in the observed 200 spreadsheets.

Table 2 presents a list of selected statistical numbers for the specific semantic spreadsheet patterns. The evaluation has shown that most of the observed entity lists (about 80 %) have a rather simple structure, i.e., they contain neither attribute historizations, attribute groups, nor entity groupings. Another interesting fact is that the average number of variables in time series is 13.2, which is significantly higher than the average number of attributes in entity lists (7.47). Furthermore, as the histogram in Figure 14 shows, the most used time resolutions in time series are *Month* and *Day*, while finer resolutions like *Minute* and *Second* are not common. For singletons, the average number of attributes is 5.4, while for matrices we cannot make any reliable claim because of the small sample size. For the charts, Figure 14 shows a histogram with the number of charts for each chart type. The most common chart types in spreadsheets are *Bar/Column* and *Line* charts. Less than 15 % of spreadsheet charts are neither *Bar/Column* nor *Line* charts.

Entity list	
Average number of attributes per entity list	7.5
Average number of input attributes per entity list	5.7
Average number of derived attributes per entity list	1.7
Number of entity lists with attribute historizations	38 (7 %)
Number of entity lists with attribute groups	68 (12 %)
Number of entity lists with entity groupings	13 (2 %)
Time series	
Average number of variables	13.2
Singleton	
Average number of singleton attributes	5.4
Matrix	
Average axis size	2

Table 2. A selected set of statistical measurements about the identified semantic spreadsheet patterns.

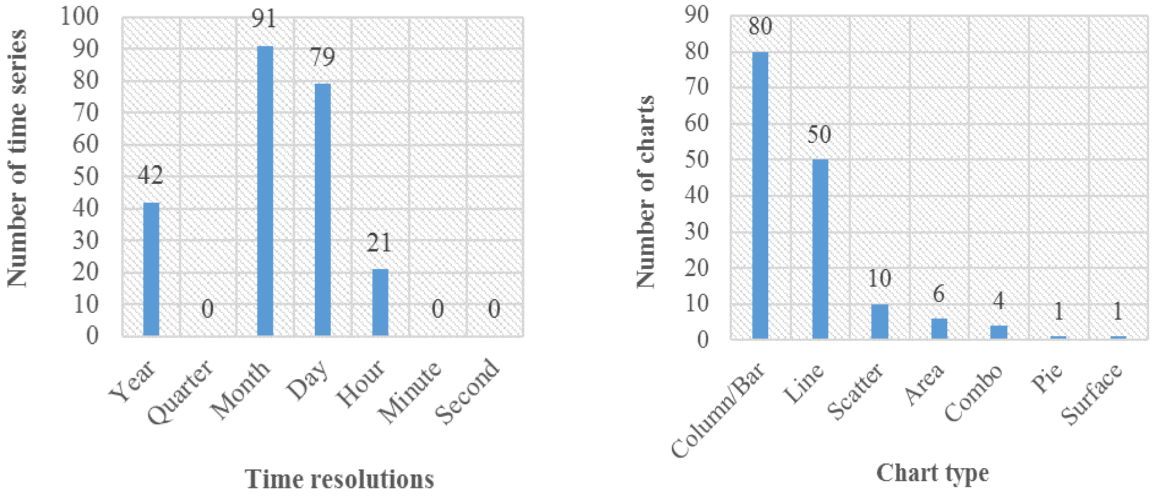


Figure 14. Two histograms showing the number of time series for different time resolutions and the number of charts for different chart types respectively.

6 Conclusion

In this paper, we have presented a meta model for describing the semantics of spreadsheets. Thereby, we describe five different semantic spreadsheet patterns which commonly occur in practice, namely entity lists, time series, singletons, matrices, and charts. Being able to explicitly describe the semantic structure of spreadsheets independently from its data directly addresses the lack of separation between structure and data as outlined by Panko and Ordway (2005). By providing an explicit definition of the semantic data model, the spreadsheet is able to automatically identify errors and inconsistencies of the data regarding the defined model at design-time. This reduces the risk of spreadsheet errors significantly (Erwig *et al.*, 2005). Although existing approaches propose similar methods for defining the structure of spreadsheets, the paper at hand is novel in the sense that it provides a holistic meta model for spreadsheets which not only captures structural but also semantic aspects based on an empirical investigation of a spreadsheet corpus (Hermans and Murphy-Hill, 2014).

A critical reflection of the present work reveals potential threats to the validity of this paper's contribution. First of all, the evaluation as part of the design-science research method was performed by applying the semantic meta model to the spreadsheets of the Enron corpus. In this sense, the semantics of each spreadsheet was solely derived from its structure, while the original intention of the spreadsheet's designer obviously could not be taken into account. Therefore, the derived semantic model of an observed spreadsheet might differ from the corresponding spreadsheet designer's mental model. However, we believe that in a majority of cases we were able to capture the actual semantics of the spreadsheets, and that the basic structure of the semantic meta model is actually valid.

Furthermore, as already described in Section 5, we applied the semantic meta model not to all 15,929 spreadsheets of the Enron corpus, but only to 200 randomly selected spreadsheets. Therefore, the semantic meta model might still have flaws in the sense that there might be spreadsheets in the corpus which cannot be described by the presented meta model. However, while the evaluation of the meta model as part of the design-science research method with the first 100 spreadsheets frequently revealed a need for refinement of the meta model, this was not the case for the other 100 spreadsheets, i.e., we already reached a stable version of the semantic meta model after manually processing 100 spreadsheets. Therefore, we believe that the meta model (or at least its basic structure) would not change even by performing an evaluation with the complete Enron spreadsheet corpus.

Apart from this, we do not claim to have a complete meta model anyway, and that for specific cases it has to be extended by additional concepts. For this purpose, the semantic meta model provides certain extension points. For example, specific *AttributeConstraints* can define restrictions for certain attribute types, e.g., for string attributes (by specifying a proper regular expression), numerical attributes (by specifying a range), and reference attributes (e.g., by specifying to which elements the attribute can refer to). Other examples for specializations of the meta model are the adaption of the enumeration types in Figure 3, the integration of the composite pattern into the matrix pattern, or the extension of the matrix pattern by a more specific correlation matrix pattern. However, we found no evidence in related literature or the observed spreadsheets of the Enron corpus to add those extensions and concepts in the general semantic meta model as proposed by this work. Therefore, although the semantic meta model is not complete in the sense that it can describe the full semantics of each spreadsheet, it already provides a basic structure which is not only evaluated by its application on real-world spreadsheets, but also derived from related literature.

Based on the semantic meta model, future research activities can focus on model-driven approaches to spreadsheets by incorporating the concepts of the meta model on the one hand, and addressing the identified semantic spreadsheet patterns on the other hand. Based on shared understanding of semantic spreadsheet patterns, researchers and practitioners are able to focus on pattern-specific characteristics, and to work on novel approaches for specific semantic patterns. For example, novel model-driven approaches to spreadsheets can address specific issues of entity lists while neglecting issues of other semantic patterns.

References

Abraham, R. and Erwig, M. (2004), “Header and Unit Inference for Spreadsheets Through Spatial Analyses”, *Proceedings of the Symposium on Visual Languages and Human-Centric Computing*.

Abraham, R., Erwig, M., Kollmansberger, S. and Ethan, S. (2005), “Visual Specifications of Correct Spreadsheets”, *Proceedings of the Symposium on Visual Languages and Human-Centric Computing*.

Basel Committee on Banking Supervision (2004), *Basel II*.

Bradley, L. and McDaid, K. (2009), “Using Bayesian Statistical Methods to Determine the Level of Error in Large Spreadsheets”, *Proceedings of the International Conference on Software Engineering*, pp. 351–354.

Bretz, J. (2009), “OpRisk individuelle Datenverarbeitung. Management der operationellen Risiken aus dem Einsatz individueller Datenverarbeitung (IDV)”, *Bank-Praktiker rechtssicher, revisionsfest, risikogerecht*, Vol. 6, pp. 294–298.

Bretz, J. (2012), *Prüfung IT im Fokus von MaRisk und Bundesbank: Verstärkter IT-Fokus in Sonderprüfungen*, Finanz Colloquium Heidelberg, Heidelberg.

Brockwell, P.J. and Davis, R.A. (2009), *Time Series: Theory and Methods*, Springer Science & Business Media.

Brown, P.S. and Gould, J.D. (1987), “An Experimental Study of People Creating Spreadsheets”, *ACM Transactions on Information Systems*, Vol. 5 No. 3, pp. 258–272.

Burnett, M., Cook, C., Pendse, O., Rothermel, G., Summer, J. and Wallace, C. (2003), “End-User Software Engineering with Assertions in the Spreadsheet Paradigm”, *Proceedings of the International Conference on Software Engineering*.

Caulkins, J.P., Morrison, E.L. and Weidemann, T. (2007), “Spreadsheet Errors and Decision Making: Evidence from Field Interviews”, *Journal of Organizational and End User Computing*, Vol. 19 No. 3, pp. 1–23.

Cunha, J., Erwig, M., Mendes, J. and Saraiva, J. (2014a), “Model Inference for Spreadsheets”, *Automated Software Engineering*, pp. 1–32.

Cunha, J., Fernandes, J.P., Mendes, J., Pacheco, H. and Saraiva, J. (2012a), “Bidirectional Transformation of Model-Driven Spreadsheets”, *Theory and Practice of Model Transformations*.

Cunha, J., Fernandes, J.P., Mendes, J., Pereira, R. and Saraiva, J. (2014b), “Embedding Model-Driven Spreadsheet Queries in Spreadsheet Systems”, *Proceedings of the Symposium on Visual Languages and Human-Centric Computing*.

Cunha, J., Fernandes, J.P., Mendes, J. and Saraiva, J. (2012b), “A Bidirectional Model-Driven Spreadsheet Environment”, *Proceedings of the International Conference on Software Engineering*.

Cunha, J., Fernandes, J.P., Mendes, J. and Saraiva, J. (2012c), “MDSheet: A Framework for Model-driven Spreadsheet Engineering”, *Proceedings of the International Conference on Software Engineering*, pp. 1395–1398.

Cunha, J., Fernandes, J.P. and Saraiva, J. (2012d), “From Relational ClassSheets to UML+ OCL”, *Proceedings of the Symposium on Applied Computing*.

Erwig, M., Abraham, R., Cooperstein, I. and Kollmansberger, S. (2005), “Automatic Generation and Maintenance of Correct Spreadsheets”, *Proceedings of the International Conference on Software Engineering*, pp. 136–145.

Erwig, M. and Burnett, M. (2002), “Adding Apples and Oranges”, *Proceedings of the International Symposium on Practical Aspects of Declarative Languages*, pp. 173–191.

Erwig, M. and Engels, G. (2005), “ClassSheets: Automatic Generation of Spreadsheet Applications from Object-Oriented Specifications”, *Proceedings of the International Conference on Automated Software Engineering*, pp. 124–133.

Fisher, M. and Rothermel, G. (2005), “The EUSES Spreadsheet Corpus: A Shared Resource for Supporting Experimentation with Spreadsheet Dependability mechanisms”, *Proceedings of the Workshop on End-User Software Engineering*, pp. 47–51.

Gable, G.G., Yap, C.M. and Eng, M.N. (1991), “Spreadsheet Investment, Criticality, and Control”, *Proceedings of the Hawaii International Conference on System Sciences*, pp. 153–162.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1994), *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Grossman, T.A., Mehrotra, V. and Özlük, Ö. (2005), *Spreadsheet Information Systems are Essential to Business*, working paper.

Hermans, F. and Murphy-Hill, E. (2014), *Enron’s Spreadsheets and Related Emails: A Dataset and Analysis*.

Hermans, F., Pinzger, M. and van Deursen, A. (2010), “Automatically Extracting Class Diagrams from Spreadsheets”, *Proceedings of the European Conference on Object-Oriented Programming*, pp. 52–75.

Hermans, F., Pinzger, M. and van Deursen, A. (2012), “Detecting Code Smells in Spreadsheet Formulas”, *Proceedings of the International Conference on Software Maintenance*.

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004), “Design Science in Information Systems Research”, *Management Information Systems Quarterly*, Vol. 28 No. 1, pp. 75–105.

Knight, B., Chadwick, D. and Rajalingham, K. (2000), “A Structured Methodology for Spreadsheet Modelling”, *Proceedings of the European Spreadsheet Risks Information Group*.

Mittermeir, R. and Clermont, M. (2002), “Finding High-Level Structures in Spreadsheet Programs”, *Proceedings of the Working Conference on Reverse Engineering*, pp. 221–232.

Object Management Group (2012), *OMG Object Constraint Language (OCL)*, 2.3.1st ed., available at: http://www.omg.org/spec/UML/2.4.1.

Object Management Group (2014), *Meta Object Facility (MOF) v2.4.2*, 2.4.2nd ed., available at: http://www.omg.org/spec/MOF/2.4.2/.

Panko, R.R. (2000), “Spreadsheet Errors: What We Know. What We Think We Can Do”, *Proceedings of the European Spreadsheet Risks Information Group*, pp. 7–17.

Panko, R.R. (2006), “Facing the Problem of Spreadsheet Errors”, *Decision Line*, Vol. 37 No. 5, pp. 8–10.

Panko, R.R. and Halverson Jr., R. (1996), “Spreadsheets on Trial: A Survey of Research on Spreadsheet Risks”, *Proceedings of the Hawaii International Conference on System Sciences*, pp. 326–335.

Panko, R.R. and Ordway, N. (2005), “Sarbanes-Oxley: What About all the Spreadsheets?”, *Proceedings of the European Spreadsheet Risks Information Group*.

Pemberton, J.D. and Robson, A.J. (2000), “Spreadsheets in Business”, *Industrial Management & Data Systems*, Vol. 100 No. 8, pp. 379–388.

Powell, S.G., Baker, K.R. and Lawson, B. (2008), “A Critical Review of the Literature on Spreadsheet Errors”, *Decision Support Systems,* Vol. 46 No. 1, pp. 128–138.

Powell, S.G., Baker, K.R. and Lawson, B. (2009), “Impact of Errors in Operational Spreadsheets”, *Decision Support Systems,* Vol. 47 No. 2, pp. 126–132.

Reschenhofer, T. and Matthes, F. (2015a), “A Framework for the Identification of Spreadsheet Usage Patterns”, *Proceedings of the European Conference on Information Systems*.

Reschenhofer, T. and Matthes, F. (2015b), “An Empirical Study on Spreadsheet Shortcomings from an Information Systems Perspective”, *Proceedings of the International Conference on Business Information Systems*.

United States Congress (2002), *Sarbanes-Oxley Act*.