Analysing the cognitive effectiveness of the WebML visual notation

David Granada · Juan Manuel Vara · Marco Brambilla · Verónica Bollati · Esperanza Marcos

Abstract WebML is a domain-specific language used to design complex data-intensive Web applications at a conceptual level. As WebML was devised to support design tasks, the need to define a visual notation for the language was identified from the very beginning. Each WebML element is consequently associated with a separate graphical symbol which was mainly defined with the idea of providing simple and expressive modelling artefacts rather than by adopting a rigorous scientific approach. As a result, the graphical models defined with WebML may sometimes prevent proper communication from taking place between the various stakeholders. In fact, this is a common issue for most of the existing model-based proposals that have emerged during the last few years under the umbrella of model-driven engineering. In order to illustrate this issue and foster in using a scientific basis to design, evaluate, improve and compare visual notations, this paper analyses WebML according to a set of solid principles, based on the theoretical and empirical evidence concerning the cognitive effectiveness of visual notations. As a result, we have identified a set of possible improvements, some of which have been verified by an empirical study. Furthermore, a number of findings, experiences and lessons learnt on the assessment of visual notations are presented.

Keywords Web modelling language (WebML) · Visual notation · Cognitive effectiveness · Visual communication · Visual syntax · Concrete syntax

D. Granada · J. M. Vara (✉) · V. Bollati · E. Marcos
Department of Computing Languages and Systems II,
University Rey Juan Carlos, Madrid, Spain
e-mail: juanmanuel.vara@urjc.es

D. Granada
e-mail: david.granada@urjc.es

V. Bollati
e-mail: veronica.bollati@urjc.es

E. Marcos
e-mail: esperanza.marcos@urjc.es

M. Brambilla
Dipartimento di Elettronica e Informazione,
Politecnico di Milano, Milan, Italy
e-mail: marco.brambilla@polimi.it

1 Introduction

Web modelling language (WebML), which was initially defined as a conceptual model with which to design data-intensive Web applications [12], has now evolved into a domain-specific language (DSL) [37] that can be used to design complex, distributed, multi-actor, and adaptive applications deployed on the Web and in service-oriented architectures (SOA) [2].

Like any other DSL, WebML is defined through the core ingredients shown in Fig. 1: a DSL is mainly based upon a **metamodel**, which collects the **abstract syntax** of the language. It specifies the vocabulary of concepts or language elements provided by the language and how they may be combined to create models. The meaning of those concepts and their connections is referred to as the **semantics** of the language [15]. Finally, **concrete syntax** provides a notation that facilitates the presentation and construction of models or programs in the language. There are two main types of con-crete syntax typically used by languages: textual syntax and visual syntax. The latter, which is commonly referred to as visual notation in the literature related with cognitive effec-tiveness [44], consists of a set of graphical symbols used to represent the concepts collected in the metamodel and a set of compositional rules. In short, this paper will concentrate on

This is a post-peer-review, pre-copyedit version of an article published in SOFTWARE AND SYSTEMS MODELING, 2017, 16:1, 195-227. The final authenticated version is available online at: http://dx.doi.org/10.1007%2Fs10270-014-0447-8

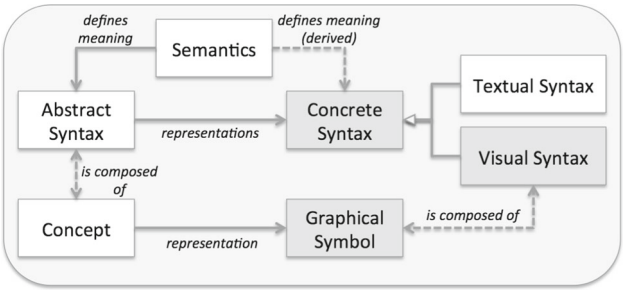


Fig. 1 Syntaxes and semantics of a DSL (adapted from [10])

the visual notation and graphical symbols used in WebML, and the right-hand side of Fig. 1 illustrates the scope of this paper.

Most of the WebML graphical symbols were defined by taking into consideration that simple and expressive concepts should be provided in order to ease the task of designing Web applications with the language, the final decisions being merely based on intuition and best practices [12]. Although these symbols have evolved over time, the criteria used to define them, such as simplicity and intuitiveness, have remained. Nevertheless, these criteria are highly subjective and may consequently result in the definition of models that prevent proper communication between stakeholders [39].

In order to address this kind of scenario, cognitive effectiveness should be taken into account when designing a visual notation, i.e. the speed, ease and accuracy with which a representation can be processed by the human mind [62]. Note that this concept is not intrinsic in any visual representation, and it must therefore be explicitly considered when designing, evaluating and comparing visual notations. The design of visual notations and the choice of graphical conventions should therefore be based on theoretical principles and empirical evidence of cognitive effectiveness rather than on best practices, common sense or social opinion [57].

Bearing all this in mind, this paper analyses the WebML visual notation used in WebRatio [1], the integrated development environment (IDE) based on the Eclipse framework which supports the model-driven specification of WebML models. The subject of the analysis is the implementation of the language by the tool and not the language itself since, as has traditionally occurred in modelling languages in software engineering (SE), the specifications tend to evolve independently of their corresponding implementations, and the language is rarely updated to reflect this evolution. This is the classic scenario encountered when analysis and design models do not reflect the current state of the system, whose working-code has evolved after the specification stages [36].

The underlying objective of this paper is to foster the interest in using a scientific basis to design, evaluate, improve and compare visual notations. This is particularly relevant in the context of model-driven engineering (MDE), since graphical DSLs play a cornerstone role in almost all existing MDE pro-posals [52]. Moreover, visual syntax has a large influence on the effectiveness of modelling processes, that is equal to (if not greater than) decisions concerning semantics [30]. How-ever, historically there has been very little interest in this issue, and very few studies are focused on ensuring that a particular visual syntax is good or on providing approaches for the assessment of cognitive effectiveness.

Note also that our intention is not simply to focus exclusively on the visual deficiencies of the language, but rather to make a constructive analysis using a scientific basis in order to improve the communication effectiveness of WebRatio's visual notation. Due to the low priority that has been given historically to the analysis of cognitive effectiveness of visual syntax, we wanted to perform this work on probably one of the most successful tools in the context of MDE [11], judged by the number of partners, adopters and success stories of the framework (see WebRatio's Web site: http://www.webratio. com/).

The remainder of this paper is structured as follows: Sect. 2 provides an overview of WebML and the WebRatio environment. Section 3 reviews existing proposals for the assessment of visual notations with special attention paid to the Physics of Notations. Section 4 presents the analysis of WebML notation under the principles of the Physics of Notations. Sec-tion 5 validates some of the conclusions gathered from the analysis by means of an empirical study. Section 6 discusses the main lessons learnt from this study in order to foster an interest in using a scientific basis to design, evaluate, improve and compare visual notations. Finally, Sect. 7 concludes by summarizing the main findings and providing directions for future work.

2 Web applications development with WebML and WebRatio

The WebML methodology for developing applications consists of an incremental and iterative process, in which the various stages are repeated and refined until results meet the requirements of the application. Note that an iterative and incremental life cycle fits perfectly with the nature of Web applications development (short time-to-market and evolving requirements). The stages of this method are shown in Fig. 2.

The scope of this work is limited to the analysis of the conceptual modelling stage, which consists of the definition of a set of conceptual schemas that reflect the structure of the application at a high abstraction level, i.e. without taking into consideration the technical details. More concretely, the conceptual modelling stage comprises the Data design and the Hypertext design. The former consists of the orga-

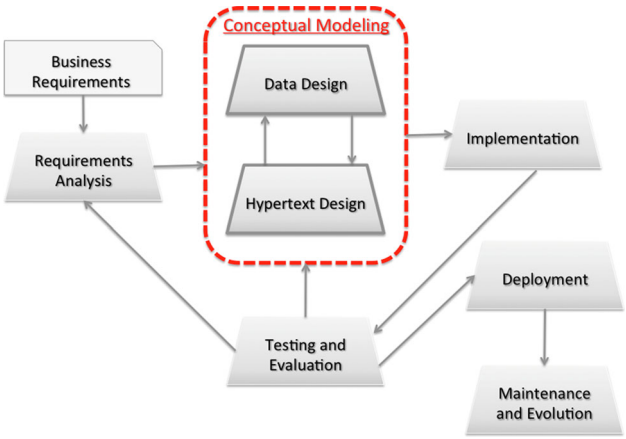


Fig. 2 Stages in the WebML development process (adapted from [2])

nization of the core information objects into a data schema that may be enriched through derived objects. The latter produces site view schemas on top of the data schema previously defined.

The modelling activities comprised in the conceptual modelling stages are the most relevant, as they drive the rest of the development process, dictating what the final result will be. This work focuses on the artefacts handled during the conceptual modelling stage. As a matter of fact, the WebML language that is introduced in the next section was particu-larly intended to support the modelling activities performed during this stage.

2.1 WebML overview

WebML is a visual language for specifying the content structure of a Web application and the organization and presentation of such content in a hypertext [12]. To that end, WebML was designed by reusing conceptual data models and proposing an original notation to express the navigation and composition features of hypertext interfaces.

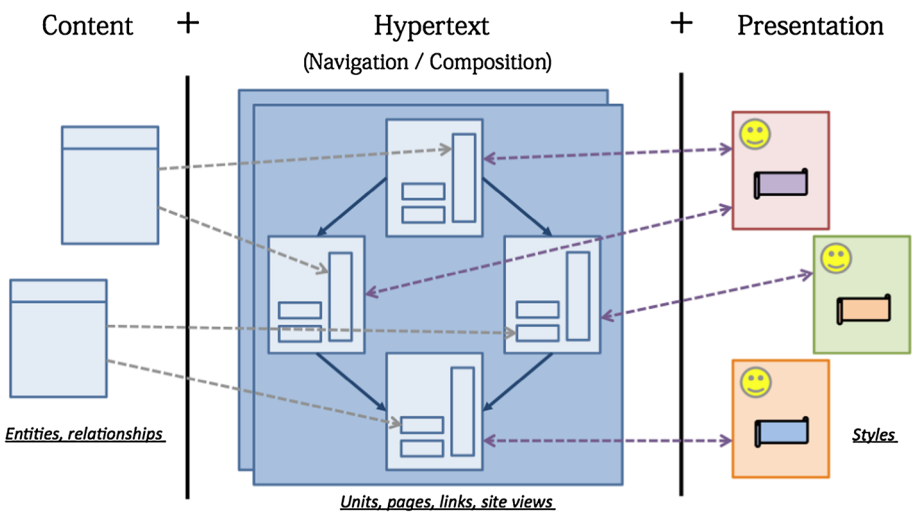
As Fig. 3 shows, the specification of a Web application in WebML considers a set of orthogonal dimensions, namely the *Content*, the *Hypertext* (*Navigation/Composition*) and the *Presentation*.

The *Content* is modelled by adopting entity/relationship primitives [13] (or, equivalently, UML class diagrams) to represent the organization of the application data resulting in the *Content* model. Thereby, the main elements of this model are the entities, which are defined as containers of data elements, and the relationships, which are defined as semantic connections between entities.

The *Hypertext* is modelled in a number of *Navigation* and *Composition* models that describe how the data specified in the *Content* model are published in the application by means of pages which are linked to each other. To that end, a hyper-text structure is defined in terms of Site Views, Areas, Pages, Units and Links. Actually, the definition of this structure is divided into two types of models as follows: *Composition* models specify the Pages bundled in the application as well as their internal organization in terms of units; on the other hand, *Navigation* models define the set of Links that connect the different Units and Pages collected in the Composition model. Both models provide a complete specification of the application front-end.

Presentation deals with the graphic appearance of the application by defining how the pages defined in the *Hypertext* model are rendered according to a set of style sheets which dictate their layout. WebML does not include a specific model for expressing presentation at the conceptual level, but leverages standard approaches, more familiar to graphic and communication experts. Actually, since WebML specifications can be represented using XML, presentation is considered like a document transformation mapping the WebML specification of a page into a page written in a con-

Fig. 3 Dimensions of the specification of a web application in WebML



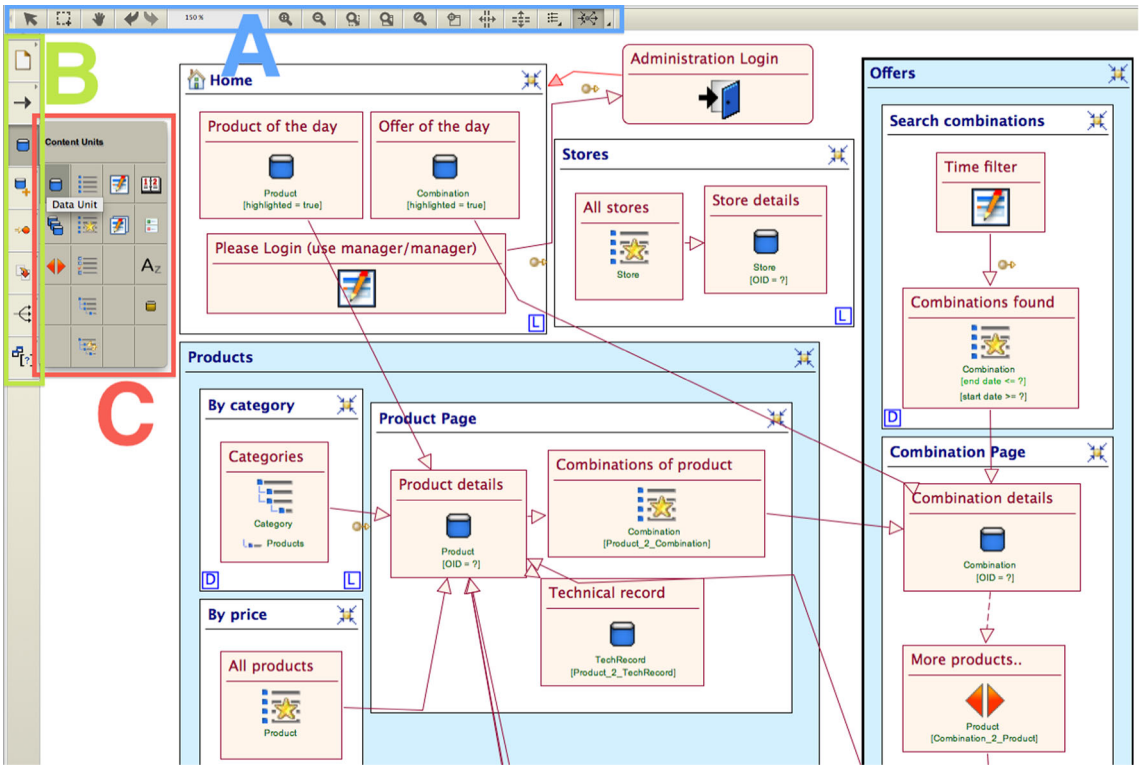


Fig. 4 Hypertext model editor in WebRatio

crete implementation language like JSP or ASP.NET. Consequently, presentation is addressed in WebML by attaching XSL style sheets to site views, pages, units and unit subelements.

The main innovation in WebML is actually the hypertext modelling notation (*Navigation* and *Composition* models), which was patented in 2003, and the focus of the analysis performed in this paper uses it as the main building block of the proposal. Furthermore, note that the *Content* model is largely based on the well-known E/R model, whose understandability has been previously assessed in a number of works, such as [20], whereas *Presentation* rests on the application of style sheets for which an analysis of its visual syntax makes no sense since they are simply built atop of an XML abstract syntax.

2.2 WebRatio: industrializing WebML

Enhancing the role of models and increasing the level of automation are probably the main principles of MDE [52]. The latter is even more relevant in the context of WebML, since high-level specifications are to be directly translated into working-code. WebRatio, which was originally a Java stand-alone tool that later turned into an Eclipse-based tool, was devised to answer this need by automating the WebML proposal. To that end, WebRatio bundles a mod-elling toolkit that implements the WebML models, plus a set of code generators to translate such models into running applications.

Probably the most interesting component of WebRatio from the point of view of this work is the *diagram editor*, which supports the data and hypertext models. Figure 4 shows a screen-capture of the editor including an excerpt from the hypertext model of a project related to a small furniture company's website.

The upper left-hand side of the figure shows the main controls of the diagram editor, such as the classical toolbar for selecting, moving, re-sizing and aligning elements of the diagrams (A). Another toolbar contains a button for each of the elements' categories of the language, namely Containers, Links, Data Units, Operation Units, Session Units, Service Units, Units and Flow Control Utility Units (B). When clicked, each button invokes the palette containing the controls that allow instantiating the elements of the category. The Data Units palette is shown to illustrate the use of these controls (C).

The model excerpt depicted in the drawing pane is part of the project focused on publishing the catalogue of the aforementioned furniture company. Note that the Home Page will publish information about products and special offers of the day. Such products are linked to a technical record which is published in the Product Page which is contained in the Products Area. Also note that Product Pages can be displayed according to their Category or Price. On the other hand, the

Offers Area on the upper right-hand side of the figure shows that offers are combinations of products sold at a discounted price during a specified period.

3 Research framework

Software Engineering currently has a number of established methods that are used to evaluate the semantics of the concepts used in different languages, but it lacks equivalent methods with which to evaluate their visual syntax, whose relevance has historically been undervalued, probably because visual notations have been traditionally considered as an informal concept, contrary to what occurs with semantics.

The case of UML, the most widely adopted modelling language (although its practical usage has been frequently put into question), serves to illustrate this situation: despite the number of existing studies on UML (see [46,54] for instance), most of them are focused on analysing the semantics of the language, whereas almost none have considered the correctness of its visual syntax (see [18,55] for instance).

Still, there are some proposals which can be applied to assess the visual syntax of a given modelling language. Considering that this is the main goal of this research, in the following we briefly describe them to motivate the final adoption of Moody's Physics of Notations, which is later introduced in Sect. 3.2.

3.1 Proposals for notation analysis

Among the few pieces of research focused on the definition of a framework to support the analysis of a given notation, it is worth mentioning the one from Krogstie, Sindre and Jorgensen on semiotic quality (SEQUAL) [29]. SEQUAL is based on semiotic theory and provides a list of properties to evaluate the quality of models and modelling languages, defining an extensive ontology of modelling language quality concepts such as: *physical*, *empirical*, *syntactical*, *semantic*, *perceived semantic*, *pragmatic*, *social*, *knowledge*, *language* and *organizational* quality. The first version of SEQUAL dismissed the relevancy of cognitive effectiveness, although some principles, like expressive economy, were still considered as a means to reach *pragmatic quality* [31]. The latest version of the framework goes a step beyond and almost com-pletely ignores the type of features that we aimed to assess in this work [29]. It was therefore discarded as an assessment framework.

On the other hand, Schuette and Rotthowe [51] presented a set of guidelines of modelling (GoM), which aimed at defining a framework to improve the quality of information models by running a set of syntactic rules based on six principles: *Accuracy*, *Relevance*, *Economic Efficiency*, *Clarity*, *Comparability* and *Systematic design*. Although it was created with the intention of being used by modellers, the proposal ignores the complexity and limitations of the human mind. Moreover, it cannot be applied as is to evaluate models expressed with different languages but needs to be adapted and refined for each particular language.

The cognitive dimensions (CDs) proposal, which was first introduced by Green [25] as a set of features that provide a language to compare the form and structure of programming languages, has been the most referenced approach by researchers on the usability of visual languages. Actually, the application of that proposal to analyse the usability of visual programming environments presented by Green and Petre [24] yielded the Cognitive Dimensions framework, which has later been applied to evaluate many different types of notations. In short, the framework provides a vocabulary of terms (or dimensions) to specify the details of the structure of cognitive artefacts. The main terms are shown in Table 1, along with their informal definition.

Existing literature states that the CD framework presents some flaws from the point of view of this work, some of them acknowledged by the authors:

- It is too generic, since it was devised to be used in any type of domain [26], from spreadsheets to programming languages. In particular, it was not intended to work for visual modelling languages.
- The definitions of dimensions (main basis of the proposal) are not very precise, which, next to the lack of a well-defined procedure, leads to confusion and misunderstanding at the time of using them [16].

Table 1 Main cognitive dimensions of CDs framework

Cognitive dimension	Definition
Abstraction gradient	Availability and types of abstraction mechanisms
Closeness of mapping	Nearness of representation to domain
Consistency	Similar semantics are defined in similar syntactic forms
Diffuseness	Conciseness of language
Hard mental operation	Processes that place a high demand on working memory
Hidden dependencies	Significant links between entities are not visible
Progressive evaluation	Effort required to achieve a goal
Role-expressiveness	The purpose of a program component is easily deduced
Secondary notation	Additional information in means other than program syntax
Viscosity	Resistance to alteration
Visibility	Capacity to view components easily

Table 2 Number of documents citing Moody's work [39] and Green and Petre's work [24]

Source: Scopus

	1996–2009	2009–2013	Total	2009	2010	2011	2012	2013
Moody's	–	136	272	3	15	30	39	49
Green and Petre	210	167	544	45	41	32	26	23

- The proposal lacks theoretical and empirical foundation [16].
- The number of dimensions has grown since the appearance of the framework, resulting in an unmanageable set of dimensions arising from the simplification of the framework to target non-skilled potential users.

Finally, Moody's Physics of Notations theory [39] is a framework which establishes nine principles to design, evaluate, compare and improve visual notations which has garnered a lot of attention during the last number of years.

A quick look at some data gathered from SCOPUS (see Table 2) shows that the number of references to Moody's work, since it was first published, is almost the same as the number of references to Green and Petre's work. Note that the latter was published more than 10 years before Moody's work. Note also the upward trend in the number of references to Moody's work against the downward trend for Green and Petre's.

This general reflection may serve to show that with the advent of MDE, where visual modelling languages have become even more relevant, Moody's proposal has attracted MDE practitioners to use it as an evaluation technique, since it fits perfectly with the nature of these languages while preserving the complexity of application under certain admissible levels. However, remember that this information does not reflect conclusive statistical data, but allows us to get an idea about the trend in this context.

As a matter of fact, when Green and Petre were asked about their personal reflections on CDs 10 years after they first introduced it, they opted to present their reflections separately [26,50]. However, both of them started their respective contributions by pointing out that the main purpose of CDs was not to provide a framework to evaluate the qual-ity (actually the usability) of information artefacts, but to provide a vocabulary of terms to ease the discussion around human–computer interaction (HCI).

We interpret this as an evidence of the lack of proper methods and techniques to evaluate visual notations: under the absence of a suitable proposal to that end, researchers have been forced to adopt a generic proposal, such as CD's, which was not intended to work particularly well for visual notations (indeed, it was thought to evaluate non-visual notations), and adapt it to their needs.

By contrast, Moody's proposal was specifically designed to define principles for correct development of visual languages. In this sense, recall that we aim to evaluate the effectiveness of WebML's visual notation, not the usability of its supporting tool, a task for which CD's will definitively work better. Using CD's here would serve, consequently, to complement the analysis performed using the Physics of Notations theory.

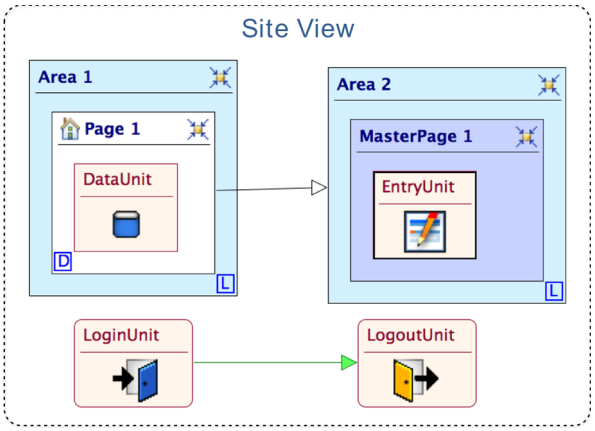


Fig. 5 Main composition rules of WebML visual notation

However, being exclusively focused on the best way to represent a set of constructs visually, the Physics of Notations has also some limitations. In particular, it does not propose any principles with which to assess the effectiveness of the composition rules of the language, which may result in cognitively inefficient diagrams.

In our case, this is not particularly disturbing, since the set of WebML composition rules is merely limited to those shown in Fig. 5: a Site View is composed mainly of Areas and Units; Areas contain Pages that are made by other Units and, finally, each of these elements may be interconnected by Links.

Note that these composition rules are mainly inherited from the definition of the abstract syntax of the language (metamodel plus additional restrictions) which, in the context of MDE, is typically developed prior to addressing the development of the concrete syntaxes of the language [10], and usually in an isolated way. It is therefore much more complicated to act on these composition rules at the time of assessing the visual syntax of the language. Just think about the impact that the evolution of metamodels has had over the ecosystem of related models and transformations.

As a matter of fact, some initiatives towards moving the focus of DSLs development to its notation instead of its metamodel have recently emerged in response to this scenario [61]. Applying the cognitive dimensions framework to

drive the development of the language would be much more affordable when adopting a notation-driven approach, since it eases the task of rethinking the composition rules of the language.

By contrast, we believe that the Physics of Notations theory fits better with the metamodel-driven approach adopted by most of the existing DSLs. As we will mention sev-eral times throughout this paper, adopting Moody's proposal does not ensure a cognitively efficient language but provides certain levels of confidence without compromising the balance between effort and reward when performing the kind of assessments presented in this paper.

Given all this, we could say that there is no perfect tool for the job. Using either the Cognitive Dimensions or the Physics of Notations, some aspects will not be properly covered by the analysis. However, we believe the latter to fit bet-ter with the purpose of this paper, thus fewer aspects remain uncovered.

Note also that even though they have been applied to similar purposes, their main goal was not exactly the same. Whereas the Cognitive Dimensions was particularly intended for usability assessment, the Physics of Notations was devised for cognitive effectiveness assessment. In this sense, both frameworks could be seen as complementary.

To conclude this section, we would like to mention that while we have presented some of the main frameworks that can be used to evaluate the visual syntax of a modelling language, more and more research in the area is emerg-ing. See for instance the works from Baar et al. [3,19], which propose concrete syntax to be defined by extending the metamodel that defines the abstract syntax and provide a technique to ensure consistency between both definitions, or the works from Bottoni et al. [8,9] on constraining the definition of concrete syntaxes by means of metamodelling techniques.

3.2 The Physics of Notations

As mentioned before, the so-called Physics of Notations theory [39] from Moody is a framework exclusively devoted to designing, evaluating, comparing and improving visual notations. In this work, Moody establishes a set of nine principles defined from theory and empirical evidence brought from different disciplines such as: cognitive and perceptual psychology, graphic design, communication theory, cartography, etc. Indeed, these principles have been already used in several works to evaluate and improve other visual languages such as ArchiMate [41], i* [42], BPMN [21], UML [40] and UCM [22].

Each of the principles of the Physics of Notations contain the design strategies which may contribute to improving visual notations regarding such principles; a different evaluation procedure or metric that can be used to compare different notations; examples of notations that satisfy or violate the principle. These nine principles are:

1. The Principle of **Semiotic Clarity**: there should be a one-to-one correspondence between elements of the language and graphical symbols.
2. The Principle of **Perceptual Discriminability**: different symbols should be clearly distinguishable from each other.
3. The Principle of **Visual Expressiveness**: the use of the full range and capacities of visual variables.
4. The Principle of **Semantic Transparency**: the use of visual representations whose appearances suggest their meaning.
5. The Principle of **Complexity Management**: include explicit mechanisms when dealing with complexity.
6. The Principle of **Cognitive Integration**: include explicit mechanisms to support the integration of information from different diagrams.
7. The Principle of **Dual Coding**: use text to complement graphics.
8. The Principle of **Graphic Economy**: the number of different graphical symbols should be cognitively manageable.
9. The Principle of **Cognitive Fit**: use different visual dialects for different tasks and audiences when required.

Before addressing the assessment of WebML visual notation according to Moody's principles, we would like to introduce the concept of *visual variables*, a set of elementary building blocks that can be used to graphically encode information, which are often used and referenced in each of the principles.

Studies conducted on the nature of graphical symbols have identified eight different visual variables (see Fig. 6) that can be used to encode information. These variables can be defined in two subsets: *planar* and *retinal*. The most important work in this regard is the seminal work of Bertin [5], which is considered to be to graphic design what the periodic table is to chemistry.

Each of these visual variables has a set of properties that are used to encode certain types of information, and these properties must therefore be known, if effective choices are to be made. From here on, each time that a visual variable is mentioned, we shall use the convention of underlining its name to ease their identification.

4 Analysis of WebML visual notation

In this section, we evaluate the various WebML graphical symbols implemented in WebRatio through the principles

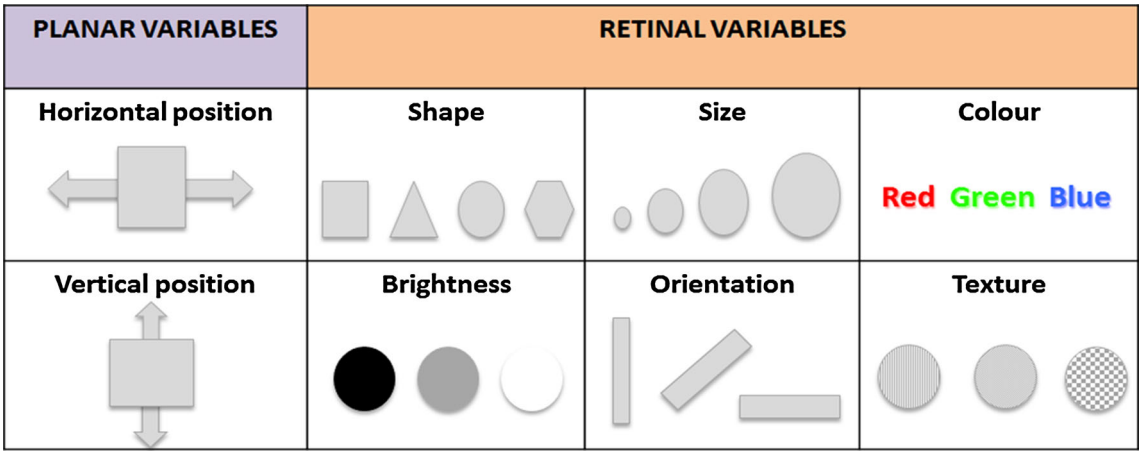


Fig. 6 Visual variables used to construct visual notations (adapted from [5])

mentioned in the previous section. Recall that, as mentioned before, WebML comprises different types of models, but considering that the *Content* model is based exclusively on E/R diagrams and the *Presentation* model is based on the use of style sheets, this analysis focuses just on the *Hypertext* model, the most relevant in the development of a Web application with WebML.

The following subsections show the main results of the analysis, which are organized by principle. This way each subsection contains a brief description of the principle, which is text excerpted directly from the Moody's Physics of Notations theory [39], next to a summary of WebML's assessment with regard to the principle and some suggestions as to how the language could be improved according to the assessment results.

Before diving into the assessment, it is worth mentioning that most of the principles are somehow related. The findings and particularly the suggestions made for one principle may consequently depend on some others. Likewise, it should be noted that in order to make suggestions for improvement related to certain principles, a more detailed and comprehensive study of the language is required. In our case, we have identified that there is room for improve-ment in some aspects, and we have validated some of our proposals made throughout this paper through an empirical study.

Finally, it is worth mentioning that the reflections that follow are the result of debates and discussions that were held by the authors during the development of this work. And to contrast these discussions, we decided to conduct an empirical study to validate some of the most relevant findings. In particular, the empirical validation has focused solely on the recommendations for the iconic representations, which are the ones that result best candidates for refining: icons can be easily updated without concerning too much about implementation details.

4.1 Principle of semiotic clarity

4.1.1 Description

According to Goodman's theory of symbols [23], for a notation to satisfy the requirements of a notational system, there must be a one-to-one correspondence between symbols and their referent concepts. When there is not a one-to-one correspondence between constructs and symbols, one or more of the following anomalies can occur:

- Symbol redundancy occurs when multiple graphical symbols can be used to represent the same semantic construct. These symbols are called synographs [45].
- Symbol overload occurs when two different constructs can be represented by the same graphical symbol. These symbols are called homographs.
- Symbol excess occurs when graphical symbols do not correspond to any semantic construct.
- Symbol deficit occurs when there are semantic constructs that are not represented by any graphical symbol [39].

4.1.2 Assessment

To identify anomalies related with the Semiotic Clarity of WebML, it has been necessary to develop a list of all of its elements and their corresponding symbols in order to contrast them. This has been done by studying the *non-abstract* elements of the language in the WebML metamodel and the available symbols in the WebRatio tool. For instance, the *ContentUnitView* package of WebML's metamodel contains 24 metaclasses. Eight of these metaclasses are abstract while just 15 of them have a graphical

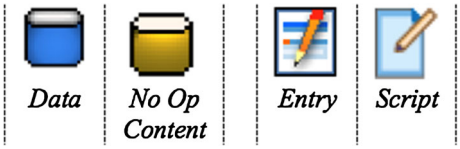


Fig. 7 Symbol overload occurrences

symbol assigned in WebRatio. The analysis of the whole metamodel reveals a total of 41 *non-abstract* elements of the language, and 70 graphical symbols were obtained, which correspond to the occurrences of 0*symbol redundancies*, 5 *symbol overloads*, 42 *symbol excesses* and 4 *symbol deficits*.

4.1.3 Recommendations

- *Symbol overload* (5 occurrences) could be resolved by differentiating between the symbols used to represent different elements of the language, and the principle of **Perceptual Discriminability** is again useful for this differentiation. For instance, this anomaly occurs in cases such as *Data-NoOpContent* and *Entry-Script* as shown in Fig. 7. Note that the similarities between each pair of symbols hamper distinction between the concepts they were devised to represent.
- *Symbol excess* (42 occurrences) could be resolved by removing any unnecessary symbols. Unnecessary symbols increase graphic complexity and hamper the understanding of appropriate ones [45].
- *Symbol deficit* (4 occurrences) signifies that some metaclasses do not have a corresponding symbol. For instance, this anomaly occurs in cases such as *Transaction* and *Precondition*. This could be solved by creating new symbols or modifying the WebML metamodel in order to remove, where appropriate, some of the elements of the language.

As the data above shows, the principal problem of WebML in this aspect is *symbol excess*, and this problem has arisen because language specification has not undergone the same rate of development and maintenance as the tool that implements the language. However, analysing whether or not changes to the semantics are necessary is not within the scope of the Physics of Notations, i.e. we work under the assumption that WebML semantic concepts are appropriate and have been well defined, thus the only artefacts that might be modified are the symbols used to represent them. Such modifications should be driven towards addressing the issues presented in this work.

4.2 Principle of perceptual discriminability

4.2.1 Description

> **Perceptual Discriminability** is the ease and accuracy with which graphical symbols can be differentiated from each other. This relates to the first phase of human visual information processing: perceptual discrimination. Accurate discrimination between symbols is a pre-requisite for accurate interpretation of diagrams [60]. Discriminability is primarily determined by the visual distance between symbols. This is measured by the number of visual variables on which they differ and the size of these differences. In general, the greater the visual distance between symbols, the faster and more accurately they will be recognized [39,59].

4.2.2 Assessment

WebML symbols can be grouped into three categories: **Containers** (e.g. *Page*, *Area* and*Master Page*), **Units** (e.g. *Contents*, *Operations*, *Service* and *Session*) and **Links** (e.g. *KO* and *OK*). We shall first analyse **Perceptual Discriminability**. To that end, Table 3 lists some of these symbols, along with the values of their visual variables and their semantic carrier (SC), i.e. whether the visual variables of each symbol are related to their semantics.

- WebML **Containers** such as *Page* and *Area* use only two visual variables to carry semantic information: Shape and Colour. In other words, the Shape and the Colour used to represent Containers allow us to distinguish them from other types of WebML elements. More concretely, they share the same Shape: a quadrilateral, and use a different Colour for the background of the different types of Containers.
- WebML **Links** use three visual variables to carry semantic information: Shape, Colour and Texture. They share the same Shape, an arrow, and use different Colours and Textures to distinguish between the types of Links.
- WebML **Units** use three visual variables to carry semantic information: Location (horizontal and vertical position), Shape and Colour. Content Units differ from other types of Units in the Shape used to represent them (rectangle and roundtangle) and in the fact that the former have an inclusion relationship with **Containers**. Their planar variables thus depend on the Location of their **Containers**. The Colour of all Units is the same. Moreover, each of the Units uses a different *iconic marker* (this is the term used in the literature to refer to icons located inside a graphical symbol), which is discussed in the principle of **Semantic Transparency**.

Table 3 Visual variable values and semantic carriers (SC) for some WebML symbols

		Location		Shape		Size		Colour		Brightness		Orientation		Texture	
		Value	(SC)	Value	(SC)	Value	(SC)	Value	(SC)	Value	(SC)	Value	(SC)	Value	(SC)
Containers		Variable	(N)	Quadrilateral	(Y)	Variable	(N)	White	(Y)	N.A.	(N)	N.A.	(N)	Thin border	(N)
		Variable	(N)	Quadrilateral	(Y)	Variable	(N)	Turquoise	(Y)	N.A.	(N)	N.A.	(N)	Thin border	(N)
Links		Variable	(N)	Arrowhead link	(Y)	Variable	(N)	B/W	(Y)	N.A.	(N)	Variable	(N)	Single dotted	(Y)
		Variable	(N)	Arrowhead link	(Y)	Variable	(N)	Green	(Y)	N.A.	(N)	Variable	(N)	Single thin	(Y)
		Variable	(N)	Arrowhead link	(Y)	Variable	(N)	Red	(Y)	N.A.	(N)	Variable	(N)	Single thin	(Y)
Units		Inclosure	(Y)	Quadrilateral	(Y)	Fixed	(N)	Pale peach	(Y)	N.A.	(N)	N.A.	(N)	Thin border	(N)
		Variable	(Y)	Roundtangle	(Y)	Fixed	(N)	Pale peach	(Y)	N.A.	(N)	N.A.	(N)	Thin border	(N)

4.2.3 Recommendations

Size is a variable that influences discriminability. Larger sym-bols immediately attract the reader's attention [48]. However, WebRatio only allows the re-sizing of the symbols that may contain other symbols according to an inclusion relationship. Furthermore, Size also depends on the amount of text used to attach a name to the symbols, i.e. if the text inside the symbol is too long, the symbol is automatically resized to fit the text. One recommendation would be to provide the possibility of being able to customize the Size of each symbol, so that the designer can decide on that visual variable.

The Colour variable is one of the most cognitively effective variables [34]. It is used with a different value for each Container and for each Link. In contrast, all Units have the same background Colour. The Physics of Notations theory proposes that the choice of Colour should be related to some kind of relationship between the symbol and the concept it represents. One proposal would be to use one different Colours to differentiate between the seven categories of Units. Thus, we would use a number of colours that are within the limits of the capacity of the visual variable, i.e. the number of different perceptible steps by the human mind. But if we surpass the capacity limits of a visual variable, we might encounter saturation problems. This concept is explained in detail in the next section. However, note that despite Colour being one of the most effective visual variables for the human visual system, it should not be used as the sole basis for distinguishing between symbols, as it is sensitive to variations in visual perception (e.g. colour blindness), in screen/printer characteristics (e.g. B/W printer) and in representational mediums (e.g. whiteboards/paper).

The Shape plays a key role and is the primary basis for discrimination between symbols. This means that it must be possible to clearly distinguish between all the Shapes used for the various symbols. For example, 3D Shapes could be used for those symbols that represent Containers because these forms suggest the ability to contain objects. Moreover, in WebML, most symbols are variants of the rectangle (see first column of Table 3).

Therefore, another recommendation would be to use different basic geometric Shapes, such as ellipses, triangles and cylinders instead of using only quadrilaterals, in order to increase discriminability between the symbols used either by different categories or by different elements in the same category.

4.3 Principle of visual expressiveness

4.3.1 Description

Visual Expressiveness is defined as the number of visual variables used in a notation. Visual expressiveness partitions the set of visual variables into two subsets:

- **Information-carrying variables** Variables used to encode information in a notation.
- **Free variables** Variables not (formally) used.

As well as using only a limited range of the visual variables available, SE notations also use only a limited range of the possible values of each variable (capacity). For example, they use a very limited repertoire of shapes, mostly rectangle variants [49]. These are the least effective shapes for human visual processing, and empirical studies show that curved, 3D and iconic shapes should be preferred [4,28,60]. Colour is one of the most cognitively effective of all visual variables: the human visual system is highly sensitive to variations in colour and can quickly and accurately distinguish between them [34]. Differences in colour are detected three times faster than shape and are also more easily remembered [32].

Table 4 Visual variables capacity (adapted from [39])

Variable	Capacity
Horizontal pos.	10–15
Vertical pos.	10–15
Size	20
Brightness	6–7
Colour	7–10
Texture	2–5
Shape	Unlimited
Orientation	4

Different visual variables have different capacities (number of perceptible steps) [5]. The properties of each visual variable have been established by research in psychophysics (summarized in Table 4) [39].

4.3.2 Assessment

In the WebML symbols, some visual variables contain information related to the semantics of the symbol. More concretely, the *information-carrying variables* in WebML are Location, Shape, Colour and Texture, while Brightness, Size and Orientation do not carry semantic information and can be defined consequently as *free variables*. Table 5 shows a summary of each information-carrying variable. In this table, we can see the values used in WebML for each of the visual variables and their corresponding *saturation*, i.e. the ratio between the number of values used and the *capacity* of each variable. This ratio illustrates to what extent the visual variable is efficiently used in WebML.

The only value of Location (horizontal and vertical position) that carries semantic information in WebML is *enclosure* (i.e. symbols contained in others symbols); apart from this, the Location of a given symbol provides no semantic information. Therefore, to calculate the Location's *saturation*, we must divide the quantity of values used by the language (1) between the upper limit (15) and the lower limit (10) of the *capacity* of this visual variable, whereby we get the range saturation of this visual variable ($1/15 = 7\,\%$ and $1/10 = 10\,\%$). This indicates that there is room for using other values of this visual variable and thus increase the visual expressiveness. Next, the different WebML symbols use nine different Colours. Considering that the *capacity* of the Colour variable ranges between 7 and 10, *saturation* ranges from 90 to 100 %. This indicates that using a greater range of Colours could complicate a correct discrimination by users. Then, Texture is only used to differentiate between two types of Links that yield a *saturation* point of 40–100 % (i.e. it ranges from 2/2 to 2/5 since Texture capacity ranges between 2 and 5). Finally, Shapes are mainly limited to two categories: quadrilateral and arrows. In this case, a *saturation* point cannot be derived since the Shape variable has unlimited *capacity*.

4.3.3 Recommendations

The Location variable could be used to represent an interval if required, and not only to represent the enclosure. A possible value for this variable would be visual e.g. the possibility to represent the internal pages under their main webpages. In this way, the user, by Location, can have an immediate idea of the main content of a webpage.

Texture in WebML is only used to distinguish between some types of Links. The data above shows that according to the lower limit of texture capacity, there is room to use three more Textures. These free Texture values could consequently be used to distinguish between the three categories of symbols that use a quadrilateral Shape with a single line border, i.e. *Containers*, *Content Units* and *Other Units* (see Table 3).

WebML symbols use mainly two types of Shapes. This has therefore led to the possible improvement discussed in Sect. 4.2.3. However, replacing the symbols using an iconic marker by the marker itself would be more highly recommendable. This way, the symbol becomes an icon itself, widening the range of values for the Shape variable.

With regards to Colour, according to Table 3, the range of Colours used in WebML notation is appropriate for the differentiation capacity of the human mind. However, these colours are mainly used to differentiate the different types of Containers, whereas the same Colour (peach) is used for the background of all the types of Units. Therefore, our recom-

Table 5 Information-carrying variables in WebML visual notation

Variable	Capacity	WebML values—(quantity)	Saturation (%)
Horizontal pos.	10–15	Enclosure—(1)	7–10
Vertical pos.	10–15	Enclosure—(1)	7–10
Colour	7–10	White, black, periwinkle, grey, turquoise, silver, red, green, peach—(9)	90–100
Texture	2–5	Single dotted, single thin—(2)	40–100
Shape	Unlimited	Quadrilateral, arrowhead, roundtangle—(3)	–

mendation is to use different colours to distinguish between the Unit types also.

Finally, Brightness, Orientation and Size are not used in the symbols and could be used in order to increase **Visual Expressiveness** and**Perceptual Discriminability**, since as mentioned before these two principles are directly related.

4.4 Principle of semantic transparency

4.4.1 Description

Semantic Transparency is defined as the extent to which the meaning of a symbol can be inferred from its appearance. The concept of semantic transparency formalizes informal notions of "naturalness" or "intuitiveness" that are often used when discussing visual notations, as it can be evaluated experimentally. Semantic transparency is not a binary state but a continuum:

- A symbol is semantically **immediate** (I) if a novice reader would be able to infer its meaning from its appearance alone.
- A symbol is semantically **opaque** (O) if there is a purely arbitrary relationship between its appearance and its meaning.
- A symbol is semantically **perverse** (P) if a novice reader would be likely to infer a different meaning from its appearance.
- In between semantic immediacy and opacity, there are varying degrees of semantic **translucency** (T).

Iconic representations speed up recognition and recall and improve intelligibility of diagrams to naive users [35]. They make diagrams more visually appealing: people prefer real objects to abstract shapes [4,39,49].

4.4.2 Assessment

There are three different types of Links: Normal, KO and OK. As shown in Table 3, the Shape used to represent them is the arrow, whereas different colours allow us to distinguish them easily: green is used for the OK Link, red for the KO Link and black for the Normal Link. The arrow is the most common and intuitive Shape to represent a link, and green and red Colours own intrinsic meanings in most domains, e.g. right/wrong, threat/safe, stop/go, etc. The combination of the Shape and Colour used is therefore considered to result in graphical symbols that represent the different types of Links with an immediate Semantic Transparency.

The main Shapes used in all other elements of WebML are a variant of a rectangle, which is very common in the visual notations used in modelling language in SE [49]. These Shapes are semantically opaque since there is usually no rela-tionship between their appearance and the meaning of the concept they represent. However, each of the rectangles used to represent WebML Units contain a different iconic marker. Therefore, we will analyse the iconic markers for the ele-ments of the different Units categories here, bearing in mind that the level of semantic transparency depends on different factors (e.g. expert/novice users) and that a meaning of a sym-bol may therefore be immediate for some users and opaque for others. The assessment below is not therefore meant to be definitive. The following statements are merely based on direct observation. For this reason, we have carried out an empirical study to validate some of the proposals made in relation to this principle. The details and results of this study can be found in Sect. 5.

Figure 8 shows the icons for the Content Units. The meaning (also known as the referent concept) of the icons such as *Power Index*, *Recursive Hierarchical Index* and *Multi Message* is not so obvious or immediate. For example, some lines arranged horizontally with green and red boxes could refer

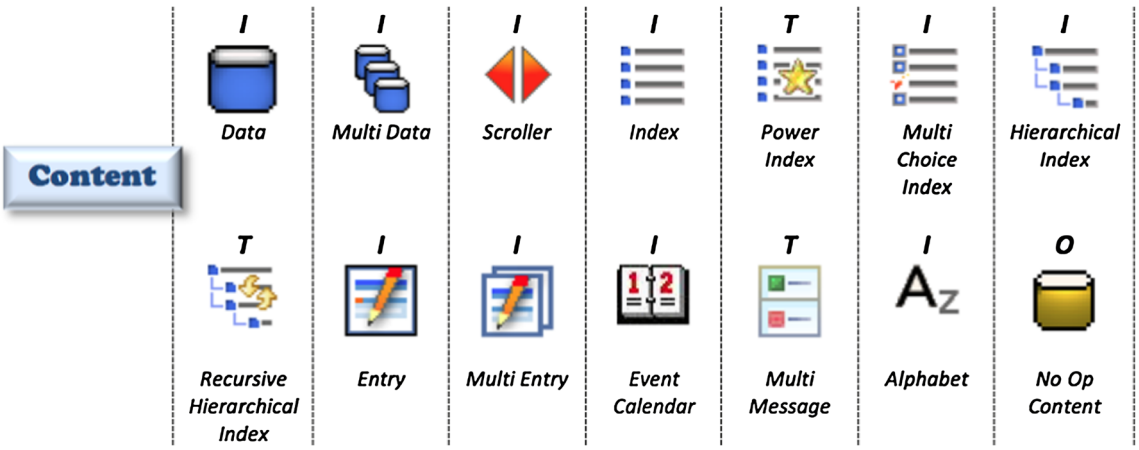


Fig. 8 Content Units—icons

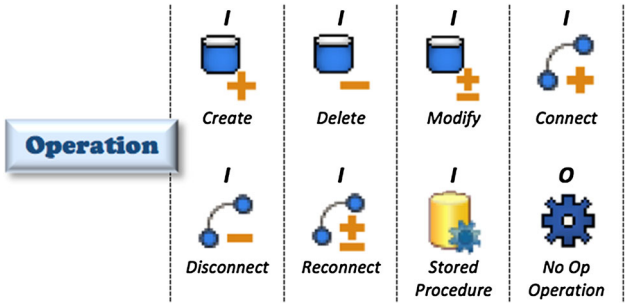


Fig. 9 Operation Units—icons

to the concept of a to-do list instead of its intended meaning, which is *Multi Message*. Moreover, the Shape chosen for the so-called *No Op Content* icon is in no way related to its semantics, i.e. to the notion of a Unit that represents the concept of "no business logic", and it is thus semantically opaque. The remaining referent concepts for the other icons are obvious, signifying that the symbols have been well chosen. However, there is room for improvement, as will be shown in the following section.

Regarding Operation Units, Fig. 9 shows the corresponding icons. The Shape chosen for the so-called *No Op Operation* icon is in no way related to its semantics, i.e. to the notion of a Unit that represents the concept of "no business logic" and it is thus semantically opaque, as it happened with the *No Op Content* icon. The remaining referent concepts for the other icons are well chosen.

The Session Units' icons are shown in Fig. 10. The referent concepts for the first three icons are semantically translucent since it is not trivial to work out their exact meaning. By way of illustration, the *Get* icon is very similar to that used for *Jump* (see Fig. 12). Furthermore, an arrow Shape usually suggests a type of link or relationship. The last three are semantically immediate since the door icon is much more intuitive.

Regarding Service Units, it is not easy to immediately understand the meaning of the referent concepts for the *Get XML* and *Schedule Job* icons shown in Fig. 11. For instance, a few lines written on a sheet with an XML text annotation evoke the concept of XML source code, but no evocation of the *Get* action is transferred to the modeller. In contrast, the other icons are well chosen and are semantically immediate.

Moving to Control Flow Units, the referent concepts for the *Switch* and *Loop* icons shown in Fig. 12 are well chosen and are semantically immediate. However, the other icons are not as obvious or immediate, and are therefore semantically translucent. For instance, an arrow entering a circle that is subsequently divided into two arrows might evoke the concept of alternative paths, which is far from the actual referent concept that is (parameter) collector and it is similar to the *is not null* symbol.

Next, Fig. 13 shows the icons for the Utility Units. The referent concepts for icons such as *Script* and *Query* are not so obvious or immediate. For example, a pencil writing on a sheet of paper is virtually identical to the *Entry* icon. This results in the so-called *synograph* anomaly. Moreover, the

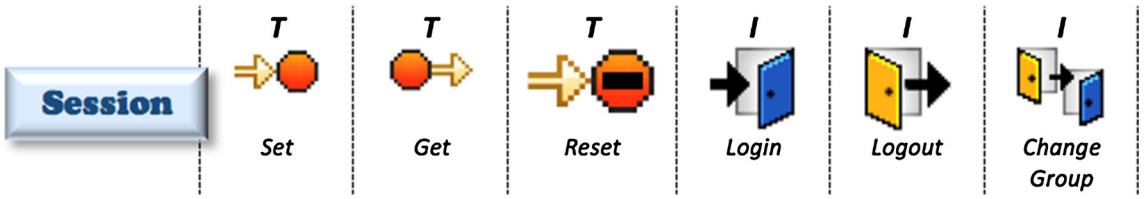


Fig. 10 Session Units—icons

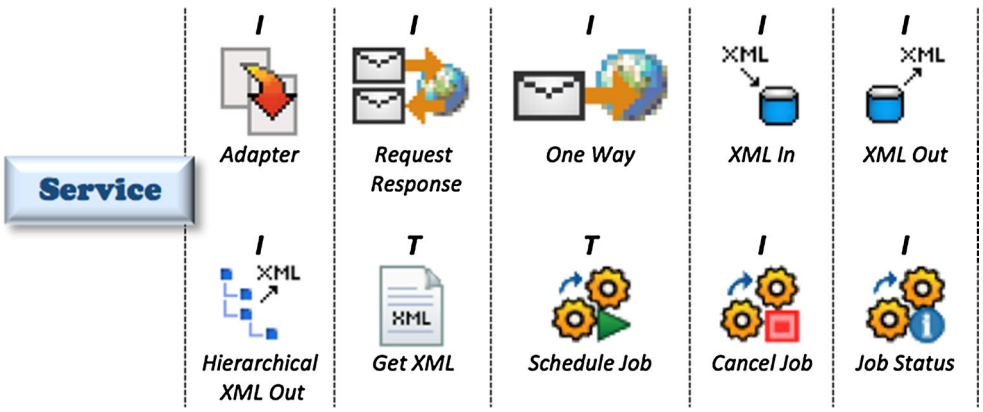


Fig. 11 Service Units—icons

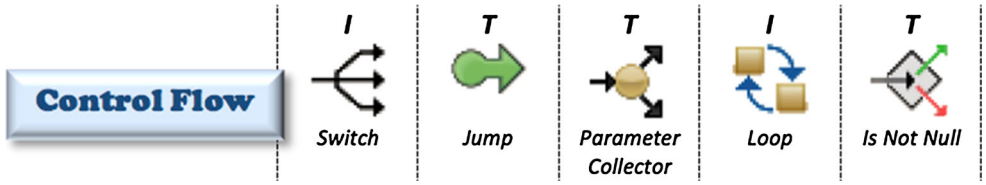


Fig. 12 Control Flow Units—icons

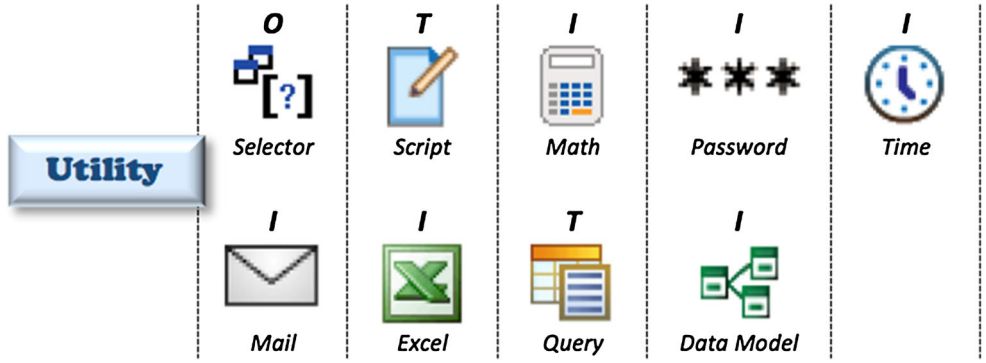


Fig. 13 Utility Units—icons

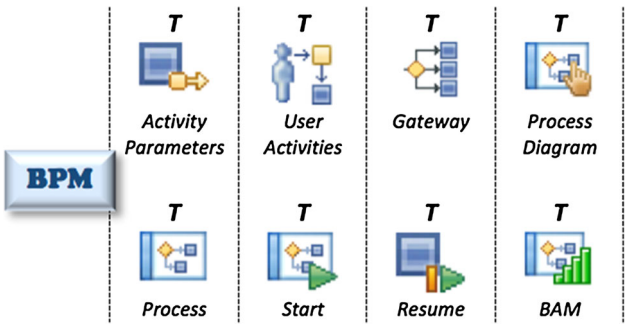


Fig. 14 BPM Units—icons

Shape chosen for the *Selector* icon is in no way related to its meaning and is thus semantically opaque. The remaining referent concepts for the other icons come easily to the user's mind, so that the symbols are considered to be well chosen. Note, however, that there is always room for improvement. For instance, while the Loop Unit traverses an array look-ing for an object which is returned aligned with its index in case of success, the Loop symbol might evoke a closed loop. Some kind of arrow pointing to an outgoing object might illustrate this behaviour. We believe, however, that the symbol works fine to represent at least the Unit's main functionality.

Finally, Fig. 14 shows the icons for the BPM Units. Icons in this category are semantically translucent. The main problem in this set is that some of its Shapes are very similar, and in some cases, the differences between them are too small, which creates problems also related to the principle of **Perceptual Discriminability**.

4.4.3 Recommendations

As mentioned previously, semantic immediacy plays a cornerstone role in order for any user to understand properly a given model. Previous paragraphs have shown that some WebML symbols leave room for improvement in this sense. Figure 15 therefore shows some proposals for new icons that improve semantic immediacy according to the commonly acknowledged idea that users prefer real objects to abstract forms [4].

In the subsequent paragraphs, the proposals for the new icons that improve semantic immediacy are described. The underlying idea is to re-use those graphical elements that are commonly used to bring to mind the same con-cept in completely different domains. Note that these might not be the best options, but are possible enhance-ments. The relevant finding here is that there is room for improvement.

- *Data* a series of stacked discs is usually used to represent this information.
- *Event Calendar* the current symbol used in WebML might appear to be a textbook and could be improved by using an icon that clarifies that we are referring to a calendar.
- *Power Index* a star is generally used to represent the concept of "favourite" and might be misleading. By contrast, the lightning symbol makes it semantically clearer that the concept of "power" is being represented.
- *Multi Message* rather than the current symbol that might evoke something related to a traffic light, a series of letters

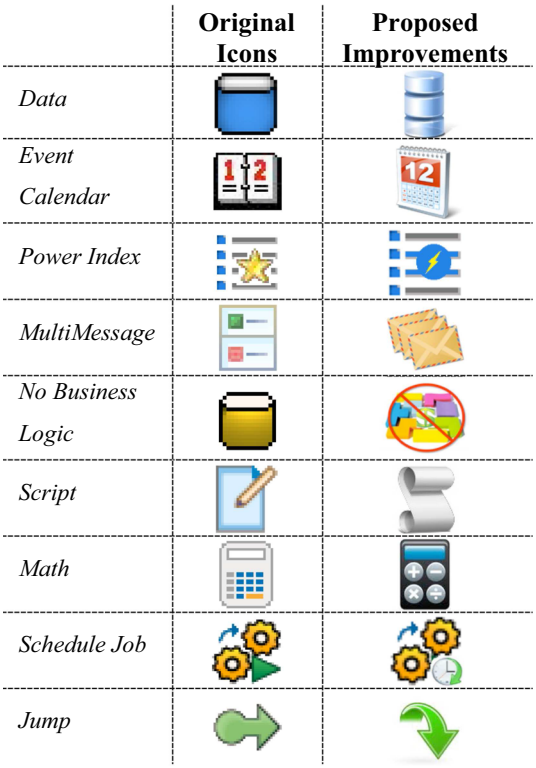


Fig. 15 Improving semantic transparency

might result in a much more intuitive symbol to represent multi-message objects.

- *No Business Logic* a complex icon is proposed containing a mix of the graphics used to symbolize prohibition (not), dollar (business) and Tetris game elements (logic).
- *Script* instead of the current icon that appears to be a notebook and is very similar to the *Entry* icon, the new proposed icon is far more frequently recognized in many areas of computing to represent code excerpts and programming languages.
- *Math* the proposed icon improves slightly on the current one and since it is close to the result, it is semantically transparent.
- *Job Schedule* rather than the classic graphic used to represent the "play" action, a clock with an arrow to represent the concept of "schedule" might result in it being semantically clearer.
- *Jump* rather than the arrow currently used that evokes the idea of moving forward, the proposed arrow is more related to the idea of jumping.

Remember that these proposals, of new graphical symbols, have been evaluated through an empirical study and the results of this study can be found in Sect. 5.

Note however that, just as it happens with the rest of principles, there might be some particular reason to maintain a given symbol as is, even though it was considered to impact negatively on Semantic Transparency. For instance, the advantages brought to cognitive effectiveness by adhering to an existing standard could be greater than that obtained by changing a symbol in order to make it more transparent. This would be the case of the symbols used by WebML to represent BPM Units if they were based on those defined by the BPMN standard. Even though they are found to be semantically translucent, this would not be a problem if the users are previously used to them because of previous knowledge of the standard. Unfortunately, this is not the case since these symbols are different from those found in the BPMN standard.

Finally, in order to decide on whether a symbol should be modified in order to improve the semantic transparency of the notation, one might wonder about the actual semantics of the concept represented by the symbol, which might be misleading due to wrong naming practices. For instance, think about the symbol used to represent the Power Index. Our proposal for improvement tries to invoke the concept of Power, whereas probably Enhanced or Multi are terms which better reflect the nature of the Power Index. Nevertheless, this kind of semantics analysis falls outside of the scope of this paper.

4.5 Principle of complexity management

4.5.1 Description

Complexity management refers to the ability of a visual notation to represent information without overloading the human mind. To effectively represent complex situations, visual notations must provide mechanisms for modularization and hierarchically structuring: These correspond to subsystems and level structures in ontological theory [39,59].

4.5.2 Assessment

One of the features supported by WebRatio is the ability to create reusable modules for different sets of Units (see Fig. 16). A reusable module is a kind of pattern that can be used everywhere in the project without having to copy all the pattern elements but just referencing it. To that end, two or more elements from the diagram are first selected and then grouped together into a single module by means of a context menu. From there on, the module can be reused elsewhere in the diagram. As a result, the need for recreating the very same set of elements at different parts of the diagram is avoided while complexity is reduced. Users can then check the module's content at any time just by double clicking on it.

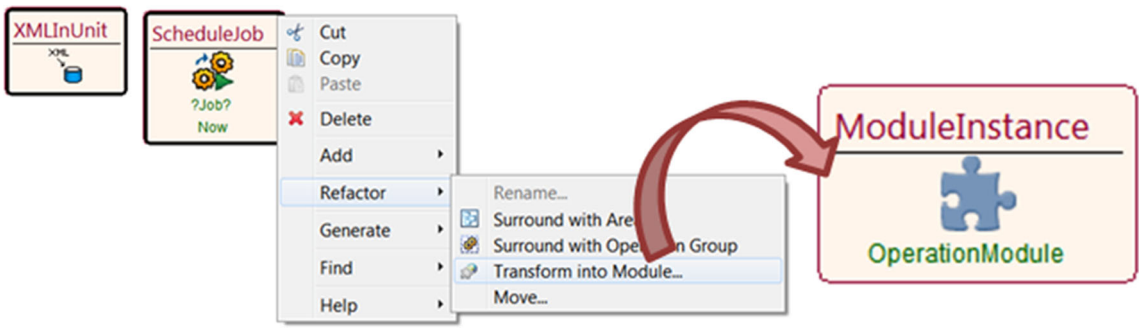


Fig. 16 Grouping different Units into a Module

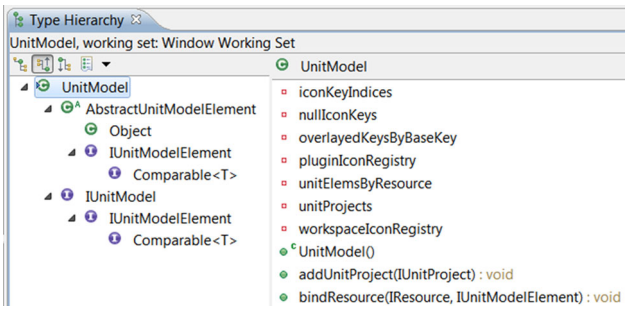


Fig. 17 Type hierarchy view on WebRatio

Another mechanism that is used to manage complexity is the hierarchical structure. WebRatio provides the Type Hierarchy View (see Fig. 17). This feature makes it possible to view classes, subclasses and members in a variety of different tree view ways.

Finally, another mechanism for managing complexity in WebML is the Master Page element. A Master Page is a special page that contains units, sub-pages and alternative pages. Master Pages have a hierarchical visibility, i.e. for each level, only one Master Page can exist, and a Master page defined in a lower level has higher priority than Master Pages defined on upper levels. Figure 18 shows an example of using the Mas-ter Page. The upper diagram-excerpt (without Master Page) shows that the elements on the right-hand side have to be replicated in all the containers. On the other hand, the lower picture shows that once such elements are included in a Master Page object, they are automatically included in all the containers of the diagram, eliminating the need for replication.

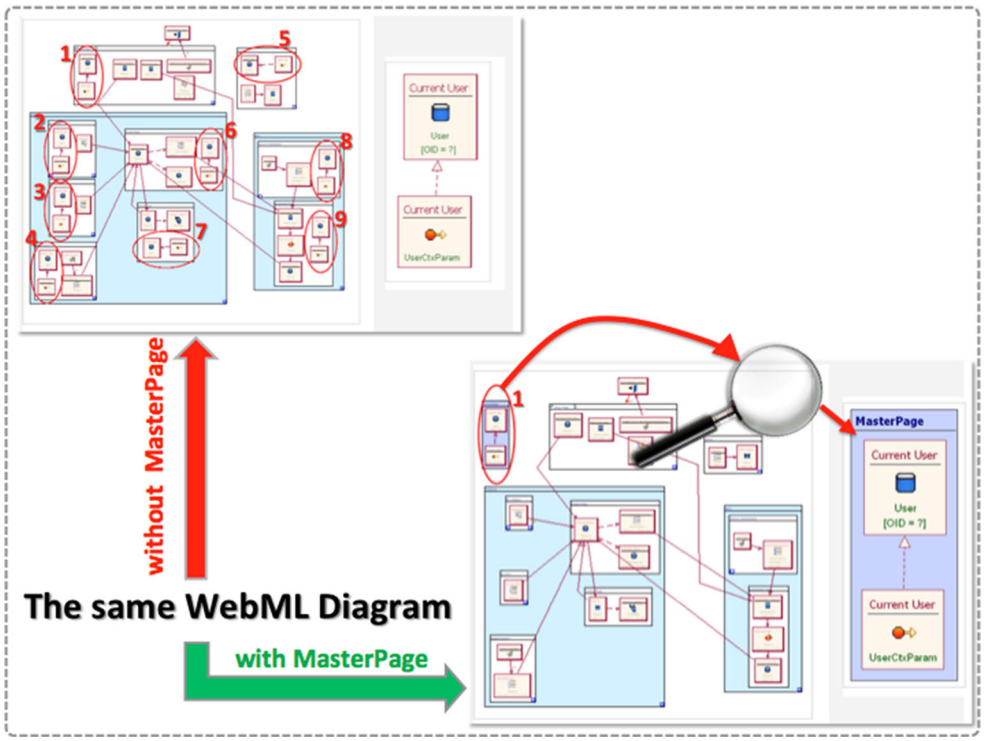


Fig. 18 Inheritance on Master Page elements

4.5.3 Recommendations

There is no recommendation in order to improve WebML symbols according to this principle, given that the tool that implements the language already supports the concepts of modularization and hierarchy.

4.6 Principle of cognitive integration

4.6.1 Description

Cognitive integration only applies when multiple diagrams are used to represent a system. For multi-diagram representations to be cognitively effective, they must include explicit mechanisms to support

- Conceptual integration: Mechanisms to help the reader assemble information from separate diagrams into a coherent mental representation of the system.
- Perceptual integration: Perceptual cues to simplify navigation and transitions between diagrams [39].

4.6.2 Assessment

In the software development process supported by WebRatio, it is possible to have a combination of different models created with UML, E/R, BPMN and WebML. BPMN is used to support the specification of business processes, WebML for specifying the conceptual model of Web applications, and standard E/R or UML class diagrams for specifying the data model. The tool therefore supports these languages. Unfortunately, their visual notations have graphical symbols that share the use of some basic geometric Shapes such as variants of the rectangle, diamonds and simple lines to represent links. For this reason, the combination of UML, E/R, BPMN and WebML models leads to various problems that are principally related to the creation of synographs (the graphical equivalent of synonyms) and homographs (the graphical equivalent of homonyms).

4.6.3 Recommendations

WebML does not leave much room for the analysis of homogeneous integration since scenarios where diagrams of the same type are composed is extremely rare. On the other hand, heterogeneous integration is much more common due to Data and Hypertext models being combined. Unfortunately, the study of the problems related to heterogeneous integration is not within the scope of this work: before making recommendations on the visual notations used by those modelling languages, it would be appropriate to study them from the point of view of the Physics of Notations theory; just as it has been done in this work for WebML.

4.7 Principle of dual coding

4.7.1 Description

According to dual coding theory [47], using text and graphics together to convey information is more effective than using either on their own. When information is presented both verbally and visually, representations of that information are encoded in separate systems in working memory and referential connections between the two are strengthened. This suggests that textual encoding is most effective when it is used in a supporting role: to supplement rather than to substitute for graphics.
Including textual explanations (annotations) can improve understanding of diagrams in the same way that comments can improve understanding of programs. Textual encoding can be used to reinforce and expand the meaning of graphical symbols (hybrid symbols). In the hybrid representation, the text both expands and reinforces the meaning of the graphics [39].

4.7.2 Assessment

The graphical symbols of WebML use a limited amount of text. In particular, they are used by XML-related symbols representing Session Units, to emphasize the XML nature of the objects represented (see Fig. 19a). This is an exam-ple of a hybrid representation. Furthermore, WebML uses text annotations to provide some relevant information about the unit represented by a given symbol. For instance, in the first element of Fig. 19b, a textual annotation informs the user about the title of an Index unit; in the second one, the annotation states which operation is performed by the Math unit; the last one uses the annotation to inform the user about the data items contained in the Multidata unit.

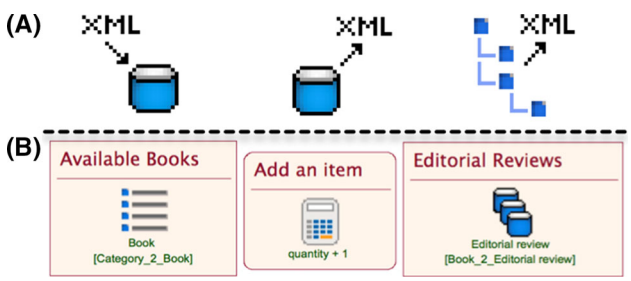


Fig. 19 Dual coding in WebML graphical symbols

4.7.3 Recommendations

We recommend a further study of potential hybrid representations that could be added to WebML visual notation be carried out. The idea would be to eliminate the need for some symbols by using hybrid representations, thereby improving **Graphic Economy**, which is presented in the next section and is mainly based on the total number of different symbols used by the language. The use of more hybrid representations would serve to improve the user's understanding of the annotated symbols. This would be particularly helpful to deal with symbols that are not semantically transparent, as discussed in Sect. 4.4.

4.8 Principle of graphic economy

4.8.1 Description

> Graphic complexity is defined by the number of graphical symbols in a notation: the size of its visual vocabulary [45]. Graphic complexity affects novices much more than experts, as they need to consciously maintain meanings of symbols in working memory. The human ability to discriminate between perceptually distinct alternatives (span of absolute judgment) is around six categories [38]: This defines an upper limit for graphic complexity. Many SE notations exceed this limit by an order of magnitude: For example, UML Class Diagrams have a graphic complexity of over 40. Interestingly, the two most commonly used notations, in practice (DFDs and ER), do satisfy this principle, which may partly explain their longevity and continued popularity in practice. SE notations tend to increase inexorably in graphic complexity over time, primarily due to efforts to increase their semantic expressiveness [39].

4.8.2 Assessment

We have identified that the current *graphic complexity* of WebML is 70. As mentioned in the description of this principle, the human ability to discriminate between perceptually distinct alternatives is around six categories [38], but this number indicates a limit for immediate recall and is not related to the ability of the human mind to understand a graphic design in its entirety. WebML's graphic complex-ity is therefore about 12 times greater than this limit related to immediate recall. However, it seems that a major factor determining a viewer's capacity to make effective use of a diagram is how much that person already knows about the sort of subject matter depicted in the diagram and the specific method of depiction [14]. Therefore, we can say that the correct understanding of a diagram obviously depends on the user's prior knowledge and the assistance of the tool in this context. What is important when calculating the graph-ical complexity of a language is that we can compare it with other visual languages and be aware of the complexity that the generated diagrams could have. Thus, we could consider this obtained number to define visual dialects with a lower graphical complexity and with a different level of detail according to the user's expertise either with the modelling language (or supporting tool) or the domain to target in the model. This reasoning can be done with any visual language that has a number of graphical symbols greater than the limit set by the working memory. Among the few visual languages that conform to this limit are the Data Flow diagrams (complexity of 4) or E/R diagrams (complexity of 5).

4.8.3 Recommendations

The number of graphical symbols used in WebML has evolved and increased over time. In fact, the first implementation of WebML had a *graphic complexity* of twenty [12] but the evolution of the language, mainly due to the development of the supporting IDE, WebRatio, has resulted in a dramatic increase of the *graphic complexity*. The extensibility of the language has also contributed to the increase in its *graphic complexity*. This feature has led to the development of a large number of new Units, which, in order to alleviate the increasing complexity, can be arranged in packages so that each user can decide whether to use them or not.

If this area is to be improved upon, the first step should be to conduct a detailed study on the use of different symbols in common use cases, in which the subjects will be WebRatio users. This study would help to differentiate between the use of the language by novice and expert users [33], in order to propose different modifications to the language that would result in different *graphic complexities* for each type of user. For example, novice users usually develop relatively sim-ple Web applications for which simpler models are needed. In particular, the core of the language, which has a *graph-ical complexity* of thirty, is enough to elaborate such sim-ple models. Therefore, it would be advisable to consider the possibility of providing different sets of WebML elements to deal with the different complexities of different types of applications. This is partially supported in the current ver-sion of WebRatio, since expert users, who are ready to deal with higher levels of graphic complexity, can download dif-ferent packages at will. However, the current version of the tool implies the use of the built-in package whose graphic complexity is higher than that of the hypothetical *core* package.

Another possible way in which to improve the **Graphic Economy** would be to add a *symbol deficit* and do not support the representation of some elements graphically, using tex-

tual annotations to represent them instead. Finally, a further option would be to increase the **Visual Expressiveness** of symbols, thus reducing the number of symbols and increasing the human ability of discrimination.

4.9 Principle of cognitive fit

4.9.1 Description

> Cognitive fit theory states that different representa-tions of information are suitable for different tasks and different audiences. Problem solving performance (which corresponds roughly to cognitive effectiveness) is determined by a three-way fit between the problem representation, task characteristics, and problem solver skills. There are well-known differences in the way experts and novices process diagrams [14,59]. For non-experts, interpretation is slower, more error-prone, and requires conscious effort. The well-documented differences between experts and novices suggest the need for at least two different visual dialects: an expert ("pro") and a novice ("lite") one.
> Another situation that may require different visual dialects is the use of different representational media. In particular, requirements for sketching on whiteboards or paper (an important use of visual notations in early design stages) are different to those for using computer-based drawing tools. In fact, most SE visual notations seem designed for the pre-computer era, as they make little use of the powerful capabilities of modern graphics software: Effectively, they are designed for pencil-and-paper [39].

4.9.2 Assessment

WebML models are abstract and technology independent, i.e. they do not deal with the details of the targeted platform where the application under development will be deployed. These models can therefore be used to capture the requirements, design the information, design the navigation, etc., and their users may therefore be developers, designers or end customers. Moreover, the tool that supports WebML provides different views focused on different aspects (data, presentation, behaviour). These features facilitate the understanding of the models by users with different capabilities.

Another aspect to be considered is the differences involved in representing the language symbols. Textbooks contain fewer symbols that are in B/W, and most of them are different from those supported by WebRatio. Users who study the symbols in books may therefore encounter initial difficulties when using the supporting tool.

4.9.3 Recommendations

The aforementioned differences in abilities to discriminate, understand and analyse between experts and novice users [14] have again led us to think about the need to analyse the possibility of creating visual dialects for various types of users. It would thus be possible to obtain a number of sets of more/less cognitively effective graphical symbols. For novice users who may be end users or business experts, it is important to bear in mind, for example, the need to use more discriminable symbols, to reduce complexity and to simplify visual vocabularies, and it is therefore necessary to enhance principles such as **Graphic Economy**, **Semantic Transparency** and **Perceptual Discriminability**, while for expert users who need more details, it is important to emphasize principles such as **Semiotic Clarity**, **Dual Coding** and **Complexity Management**.

4.10 Summary and discussion

In order to provide an overview of the main results of the analysis, Table 6 enumerates the main findings related to each principle, along with some of the recommendations proposed to improve the visual notation of the language regarding this principle. After analysis and discussion by the authors, among whom we can count one of the creators of the WebML language, we wanted to provide an indicator (compliance), whose aim is to reflect to what extent the language is aligned with each of the principles, in order to provide a quick sum-mary of our assessment of WebML in respect of each of the principles.

As the Recommendations column in Table 6 shows, some principles are highly interconnected and indeed, there are many more interactions of this type. For example, Fig. 20 shows some of these interactions, which, as can be seen, are not symmetric interactions. Some of the most important are:

- **Perceptual Discriminability** increments **Visual Expressiveness**, as it implicates the use of more visual variables.
- Reducing the graphic complexity (**Graphic Economy**) exacerbates the **Visual Expressiveness**, and improving **Visual Expressiveness** positively affects the **Graphic Economy**.
- **Perceptual Discriminability**, **Semantic Transparency** and **Complexity Management** can improve effectiveness for novice users, though **Semantic Transparency** can reduce effectiveness for expert users (**Cognitive Fit**).

It is therefore important to master each of the principles and their possible interactions in order to avoid certain conflicts and exploit certain advantages. Which principle should prevail in each case is a matter of decision making for which no

Table 6 WebML assessment overview

Principle	Assessment	Recommendations	Compliance[a]
Semiotic clarity	Symbol anomalies: redundancy, overload, excess and deficit	Eliminate anomalies	*
Perceptual Discriminability	Few visual variables are used: Shape, Colour and Texture	• Use more visual variables: Size, Location • Use Colour to differentiate categories of Units • Use different Shapes: 3D, ellipses	**
Visual Expressiveness	Information-carrying variables: Location, Shape, Colour and Texture	Exploit the potential of the visual variables and uses that are unused	**
Semantic transparency	Some symbols are semantically opaque and translucent	Define new semantically immediate icons	**
Complexity management	Creation of modules for a set of Units and hierarchical structure	None	***
Cognitive Integration	Combination of different models creates synographs and homographs	Perform a complete and in-depth analysis in future works	–
Dual coding	Textual annotations only in XML symbols and to represent some properties	Add more textual annotations to improve **Graphic Economy**	***
Graphic economy	Graphic complexity of 70	• Different graphic complexity for novice and expert users • Add symbol deficit • Increase **Visual Expressiveness**	*
Cognitive Fit	• Same models for different types of users • Difference in representational media	Visual dialects	**

[a] Legend (for weightable fields): – none, * poor, ** good, *** excellent

Fig. 20 Interaction among some principles (adapted from [39])

generic statement can be made. As a matter of fact, the issue of dealing with the trade-offs among principles, best practices, rules or whichever building block of the framework is inherently associated with the application of assessment frameworks. See for instance, the reflections in this sense made by Petre [50] or Blackwell and Green [7] regarding the trade-offs between dimensions proper of the cognitive dimensions framework.

Finally, it is worth noting that the aim of the analysis was only to detect the main flaws (if any) of WebML's visual notation and to prove that they do exist. While some indications of possible improvements to address these flaws have been given, the goal was not to identify the set of specific refinements that would solve the problems detected. In fact, there is no one specific set of refinements since, as we have just mentioned, the trade-offs between principles make it impossible to provide a complete solution.

5 Empirical validation study

This work subjectively establishes a series of anomalies related to the **Semantic Transparency** of WebML icons. In this section, we present an empirical validation that sup-ports the identification of these anomalies (only those related with Semantic Transparency), which have led us to propose

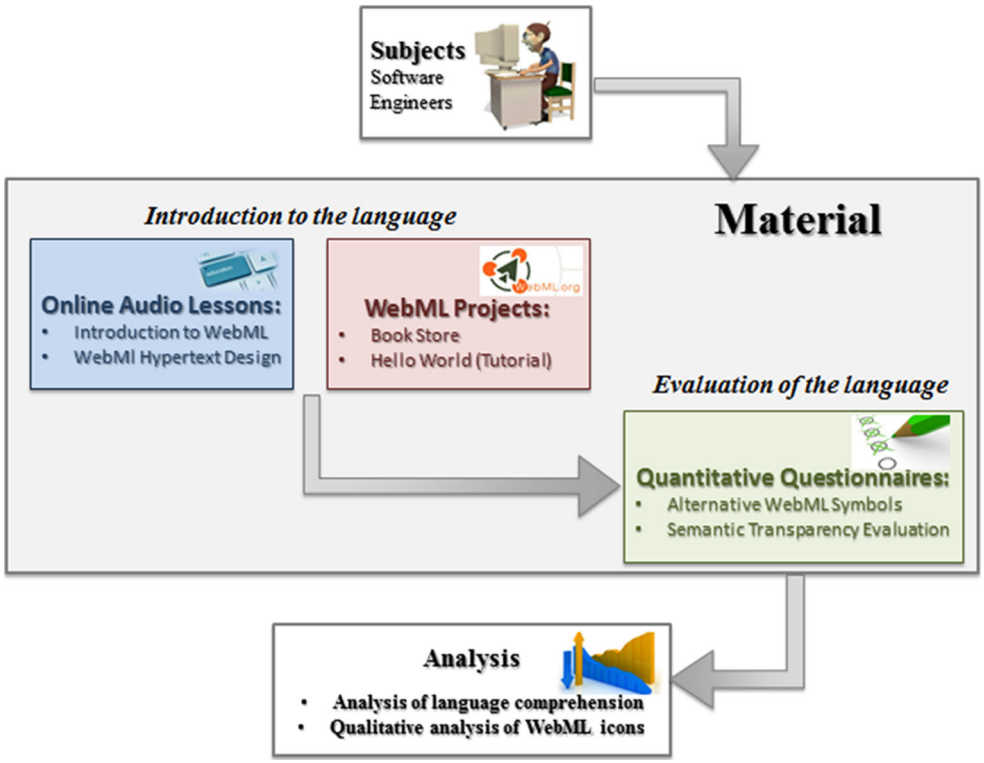


Fig. 21 Summary of the empirical study

a series of new graphical symbols for WebML. This will make it possible for us to validate whether there is room for improvement in the visual notation of the language. We focused on the validation of Semantic Transparency, as it is one of the principles that allows us to obtain several improve-ments that could be implemented, with a certain balance between effort and results achieved, in improving the cog-nitive effectiveness of language. However, it is worth men-tioning that it would be possible to conduct an empirical evaluation of other principles of the theory of Moody, but some of them would require a remarkable effort and the return could be minimal. This is mainly due to the diffi-culty in applying some of the improvements suggested in this study in such a complex and consolidated tool as WebRa-tio. For example, to make improvements to the Cognitive Integration, would mean a re-structuring of the entire archi-tecture of the tool that supports WebML. For this reason, it is worth remembering that we want to encourage this kind of analysis from the early stages of creating a modelling language.

Thus, as in any empirical scientific research, we have followed certain guidelines that establish the need to describe the approach, the materials used, the method and the analysis of the results [53]. The empirical study has been carried out with the collaboration of a group of postgraduate students at the University of Rey Juan Carlos and researchers from the Kybele Research Group.

5.1 Planning

The empirical study was carried out by following the recommendations and templates proposed by Mora et al. [43]. Figure 21 shows an overview of the structure of the study.

This structure consists of three main components: subjects, material and analysis.

- The subjects who collaborated in the study were a team of software engineers who, according to their profiles, could be divided into two different groups: postgraduate students and experts in software engineering.
- The material used consisted of a set of online audio lessons and WebML projects to introduce the language, while two quantitative questionnaires were used as material for assessing the language.
- Finally, two analyses were carried out: one concerning the basic understanding of the language, and other a qualitative analysis related to the Semantic Transparency of certain WebML graphical symbols.

Each of the phases is discussed in greater detail in the following sections.

5.2 Subjects

The subjects who collaborated in the empirical study were 45 software engineers. Their profiles enabled them to be divided

into two different groups, the first of which was composed of 30 students on the Information Management module of the Master's in Information Systems Engineering at the Univer-sity of Rey Juan Carlos (*Students group*), and the second of which was composed of 15 experts in Software Engineering who are part of the Kybele research group (*Software experts group*). The difference in the subjects' profiles has been taken into account in the analysis and presentation of the results, as will be noted in Sect. 5.4.

The requirements for the participants were limited to being software engineers. However, the content of some of the Master's modules (in particular the module on "New trends in Information Systems Engineering") and the previous works of the researchers enabled us to ensure beforehand that all the participants had some previous knowledge of Web engineering and MDE. What is more, all the subjects had previously used Eclipse, the framework over which WebRatio runs.

5.3 Material

The experiment was conducted in one of the computer labs at the University of Rey Juan Carlos. This lab has seventy Dell Optiplex GX280 computers with the WebRatio tool v. 7.0.1 Personal Edition installed.

The material distributed to the participants consisted of basic learning documentation related to the WebRatio tool and the WebML language, a modelling exercise, an example of a complete project and two quantitative questionnaires related to the visual notation of the language. Each of these materials is described in greater detail in the following subsections.

5.3.1 Introduction to the language

Online audio lessons on WebML As learning documentation related to WebRatio and WebML, the participants were provided with two online lessons which are available at the following links:

- http://home.deib.polimi.it/mbrambil/webml/lesson1/
- http://home.deib.polimi.it/mbrambil/webml/lesson3/

The first is an introductory presentation lasting 28 min and dealing with WebML and its implementation in WebRatio. The content of this first presentation is:

- *Advantages of the model approach in Internet development*
- *Why WebML?*
- *What is WebML?*
- *Models: structure, composition, navigation*
- *WebRatio Tool overview*

The second lesson relates solely to the details of the *Navigation/Composition* model, which has been the object of analysis of our work. This presentation lasts 41 min, although it was possible to omit some segments, thus reducing the duration to 33 min. The total initial learning time was therefore approximately 1 h.

WebML projects Once the two online lessons had been completed, we provided all the subjects with two WebML projects in order to have practical contact with the language and the tool that implements it. The subjects have therefore acquired new useful knowledge for our empirical study and for prob-able future empirical studies on the cognitive efficacy of this language.

The first project that we provided them with was a short tutorial to develop a basic "Hello World" Web application. This tutorial allowed the participants to develop a small appli-cation on WebRatio in eight simple steps. This application makes it possible to create a Web page that shows a text chosen by the developer. The eight steps in the tutorial are summarized in "Appendix 1".

After this simple project, each participant was given a document with an example of a full Web application project, which shows how to develop the catalogue for a collection of books. Figure 22 shows an excerpt from the model that represents this Web application.

This last project was much more complex than the previous one, but the goal, as mentioned above, was for the subjects to have practical contact with the visual notation of WebML and thereby obtain an empirical study based on the responses of engineers with a basic knowledge of the language.

5.3.2 Evaluation of the language

Quantitative Questionnaires The last set of materials that each of the participants were provided with were two quantitative questionnaires: one of them contained multiple choice questions while the other contained rating questions. Appendix 2 contains the full questionnaire, which consists of a set of language elements with a brief description and two possible graphical representations: that currently is used by WebRa-tio and a new proposal based on the main findings raised by the analysis presented in Sect. 4. The participants were thus able to choose which of the two graphical symbols they con-sidered to be most convenient to represent the element. The document also contained a space in which the participants were able to specify any comments regarding those symbols. The study of past projects signifies that the participants have seen many of the graphical symbols of the language, but not its complete visual notation. For this reason, and in order to avoid influencing the answers in this respect, we decided to place the graphical symbols in the questionnaire in random order so that if the participant did not know the implemen-

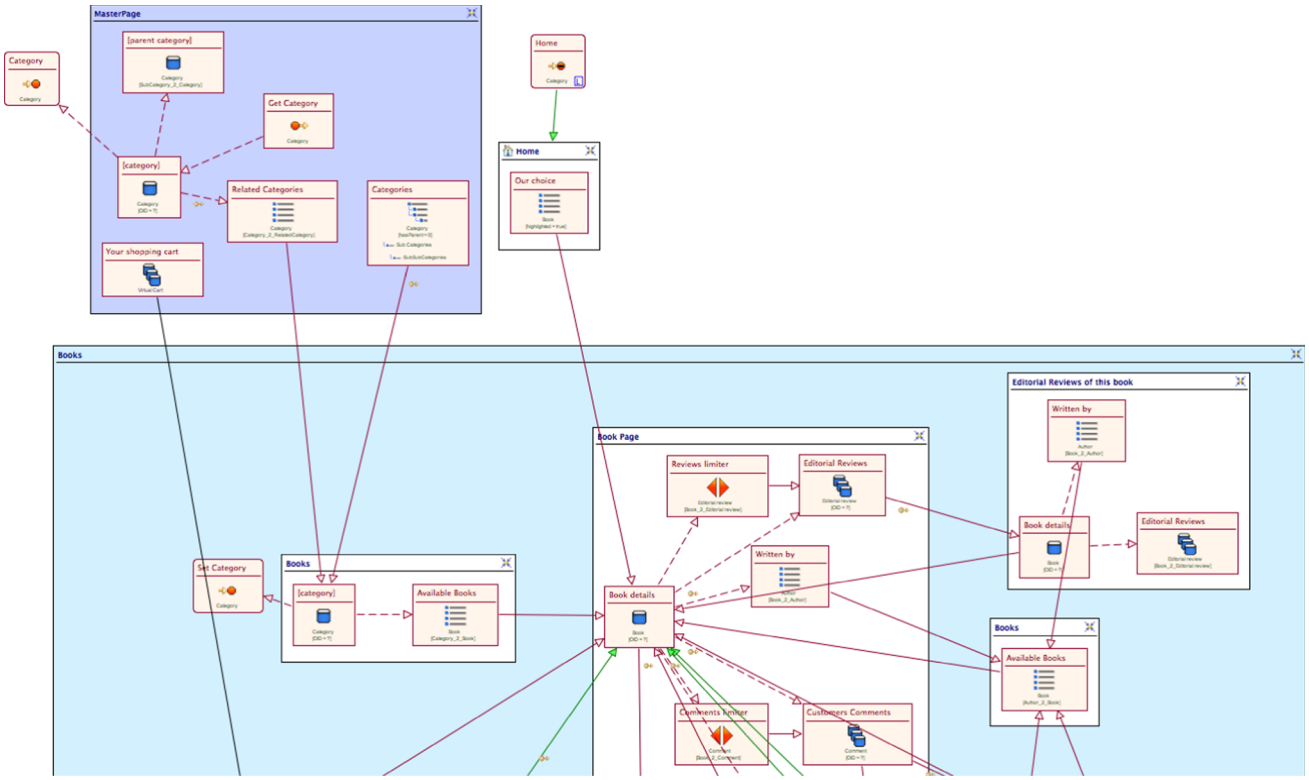


Fig. 22 Excerpt of the navigation/composition model of the book store WebML project

tation language, she/he could not figure out which was the original symbol and which was the proposed improvement.

The second questionnaire contained different icons used in the implementation of WebML. Appendix 3 contains the full questionnaire. For each of the icons proposed, the participants had to specify whether or not they considered the graphical representation of the elements appropriate. They could choose from a range of values from 1 to 5, where one signified highly inadequate, and five signified highly suitable. As with the previous questionnaire, the participants could make comments about their subjective assessments. The comments made when the evaluation of the symbol was negative allowed us to analyse whether there was an arbitrary relationship between the symbol and its meaning, or whether the participant understood the opposite meaning from its graphical representation.

5.4 Results and analysis

5.4.1 Language comprehension

The learning material that the participants were provided with gave them a basic understanding of the WebML language and, more specifically, learning related to the *Navigation/Composition* model. After about an hour of receiving two online lessons about the language, it was found that all the participants were able to develop a simple Web application using WebRatio and that they understood the example of the complex project provided.

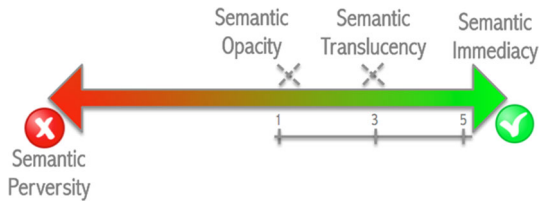


Fig. 23 Degrees of semantic transparency (adapted from [39])

5.4.2 Semantic transparency evaluation

The questionnaire shown in "Appendix 3" allowed the subjects of the empirical study to evaluate 21 graphical symbols used in WebRatio. Each of these symbols may have one of the states shown in the continuum scale of Fig. 23. As mentioned earlier, the four possible states of a graphical symbol in terms of **Semantic Transparency** are:

- *Immediacy*: meaning can be inferred from appearance without explanation
- *Translucency*: appearance provides a cue to meaning

Table 7 Summary of the results of the semantic transparency evaluation questionnaire

Element	Icon	Students group average	SW Experts group average	Semantic transparency assessment
Power Index		1.90	2.67	*Translucent*
Recursive Hierarchical Index		1.80	2.73	*Translucent*
Alphabet		4.33	4.40	*Immediate*
Multi Message		2.53	3.13	*Translucent*
No Op Content		1.47	1.70	*Opaque*
No Op Operation		1.33	1.57	*Opaque*
Connect		4.53	4.73	*Immediate*
Set		3.03	3.53	*Translucent*
Get		2.83	3.73	*Translucent*
Reset		3.10	3.40	*Translucent*
Login		4.33	4.53	*Immediate*
Get XML		3.13	3.60	*Translucent*
Schedule job		3.10	3.40	*Translucent*
Jump		2.17	3.27	*Translucent*
XML In		4.37	4.60	*Immediate*
Parameter collector		2.03	2.53	*Translucent*
Is not null		1.80	2.67	*Translucent*
Selector		1.43	2.53	*Opaque*
Password	***	4.37	4.47	*Immediate*
Script		3.07	3.27	*Translucent*
Query		2.00	3.40	*Translucent*

- *Opacity*: arbitrary relationship between appearance and meaning
- *Perversity*: appearance suggests different or opposite meaning

However, after our analysis of this principle, we consider that none of the WebML graphical symbols has a *Semantic Perversity*. We have therefore limited ourselves to studying the other three possible states.

The main factor in the relationship between a graphical symbol and its meaning is the perceptual resemblance, but there are other influences such as functional similarities, metaphors or cultural associations. Therefore, **Semantic Transparency** cannot be considered a binary state but a continuum state. Thus, from the values obtained from the questionnaires, we estimated the Semantic Transparency of each icon according to the continuum scale of Fig. 23, where approximately values between 1 and 2 represent opacity, values between 2 and 4 indicate translucency, and values above 4 suggest immediacy.

Table 7 shows a summary of the results obtained from the questionnaire related to the **Semantic Transparency** evaluation, in which we asked the participants to assess the icons used by the language and deduce their meaning. The values obtained allowed us to assess whether the meaning of the graphical symbols could be deduced from their appearance and therefore to formalize the concept of naturalness or intuition according to the principle of **Semantic Transparency**.

The results obtained, and a consideration of the average value of each of the elements, allowed us to verify that some of the graphical symbols have anomalies related to *translucency* and *opacity*. More specifically, thirteen graphical sym-

Table 8 Summary of the results of the first questionnaire

	Subjects	Total responses	Choice of the original symbol	Choice of the new symbol
Students group	30	240	27 (11.2 %)	213 (88.8 %)
SW experts group	15	120	11 (9.7 %)	109 (90.3 %)
Total	45	360	38 (10.6 %)	322 (89.4 %)

bols are *translucent*, three are *opaque* and the remaining five symbols can be considered *immediate*.

It should be noted that, in the assessment of all the elements, the average obtained from the *Software Experts Group* is higher than that obtained from the *Students Group*.This signifies that, in general, the software engineering experts have a greater ability to infer the meaning of the element from the appearance of the graphical symbol.

5.4.3 Alternative WebML graphical symbols

The results obtained from the questionnaire concerning the alternative graphical symbols have shown that most of the participants in the study preferred the new ones.

Table 8 sums up the data obtained from this quantitative questionnaire. It shows that the newly proposed symbols are preferred over the ones that WebML is currently using. We have again opted to present the results for each of the groups of subjects separately in order to analyse whether the participants' profiles implies any notable difference. More concretely, 88.8 % of the responses from the participants of the *Students Group* and 90.3 % of responses from the participants in the *Software Experts Group* favoured the new symbols.

In order to assess whether these results were statistically significant, the Stata v12 statistical software[1] was used to run a *t* distribution test, which is one of the probability distributions best suited when there are statistical results from a small number of subjects. The test yields a *p* value, which is the probability that a variable would assume a value greater than or equal to the observed one strictly by chance. Recall that statistical significance is reached when the value of *p* is less than 0.05 ($p < 0.05$).

The *t* distribution test was first used to verify that there were no statistically significant differences between the results obtained by each of group. The test was then run again considering the overall results for the 45 subjects (last line of the Table 8). The following two hypotheses were specified:

- *Null hypothesis* (**Ho**): there is no difference between the two options (former or new) of graphical symbols, i.e. subjects choose between the two options indistinctly.
- *Alternative Hypothesis* (**Ha**): there are differences between the two options of graphical symbols, i.e. subjects choose between the two options differently.

Table 9 *T* distribution test on symbols preferences

	Choice of the new symbol	Choice of the original symbol
Number of total responses	322	38
M	0.89	0.16
SD	0.02	0.06
95 % CI	0.86–0.92	0.04–0.27
t test	$p < 0.0000$	

M mean, *SD* standard deviation, 95 % *CI* 95 % confidence interval

Taking into account the considerations presented above, Table 9 shows the data obtained from running the test.

The test yields a *p* value $0.0000 < 0.05$. We can then reject the null hypothesis and conclude that there are differences in the choice of symbols.

6 Lessons learnt

As mentioned in the introduction, the main goal of this paper was not only to assess the visual notation of WebML but also to foster the interest in using a scientific basis to design, evaluate, improve and compare visual notations.

This section gathers some reflections and conclusions extracted from the development of the analysis. The assump-tion is that they may help researchers (in particular MDE practitioners) willing to consider these kinds of quality fea-tures when developing the modelling languages bundled in their proposals. Note that this is a line in which much work needs to be done, since so far MDE proposals used to be authored by developers with a technical background but with-out in-depth experience of human–computer interaction or cognitive issues [58].

These conclusions have been structured into three different groups from most to least generic according to their scope of application: those related to the assessment of a DSL visual notation; those related to the use of a particular framework (Physics of Notations) for the assessment of visual notations; and finally, those related to potential improvements to WebML.

[1] http://www.stata.com/.

6.1 On the assessment of a DSL visual notation

While some similar studies exist which apply the Physics of Notations theory for the assessment of a language's visual notation, the subjects of analysis are well-established languages such as UML, BPMN or i*. Note that all of these languages were defined some time ago; they enjoy certain levels of adoption; and they are *just* modelling languages in the sense that they were not intended to be used for code generation or subsequent transformations. Our reflection in this sense is that the visual notations of these languages have been analysed for two main reasons: there has been time to do so and they are popular languages.

However, little attention has been paid so far to perform this type of analysis with immature DSLs, which are mainly intended to define models that will be subsequently transformed or used as input for code generation. These are the type of languages which constitute the basis of the huge num-ber of model-based proposals that have emerged during the last number of years under the wing of MDE.

In this sense, this analysis serves to show that it is not only feasible to perform this type of analysis in order to identify potential flaws for any given (visual) DSL, but also that it is feasible to do so while keeping a reasonable balance in terms of time and effort. This is particularly relevant bearing in mind that current practices around MDE are more oriented towards the use of small and focused DSLs, such as WebML, than to the use of "macro" modelling languages, such as UML [17].

By contrast, another relevant lesson learnt from this expe-rience for MDE practitioners, who produce new modelling languages regularly, is that this type of reflection, on whether the language is cognitively efficient, usable, etc., should be taken into consideration from the early stages of the development, since doing so once the proposals have been implemented and distributed is either impossible or at best requires too much effort. For instance, in the particular case of WebML, turning the conclusions about WebML's visual notation gathered from the analysis performed in this paper into real actions over WebRatio would imply producing and distributing a widely refined version of WebRatio, an industrial tool that is now at the core of all the Web develop-ment efforts of a good number of IT organizations. Therefore, it would be advisable to consider these aspects from the early stages of the development of new DSLs, so that good decisions related to cognitive effectiveness would be translated throughout the different stages of the develop-ment until the working implementation. Once the DSLs have already been developed and distributed, think about the impact that the evolution of metamodels has over the ecosystem of related models and transformations when the traditional metamodel-driven approach for the development of DSLs has been adopted. By contrast, if the notation of WebML would have been defined taking Moody's, Green and Petre's or whichever set of principles for good designs in to consideration, this *good* design would have been translated into WebRatio's implementation. As a matter of fact, a similar study on IFML's visual notation according to the Physics of Notations theory is being performed by us in order to act either over the language or over its implementation now that they are still emerging proposals.

Another of the ideas presented by researchers using the Physics of Notations theory or any other similar proposal is that the result of the assessment process is by no means enough to assert that the language hasn't any problems. In some sense, the idea is similar to that guiding software testing: 100 % code coverage is either not feasible or requires too much effort. Note that this problem would arise despite which assessment framework is used. In fact, studies on the Cog-nitive Dimensions framework acknowledged the very same problem: Cognitive Dimensions are not useful as a tool for acceptance [16]. Of course, there are ways to improve the *code coverage*, such as performing two different analyses: one against the Physics of Notations theory and another one applying Cognitive Dimensions. A set of empirical studies with final users would be even better in terms of ensuring that the language fulfils users' expectations. However, *who*, *when* and *how* the language is used has also been acknowl-edged to influence greatly the finding of problems [16]. So that the set of studies that should be accomplished in order to reach certain levels of *coverage* would simply do not make for the effort needed to perform such studies. In this context, the use of the Physics of Notations theory, Cognitive Dimen-sions or whichever similar assessment framework constitutes a practical solution which keeps a balance between the effort required and the results.

6.2 On the use of the Physics of Notations

A number of reflections follow on from the experiences gath-ered when using a particular framework, the Physics of Nota-tions theory, for the analysis of WebML.

The last reflection made on the previous section stated that every proposal has its own flaws. In the case of the Physics of Notations, the fundamental flaw is probably that its approach does not consider the language as a whole but as set of nota-tion elements which are analysed in an isolated way (see the discussion that ensued on the lack of support for the assess-ment of combination rules at the end of Sect. 3.1). This non-integrated view might result in cognitively ineffective notations, even if only Moody's principles were considered when designing the notation. By contrast, Moody's princi-ples should be considered along with other factors that might have a negative or positive influence on the notation being cognitively effective. Some of them have been mentioned throughout the paper, such as similarity with existing stan-

dards, previous knowledge of the domain, familiarity with similar notations, which might contribute to improving the cognitive effectiveness of the notation even when they were not fully aligned with Moody's principles.

On the other hand, the main lesson learnt from our experience is that we have found the Physics of Notation theory as a suitable tool for the assessment of DSLs' visual notations. As MDE practitioners ourselves, who have never been concerned about the type of features analysed in this work, it has been relatively easy to be able to generate checklists as well as to define procedures, metrics and numerical indicators to assess these features, identify potential flaws and even propose refinements for those elements that leave room for improvement.

In fact, Moody's work has positively influenced MDE practitioners to start thinking about whether the languages they develop as part of their proposals can be efficiently processed and used by external users or whether their propos-als are somehow aligned with certain quality criteria. Despite the fact that the number of studies has not increased signifi-cantly, it has been found that more and more works have at least mentioned the need to consider some way of ensuring that the visual syntaxes should consider Moody's principles [10].

However, most of the existing analyses of modelling languages according to Moody's principles have been authored either by Moody or by some of his co-authors. There are still very few works using this theory to improve the result of their proposals. By contrast, the authors of this paper have no previous experience at all with HCI or cognitive effectiveness issues since we were merely MDE practitioners, and these were our first steps into these areas. We believe that this work could serve then to foster the interest of MDE practitioners in addressing this type of analysis based on the use of Moody's proposal. As mentioned before, they will not be able to ensure that the languages they develop are completely correct but at least they could discard some particular issues which are frequently suffered by the DSLs that have been emerging recently. Besides, we consider that the fact that this analysis has been authored by researchers not related with Moody's proposal serves also to provide a kind of pragmatic and holistic validation of the proposal in terms of usability.

Furthermore, every conclusion derived from the analysis of a language's visual notation should later be subject to the consideration of domain experts. For instance, as we have mentioned in Sect. 4.3, most of the visual notations used by existing modelling languages might be considered deficient under the principle of Visual Expressiveness, due to the defi-cient use of the Shape visual variable, typically limited to the use of rectangles or some variation. According to some stud-ies on HCI, the use of geons (simple 2D or 3D forms) instead of boxes and arrows diagrams would contribute to improve user perception since structural representations of objects in the brain are acknowledged to be composed of geon objects and relationships [6]. Nevertheless, the very same authors acknowledge that the use of geons for diagramming tasks raises a number of issues, such as the labelling of nodes and relationships and the corresponding impact on the layout, that may result in unmanageable diagrams when a certain amount of information is to be displayed [28]. Besides, the importance of final users' habits should not be dismissed: even though geons could be perceived better by the average human brain, SE modellers are used to boxes and arrows dia-grams. Therefore, at the time of dealing with a new notation, they will feel much more comfortable if is based on abstrac-tions with which they are used to working with on a daily basis. This is the case of WebML, where newcomers are usually familiarized with some SE notation, such as UML or BPMN.

We have also found that the main limitations of the frame-work are related with the lack of support to analyse the com-position rules of the visual notation and the need for empirical studies with which the findings and predictions produced by the analysis could be contrasted. In this sense, a clear contribution would be the definition of techniques and processes specifically designed to conduct these studies according to the particular features of the Physics of Notations theory.

Apart from the above reflections, some concrete findings gathered from our experience using the Physics of Notations follows.

We have found that the tasks which are more easily accomplished among those found in the application of Moody's proposal are: identifying anomalies between the elements of the language and its graphic symbols (Semiotic Clarity); determining the visual distance between the graphical symbols of the language (Perceptual Discriminability); computing the saturation of visual variables (Visual Expressiveness); verifying the appropriate use of hierarchy and modularity relationships (Complexity Management); and deriving graphic complexity (Graphic Economy).

By contrast, we have found the more complex tasks to be identifying the relationships between graphical symbols and their meaning (Semantic Transparency); assessing the proper integration of the different types of diagrams (Cognitive Integration); checking the use of text in graphical symbols (Dual Coding); thinking of different visual dialects to target different audiences (Cognitive Fit).

Finally, another relevant lesson learnt is that, accord-ing to our own experience, there are some principles over which it is more feasible to act, i.e. for which little effort would be required to improve the language according to that principle. In particular, the re-distribution of the visual variables between the graphical symbols can significantly improve the visual distance between them, contributing to a faster graphic understanding, thus improving Perceptual

Discriminability; an appropriate use of the capacity of each visual variable would result in much better saturation values, which contribute in terms of Visual Expressiveness; Semantic Transparency would benefit from the refinement of opaque, translucent and perverse symbols in order to facilitate the derivation of the meaning of a symbol from its representation; lastly, decreasing the number of graphical symbols used by the language would reduce Graphic Complexity, thereby improving the cognitive recognition process.

6.3 On WebML improvements

As it has been mentioned a number of times throughout this work, most of the improvements derived from the findings of the analysis performed should have been carried out in the early stages of WebML development. Now that it has become a well-established and distributed product, most of them are just unaffordable.

Nevertheless, we would like to conclude this section by highlighting those that we have found to be the most reasonable steps in order to increase the cognitive effectiveness of WebML at a balanced cost

- To get rid of the symbol overload flaws by modifying those graphical symbols which are too similar and yet represent different elements of the language.
- To support the re-sizing of any graphical symbol.
- To increase the saturation of some visual variables, such as Location, Texture and Brightness.
- To address the issues detected with translucent graphical symbols according to the empirical study summarized in this paper.
- To lower the graphic complexity of the language by creating two built-in packages, basic and advanced, to target at least two types of users (novice and advanced users).

7 Conclusion and further work

In this work, we have carried out an analysis of the cognitive effectiveness of the WebML visual notation. The definition of cognitive effectiveness involves different objectives which may contradict each other. Most of the problems in this respect are caused by a lack of theoretical and scientific prin-ciples with which to define, improve, evaluate or compare visual notations.

In this analysis, we have used the Physics of Notations theory [39], which establishes a set of principles based on theory and empirical evidence that can be used to assess the cognitive effectiveness of visual notations.

The data obtained has allowed us to detect various problems and provide certain recommendations on how to improve the visual notation of the language. Moreover, the analysis will serve to foster the discussion on other possible improvements. Another relevant conclusion is the fact that, as occurs with almost every modelling language, there cannot be an ideal language for all types of users. A thorough analysis would therefore be required to determine the possible visual dialects that WebML could provide to become more user-friendly for different types of users.

What is more, in order to be able to analyse the WebML visual notation, we have analysed the implementation of the language made by WebRatio, which in turn could be considered as the tool that proves the core value of MDE. Since models and modelling languages are the core building blocks of any MDE proposal, and since most of them are graphical, MDE may constitute a scenario in which the already acknowledged impact of the cognitive effectiveness of visual notations reaches new limits. To date, little attention has been paid by the MDE community to formal or scientific ways in which to define modelling languages. In particular, few analysis of the cognitive effectiveness of modelling lan-guages have been made, since the issues related to visual syn-tax have historically been undervalued. Nevertheless, now that technology is achieving certain levels of maturity, it is time to start considering these issues. This work therefore aims to foster the interest in the topic and provide new point-ers for future works in the area.

It should be noted that we have verified that there is room for improvement in the WebML visual notation, and it was for this reason that we wished to carry out an empirical study to verify the validity of these proposed improvements. However, now that some data have been gathered as regards the extent to which WebML is aligned with the Physics of Notation theory, we are ready to address the development of more experiments that will allow us to identify exactly which improvements would be most welcomed by different types of users.

Finally, the other direction for future work consists of translating the improvements made to WebML to the Interaction Flow Modelling Language (IFML) [27], formally adopted as a standard by the OMG last March, which covers similar objectives to those of WebML, by which it was greatly inspired. We therefore intend to carry out a complete analysis of IFML cognitive effectiveness based on the main findings about and improvements to WebML.

Acknowledgements This research has been partially funded by the Regional Government of Madrid under project SICOMORo-CM (S2013/ICE-3006) and by the MASAI (TIN-2011-22617) and ELASTIC (TIN2014-52938-C2-1-R) projects, financed by the Spanish Ministry of Science and Innovation. We want to thank the students on the Information Management module of the Master's in Information Systems Engineering at the University of Rey Juan Carlos and the researchers who are part of the Kybele research group, for their cooperation in conducting the tests to carry out the empirical study that is part of this work.

Appendix 1

WebML Project—Hello World!

Steps	Brief description
1. Open the WebRatio perspective	Select Window > Open Perspective > WebRatio
2. Create a Web Project	File > New > Web Project item. Enter "Hello World" for the project name, then click Finish
3. Create a New Site View	In the work area, right-click and choose the Add Site View command. Enter "Hello World" for the site view name, then click Finish
4. Add a Page	Select the Page icon from the left side palette and then click in the site view work area in which you wish to place the page
5. Editing Page Properties	Edit the Page properties in the Properties View. Enter "Hello World!" for the page name and check the Home property to mark the new page as the home page of your site view
6. Add a unit	Add a unit to the page. Select the multi message unit from the palette and then left click on to the page
7. Editing unit Properties	Enter "Hello World!" for the unit name. Click on the Edit button next to the Default message property to set the default message to be shown. Write "Hello World!"
8. Generate your Application	Click the Generate Full Project button on the main toolbar. Start Tomcat, then open your browser and type the following url: http://www.localhost:8080/HelloWorld

Appendix 2

Questionnaire 1

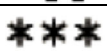
Evaluation of alternatives WebML graphical symbols: use the values 1 or 2 to choose the graphical symbol you consider most appropriate for the language element:

Element	Brief Description	Graphic Symbol 1	Graphic Symbol 2	Choice		Comments
Data	It publishes a single object of a given entity.			1	2	
Event Calendar	It shows a perpetual calendar, possibly enhanced with the list of events assigned to each day of the months			1	2	
Power Index	It provides commands to scroll through the available objects and to dynamically order the shown instances.			1	2	
Multi Message	It lets the user print out messages on the page.			1	2	
Script	It is designed to execute a block of arbitrary code.			1	2	
Math	It is designed to parse a mathematical expression, replace a set of operands, evaluate the expression, and propagate the result as an output parameter.			1	2	
Schedule Job	This Unit permits to schedule a job			1	2	
Jump	This Unit permits to jump forward and backward within the application without modelling an explicit link.			1	2	

Appendix 3

Questionnaire 2

Evaluation of some WebML graphical symbols: use the values of 1 (highly inappropriate) to 5 (highly appropriate) to evaluate the graphical symbols presented below:

Element	Brief Description	Graphical Symbol	Evaluation	Comments
Power Index	It provides commands to scroll through the available objects and to dynamically order the shown instances.		1 2 3 4 5	
Recursive Hierarchical Index	It shows a hierarchy of objects belonging to an entity.		1 2 3 4 5	
Alphabet	It searches the instances of an entity and shows the starting characters of a specific attribute.		1 2 3 4 5	
Multi Message	It lets the user print out messages within the page.		1 2 3 4 5	
No Op Content	No Op Content is a placeholder for a content unit with no business logic.		1 2 3 4 5	
No Op Operation	No Op Operation is a placeholder for an operation unit with no business logic.		1 2 3 4 5	
Connect	A connect unit creates new instances of a relationship.		1 2 3 4 5	
Set	A set unit assigns a value to a global parameter.		1 2 3 4 5	
Get	A get unit retrieves the value of a global parameter.		1 2 3 4 5	
Reset	Reset units allow removing session parameters from the session context.		1 2 3 4 5	
Login	The login unit verifies the identity of a user accessing the site.		1 2 3 4 5	
Get XML	A Get XML Unit retrieves XML content from a local or remote URL.		1 2 3 4 5	
Schedule Job	This Unit permits to schedule a job.		1 2 3 4 5	
Jump	It permits to jump forward and backward within the application without modelling an explicit link.		1 2 3 4 5	
XML In	An XML In Unit is able to transform an XML document into relational data.		1 2 3 4 5	
Parameter Collector	It permits to collect a set of parameters coming from multiple sources and to redistribute them to multiple targets.		1 2 3 4 5	
Is Not Null	It checks the value of its unique input parameter and to follow either its outgoing to OK or KO link.		1 2 3 4 5	
Selector	It specifies the logic of a selector in a hypertext without the necessity to display in the page the extracted values.		1 2 3 4 5	
Password	The password unit is able to randomly generate a password having a specified length.		1 2 3 4 5	
Script	It executes a block of arbitrary code.		1 2 3 4 5	
Query	It permits to execute a custom edited query using the HQL or SQL query language.		1 2 3 4 5	

References

1. Acerbis, R., Bongio, A., Brambilla, M., Butti, S.: WebRatio 5: An eclipse-based CASE tool for engineering web applications. In: ICWE, pp. 501–505 (2007)
2. Acerbis, R., Bongio, A., Brambilla, M., Butti, S., Ceri, S., Fraternali, P.: Web applications design and development with WebML and WebRatio 5.0. TOOLS (46), 392–411 (2008)
3. Baar, T.: Correctly defined concrete syntax for visual modeling languages. In: MoDELS 2006, pp. 111–125 (2006)
4. Bar, M., Neta, M.: Humans prefer curved visual objects. Psychol. Sci. **17**(8), 645–648 (2006)
5. Bertin, J.: Semiology of Graphics: Diagrams, Networks, Maps. University of Wisconsin Press, Madison (1983)
6. Biederman, Irving: Recognition-by-components: a theory of human image understanding. Psychol. Rev. **94**(2), 115–147 (1987)
7. Blackwell, A.F., Green, T.R.G.: Notational systems—the cognitive dimensions of notations framework. In: Carroll, J.M. (ed.) HCI Models Theories and Frameworks, pp. 103–134. Morgan Kaufmann, San Francisco (2003)
8. Bottoni, P., Grau, A.: A suite of metamodels as a basis for a classification of visual languages. In: VL/HCC 2004, pp. 83–90 (2004)
9. Bottoni, P., Quatrocchi, P., Ventriglia, D.: Constraining concrete syntax via metamodel information. In: VL/HCC 2006, pp. 85–88 (2006)
10. Brambilla, M., Cabot, J., Wimmer, M.: Model-driven software engineering in practice. Morgan & Claypool, USA (2012)
11. Brambilla, M., Fraternali, P.: Large-Scale Model-Driven Engineering of Web User Interaction: The WebML and WebRatio experi-ence. Science of Computer Programming. Elsevier, Amsterdam (2013)
12. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, Los Altos (2002)
13. Chen, P.: The entity-relationship model—toward a unified view of data. ACM Trans. Datab. Syst. **1**(1), 9–36 (1994)
14. Cheng, P.C., Lowe, R.K., Scaife, M.: Cognitive science approaches to understanding diagrammatic representations. Artif. Intell. Rev. **15**(1/2), 79–94 (2001)
15. Clark, T., Evans, A., Sammut, P., Willans, J.: Applied Metamodelling—A Foundation for Language Driven Development, 2nd edn. http://itcentre.tvu.ac.uk/~clark/book.html (2008)
16. Dagit, J., Lawrance, J., Neumann, C., Burnett, M., Metoyer, R., Adams, S.: Using cognitive dimensions: advice from the trenches. J. Vis. Lang. Comput. **17**, 302–327 (2006)
17. Davies, J., Gibbons, J., Welch, J., Crichton, E.: Model-driven engineering of information systems: 10 years and 1000 versions. Sci. Comput. Program. **89B**, 88–104 (2013)
18. Fish, A., Störrle, H.: Visual qualities of the modeling language: deficiencies and improvements. IEEE symposium on visual languages and human-centric computing. VL/HCC 2007, pp. 41–49 (2007)
19. Fondement, P., Baar, T.: Making metamodels aware of concrete syntax.In: ECMDA-FA 2005, pp. 190–204 (2005)
20. Genero, M., Poels, G., Piattini, M.: Defining and validating metrics for assessing the understandability of entity-relationship diagrams. Data Knowl. Eng. **64,3**, 534–557 (2008)
21. Genon, N., Heymans, P., Amyot, D.: Analysing the cognitive effectiveness of the BPMN 2.0 visual notation. In: Software Language Engineering. Lecture Notes in Computer Science, vol. 6563, pp. 377–396 (2011)
22. Genon, N., Amyot, D., Heymans, P.: Analysing the cognitive effectiveness of the UCM visual notation. In: System Analysis and Modeling: About Models, Lecture Notes in Computer Science, vol. 6598, pp. 221–240 (2011)
23. Goodman, N.: Languages of Art: An Approach to a Theory of Symbols. Bobbs-Merrill, Indianapolis (1968)
24. Green, T.R.G., Petre, M.: Usability analysis of visual programming environments: a 'cognitive dimensions' framework. J. Vis. Lang. Comput. **7**, 131–174 (1996)
25. Green, T.R.G.: Cognitive dimensions of notations. In: Sutcliffe, A., Macaulay, L. (eds.) People and Computers, vol. 5, pp. 443–460. Cambridge University Press, Cambridge (1989)
26. Green, T., Blandford, A., Church, L., Roast, C., Clarke, S.: Cognitive dimensions: achievements, new directions, and open questions. J. Vis. Lang. Comput. **17**, 328–365 (2006)
27. Interaction Flow Modeling Language (IFML) beta specification. OMG Document: http://www.omg.org/cgi-bin/doc?ptc/13-03-08
28. Irani, P., Ware, C.: Diagramming information structures using 3D perceptual primitives. ACM Trans. Comput. Hum. Interact. **10**(1), 1–19 (2003)
29. Krogstie, J., Sindre, G., Jorgensen, H.: Process models representing knowledge for action: a revised quality framework. Eur. J. Inf. Syst. **15**, 91–102 (2006)
30. Larkin, J.H., Simon, H.A.: Why a diagram is (sometimes) worth ten thousand words. Cogn. Sci. **11**(1), 65–100 (1987)
31. Lindland, O.I., Sindre, G., Sølvberg, A.: Understanding quality in conceptual modelling. IEEE Softw. **11**(2), 41–49 (1994)
32. Lohse, G.L.: A cognitive model for understanding graphical perception. Hum. Comput. interact. **8**(4), 353–388 (1993)
33. Lohse, G.L.: The role of working memory in graphical information processing. Behav. Inf. Technol. **16**(6), 297–308 (1997)
34. Mackinlay, J.: Automating the design of graphical presentations of relational information. ACM Trans. Graph. **5**(2), 110–141 (1986)
35. Masri, K., Parker, D., Gemino, A.: Using iconic graphics in entity relationship diagrams: the impact on understanding. J. Datab. Manag. **19**(3), 22–41 (2008)
36. Mens, T., Wermelinger, M., Ducasse, S., Demeyer, S., Hirschfeld, R., Jazayeri, M.: Challenges in software evolution. In: IWPSE, pp. 13–22. Lisbon, Portugal (2005)
37. Mernik, M.: When and how to develop domain-specific languages. ACM Comput. Surv. (CSUR) **37**(4), 316–344 (2005)
38. Miller, G.A.: The magical number seven, plus or minus two: some limits on our capacity for processing information. Psychol. Rev. **63**, 81–97 (1956)
39. Moody, D.: The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**(6), 756–779 (2009)
40. Moody, D., van Hillegersberg, J.: Evaluating the visual syntax of UML: an analysis of the cognitive effectiveness of the UML family of diagrams. In: LNCS, vol. 5452, pp. 16–34. Springer (2009)
41. Moody, D.L.: Review of ArchiMate: the road to international standardisation. Technical report, Report commissioned by the ArchiMate Foundation and BiZZDesign B.V., Enschede (2007)
42. Moody, D.L., Heymans, P., Matulevicius, R.: Improving the effectiveness of visual representations in requirements engineering: an evaluation of i* visual syntax. In: Proceedings of the 17th IEEE International Requirements Engineering Conference (RE 2009), pp. 171–180 (2009)
43. Mora, B., García, F., Ruiz, F., Piattini, M.: Graphical vs textual software measurement modelling: an empirical study. Softw. Qual. J. **19**(1), 201–203 (2011)
44. Morris, S.J., Gotel, O.C.Z.: Flow diagrams: rise and fall of the first software engineering notation. In: Diagrammatic Representation and Inference. Lecture Notes in Computer Science, vol. 4045, pp. 130–144 (2006)
45. Nordbotten, J.C., Crosby, M.E.: The effect of graphic style on data model interpretation. Inf. Syst. J. **9**, 139–156 (1999)

46. Opdahl, A.L., Henderson-Sellers, B.: Ontological evaluation of the UML Using the Bunge–Wand–Weber model. Softw. Syst. Model. **1**(1), 43–67 (2002)
47. Paivio, A.: Mental Representations: A Dual Coding Approach. Oxford University Press, Oxford (1986)
48. Patrignani, M.: Visualization of Large Graphs. Dottorato di Ricerca (doctoral dissertation), IngegneriaInformatica. Univ. degliStudi di Roma (2003)
49. Petre, M.: Why looking isn't always seeing: readership skills and graphical programming. Commun. ACM **38**(6), 33–44 (1995)
50. Petre, Marian: Cognitive dimensions 'beyond the notation'. J. Vis. Lang. Comput. **17**(4), 292–301 (2006)
51. Schuette, R., Rotthowe, T.: The guidelines of modeling: an approach to enhance the quality in information models. In: LNCS, vol. 1507, pp. 240–254. Springer, Heidelberg (1998)
52. Selic, B.: The pragmatics of model-driven development. IEEE Comput. **20**(5), 19–25 (2003)
53. Shull, F., Carver, J.C., Vegas, S., Juzgado, N.J.: The role of replications in empirical software engineering. Empir. Softw. Eng. **13**(2), 211–218 (2008)
54. Siau, K., Cao, Q.: Unified modeling language: a complexity analysis. J. Datab. Manag. **12**(1), 26–34 (2001)
55. Störrle, H., Fish, A.: Towards an operationalization of the "Physics of Notation" for the analysis of visual languages. In: Moreira, A., Schätz, B., Gray, J., Vallecillo A., Clarke, P. (eds.) MoDELS. Lecture Notes in Computer Science, vol. 8107, pp. 104–120. Springer, Berlin, Heidelberg (2013)
56. Weber, R.A.: Ontological Foundations of Information Systems (Coopers and Lybrand Accounting Research Methodology Monograph No. 4). Coopers and Lybrand, London (1997)
57. Wheildon, C.: Type and layout: are you communicating or just making pretty shapes?. Worsley Press, Australia (2005)
58. Whittle, J., Hutchinson, J., Rouncefield, M.: Industrial adoption of model-driven engineering: are the tools really the problem? In: Moreira, A., Schätz, B., Gray, J., Vallecillo, A., Clarke, P. (eds.) Model-Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 8107, pp. 1–17. Springer (2013)
59. Winn, W.D.: An account of how readers search for information in diagrams. Contemp. Educ. Psychol. **18**, 162–185 (1993)
60. Winn, W.D.: Encoding and retrieval of information in maps and diagrams. IEEE Trans. Prof. Commun. **33**(3), 103–107 (1990)
61. Wouters, L., Gervais, M.-P.: Notation-driven vs metamodel-driven development of domain-specific modeling languages. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing—SAC'13 (p. 1145). ACM Press, New York (2013)
62. Zhang, J., Norman, D.A.: Representations in distributed cognitive tasks. Cogn. Sci. **18**(1), 87–122 (1994)