Journal of Visual Languages and Computing 21 (2010) 359–373

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

Cognitive effectiveness of visual instructional design languages

Kathrin Figl [a,*], Michael Derntl [b], Manuel Caeiro Rodriguez [c], Luca Botturi [d]

[a] WU—Vienna University of Economics and Business, Institute for Information Systems & New Media, UZAII, Augasse 2-6, A-1090 Vienna, Austria
[b] University of Vienna, Faculty of Computer Science, Austria
[c] Department of Telematic Engineering, University of Vigo, Spain
[d] Dipartimento Formazione e Apprendimento—SUPSI, Switzerland

ARTICLE INFO

Keywords:
Visual design languages
Cognitive effectiveness
Instructional design
Visual notations
E^{2}ML
PoEML
CoUML

ABSTRACT

The introduction of learning technologies into education is making the design of courses and instructional materials an increasingly complex task. Instructional design languages are identified as conceptual tools for achieving more standardized and, at the same time, more creative design solutions, as well as enhancing communication and transparency in the design process. In this article we discuss differences in cognitive aspects of three visual instructional design languages (E^{2}ML, PoEML, coUML), based on user evaluation. Cognitive aspects are of relevance for learning a design language, creating models with it, and understanding models created using it. The findings should enable language constructors to improve the usability of visual instructional design languages in the future. The paper concludes with directions with regard to how future research on visual instructional design languages could strengthen their value and enhance their actual use by educators and designers by synthesizing existing efforts into a unified modeling approach for VIDLs.

1. Introduction

When an architect is in charge of designing a new house, s/he usually starts – right after what an engineer would refer to as a requirements analysis – with some sketch about the division and uses of the available space. The architect would then refine this and translate the design solution into a visual representation that the client could see, understand and discuss, and then into some executive plans that s/he would hand out to the construction staff. Architects exploit a number of such visual representations as part of the process of analyzing design problems, thinking about solutions, and communicating with stakeholders and other partners. Examples include blueprints, structural drawings, electrical wiring schemas, and three-dimensional displays of the house. The ability to use such representations, as part of their design language, is very important for architects—as it is for industrial and graphic designers, software architects and designers, musicians, and for all those involved in a design activity with a long tradition.

For instructional designers — architects of learning environments — using a visual instructional design language (VIDL) for modeling different aspects of courses involving the use of new media, has similar advantages. The contemporary rise of new, advanced learning technologies such as e-learning, mobile learning, serious gaming, and simulations — often in combination with the introduction of "new learning" models such as problem-based learning, case-based learning, competency-based learning, etc. — has significantly increased the complexity of teaching and learning processes [1]. This requires more advanced design and development processes in which communication is supported by the use of shared design languages that are detailed and formal. In response, VIDLs for instructional designers and

[*] Corresponding author. Tel.: +43 1 31336 4467; fax: +43 1 31336 90 4467 3.
E-mail addresses: kathrin.figl@wu.ac.at (K. Figl), michael.derntl@univie.ac.at (M. Derntl), mcaeiro@det.uvigo.es (M.C. Rodriguez), luca.botturi@usi.ch (L. Botturi).

1045-926X/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jvlc.2010.08.009

developers are emerging as a new conceptual tool in order to deal with this complexity. For example, two handbooks on instructional design languages [2,3] and a chapter on the same topic in the AECT Research Handbook [4] have been published recently.

However, until now, there has been a discrepancy between the attention paid to VIDLs in research and their actual usage by instructional designers. In practice, instructional designers find it difficult to use VIDLs due to their unfamiliarity and to the intrinsic complexity of the languages used [5]. Therefore, conceptions about the usability and cognitive effectiveness of VIDLs are of practical relevance in order to provide a solid basis for evaluating and comparing existing VIDLs and guiding practitioners in choosing an appropriate language. As previous research has demonstrated for a range of products, design aesthetics positively influence perceived usability [6], and it is likely that the design of VIDLs influences user's desire to become familiar with a VIDL. Existing literature comparing VIDLs [7–9] focuses mainly on formal aspects of the languages; evaluations from the user point of view are rare up to now. There are a few studies that assess the usability of specific VIDLs (e.g. [10]), but little research has been conducted on comparative evaluation of VIDLs.

To fill this research gap, in this article we investigate different VIDLs according to their cognitive effectiveness. We aim to bridge the gap between the theoretical descriptions and the specifications of VIDLs, and the practical application of those languages in design processes. Previous research on constructing domain specific visual (modeling) languages has shown that it is difficult to choose the appropriate concepts for visualization without emphasizing too specific concepts or too general ones [11], which may lead to low cognitive effectiveness resulting in low adoption rates. To take this into account, we specifically focus on the way VIDLs deal with the complexity of the educational domain (e.g. what perspectives or model types they provide). In this article, the discussion and evaluation of three selected VIDLs is theoretically grounded on a recently published framework on the desirable properties of visual languages [12].

The remainder of this paper is structured as follows. First, we begin with a general introduction and overview of VIDLs and their purposes. Then, we present a review of relevant theoretical perspectives on the cognitive effectiveness and management of the design complexity of visual modeling languages, with a specific focus on complexity management for the educational domain. We then continue by discussing selected VIDLs based on considerations of cognitive effectiveness and presenting the results of the user evaluation. Finally, conclusions are drawn and directions for further research are presented.

2. Visual instructional design languages (VIDL)

A design language is defined as a set of concepts that support structuring design (i.e. specification) or development (i.e. production) and help conceiving innovative solutions [13]. Although a design language is a mental construct, it can be expressed, and thus turned into a means of communication, through visual notation. A visual notation/language includes "…a set of graphical symbols, a set of compositional rules for how to form valid visual sentences, and definitions of their meanings" [12, p.756].

Design languages are of interest to a broad audience in different disciplines (e.g. [14,15]). In comparison to general-purpose modeling languages like Unified Modeling Language (UML) [16], VIDLs are domain-specific modeling languages for the instructional domain. The aim of VIDLs is similar to educational modeling languages, which have been proposed as providing a "…semantic information model and binding, describing the content and process within a 'unit of learning' from a pedagogical perspective in order to support reuse and interoperability" [17, p.10]. In contrast, however, VIDLs do not necessarily provide a binding of the conceptual meta-model underlying the language to a domain-specific or machine-readable format (e.g. XML).

2.1. Purpose of VIDLs

For a discussion or evaluation of VIDLs, we need to clarify their intended purpose [7]. From a practical point of view, a language is fundamental in order to allow a community to share their practices [18]. Using VIDLs is the first step in narrating such practices, and therefore to engage in reflective thinking as presented, for example, in Schön's "reflection on action" [19]. Visual models may help by providing a working space for problem solving in exploring, creating, refining, and redesigning design solutions. A common language means that a community has a means to name and describe its environment and its inner dynamics, to identify problems – design problems in this case – analyze them, and describe design solutions. A language is the medium for the creation of a common ground [20], i.e. a shared understanding of a problem and of its possible solutions, and eventually of a shared culture, in terms of the collection and construction of solutions and principles over time. Therefore, the language may improve communication, e.g. in design team meetings with fewer misunderstanding between experts and stakeholders due to the existence of a consistent terminology [9]. Further purposes of VIDLs include the documentation, sharing and reuse of final design solutions. VIDLs may facilitate the investigation and diagnosing of different e-learning settings according to their quality, and comparing them with respect to course design principles, as for example the alignment of face-to-face and online activities. In this way, instructional models expressed with a VIDL can support a more profound understanding of e-learning scenarios.

The use of design languages further allows designers and developers to generate and share design patterns. A design pattern captures the essential bits and pieces of a design solution to be adapted and reused over and over again for similar problems [21,22]. VIDLs can be used to complement the textual description of the design solution using visual models and illustrations.

Last, but not least, by specifying educational requirements in specific e-learning settings, VIDLs may help to bridge the gap between design and implementation. The production of a detailed and unambiguous model of instruction could then eventually be fed into an application (such as a learning management system) in order to generate a digital learning environment, although not all VIDLs support this aspect by offering a machine-readable binding.

3. Cognitive aspects of visual languages

A VIDL will only find acceptance when it supports educational designers and teaching practitioners. From a cognitive point of view, the interaction with VIDLs includes two main aspects, namely (a) the creation (authoring) of models and (b) the understanding (reading) of models [23]. Not all VIDLs require the same effort (e.g. time, subjective ease-of-use) to learn the language and to construct models. Additionally, models from different VIDLs are likely to differ according to the effort required to interpret them and to develop an understanding; VIDLs may also differ in the perceived difficulty of obtaining information through their visual representation. These aspects show the complex interplay between human cognitive models and visual instructional design models. A higher degree of match between the designer's mental image and the visual model of a learning design "…can facilitate comprehension and eliminate needless mental transformation" [24, p. 63]. That is, cognitive effectiveness is embodied in the ability of a VIDL to support appropriate translations between cognitive and visual models. Up to now, a variety of underlying cognitive theories have been adopted with regard to the context of visual modeling, often in an attempt to explore potential benefits of the visual representation. Examples include cognitive load theory [25], cognitive fit theory [26], cognitive dimensions framework for notational systems [27] and the theory of multimedia learning [28]. The form of visual information representation can have a significant impact on the efficiency of information search, the explicitness of information, and problem solving [29]. Moody [12] proposed 9 principles for the cognitively effective design of visual languages: semiotic clarity, graphic economy, perceptual discriminability, visual expressiveness, dual coding, semantic transparency, cognitive fit, complexity management, and cognitive integration. These principles are described in more detail in the following subsections.

3.1. Semiotic clarity and graphic economy

Semiotic clarity refers to the importance of a one-to-one correspondence between selected concepts and their visual representation by a symbol. Anomalies such as symbol redundancy (more than one symbol representing the same concept), overload (one symbol representing more than one concept), symbol excess, and deficit (when there are graphical symbols without a correspondence to a semantic construct, or vice versa) should be avoided, since they lead to ambiguity and additional unnecessary cognitive load for the user [12]. Research on the creation of domain-specific modeling languages reveals typical problems, e.g. that too many generic concepts for the domain or too many semantically overlapping concepts are chosen for a language; or that the language developer puts too much emphasis on specific concepts while neglecting other equally important concepts [11]. A reasonable balance between the expressiveness of a language and the number of the symbols is demanded by the principle of *graphic economy*.

3.2. Perceptual discriminability, visual expressiveness and dual coding

Perceptual discriminability is the "…ease and accuracy with which graphical symbols can be differentiated from each other" [12, p.762]. Visual languages which fully exploit the range of visual variables (e.g. spatial dimensions, shape, size, color, brightness, orientation, and texture) for their symbols offer a greater amount of *visual expressiveness*. If symbols differ according to several visual variables (e.g. color and size), they can be easily distinguished, and if a symbol has a unique value in the form of a visual variable, it is easily recognized. In comparison to a textual representation, which is encoded verbally in the reading direction, visual symbols are internally encoded in their spatial arrangement [30]. Therefore, the use of spatial dimensions (e.g. swimlanes in UML activity diagrams) can be especially recommended for visual languages. A wise combination of text and graphical representation is referred to as *dual coding*, and represents a further dimension for cognitively effective visual languages [12].

3.3. Semantic transparency

Semantic transparency describes whether symbols and their corresponding concepts are easily associated [12]. Icons, for example, are easily associated with their referent real-world concepts. Concerning the modeling of sequential learning activities, natural interpretations of the spatial relationships of symbols can be taken advantage of, e.g. elements on the left or above other elements are likely to imply some cause or one being a predecessor of the other [31]. Additionally, a visual depiction of nodes and edges is likely to be intuitively understandable because of its similarity to internal mental representations of concepts and their relationships [32].

3.4. Cognitive fit

Cognitive fit refers to the fit between the problem representation and the strategies required to perform a specific task [33]. Therefore, the cognitive effectiveness of a visual language might be different for experts and for beginners, or for sketching on paper versus using a modeling software application. A single VIDL could provide different visual dialects for each relevant user group, or task, as a means of improving its cognitive fit [12].

3.5. Complexity management and cognitive integration

According to Moody [12, p.765] complexity management "…refers to the ability of a visual notation to represent information without overloading the human mind". Cognitive load is determined by the number of elements that should be considered simultaneously [34], and there is a natural limit with regard to the capacity of the human short-term memory of approximately 7 ± 2 elements [35]. However, although the number of elements is limited, their size and complexity is not. Chunking expands the capacity of short-term memory, because information units belonging together are chunked into one unit [36]. A language should provide mechanisms to manage complexity in order to impose as low a cognitive load on users as possible, so that individual models do not overwhelm users by exposing them to too much complexity.

There are two main mechanisms that can be applied to manage complexity: modularization and hierarchical structuring. Modularization works by dividing complex domains into smaller parts ("chunks"). Languages may provide different subsystems or level structures. A larger problem becomes more easily manageable if it is broken down into separate parts. A lack of modularization and too high coupling between interconnected diagrams, may cause difficulties in maintaining models [11]. Hierarchical structuring provides different levels of detail (abstraction/summarization versus decomposition/refinement), which makes complex concepts more easily understandable for humans [12].

Modularization, or the intent to provide different perspectives, leads to multiple diagrams which belong together and represent a domain. The principle of *cognitive integration* [12] is important in terms of supporting the understanding of relationships between different models. Important methods to support cognitive integration are the provision of summary (overview) models and the showing of the context of the whole system in each single model, each of which represents only a smaller, specific part [37]. Additionally, navigational maps depicting all models and their relationships, as well as the clear labeling and numbering of levels, supports the viewers' orientation [12].

3.5.1. Complexity management in visual instructional design languages

In the following section, we present a framework for analyzing the complexity management of VIDLs, partly building on the work in [38], and partly based upon the observation that different diagram types of VIDLs address different ways of thinking, take different perspectives and focus on different aspects of the domain. Previous research on the comparison and the evaluation of VIDLs [7,9] provides a thorough basis for selecting dimensions of complexity management. Existing efforts will be briefly described and embedded in the context of the selected dimension.

Although complexity management in general is not specific to the instructional design domain, how this domain is captured and conceptualized by VIDLs is of specific interest. We identify three dimensions that reflect the characteristic management of domain complexity in VIDLs: (1) stratification, (2) elaboration, and (3) perspective. Stratification (organization) and elaboration (level of detail) have already been identified by [5] as important variables for improving the organization of design documents using a layered design architecture. The dimensions are explained in the following subsections.

3.5.2. Dimension 1: elaboration (hierarchical structuring)

The "elaboration" dimension relates to ways how VIDLs enable us to represent levels of abstraction, depending on the proximity of the modeled concepts to the actual implementation. A language may support one or more degrees of elaboration of design. Each particular diagram type of a VIDL is able to represent and describe more or less detail of a particular design artifact. We propose three levels of elaboration which were adapted from Fowler [39], and which are characterized as follows:

1. The *conceptual level* allows for a general, aggregate view of the design, indicating its rationale and main elements. This degree of elaboration is particularly suited for early design stages and idea generation.
2. The *specification level* provides means for a more comprehensive description, including the design elements at more specific levels. This degree of elaboration is suited for adding more detail to conceptual representations in order to achieve a better understanding of higher-level concepts. It can also be used to prepare the transition to the development stage.
3. The *implementation level* represents the highest level of detail. This degree of elaboration is typically required for the development of design artifacts (e.g. learning objects).

3.5.3. Dimension 2: stratification (modularization)

Stratification refers to domain-specific complexity management through modularization, by structuring the domain according to different design layers. For instance, Gibbons [40] proposes the following structure of seven design layers for instructional design, in order to organize the discussion about instructional design languages:

1. *Content layer*: analysis of the content and structure of the domain.
2. *Strategy layer*: design of the instructional tasks and activities required to achieve the instructional goals.
3. *Control layer*: design of the learner interaction with the instructional system (actions, control flow, etc.).
4. *Message layer*: design of the messages (information presented to the learner) as indicated by the strategy layer.
5. *Representation layer*: design of the media, tools, and methods that represent (e.g. visualize) the design.
6. *Media logic layer*: design of the logic of the instructional application (software architecture, learning objects logic, etc.).

7. *Management layer*: design of the data management and administration processes.

Some researchers have tried to classify VIDLs according to design layers, because many languages do not cover all layers. For instance in [41,42], the authors distinguish three different types of VIDLs which focus on different layers: content structuring languages (focus on the content layer), activity languages (focus on the strategy layer), and evaluation languages. *Evaluation languages* cannot be directly mapped to the seven layers listed above. However, evaluation seems to be another important layer, targeting issues of problem-solving and question–answering in the learning process.

Any particular VIDL can be either *single-layered* (i.e. applicable to exactly one of the seven layers) or *multi-layered* (i.e. applicable to more than one layer). A multi-layered language offers a set of visual representations for describing entities of different types, such as people and roles, activities, or learning materials, at different layers of design. Each layer exposes a different set of design goals, problems, structures, and terms that would need to be addressed and supported by the design language. Consequently, while multi-layered languages can be more expressive and detailed, they also require more effort to support the cognitive integration of different model types. Single-layered languages are easier and more straightforward to use, while limiting the number of "views" on design solutions.

3.5.4. Dimension 3: perspective

As outlined in [12], visual languages often do not only provide hierarchical structuring or modularization, but also provide heterogeneous model types, e.g. for representing and visualizing different perspectives. A VIDL can offer one *single* or *multiple* perspectives on the same concept or model. Multiple-perspective languages offer different tools for representing more than one view on the same set of entities. For example, one language can have representations both for chronological relationships between learning activities as well as for structural relationships between learning activities. Further concrete instances of perspectives are, for example, the learners' or teachers' points of view.

Note that both perspectives could be at the same level of elaboration and could be located on the same layer; that is, the perspective dimension is independent of stratification and elaboration. While each additional perspective adds more detail and facets to the entity under consideration, the cognitive integration of perspectives becomes increasingly difficult. Depending on the use of the language, an additional perspective can be used to clarify ambiguities about particular concepts among different designers. An additional perspective might also be some required artifact needed to satisfy computational or execution constraints.

4. Evaluation of cognitive effectiveness of selected VIDLs (E^{2}ML, PoEML, CoUML)

This section presents three VIDLs and discusses their diagram types according to criteria for cognitive effectiveness as presented in the theoretical part of the paper. First, we outline the method used to perform the user evaluation. Then, we describe the selected VIDLs and discuss their main diagram types in terms of salient positive and negative aspects raised during the user evaluation. Therefore, not all nine criteria for cognitive effectiveness as defined by [12] are discussed for each diagram type. Rather, the focus is particularly on examples of good design as well as violations of cognitive effectiveness. The section concludes with a presentation of results and the findings arising from the user evaluation.

4.1. Method

The evaluation of the VIDLs was based on two aspects. In the first qualitative part ("*the creation of diagrams*"), users were asked to acquaint themselves with the languages and to actively create models of course designs. In the second, more quantitatively oriented part ("*the evaluation of diagram types*"), the cognitive effectiveness and usefulness of a set of existing diagrams modeled in different languages was rated by a different sample of users in a web-based questionnaire. Thus, the evaluation involved the main cognitive activities in terms of the creation and interpretation of diagrams. Additionally it included a few users with knowledge of the languages for the qualitative evaluation, as well as a larger sample of users for the quantitative evaluation.

The creation of diagrams: Five independent experts (3 graduate students with backgrounds in information systems modeling and new media, and 2 course instructors from an information systems department), who were familiar with the cognitive effectiveness criteria, but unfamiliar with the languages, were asked to become acquainted with the language descriptions. After learning the languages, they modeled two courses using the provided diagram types in each of the languages. Then they provided feedback on the languages. Since the modeling process (in particular the tools provided) is supported quite differently by different languages, these evaluations are not immediately comparable. Nevertheless, the qualitative evaluations revealed several problems that beginners might face when learning these languages. A variety of points for improvements were identified and included in the discussion of the languages.

The evaluation of diagram types: For this evaluation, three different diagram types were selected for each language, and a web-based questionnaire instrument was created. Since there were no existing scales for the cognitive effectiveness criteria, two-item scales were constructed for each criterion that could be evaluated for each given diagram. In order to evaluate cognitive fit, complexity management and cognitive integration, knowledge of more diagram types and their relationships would be necessary. Therefore, these criteria were not included in the questionnaire. Additionally a scale on the perceived usefulness of diagram types as proposed by [43] was adapted for VIDLs and included in the questionnaire. We ran a pre-test with 3 participants for ensuring content validity and for ensuring the understandable formulation of items before administering the questionnaire. Reliability analysis revealed adequate internal consistency for all scales (Cronbach's $\alpha > 0.8$, with the exception of visual expressiveness (Cronbach's α=0.6) and semiotic clarity (Cronbach's α=0.2)), for which we then analyzed single item scores.

The final sample consisted of 20 participants (11 males, 9 females), aged 34 years on average. Most participants were course instructors (11), while some were members of the e-learning support team of universities (3) or researchers in the context of instructional design (6). The participants had already been involved in the creation of 5 different instructional designs (e.g. courses) on average.

4.2. E^{2}ML—educational environment modeling language

E^{2}ML [44,45] was developed mainly as a thinking tool for instructional designers and for enhancing communication within large e-learning projects. The result is a language with a very limited number of symbols, and with a set of diagram types that cover two different layers of detail (overview and action detail) and two perspectives (temporal and structural). Learning goals, requirements and the design of teaching and learning activities can be modeled. There is a more specific tool for goal classification that was developed in conjunction with E^{2}ML: the Quail model [46], which is a visual model for the definition and classification of high-level learning goals. E^{2}ML modeling starts with the definition and mapping of educational goals, then all available resources (actors, resources, tools) are listed (in tabular form) and action diagrams (learning and support activities) are modeled as the core of the design solution. Action diagrams are presented in structured tables and not by the use of visual symbols. They are the core part of E^{2}ML and represent educational activities. Relationships between actions, as for example inheritance and aggregation, can also be expressed. Finally, overview diagrams are created such as a timeline as a visualization of the "course calendar", or a structural overview of the activities (dependencies diagram) [47]. Thus, three main diagram types can be identified: (1) goal definitions, (2) action diagrams, and (3) overview diagrams (dependencies and activity flow diagram) [48] as depicted in Fig. 1.

Goal diagram: A positive aspect of the goal diagram is that it uses two spatial dimensions to classify goals. This makes it easy to compare the goal structures of several courses at a glance. On the other hand, the *perceptual discriminability* and the *semantic transparency* of the symbols used (fact, concept, procedure, etc.) are quite low—they only vary according to their form and no other visual variable such as size or color is used. *Dual coding* is realized via a legend, but it demands cognitive effort to simultaneously switch between symbols in the visual grid and descriptions of the symbols below.

Dependencies diagram: This diagram displays an overview of the actions/activities in a course block on research paper writing. It shows different kinds of dependencies among action elements (rectangles). For instance, the "collect literature", "content draft", and "paper writing" actions have the "paper writing workshop" as a prerequisite (indicated by an arrow with a dotted head). Collecting the literature and drafting the paper content produces relevant literature and a content draft as products, respectively, that are input to the "paper writing" activity (indicated by simple arrows). Finally, the presentations require the completed paper as a prerequisite. The visualization of the product relationships seems to be more easily understandable than those of the pre-requirement relationship due to the use of arrows. All the "group work" actions are represented as an aggregation box around the relevant actions. The aggregation boxes representing grouping exhibit *semantic transparency*, i.e. they can be understood without explanation.

Action diagram: The action diagram is represented in the form of a table. This provides a good overview, but designers have to remember the meaning of all the cells as there are no hints provided once a table is filled out. It is possible to decompose actions into sub-actions to model several levels of detail. Cognitive integration between action diagrams and goal diagrams is realized via a navigational cue: an identifier tag (a small rectangle attached to the action diagram with an abbreviation of the corresponding goal).

Activity flow diagram: The activity flow diagram describes the temporal and logical flow of the educational activities during a course. As opposed to common practice in process modeling, no start and end symbols and no arrows are used to visualize the control flow. As long as textual information about dates and times provide dual coded information, the flow direction should not be difficult to interpret. When activity flow diagrams as well as dependencies diagrams are used, the problem of *symbol overload* occurs: a small dot represents a join of different activity flows as well as a pre-requirement relationship between different actions, respectively.

4.3. PoEML—Perspective-oriented Educational Modeling Language

The Perspective-oriented Educational Modeling Language (PoEML) [49,50] stems from a study of the expressive power of current instructional design languages, with a specific focus on IMS Learning Design (IMS LD) [51,52] and integrates many concepts from workflow modeling and groupware. It focuses on the separation of 13 different perspectives on educational designs (e.g., structural, functional, participants, environment, data and data flow, tools, order and control flow, etc.). In constructing these perspectives, overlaps between perspectives were reduced to a minimum, so that perspectives can be modeled independently of one another. This appears to be true for most perspectives; though, for example the data perspective models are integrated with several other diagram types. Consequently, hierarchical structuring is provided by decomposition into several independent model types [53, p. 983]. Additionally, PoEML uses a second orthogonal kind of cross-cutting concerns and distinguishes between four different aspects

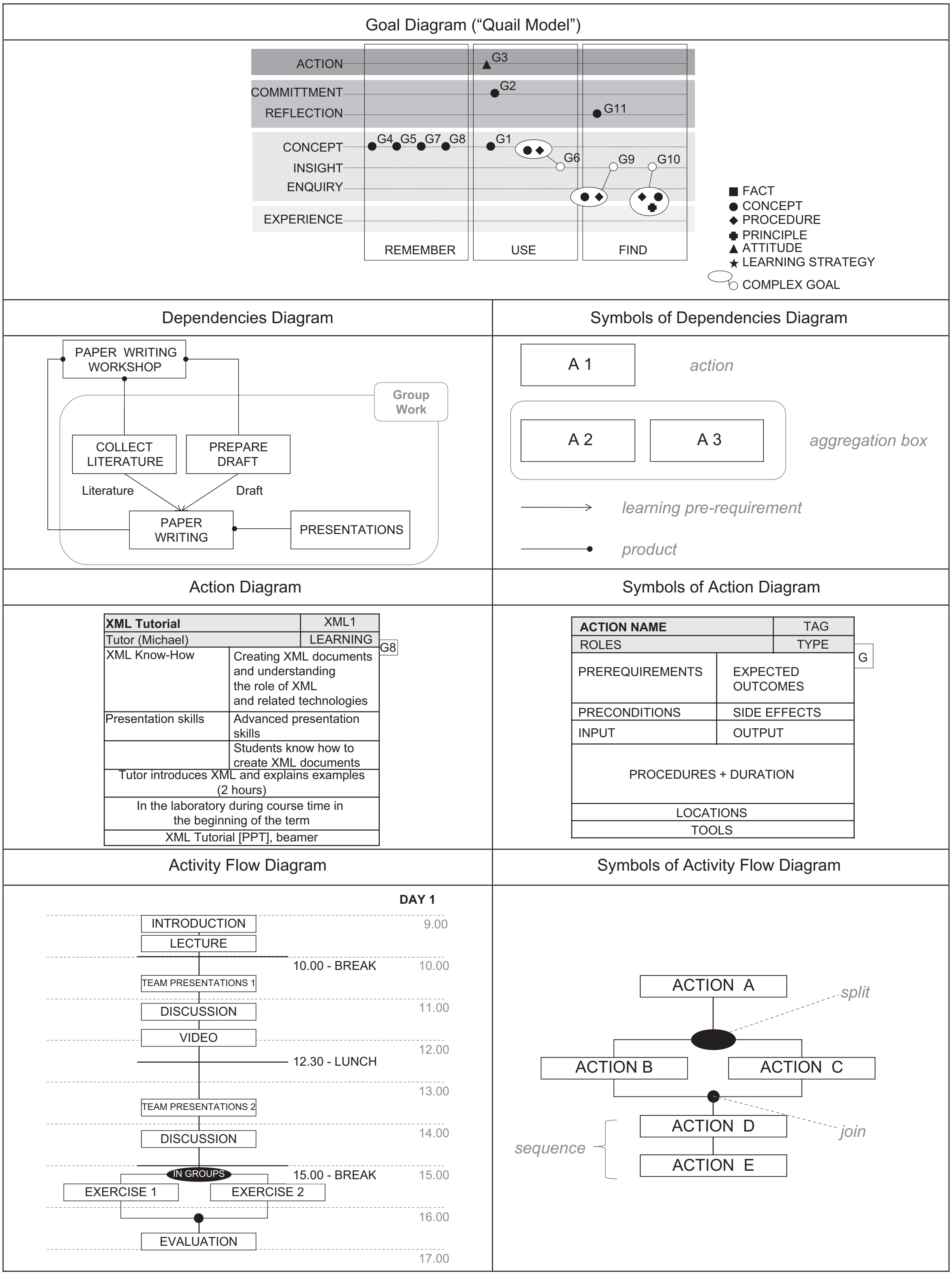

Fig. 1. Different diagram types in E^{2}ML.

(modes of control) describing how an educational unit is carried out during runtime (constant-fixed, data-based/conditioned, event-based/signaled or decision-based behaviour). The relationships between several diagram types are described in the meta-model. PoEML provides an extremely rich and expressive tool which can be used by designers to model educational scenarios on different aggregation levels (e.g. single lessons or whole curricula). It also offers a set of patterns for modeling in each of the perspectives. The output is coded in XML. Similar to IMS LD, PoEML can hardly be used without a graphical user interface application, of which a prototype is available (J-PoEML; [49]). Several diagram types of E^{2}ML are depicted in Fig. 2.

Structural perspective diagram: The structural perspective provides an overview of several elements of an educational scenario (e.g. a course). In general, the symbols used as part of the structural package provide high semantic transparency due to their iconic representation. However, not all of them are similarly intuitive. For example, for "order specification" and "causal descriptions", it might be possible to find symbols with higher perceptual immediacy.

The structural perspective allows for hierarchical aggregation and the refinement of educational scenarios, visualized in the form of a hierarchical tree, which should be easily understandable. Concerning semiotic clarity, users might be irritated that, on the highest level of detail, a different symbol is used for an educational unit/scenario than on lower levels.

Functional goal perspective diagram: Functional goals refer to the tasks that participants have to perform, and not to knowledge, skills or abilities that could be attained in an educational setting, as in the goal diagram of E^{2}ML. This is one of the few diagram types in which the visual variable color is explicitly used to convey information (mandatory, optional or hidden goals).

Participants' perspective diagram: In this diagram type, different roles are modeled (e.g. learner, instructor). Here, it is also possible to model roles and sub-roles hierarchically. The sample diagram demonstrates that a high level of detailed information and specific rules can be visualized in PoEML. For instance, the minimum and maximum number of learners and teachers is defined by the attached data element symbols. Moreover, it is modeled that a specific algorithm (first-in–first-out) is used to assign learners to exams or pairs in the practical part. The use of data elements allows the refining of a design up to a very detailed implementation level, as compared to the two other VIDLs under investigation here. On the other hand, for beginners, the great variety of symbols and connection types might be confusing.

Environments' perspective diagram: This diagram visualizes whether activities are performed in a virtual or a physical environment (e.g. a laboratory) and which tools (e.g. a document) and artifacts (e.g. a text-editor) are used by the participants.

Order perspective diagram and temporal perspective diagram: The order diagram and the temporal diagram visualize in what logical order and under what temporal constraints educational scenarios (comparable to activities and actions in other languages) are performed, respectively. It is likely to be intuitive due to the left-to-right layout of the sequence, and the use of arrows between activities. Therefore, the meaning of the icons used to represent the start (house) and the end (flag) also becomes obvious. On the other hand, the order connectors (sequence, parallel split, and synchronization) are dispensable. Since the alignment of connecting arrows represents the same process flow, users might even get irritated due to symbol redundancy.

4.4. CoUML—Cooperative UML

CoUML is an educational modeling language that can be used to model technology-enhanced learning and

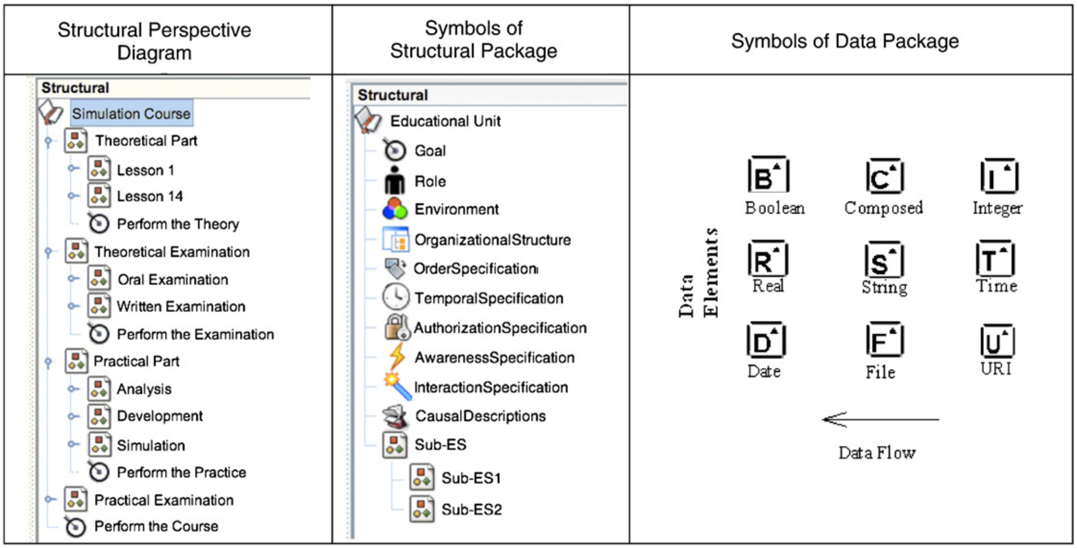

Fig. 2. Different diagram types and symbol sets in PoEML [49,53].

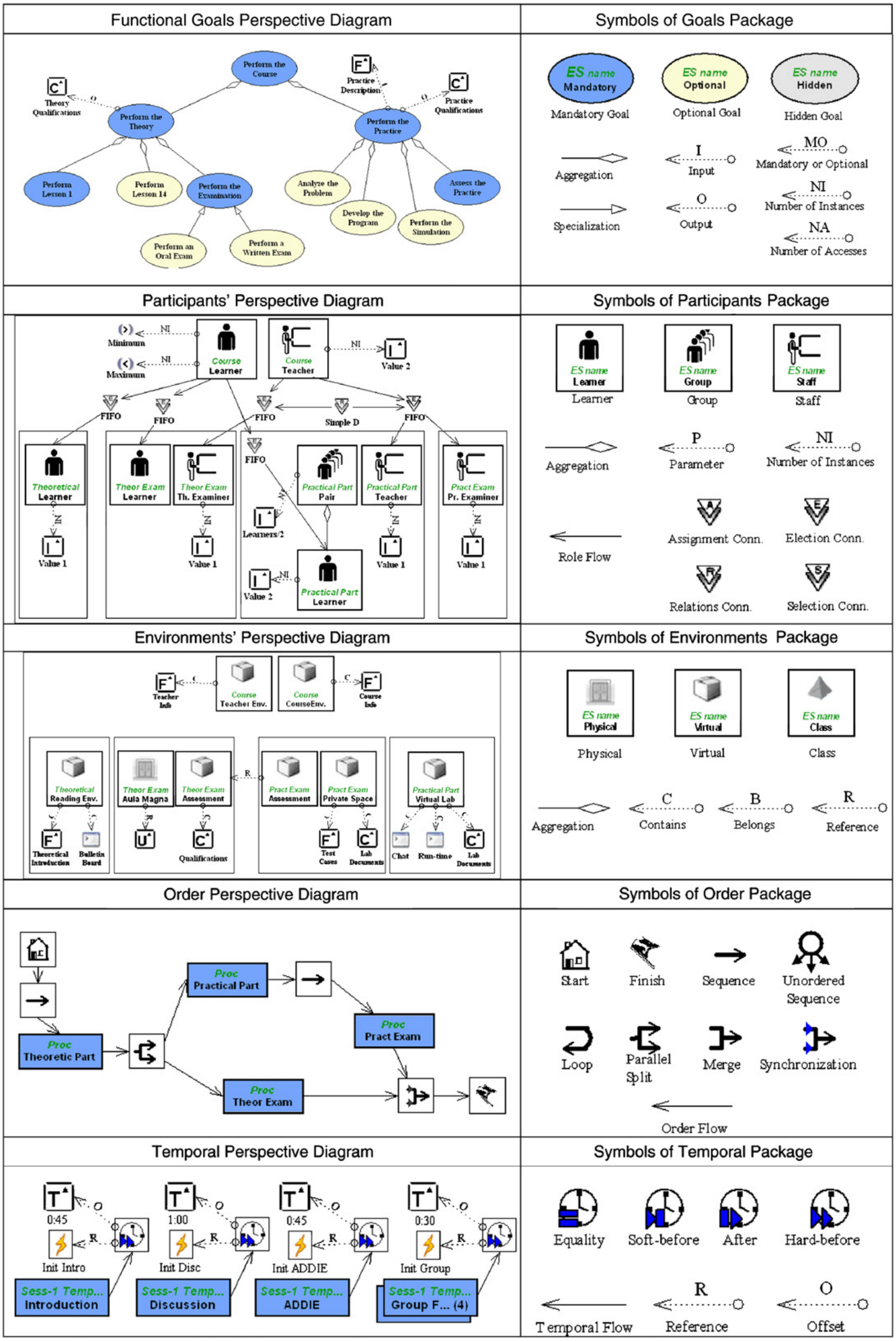

Fig. 2. (*Continued*)

cooperation environments [54]. CoUML stands for ''Co-operative UML'', indicating that its notation system is essentially an extension of the UML used to model cooperative activities and environments. The notation has been revised and improved over several years during its application in practice; it was used to model blended learning courses for documentation purposes and for finding patterns of recurring activities and structures in technology-enhanced environments. Being based on UML, it exposes a formal notation system allowing (a) the modeling of structural concepts like the documents, goals, and roles involved; and (b) the modeling of activities performed by roles in the target environment, incorporating relevant objects from the structural models (e.g., documents used in or produced by activities, or goals achieved by activities). The structural models use generalization/specialization concepts, as well as dependency relationships (e.g., include, derive, successor-of, or use) and the overview diagram shows how the diagrams relate to each other. CoUML offers the following diagram types as illustrated in Fig. 3.

Course activity diagram: Course activity diagrams are the ''primary artifacts'' of a coUML design model [55]. The course activity diagram in Fig. 3 shows a coUML model of activities performed, and documents produced by the instructor, students, and student groups. The level of detail is low, but the perspective is different: here, the focus is on the temporal aspect. This diagram shows how coUML is used to demonstrate different areas of responsibility (those of instructor, student, and group), and how activities (rounded rectangles) are arranged in chronological order (solid arrows), including the documents (rectangular boxes) produced and consumed (dotted arrows) by those activities. This model type is an extension of UML activity diagrams; the most notable extensions include the visualization of points in time and the different stereotypes for declaring activities as proceeding primarily *face-to-face*, *web-based*, or in a *blended* mode [55]. It is worthwhile to mention that a positive cognitive aspect of this diagram is the use of two spatial dimensions to depict information on roles (represented as so-called ''swimlanes'' in UML) and the temporal aspect, leading to high *visual expressiveness*. The diagram's notation is based on UML activity diagrams, which generally provide high *perceptual discriminability* of symbols [56].

Learning goals diagram: This diagram is used to model the intended learning goals (rectangles carrying the keyword ≪goal≫) to be achieved by learners. Specific goals can be generalized by higher-level goals using the UML generalization relationship (a solid-line arrow with a hollow triangle pointing to the more general goal). Aggregate goals can be decomposed into a set of sub-goals by using UML aggregation relationships (solid connectors with a hollow diamond at the aggregate end). Learning goal diagrams do not perform well on the *visual expressiveness* dimension, since goals at all levels, and of any type, have the same simple symbol. Other than that, these diagrams are *graphically economic*.

Document diagram: Document diagrams are used to model structural overviews of the documents that are provided and created during the runtime. Documents are modeled as a rectangle carrying the name of the document and the keyword ≪document≫. There are several types of relationships that can be modeled between documents: aggregation (similar to goals, see above), and a dependency between documents, which indicates that one document requires another document. This diagram type also allows for modeling the providers and consumers of documents by linking document symbols with role symbols using dotted arrows (either unidirectional or bidirectional). This notation should be easily understandable since it is *semantically transparent* and *graphically economic*.

Role diagram: The role diagram is used to model the roles that participate in and interact with each other during the instruction. It is a structural model that represents roles (e.g. instructor or student) as stick-figures. Roles can be associated with each other, either using a support dependency (a dashed arrow carrying the keyword ≪support≫) or a UML aggregation relationship, indicating that a role may be part of another role (e.g. in groupwork scenarios, students are organized in groups, introducing the group role). Role diagrams are typically simple, since most instructional designs will not involve more than a handful of different actor roles.

Course structure diagram and course package diagram: Finally, the course structure diagram acts as visual index to the course activity diagrams, and the course package diagram shows, in an overview diagram, the components of the whole design solution. Both diagram types exhibit only a small set of symbols, i.e. package symbols and rectangles with arrow connectors.

4.5. Evaluation results

This section presents the results of the user evaluation of the selected diagram types of the three languages, E^{2}ML, PoEML, and coUML. Since the different diagram types of the languages did vary to a great extent according to criteria such as *perceptual discriminability* or *semantic transparency*, it is difficult to offer a general evaluation for a language. The overall evaluation for a language may also differ from the mean value of the scores for its diagram types; for instance, *semiotic clarity* might be high for specific diagram types yet low for the whole language if a symbol has different meanings in different diagram types. Therefore, the evaluation results are presented separately for each diagram type. Table 1 shows the descriptive results of the user evaluation.

E^{2}ML evaluation: The *semiotic clarity* of the three E^{2}ML diagram types is moderately high. The scores for the absence of construct deficit range from 3.44 to 2.62. Meanwhile, the scores for the absence of construct excess vary from 3.30 to 4.36. The *graphic economy* is rated very high except in the case of the goal diagram (2.09). This result is directly correlated with the total number of symbols (7). The *perceptual discriminability* results confirm our initial assessment outlined in Section 4.2, because the goal diagram obtained a rather low score (2.11). However, the other diagram types achieve good values (3.12 for the

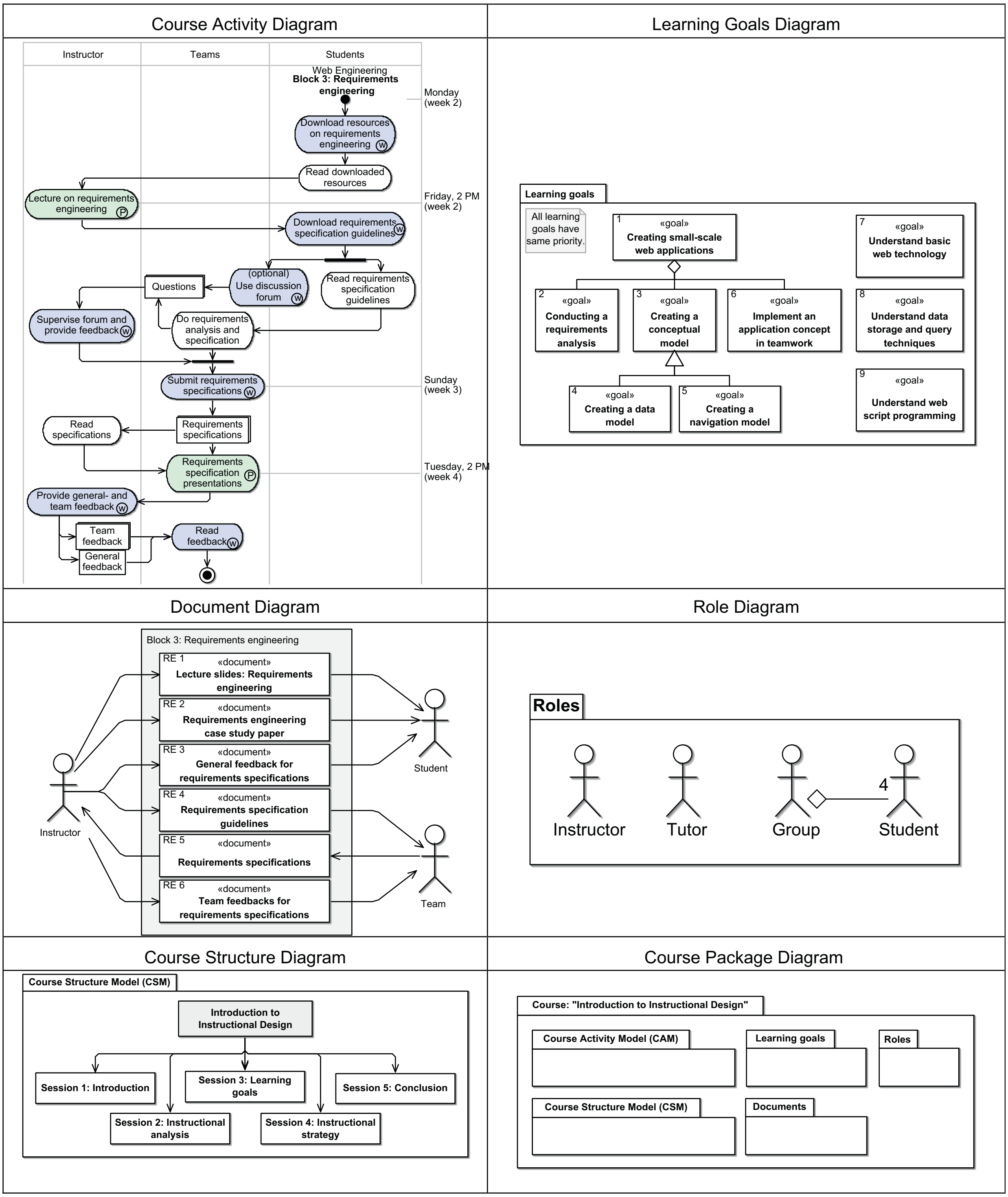
Course Activity Diagram
Instructor
Teams
Students
Web Engineering
Block 3: Requirements engineering
Monday (week 2)
Download resources on requirements engineering
Read downloaded resources
Lecture on requirements engineering
Friday, 2 PM (week 2)
Download requirements specification guidelines
(optional) Use discussion forum
Read requirements specification guidelines
Questions
Supervise forum and provide feedback
Do requirements analysis and specification
Submit requirements specifications
Sunday (week 3)
Read specifications
Requirements specifications
Requirements specification presentations
Tuesday, 2 PM (week 4)
Provide general- and team feedback
Team feedback
General feedback
Read feedback
Learning Goals Diagram
Learning goals
All learning goals have same priority.
1 «goal» Creating small-scale web applications
2 «goal» Conducting a requirements analysis
3 «goal» Creating a conceptual model
6 «goal» Implement an application concept in teamwork
4 «goal» Creating a data model
5 «goal» Creating a navigation model
7 «goal» Understand basic web technology
8 «goal» Understand data storage and query techniques
9 «goal» Understand web script programming
Document Diagram
Block 3: Requirements engineering
RE 1 «document» Lecture slides: Requirements engineering
RE 2 «document» Requirements engineering case study paper
RE 3 «document» General feedback for requirements specifications
RE 4 «document» Requirements specification guidelines
RE 5 «document» Requirements specifications
RE 6 «document» Team feedbacks for requirements specifications
Instructor
Student
Team
Role Diagram
Roles
Instructor
Tutor
Group
4
Student
Course Structure Diagram
Course Structure Model (CSM)
Introduction to Instructional Design
Session 1: Introduction
Session 3: Learning goals
Session 5: Conclusion
Session 2: Instructional analysis
Session 4: Instructional strategy
Course Package Diagram
Course: "Introduction to Instructional Design"
Course Activity Model (CAM)
Learning goals
Roles
Course Structure Model (CSM)
Documents

Fig. 3. Examples of diagram types offered by coUML.

dependencies diagram and 4.03 for the activity flow diagram). Similarly, the *visual expressiveness* was also rated lower for the goal diagram than for the other two diagrams. The *semantic transparency* criterion follows the same pattern, with a very low score for the goal diagram (1.45), a medium score for the dependencies diagram (2.43), and a good score for the activity flow (3.83). The *dual coding* dimension received very high scores ranging from 3.70 to 4.50. This could be a consequence of the use of textual legends. Summarizing the results for E^{2}ML, the

Table 1
Mean values of user evaluation of the cognitive effectiveness of diagram types (n=20) [five-point Likert scale from 1=strongly disagree to 5=strongly agree].

	1. Semiotic clarity: absence of construct deficit	1. Semiotic clarity: absence of construct excess	2. Graphic economy	[Number of different symbols]	3. Perceptual discriminability	4. Visual expressiveness	5. Dual coding	6. Semantic transparency	7. Perceived usefulness
E^{2}ML									
Goal diagram	3.44	3.30	2.09	[7]	2.11	1.93	3.70	1.45	2.30
Dependencies diagram	2.62	3.73	4.29	[4]	3.12	2.55	4.05	2.43	2.97
Activity flow diagram	3.21	4.36	4.50	[4]	4.03	3.20	3.83	3.83	3.70
PoEML									
Functional goals perspective diagram	4.08	3.50	2.97	[10]	2.92	2.88	3.75	2.15	2.98
Participant's perspective diagram	4.29	3.21	1.73	[11]	1.86	2.32	3.48	1.73	2.00
Order perspective diagram	3.43	3.80	3.94	[9]	3.25	3.15	3.75	3.10	3.17
coUML									
Role diagram	2.41	3.82	4.60	[2]	2.85	4.20	3.40	2.45	2.85
Document diagram	2.54	4.28	4.50	[3]	4.18	3.50	3.83	3.98	3.22
Course activity diagram	4.00	4.60	4.40	[9]	3.70	4.05	3.15	4.10	3.70

global perceived usefulness of the E^{2}ML diagrams is quite high, despite the goal diagram receiving a low 2.30 score.

PoEML evaluation: PoEML is notable for its extensive use of easily understandable icons (e.g. stick-figures, clocks, houses). The *semiotic clarity* of PoEML is very good, and the three evaluated diagram types achieved results ranging between 3.21 and 4.29, both in relation to the absence of construct deficit and excess. Nevertheless, since there are many diagram types and a large number of symbols, the principle of *graphic economy* is not fulfilled so well. This is particularly true for the participants' perspective diagram, which received a 1.73 score with 11 different symbols. The *perceptual discriminability* was rated quite low, especially with regard to the participants' perspective diagram (1.86). This may be due to the large number of similar symbols, e.g. many rectangles are used for different concepts, which can only be discriminated by colors and the icons inside. There is also a variety of symbols in the other diagrams that can only be distinguished by their textual annotation, e.g. a dotted arrow symbol is used to represent at least 9 different types of relationships (labeled with I, O, MO, NI, NA, P, C, B, R). Similarly, the 9 different data elements are only distinguished with single letters. This may lead to difficulties in distinguishing different relationships (dotted arrows) or data types (small boxes) from one another. On the other hand, using a similar shape for different symbols may account for recognizing them as belonging together, due to the Gestalt law of similarity [57]. This could be useful for data symbols, but less useful for the relationship symbols, as they represent quite different types of relationships. Probably as a result of this, the *visual expressiveness* aspect received medium scores ranging from 2.88 to 3.15. PoEML does reasonably well on the *dual coding* criterion, with scores ranging from 3.48 to 3.75, perhaps because it allows the use of textual annotations which are placed inside the symbols in most diagram types. Nevertheless, the *semantic transparency* of the three diagram types was rated rather low (2.15, 1.73, and 3.10, respectively). These low scores suggest that the symbols need to be complemented with icons whose appearance suggests their meaning more intuitively. Finally, the perceived usefulness of PoEML is quite good, except in the case of the participants' perspective diagram, which received the worst score of all the evaluated diagrams (2.00).

CoUML evaluation: The *semiotic clarity* of coUML is generally good, even though it exhibits a certain degree of overload, since some symbols (e.g. rectangles) are used to model different concepts. However those symbols are tagged with a keyword, so it is possible to discriminate between them. In this way, the user evaluation shows the maximum scores for the course activity diagrams: 4.00 for the absence of construct deficit and 4.60 for the absence or construct excess. CoUML's *graphic economy* is excellent as it receives very high scores for the three diagrams (4.40 to 4.60). The results indicate that the language allows the visual expression of a versatile set of concepts in detail, with a low number of visually easily discriminable symbols. The *perceptual discriminability* and *visual expressiveness* also obtained very good scores, with values greater than 3.50, except for role diagram (2.85). CoUML also does reasonably well on the *dual coding* dimension, perhaps because both text and symbols are used to represent concepts. The *semantic transparency* is also very well rated for the document diagram (3.98) and the course activity diagram (4.10), but not so well for the role diagram (2.45). Finally, the perceived usefulness of the diagrams corresponds with the results of the other criteria as the diagrams achieve very high scores (3.70 and higher), with the exception of the role diagram (2.85).

As already mentioned, some criteria could not be evaluated by users based on single example diagrams, and were consequently not included in the table; they are briefly discussed in the following. In general, the languages considered did not differ to any great extent in terms of *cognitive fit*, *complexity management*, and *cognitive integration*. Concerning *cognitive fit*, for instance, all languages provide only one visual representation of the diagram types for all user groups and tasks. Nevertheless, the literature on E^2ML shows, for example, that the language can be used for sketching on whiteboards in a very flexible manner [45]. All languages put effort into *complexity management* by providing several diagram types, including different perspectives to some degree, and supporting *cognitive integration* by the provision of overview diagrams and by enabling referencing between different diagram types. Concerning differences in stratification, E^2ML and coUML mainly provide diagrams for modeling on the strategy layer (an exception is the coUML document diagram which models aspects of the management layer). PoEML does not provide different diagram types for the layers, but in many diagrams concepts from several layers such as strategy, control, message, media logic, and management can be modeled. Different hierarchical levels are supported by all three languages, and modeling on the conceptual as well as specification layer is possible, although PoEML is the only language that enables the modeling of implementation details.

4.6. Limitations

A basic limitation of the presented evaluation is that some of the cognitive effectiveness criteria can only be evaluated after working intensively with the language. Future research could profit from including user studies involving actual designers in realistic, controlled design settings over a longer period of time, for example as demonstrated in [5]. However, we do believe that letting a larger sample of users evaluate example diagrams was consistent with the goals of the study, and provided a reasonable first test of the cognitive effectiveness and the perceived usefulness of the diagram types. The difficulty of finding test users who have a profound knowledge of the languages relates to problem of the generally low adoption of the investigated VIDLs. Looking ahead, future research needs to examine causes for low adoption and for ways of improving the achievement of higher user acceptance in the case of the existing VIDLs. Future research could also take other VIDLs into account, as there are many more available (see [2,3] for an overview). Such a complete evaluation might reveal even more usable and creative solutions for visualizing specific aspects in instructional design.

5. Conclusions

This paper presents the first study of the cognitive effectiveness of visual instructional design languages (VIDLs). Our results suggest that an evaluation from a user's point of view is useful as a means of identifying various points for improvement in terms of quality and the ease of use of VIDLs. Improvement may, then, lead to higher acceptance and actual use of VIDLs by designers in the long run.

Since there are many diagram types associated with the evaluated languages which have similar purposes (e.g. goal or learning activity diagrams), we believe that an integration of several diagram types into one single, unified modeling approach would be beneficial as a means of better supporting the instructional design community in the future. Other domains have successfully demonstrated how powerful the establishment of an accepted visual modeling standard can be, as for example the UML [16] for the software domain or Business Process Modeling Notation (BPMN) [58] for the business process domain.

Additionally, many diagram types associated with different VIDLs focus on different aspects and complement one another; their combination in a unified modeling approach would allow the modeling of an extended number of domain aspects. For instance, in early design stages, designers could use diagram types as proposed on the conceptual level in the more sketchy language E^2ML, while in later designs and in the development stages, diagram types of a language such as PoEML might be more appropriate to add more precision and detail to the creative solutions of earlier stages (cf. [59]). The provided discussion of the complexity of the domain allows an assessment of the expressiveness of existing languages and their diagram types, and might help to identify spots in the domain space that are not yet occupied. In particular, when trying to find an optimal solution, competing proposals should be compared as demonstrated by this paper, to identify strong and weak aspects of the languages concerned. New combinations of existing diagram types from different languages [2,3] could be integrated to enhance usability and to lower the cognitive demands placed on users.

In constructing a new unified modeling approach, besides combining several diagram types, efforts to align diagram types and to support cognitive integration between them seems important. Similar to the new proposal of BPMN [58], a lightweight version, including a smaller set of symbols, could be created to lower the entry barriers for beginners. A modeling standard for VIDLs could provide diagram types for a variety of specific design activities, and would enable an internationally oriented development of instructional design pattern repositories. Once in existence, such a standard could also guide (novice) designers by providing some agreed-upon structure in order to manage the complexity of the design domain.

Several possible directions for future research emerge from our user evaluation of VIDLs. Future efforts need to address why VIDLs are rarely used. Besides a lack of background in software engineering, or low interest in the more technical aspects of design languages, VIDLs may demand considerable time and effort in terms of learning, and the support of tools and documentation seems to be insufficient at this point, since usable modeling tools are missing. For E^{2}ML and coUML, for instance, power-point templates are the only available modeling tool; for PoEML there is only a Spanish modeling tool available. It is recommended that the creators of VIDLs should put an effort into lowering this threshold. For acceptance and adoption of VIDLs, the development and enhancement of automated or semi-automated software tools supporting the modeling process will be inevitable.

For researchers, the presented evaluation might also spawn similar studies on other VIDLs and facilitate the understanding and coordination of research on VIDLs.

References

[1] W.M.G. Jochems, J.J.G. van Merrienboer, R. Koper, Integrated E-learning: implications for pedagogy, Technology and Organization, Routledge, Farmer, London, 2003.

[2] L. Botturi, T. Stubbs, Handbook of Visual Languages for Instructional Design, Information Science Reference, Hershey, PA, 2008.

[3] L. Lockyer, S. Bennett, S. Agostinho, B. Harper, Handbook of Research on Learning Design and Learning Objects: Issues, Applications and Technologies, 2008.

[4] A.S. Gibbons, L. Botturi, E.W. Boot, J. Nelson, Design languages, in: M. Spector, D. Merrill, J. v. Merriënboer, M. Driscoll (Eds.), Handbook of Research in Instructional Design, AECT, 2008, pp. 633–645.

[5] E.W. Boot, J. Nelson, J.J.G. van Merriënboer, A.S. Gibbons, Stratification, elaboration, and formalization of design documents: effects on the production of instructional materials, British Journal of Educational Technology 38 (5) (2007) 917–933.

[6] A. Sonderegger, J. Sauer, The influence of design aesthetics in usability testing: effects on user performance and perceived usability, Applied Ergonomics 41 (3) (2010) 403–410.

[7] L. Botturi, Visual languages for instructional design: an evaluation of the perception of E^{2}ML, Journal of Interactive Learning Research 16 (4) (2005) 329–351.

[8] L. Botturi, M. Derntl, E. Boot, K. Figl, A classification framework for educational modeling languages in instructional design, in: Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT), Kerkrade, 2006.

[9] K. Figl, M. Derntl, A comparison of visual instructional design languages for blended learning, in: Proceedings of the ED-MEDIA, Orlando, Florida, 2006, pp. 941–948.

[10] G. Costagliola, A.D. Lucia, F. Ferrucci, C. Gravino, G. Scanniello, Assessing the usability of a visual tool for the definition of e-learning processes, Journal of Visual Languages and Computing 19 (6) (2008) 721–737.

[11] S. Kelly, R. Pohjonen, Worst practices for domain-specific modeling, IEEE Software 26 (4) (2009) 22–29.

[12] D.L. Moody, The “physics” of notations: towards a scientific basis for constructing visual notations in software engineering, IEEE Transactions on Software Engineering 35 (5) (2009) 756–779.

[13] A.S. Gibbons, E. Brewer, Elementary principles of design languages and design notation systems for instructional design, in: M. Spector, D. Wiley (Eds.), Innovations to Instructional Technology: Essays in Honor of M. David Merrill, Lawrence Erlbaum, New Jersey, 2004.

[14] J. Rheinfrank, S. Evenson, Design languages, in: T. Winograd (Ed.), Bringing Design to Software, Addison-Wesley, Reading, MA, 1996.

[15] T. Winograd, A language/action perspective on the design of cooperative work, Human–Computer Interaction 3 (1) (1987) 3–30.

[16] Object Management Group, Unified Modeling Language^{TM} (UML®) 2.2. Available: 〈http://www.omg.org/spec/UML/2.2/〉, 2009.

[17] A. Rawlings, P. van Rosmalen, R. Koper, M. Rodríguez Artacho, P. Lefrere, Survey of Educational Modelling Languages (EMLs), in: CEN/ISSS WS/LT Learning Technologies Workshop, 2002.

[18] J. Lave, E. Wenger, Situated Learning: Legitimate Peripheral Participation, Cambridge University Press, Cambridge, UK, 1991.

[19] D. Schön, The reflective practitioner, How professionals think in action, Temple Smith, London, UK, 1983.

[20] H.H. Clark, S.E. Brennan, Grounding in communication, in: L.B. Resnick, J.M. Levine, S.D. Teasley (Eds.), Perspectives on Socially Shared Cognition, American Psychological Association, Hyattsville MD, 1991, pp. 127–149.

[21] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel, A pattern language—towns, buildings, Construction, Oxford University Press, New York, 1977.

[22] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns—Elements of Reusable Object-Oriented Software, Addison-Wesley, Reading, MA, 1995.

[23] A. Gemino, Y. Wand, A framework for empirical evaluation of conceptual modeling techniques, Requirements Engineering 9 (4) (2004) 248–260.

[24] S.H. Waters, A.S. Gibbons, Design languages, notation systems, and instructional technology: a case study, Educational Technology Research and Development 52 (2) (2004) 57–69.

[25] J. Sweller, Cognitive load during problem solving: effects on learning, Cognitive Science: A Multidisciplinary Journal 12 (2) (1988) 257–285.

[26] I. Vessey, Cognitive fit: a theory-based analysis of the graphs versus tables literature, Decision Sciences 22 (2) (1991) 219–240.

[27] T.R.G. Green, M. Petre, Usability analysis of visual programming environments: a ’cognitive dimensions’ framework, Journal of Visual Languages and Computing 7 (2) (1996).

[28] R.E. Mayer, Multimedia Learning, Cambridge University Press, 2001.

[29] J.H. Larkin, H.A. Simon, Why a diagram is (sometimes) worth ten thousand words, Cognitive Science 11 (1) (1987) 65–100.

[30] J.L. Santa, Spatial transformations of words and pictures, Journal of Experimental Psychology: Human Learning and Memory 3 (1977) 418–427.

[31] W. Winn, Encoding and retrieval of information in maps and diagrams, IEEE Transactions on Professional Communication 33 (3) (1990) 103–107.

[32] A. Bajaj, S. Rockwell, COGEVAL: a. propositional framework based on cognitive theories to evaluate conceptual models, in: K. Siau (Ed.), Advanced Topics in Database Research, Idea Group Publishing, 2005, pp. 255–282.

[33] I. Vessey, Cognitive fit: a theory-based analysis of the graphs versus tables literature, Decision Sciences 22 (2) (1991) 219–240.

[34] P.A. Kirschner, Cognitive load theory: implications of cognitive load theory on the design of learning, Learning and Instruction 12 (1) (2002) 1–10.

[35] G.A. Miller, The magical number seven, plus or minus two: some limits on our capacity for processing information, Psychological Review 63 (1956) 81–97.

[36] F. Gobet, P.C.R. Lane, S. Croker, P.C.H. Cheng, G. Jones, I. Oliver, J.M. Pine, Chunking mechanisms in human learning, Trends in Cognitive Sciences 5 (6) (2001) 236–243.

[37] J. Kim, J. Hahn, H. Hahn, How Do We, Understand a system with (So) many diagrams? Cognitive integration processes in diagrammatic reasoning, Information Systems Research 11 (3) (2000) 284–303.

[38] E.W. Boot, Building-block solutions for developing instructional software. Unpublished doctoral dissertation, Open University, The Netherlands, 2005.

[39] M. Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison Wesley, Boston, MA, 2003.

[40] A. Gibbons, What and how do designers design? A theory of design structure, Tech Trends 47 (2003) 22–27.

[41] I. Martinez-Ortiz, P. Moreno-Ger, J.L. Sierra, B. Fernandez-Manjon, Educational modeling languages, in: B. Fernandez-Manjon (Ed.), Computers and Education: E-learning, From Theory to Practice, Springer, 2007, pp. 27–40.

[42] B. Fernández-Manjón, J.M. Sánchez-Pérez, J.A. Gómez-Pulido, M.A. Vega-Rodríguez, J. Bravo-Rodríguez, Educational modeling languages. a conceptual introduction and a high-level classification, computers and education, E-Learning, From Theory to Practice, Springer, Netherlands, 2007.

[43] A. Maes, G. Poels, Evaluating quality of conceptual modelling scripts based on user perceptions, Data & Knowledge Engineering 63 (3) (2007) 701–724.

[44] L. Botturi, E^{2}ML: a visual language for the design of instruction, Educational Technology Research and Development 54 (3) (2006) 265–293.

[45] L. Botturi, E^{2}ML: a tool for sketching instructional designs, in: L. Botturi, T. Stubbs (Eds.), Handbook of Visual Languages for Instructional Design: Theories and Practices, Information Science Reference, 2008, pp. 112–132.

[46] L. Botturi, The Quail model for the classification of learning goals, Università della Svizzera italiana, Lugano.

[47] L. Botturi, E2ML—educational environment modeling language, in: Proceedings of ED-MEDIA, Honolulu, Hawaii, USA, 2003, pp. 304–311.

[48] L. Botturi, K. Belfer, Pedagogical patterns for online learning, in: Proceedings of E-Learn—World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education, Phoenix, Arizona, USA, 2003.

[49] M. Caeiro-Rodríguez, PoEML: a. separation-of-concerns proposal to instructional design, in: L. Botturi, T. Stubbs (Eds.), Handbook of Visual Languages in Instructional Design: Theories and Practices, Information Science Reference, 2008, pp. 185–209.

[50] M. Caeiro-Rodríguez, A modeling language for collaborative learning educational units—supporting the coordination of collaborative activities, in: Proceedings of ICEIS, 2007, pp. 334–339.

[51] IMS Global, IMS Learning Design Information Model. Available: 〈http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html〉, (2003, August 10, 2004).

[52] IMS Global, IMS Learning Design Best Practice and Implementation Guide. Available: 〈http://www.imsglobal.org/learningdesign/ldv1p0/imsld_bestv1p0.html〉, (2003, November 21, 2003).

[53] M. Caeiro-Rodríguez, M.J. Marcelino, M. Llamas-Nistal, L. Anido-Rifón, A.J. Mendes, Supporting the modeling of flexible educational units PoEML: a separation of concerns approach, Journal of Universal Computer Science 13 (7) (2007).

[54] M. Derntl, R. Motschnig-Pitrik, coUML—a visual language for modeling cooperative environments, in: L. Botturi, T. Stubbs (Eds.), Handbook of Visual Languages for Instructional Design: Theories and Practices, Information Science Reference, 2008, pp. 155–184.

[55] R. Motschnig-Pitrik, M. Derntl, Learning process models as mediators between didactical practice and web support, in: Proceedings of International Conference on Conceptual Modeling (ER 2005), Klagenfurt, Austria, 2005, pp. 112–127.

[56] K. Figl, J. Mendling, M. Strembeck, J. Recker, On the cognitive effectiveness of routing symbols in process modeling languages, in: Proceedings of Business Information Systems (BIS), Berlin, 2010.

[57] M. Wertheimer, Laws of organization in perceptual forms, in: W.D. Ellis (Ed.), A Sourcebook of Gestalt Psychology, Routledge and Kegan Paul, London, UK, 1938.

[58] OMG, Business Process Modeling Notation (BPMN) Version 1.2, ed., 2009.

[59] M. Derntl, P. Parrish, L. Botturi, Beauty and precision: weaving complex educational technology projects with visual instructional design languages, International Journal on E-Learning 9 (2) (2010) 185–202.