The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering

Daniel L. Moody, *Member*, *IEEE*

Abstract—Visual notations form an integral part of the language of software engineering (SE). Yet historically, SE researchers and notation designers have ignored or undervalued issues of visual representation. In evaluating and comparing notations, details of visual syntax are rarely discussed. In designing notations, the majority of effort is spent on semantics, with graphical conventions largely an afterthought. Typically, no design rationale, scientific or otherwise, is provided for visual representation choices. While SE has developed mature methods for evaluating and designing semantics, it lacks equivalent methods for visual syntax. This paper defines a set of principles for designing cognitively effective visual notations: ones that are optimized for human communication and problem solving. Together these form a design theory, called the Physics of Notations as it focuses on the physical (perceptual) properties of notations rather than their logical (semantic) properties. The principles were synthesized from theory and empirical evidence from a wide range of fields and rest on an explicit theory of how visual notations communicate. They can be used to evaluate, compare, and improve existing visual notations as well as to construct new ones. The paper identifies serious design flaws in some of the leading SE notations, together with practical suggestions for improving them. It also showcases some examples of visual notation design excellence from SE and other fields.

Index Terms—Modeling, analysis, diagrams, communication, visualization, visual syntax, concrete syntax.

"Very little is documented about why particular graphical conventions are used. Texts generally state what a particular symbol means without giving any rationale for the choice of symbols or saying why the symbol chosen is to be preferred to those already available. The reasons for choosing graphical conventions are generally shrouded in mystery" [53].

1 Introduction

Visual notations form an integral part of the language of software engineering (SE), and have dominated research and practice since its earliest beginnings. The first SE visual notation was Goldstine and von Neumann's program flowcharts, developed in the 1940s [36], and the ancestor of all modern SE visual notations [91]. This pattern continues to the present day with UML, the industry standard SE language, defined as "a visual language for visualizing, specifying, constructing, and documenting software intensive systems" [98]. Visual notations are used in all stages of the SE process, from requirements engineering through to maintenance. They play a particularly critical role in communicating with end users and customers as they are believed to convey information more effectively to nontechnical people than text [2].

Visual representations are effective because they tap into the capabilities of the powerful and highly parallel human visual system. We like receiving information in visual form and can process it very efficiently: Around a quarter of our brains are devoted to vision, more than all our other senses combined [64]. In addition, diagrams can convey information more concisely [27] and precisely than ordinary language [8], [69]. Information represented visually is also more likely to be remembered due to the **picture superiority effect** [38], [71].

1.1 The Nature of Visual Languages

Visual language is one of the oldest forms of knowledge representation and predates conventional written language by almost 25,000 years [133]. Visual notations differ from textual languages both in how they encode information and how they are processed by the human mind:

1. Textual languages encode information using sequences of characters, while visual languages encode information using spatial arrangements of graphic (and textual) elements. Textual representations are **one-dimensional** (linear), while visual representations are **two-dimensional** (spatial): A widely accepted definition of a diagram is a representation in which information is indexed by 2D location [69].

• *The author is with the Department of Information Systems & Change Management, University of Twente, Enschede 7500AE, The Netherlands. E-mail: d.l.moody@utwente.nl.*

Manuscript received 30 Apr. 2008; revised 13 Mar. 2009; accepted 22 Apr. 2009; published online 21 Oct. 2009.
Recommended for acceptance by J.-M. Favre, D. Gašević, R. Lämmel, and A. Winter.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number TSESI-2008-04-0165.
Digital Object Identifier no. 10.1109/TSE.2009.67.

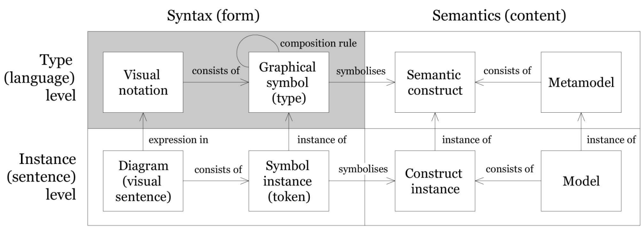


Fig. 1. Scope: this paper focuses on the top left-hand quadrant of the diagram (visual syntax): It specifically excludes semantic issues (right-hand side) and sentence-level issues (bottom).

2. Visual representations are also processed differently: according to **dual channel theory** [81], the human mind has separate systems for processing pictorial and verbal material. Visual representations are processed in parallel by the visual system, while textual representations are processed serially by the auditory system [8].

These differences mean that fundamentally different principles are required for evaluating and designing visual languages. However, such principles are far less developed than those available for textual languages [47], [148].

1.1.1 The Anatomy of a Visual Notation

A **visual notation** (or visual language, graphical notation, diagramming notation) consists of a set of **graphical symbols** (**visual vocabulary**), a set of **compositional rules** (**visual grammar**) and definitions of the meaning of each symbol (**visual semantics**). The visual vocabulary and visual grammar together form the **visual** (or **concrete**) **syntax**. Graphical symbols[1] are used to **symbolize** (perceptually represent) **semantic constructs**, typically defined by a **metamodel** [59]. The meanings of graphical symbols are defined by mapping them to the constructs they represent. A valid expression in a visual notation is called a **visual sentence** or **diagram**. Diagrams are composed of **symbol instances** (**tokens**), arranged according to the rules of the visual grammar.

Fig. 1 summarizes the scope of this paper. The paper says nothing about how to choose appropriate semantic constructs [140] or define their meaning [51], which are *semantic* issues. It only addresses how to visually represent a set of constructs once they have been defined. It also says nothing about how to effectively represent diagrams, e.g., [29], [85], which are *sentence level* issues (though decisions made at the notation level strongly determine the effectiveness of diagrams that can be produced).

1.2 The Dependent Variable: What Makes a "Good" Visual Notation?

The first problem that needs to be addressed in visual notation design is the lack of a clear **design goal** or **dependent variable**. To have any chance of success in a design task (or even to know if you have been successful), this must be clearly defined. Goals such as simplicity, aesthetics, expressiveness, and naturalness are often mentioned in the literature, but these are vaguely defined and highly subjective.

Visual notations are uniquely human-oriented representations: Their sole purpose is to facilitate human communication and problem solving [50]. To be most effective in doing this, they need to be optimized for processing by the human mind. **Cognitive effectiveness** is defined as the speed, ease, and accuracy with which a representation can be processed by the human mind [69]. This provides an operational definition of visual notation "goodness" that can be empirically evaluated. We propose this as the primary dependent variable for evaluating and comparing visual notations and the primary design goal in constructing them. Cognitive effectiveness determines the ability of visual notations to both communicate with business stakeholders and support design and problem solving by software engineers.

The cognitive effectiveness of visual notations is one of the most widely held (and infrequently challenged) assumptions in the IT field. However, cognitive effectiveness is not an intrinsic property of visual representations but something that must be *designed* into them [69]. Just putting information in graphical form does not guarantee that it will be worth a thousand of any set of words [21]. There can be huge differences between effective and ineffective diagrams and ineffective diagrams can be less effective than text.

1.3 Visual Syntax: An Important but Neglected Issue

Historically, SE researchers have ignored or undervalued the role of visual syntax. Notations tend to be evaluated based exclusively on their semantics, with issues of visual syntax rarely discussed, e.g., [99], [124]. When notations are empirically compared, differences are generally attributed to semantic rather than syntactic differences. For example, there have been many experimental studies comparing alternative notations for data modeling, e.g., [7], [122], [142]. All of these studies explain the empirical results in terms of their semantic differences, even though the differences in visual representation are just as great (if not greater). Visual representation thus acts as a significant (but unrecognized) confounding in most of these studies.

1.3.1 Visual Notation Design

Visual syntax is without question the "poor cousin" in notation design. Most effort is spent on designing semantics (what constructs to include and what they mean), with visual syntax (how to visually represent these constructs) often an afterthought. For example, ArchiMate [68] is a language recently developed as an international standard for enterprise architecture modeling. The following quote clearly shows the subordinate role of visual syntax in designing the language:

> "We do not put the notation of the ArchiMate language central but rather focus on the meaning of the language concepts and their relations. Of course, any modeling language needs a notation and we do supply a standard way of depicting the ArchiMate concepts but this is

1. The definition of graphical symbol includes 1D graphic elements (lines), 2D graphic elements (areas), 3D graphic elements (volumes), textual elements (labels), and spatial relationships. All of these types of elements can be used to construct the visual vocabulary of a notation (e.g., mind maps primarily consist of lines and labels).

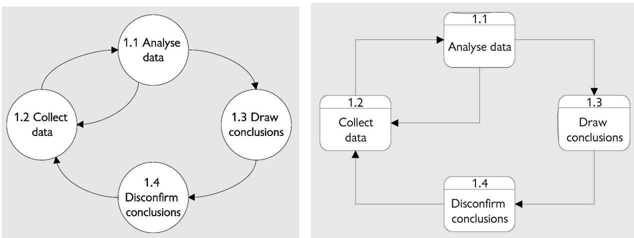


Fig. 2. Visual dialects: De Marco DFDs versus Gane & Sarson DFDs. Same semantics, different visual syntax: which is best?

subordinate to the architectural semantics of the language" [68].

Design rationale is the process of documenting design decisions made and the reasons they were made. This provides traceability in the design process and helps justify the final design [70]. Such rationale is conspicuously absent in design of SE visual notations: Graphical conventions are typically defined without reference to theory or empirical evidence, or justifications of any kind [53]. When explanations are provided (which is rare), they tend to be based on common sense. For example, the following quote comes from one of the leading notation designers in the SE field (and one of the architects of UML):

> "The selection of icons for any notation is a difficult task and not to be taken lightly. Indeed, icon design is largely an art not a science and requires a careful balance between expressiveness and simplicity. Our choice of the cloud icon suggests the boundaries of an abstraction, a concept that does not necessarily have plain or simple edges. The dashed lines indicate that clients generally only operate upon instances of a class, not the class itself" [14].

Relying on common sense to make such decisions is unreliable as the effects of graphic design choices are often counterintuitive [146]. However, even justifications of this kind are a great improvement on the usual situation where choice of symbols is not even discussed. A more common approach is to define symbols by assertion (e.g., "a class is represented by a rectangle") or examples without any rationale at all. For example, UML does not provide design rationale for any of its graphical conventions [98], suggesting that this is acceptable practice even for the industry standard language.

1.3.2 The Problem of Visual Dialects

While SE has developed mature methods for evaluating semantics of notations, comparable methods for evaluating visual notations are notably absent. As a result, many SE notations exist in multiple visual forms or **visual dialects**.[2] For example, two of the most successful notations in SE history are Data Flow Diagrams (DFDs) and ER Modeling. Both were developed in the 1970s, but a recent survey shows they are the two most commonly used modeling techniques in practice [26]. Both of these notations exist in multiple visual forms. DFDs exist in two semantically equivalent dialects (Fig. 2): the De Marco [27] dialect,

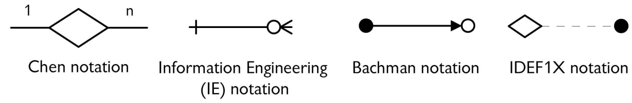


Fig. 3. Semantically equivalent forms of the ER Model. All of these visual forms express the same underlying relationship semantics.

consisting of circular "bubbles" and curved arrows, and the Gane and Sarson [31] dialect, consisting of rounded rectangles ("rountangles" [50]) and right-angled lines.

ER modeling also exists in a variety of visual dialects, with the Chen notation [18] the most commonly used in academic contexts and the Information Engineering (IE) notation [79] the most commonly used in practice. There are many other dialects associated with different CASE tools, development methodologies, and research groups. Fig. 3 shows the relationship conventions from some of the leading dialects.

Despite the fact that these notations have been used in practice for over 30 years, there is still no consensus on which is the best: Neither market forces nor Darwinian-style selection mechanisms seem to be able to identify the "fittest" alternative. Without sound principles for evaluating and comparing visual notations, there is no reliable way to resolve such debates. As a result, the choice of an appropriate dialect normally comes down to personal taste.

1.4 Why Visual Representation Is Important

The reason why visual notations have received so little attention is something of a mystery given their ubiquitous use in SE practice. One possible explanation is that researchers (especially those from mathematical backgrounds) see visual notations as being informal, and that therefore serious analysis can only take place at the level of their semantics. However, this is a misconception: visual languages are no less formal than textual ones [8], [51]. Another possible explanation is that methods for analyzing visual representations are less mature than those for analyzing verbal or mathematical representations [47], [76], [148] (which is something this paper aims to address). However, a third explanation is that SE researchers and notation designers simply consider visual syntax to be unimportant. While decisions about semantics (*content*) are treated with great care, decisions about visual representation (*form*) are often considered to be trivial or irrelevant: a matter of aesthetics rather than effectiveness [53].

This conventional wisdom is contradicted by research in diagrammatic reasoning, which shows that the *form* of representations has an equal, if not greater, influence on cognitive effectiveness as their *content* [69], [123], [152]. Human information processing is highly sensitive to the exact form in which information is presented to the senses: apparently minor changes in visual appearance can have dramatic impacts on understanding and problem solving performance [19], [69], [104], [123]. Empirical studies in SE have confirmed this: the visual form of notations significantly affects understanding especially by novices [53], [56], [57], [80], [94], [107], [108]. This suggests that decisions about visual representation are far from trivial and should be treated with as much care (if not more) as decisions about semantics.

2. As well as visual dialects (variants that exist in published form and used by a significant community of users), there are also countless **colloquial forms** (variants developed and used by particular individuals or organizations).

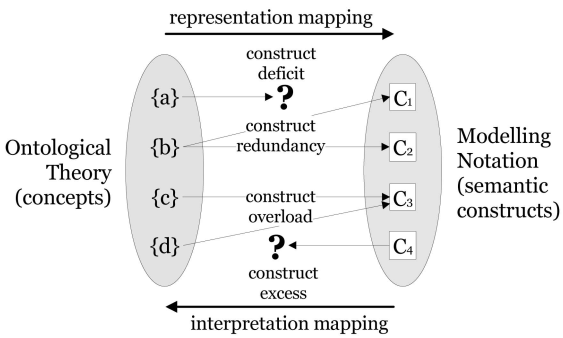


Fig. 4. Ontological analysis: There should be a 1:1 mapping between ontological concepts and notation constructs.

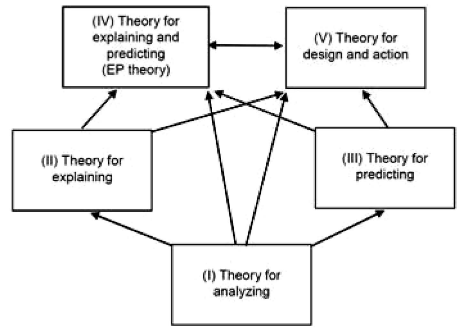


Fig. 5. Taxonomy of theory types [45]: The theory types represent a progression of evolutionary forms.

1.5 Objectives of This Paper

The use of visual notations in SE has a long history, from program flowcharts in the early days of computing (now largely extinct) to UML in the present. However, after more than half a century, the practice of designing SE visual notations lacks a scientific foundation. Currently, in evaluating, comparing, and constructing visual notations, we have little to go on but intuition and rule of thumb: We have neither theory nor a systematic body of empirical evidence to guide us.

This corresponds to what Alexander [1] calls an **unself-conscious design culture**: one that is not based on explicit design principles but on instinct, imitation, and tradition. One characteristic of such cultures is that designers are unable to explain their designs. Another is a lack of variety of different forms: Designers repeat the same patterns over and over again because they lack an understanding of the principles required to generate new ones. This may explain why SE visual notations look so similar to one another and change so little over time. For example, ER models as used in practice today look remarkably similar to Data Structure Diagrams, the first notation ever used to visualize database structures [3], [54]. Without knowledge of the underlying principles of graphic design, notation designers are unable to access the almost unlimited possibilities in the design space, and perpetuate the same (often flawed) ideas over time.

For visual notation design to progress from a "craft" to a design discipline (a **self-conscious design culture**), we need to define explicit principles for evaluating, comparing, and constructing visual notations [1]. There is a wealth of theory and empirical evidence that could be used to produce such principles, though mainly outside the SE field. One reason for the lack of progress in establishing a science of visual notation design may be the strong **Not Invented Here** (NIH) **effect** in SE research, which draws on research from other fields to only a minimal extent (it has a self-referencing rate of 98.1 percent [34]).

The goal of this paper is to establish the foundations for a science of visual notation design. It defines a theory of how visual notations communicate (Section 3) and based on this, a set of principles for designing cognitively effective visual notations (Section 4). A secondary goal is to raise awareness about the importance of visual representation issues in notation design, which have historically received very little attention.

2 Related Research

2.1 Ontological Analysis

Ontological analysis has become widely accepted as a way of evaluating SE notations [33], [120]. The leading ontology used for this purpose is the **Bunge-Wand-Weber (BWW) ontology**, originally published in this journal [140]. Many ontological analyses have been conducted on different SE notations, e.g., [41], [99], [143]. Ontological analysis involves a two-way mapping between a modeling notation and an ontology. The **interpretation mapping** describes the mapping from the notation to the ontology, while the **representation mapping** describes the inverse mapping [33]. According to the theory, there should be a one-to-one correspondence between the concepts in the ontology and constructs in the notation. If not, one or more of the following anomalies will occur (Fig. 4):

- **Construct deficit** exists when there is no construct in the notation corresponding to a particular ontological concept.
- **Construct overload** exists when a single notation construct can represent multiple ontological concepts.
- **Construct redundancy** exists when multiple notation constructs can be used to represent a single ontological concept.
- **Construct excess** exists when a notation construct does not correspond to any ontological concept.

If construct deficit exists, the notation is said to be **ontologically incomplete**; if any of the other three anomalies exist, it is **ontologically unclear**. The BWW ontology predicts that ontologically clear and complete notations will be more effective. In **Gregor's taxonomy of theory types** (Fig. 5), this represents a **Type IV theory**: a theory for explaining and predicting. Extensive empirical research has been conducted to validate the predictions of the theory, e.g., [13], [118].

Ontological analysis provides a way of evaluating the semantics of notations but specifically excludes visual representation aspects: It focuses on *content* rather than *form.* If two notations have the same semantics but different syntax (e.g., Fig. 2), ontological analysis cannot distinguish between them. Clearly, it would be desirable to have a comparable theory at the syntactic level so that visual syntax can also be evaluated in a sound manner.

2.2 Cognitive Dimensions (CDs) of Notations

Perhaps, the closest thing to a theory of visual notation design that currently exists in the IT field is the **CDs framework** [10], [42], [44]. This has emerged as the predominant theoretical paradigm in visual languages (VLs) research. However, detailed analysis shows that it has serious theoretical and practical limitations for evaluating and designing visual notations [87]:

- It is not specifically focused on visual notations and only applies to them as a special case (as a particular class of **cognitive artifacts**) [43].
- The dimensions are vaguely defined, often leading to confusion or misinterpretation in applying them [25], [43].
- The theoretical and empirical foundations for the dimensions are poorly defined [43].
- The dimensions lack clear operationalizations (evaluation procedures or metrics), which means that they can only be applied in a subjective manner [25], [43].
- It excludes visual representation issues as it is based solely on structural properties [11].
- It does not support evaluation as the dimensions simply define properties of notations and are not meant to be either "good" or "bad" [11], [44].
- It does not support design: The dimensions are not design guidelines and issues of effectiveness are excluded from its scope [11], [44].
- Its level of generality precludes specific predictions [11], meaning that it is **unfalsifiable**.

For all of these reasons, the CDs framework does *not* provide a scientific basis for evaluating and designing visual notations. In Gregor's taxonomy [45], it represents a **Type I Theory**: a theory for analyzing and describing. This is the earliest evolutionary form of theory and is appropriate when no prior theory exists. Such theories are traditionally regarded as **unscientific** as they lack testable propositions and cannot be falsified.[3] However, they play a critical role in the development of any research field: For example, the pioneering work of the early taxonomists in biology paved the way for powerful Type IV theories like Darwin's [45]. For this reason, a better term for such theories would be **prescientific**.

The CDs framework represents important pioneering work in advancing the analysis of visual notations beyond the level of intuition. However, it should not be regarded as the end point for theory development in this field but as a starting point for developing a more powerful, domain-specific theory (the goal of this paper).

3 Descriptive Theory: How Visual Notations Communicate

This section defines a theory of how visual notations communicate based on extant theories from communication, semiotics, graphic design, visual perception, and cognition. This defines a **descriptive (positive) theory** of visual notations, or, in Gregor's [45] terminology, a **Type IV theory**: a theory for explaining and predicting (Fig. 5). Only by understanding *how* and *why* visual notations communicate can we *improve* their ability to communicate: description provides the foundation for prescription.

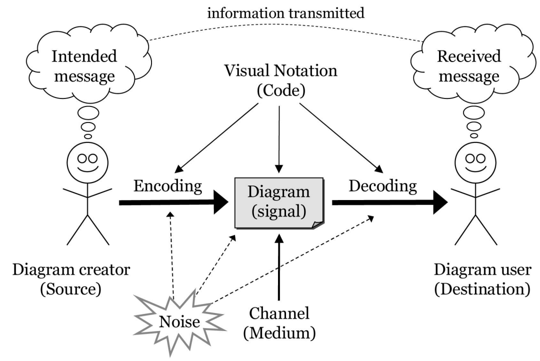


Fig. 6. Theory of diagrammatic communication: communication consists of two complementary processes: encoding and decoding.

3.1 Communication Theory

At the top level, the theory is an adaptation of Shannon and Weaver's widely accepted theory of communication [121] (or more precisely, a *specialization* of this theory to the domain of visual notations). As shown in Fig. 6, a diagram creator (**sender**) **encodes** information (**message**) in the form of a diagram (**signal**) and the diagram user (**receiver**) **decodes** this signal. The diagram is encoded using a visual notation (**code**), which defines a set of conventions that both sender and receiver understand. The medium (**channel**) is the physical form in which the diagram is presented (e.g., paper, whiteboard, and computer screen). **Noise** represents random variation in the signal which can interfere with communication. The effectiveness of communication is measured by the match between the intended message and the received message (**information transmitted**).

In this theory, communication consists of two complementary processes: **encoding** (expression) and **decoding** (interpretation). To optimize communication, we need to consider both sides:

- Encoding: What are the available options for encoding information in visual form? This defines the **design space**: the set of possible graphic encodings for a given message, which is virtually unlimited [8].
- Decoding: How are visual notations processed by the human mind? This defines the **solution space**: principles of human information processing provide the basis for choosing among the infinite possibilities in the design space.

3.2 The Design Space (Encoding Side)

The seminal work in the graphic design field is Bertin's *Semiology of Graphics* [8]. Bertin identified eight **visual variables** that can be used to graphically encode information (Fig. 7).[4] These are divided into **planar variables** (the two spatial dimensions) and **retinal variables** (features of the retinal image). Bertin's work is widely considered to be

3. Based on Popper's **criterion of falsifiability** [106], the most widely accepted criterion for distinguishing between science and pseudoscience.

4. Brightness is used instead of value (Bertin'term) to avoid confusion with everyday usage of the word "value." This variable defines a scale of relative lightness or darkness ($\text{black} = 0 \rightarrow \text{white} = 10$).

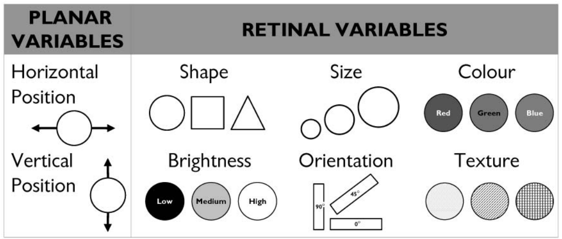


Fig. 7. Visual variables [8]: These define a set of elementary graphical techniques for constructing visual notations. A color version of this figure may be viewed at http://doi.ieeecomputersociety.org/10.1109/TSE.2009.67.

for graphic design what the periodic table is for chemistry: The visual variables define a set of atomic building blocks that can be used to construct any visual representation in the same way the periodic table can be used to construct any chemical compound. The visual variables thus define the dimensions of the graphic design space. The visual variables also define set of primitives—a **visual alphabet**—for constructing visual notations: Graphical symbols can be constructed by specifying particular values for visual variables (e.g., $\text{shape} = \text{rectangle}$, $\text{color} = \text{green}$). Notation designers can create an unlimited number of graphical symbols by combining the variables together in different ways.

3.2.1 Primary Notation, Secondary Notation, and Noise

Variations in visual variables convey meaning whether intended to or not. For example, size, color, and location of symbols have no formal meaning in UML class diagrams. However, if these variables vary (intentionally or unintentionally), they will convey information over and above the literal meaning of the diagram. There are strong perceptual interpretations associated with such variations that are difficult to consciously override. Such variations play a similar role to nonverbal communication (e.g., facial expressions and tone of voice) in speech and can either reinforce or interfere with the intended meaning.

Primary notation refers to the formal definition of a visual notation: the set of graphical symbols and their prescribed (literal) meanings. **Secondary (informal) notation** refers to the use of visual variables not formally specified in the notation to reinforce or clarify meaning, e.g., use of color to highlight information. Secondary notation is not a trivial matter: Petre [104] found that effective use of secondary notation was the major distinguishing feature between expert and novice use of a notation. **Visual noise** (accidental secondary notation) refers to unintentional use or random variation in visual variables that conflicts with or distorts the intended message [47], [97].

3.3 The Solution Space (Decoding Side)

Newell and Simon [92] showed that human beings can be considered as information processing systems. Designing cognitively effective visual notations can, therefore, be seen as a problem of optimizing them for processing by the human mind, in the same way that software systems are optimized for particular hardware. Principles of human graphical information processing provide the basis for making informed choices among the infinite possibilities in the graphic design space.

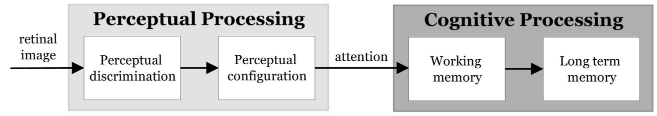


Fig. 8. The solution space: Maximizing cognitive effectiveness means optimizing notations for processing by the human mind.

Fig. 8 shows a model of human graphical information processing, which reflects current research in visual perception and cognition. Processing is divided into two phases: **perceptual processing** (seeing) and **cognitive processing** (understanding). Perceptual processes are automatic, very fast, and mostly executed in parallel, while cognitive processes operate under conscious control of attention and are relatively slow, effortful, and sequential. A major explanation for the cognitive advantages of diagrams is **computational offloading**: They shift some of the processing burden from the cognitive system to the perceptual system, which is faster and frees up scarce cognitive resources for other tasks. The extent to which diagrams exploit perceptual processing largely explains differences in their effectiveness [69], [104].

The stages in human graphical information processing are:

- **Perceptual discrimination**: Features of the retinal image (color, shape, etc.) are detected by specialized **feature detectors** [72], [131]. Based on this, the diagram is parsed into its constituent elements and separated from the background (figure-ground segregation) [101], [149].
- **Perceptual configuration**: Structure and relationships among diagram elements are inferred based on their visual characteristics [101], [149]. The **Gestalt Laws of Perceptual Organization** define how visual stimuli are organized into patterns or structures [145].
- **Attention management**: All or part of the perceptually processed image is brought into working memory under conscious control of attention. **Perceptual precedence** determines the order in which elements are attended to [66], [149].
- **Working memory**: This is a temporary storage area used for active processing, which reflects the current focus of attention. It has very limited capacity and duration and is a known bottleneck in visual information processing [65], [73].
- **Long-term memory**: To be understood, information from the diagram must be integrated with prior knowledge stored in long-term memory. This is a permanent storage area that has unlimited capacity and duration but is relatively slow [64]. Differences in prior knowledge (**expert-novice differences**) greatly affect speed and accuracy of processing.

4 Prescriptive Theory: Principles for Designing Effective Visual Notations

This section defines a set of principles for designing cognitively effective visual notations. These form a **prescriptive theory** for visual notations or, in Gregor's [45]

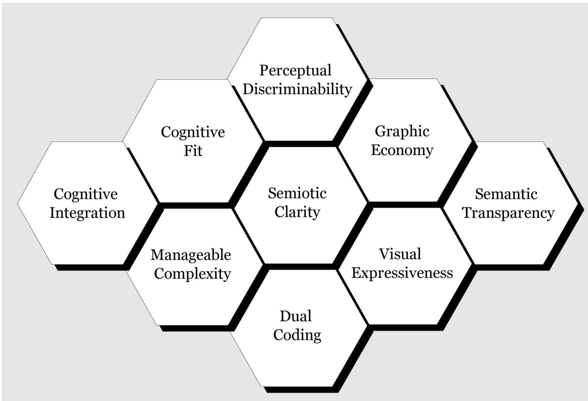


Fig. 9. Principles for designing cognitively effective visual notations: The modular ("honeycomb") structure supports modifications and extensions to the principles.

terminology, **a theory for design and action (Type V)** (Fig. 5). Defining explicit principles transforms visual notation design from an unselfconscious process (craft) into a selfconscious process (design discipline).

The principles were developed using a **best evidence synthesis** approach: They were synthesized from theory and empirical evidence about cognitive effectiveness of visual representations. This differs from approaches like [29], which rely on codifying craft knowledge or the CDs framework [44], which uses a combination of craft and scientific knowledge. The resulting principles are summarized in Fig. 9 (as a theory about visual representation should have a visual representation!). The modular structure makes it easy to add or remove principles, emphasizing that the principles are not fixed or immutable but can be modified or extended by future research. Each principle is defined by:

- Name: All principles are named in a positive sense, and represent desirable properties of notations. This means that they can be used as both evaluation criteria and design goals.
- Semantic (theoretical) definition: A one-sentence statement of what it means (included in the heading for each principle). In keeping with the prescriptive nature of a design theory, these take the form of imperative statements or "shoulds" [46].
- Operational (empirical) definition: Evaluation procedures and/or metrics are defined for most principles.
- Design strategies: Ways of improving notations with respect to the principle.
- Exemplars and counterexemplars: Examples of notations that satisfy the principle (**design excellence**) and ones that violate it (**design mediocrity**). These are drawn from SE as well as other fields.

References to principles in the following text are indicated by underlining.

4.1 Principle of Semiotic Clarity: There Should Be a 1:1 Correspondence between Semantic Constructs and Graphical Symbols

According to **Goodman's theory of symbols** [37], 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. Natural languages are not notational systems as they contain synonyms and homonyms but many artificial languages (e.g., musical notation and mathematical notation) are. The requirements of a notational system constrain the allowable expressions in a language to maximize precision, expressiveness, and parsimony, which are desirable design goals for SE notations. When there is not a one-to-one correspondence between constructs and symbols, one or more of the following anomalies can occur (using similar terms to those used in ontological analysis) (Fig. 10):

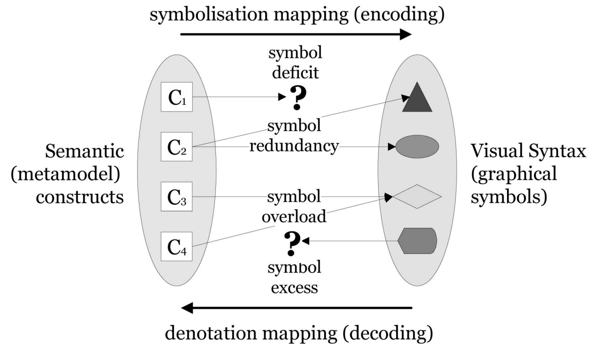


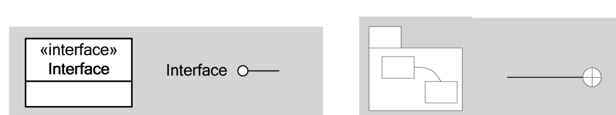
Fig. 10. Semiotic clarity: There should be a 1:1 correspondence between semantic constructs and graphical symbols.

- **Symbol redundancy** occurs when multiple graphical symbols can be used to represent the same semantic construct.
- **Symbol overload** occurs when two different constructs can be represented by the same graphical symbol.
- **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.

This principle represents an extension of ontological analysis to the level of visual syntax, though its theoretical grounding is in **semiotics** rather than ontology.

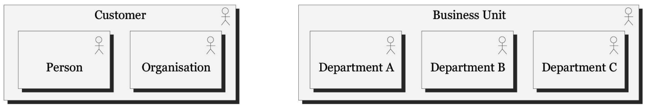
4.1.1 *Symbol Redundancy*

Instances of symbol redundancy are called **synographs** (the equivalent of synonyms in textual languages). Symbol redundancy places a burden of choice on the notation user to decide which symbol to use and on the reader to remember multiple representations of the same construct. There are many instances of symbol redundancy in UML: Fig. 11 shows an example of a construct synograph and a relationship synograph.



(a) (b)

Fig. 11. Symbol redundancy (synographs) in UML: There are alternative graphical symbols for (a) interfaces on class diagrams and (b) package relationships on Package Diagrams.



(a) (b)

Fig. 12. Symbol overload (homographs) in ArchiMate: The same graphical convention can be used to represent different types of relationships: (a) generalization and (b) composition.

4.1.2 *Symbol Overload*

Instances of symbol overload are called **homographs** (the visual equivalent of homonyms). This is the worst type of anomaly as it leads to ambiguity and the potential for misinterpretation [37]. It also violates one of the basic properties of the symbol system of graphics, **monosemy**, which means that each symbol should have a single meaning, defined in advance and independent of context [8]. ArchiMate violates this as the same graphical convention (spatial enclosure) can be used to show a range of different relationship types: inheritance, assignment, aggregation, composition, and grouping (Fig. 12).

4.1.3 *Symbol Excess*

Symbol excess adds to diagrammatic complexity (Complexity Management) and graphic complexity (Graphic Economy), which both adversely affect understanding [94]. UML includes several instances of symbol excess, the most obvious being the ubiquitous **comment**, which can appear on all diagram types (Fig. 13). These are used to clarify meaning of diagrams and perform a similar role to comments in programs. Including textual explanations on diagrams is a useful practice, but it is neither necessary nor desirable to show them using explicit symbols: This is an example of what graphic designers call "boxitis" [132], [133], [147]. Such symbols add visual clutter to diagrams and confound their interpretation by making it more likely they will be interpreted as constructs. A simple block of text would be less intrusive and less likely to be misinterpreted.

4.1.4 *Symbol Deficit*

In most SE contexts, symbol deficit is desirable to limit diagrammatic complexity (Complexity Management) and graphic complexity (Graphic Economy). Given the semantic complexity of most SE notations, it is usually counterproductive to try to show all constructs on the diagram. This represents a point of difference with ontological analysis where all deviations from a 1:1 mapping are considered harmful [33].

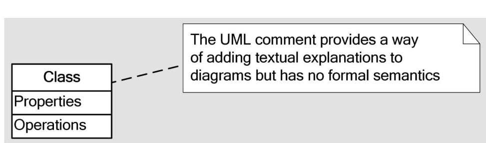


Fig. 13. Symbol excess in UML: The comment is a useful notational feature but should not be shown using a graphical symbol.

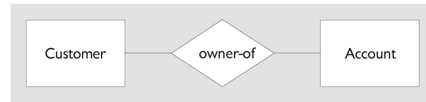


Fig. 14. Experimental studies show that rectangles and diamonds in ER diagrams are frequently confused by novices.

4.2 Principle of Perceptual Discriminability: Different Symbols Should Be Clearly Distinguishable from Each Other

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 (Fig. 8). Accurate discrimination between symbols is a prerequisite for accurate interpretation of diagrams [148].

4.2.1 *Visual Distance*

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 (measured by the number of perceptible steps). Each visual variable has an infinite number of physical variations but only a finite number of **perceptible steps** (values that are reliably discriminable by the human mind) [8]. Research in psychophysics has established **discriminability thresholds** for most visual variables, which can be used to guide choice of values [8], [126], [127]. In general, the greater the visual distance between symbols, the faster and more accurately they will be recognized [149]. If differences are too subtle, errors in interpretation may result. Requirements for discriminability are higher for novices than for experts as we are able to make much finer distinctions with experience [15].

Many SE notations have very poor discriminability, and consist of shapes and connecting lines that are visually too similar. For example, experimental studies show that entities and relationships are often confused by novices on ER diagrams (Fig. 14) [94]. The visual distance between the symbols is relatively small, as they differ on only one visual variable (shape) and the shapes belong to the same family (quadrilaterals).

4.2.2 *The Primacy of Shape*

Of all visual variables, shape plays a special role in discriminating between symbols as it represents the primary basis on which we identify objects in the real world. In fact, theories of object recognition differ only to the extent that they consider object representations to be based only on shape or if other features are also involved [9], [78], [114]. For this reason, shape should be used as the primary visual variable for distinguishing between different semantic constructs. For example, discriminability of ER diagrams could be improved by using shapes from different families (Fig. 15). It is surprising that after more than three decades

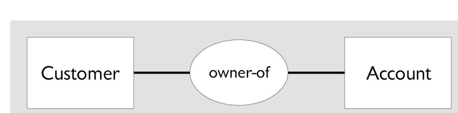


Fig. 15. Entities and relationships could be made more discriminable by using shapes from different families.

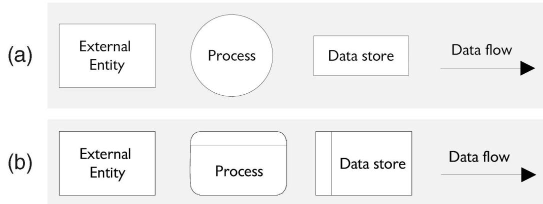


Fig. 16. Graphic excellence versus graphic mediocrity: (a) De Marco DFDs use clearly distinguishable shapes for all constructs, while (b) Gane and Sarson DFDs use rectangle variants.

Fig. 17. Redundant coding: Using multiple visual variables (shape + color) to distinguish between symbols. A color version of this figure may be viewed at http://doi.ieeecomputersociety.org/10.1109/TSE.2009.67.

TABLE 1
Relationship Types on UML Class Diagrams

Aggregation	Association (navigable)	Association (non-navigable)	Association class relationship	Composition
Constraint	Dependency	Generalisation	Generalisation set	Interface (provided)
Interface (required)	N-ary association	Note reference	Package containment	Package import (public) «import»
Package import (private) «access»	Package merge «merge»	Realisation	Substitution «substitute»	Usage «use»

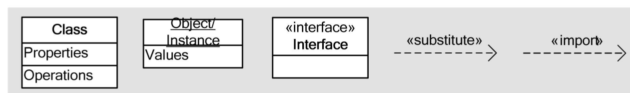


Fig. 18. Textual differentiation: UML uses labels and typographical characteristics to distinguish between symbols.

of use and the empirically established confusion between them that such a change has not been made already.

Most SE notations use a perceptually limited repertoire of shapes, mostly rectangle variants [104]. This is surprising, as shape has the largest range of values (**capacity**) of any visual variable and it is the *only* visual variable used in most SE notations (see Visual Expressiveness). As an example of **design excellence**, De Marco-style DFDs use clearly discriminable shapes to represent different constructs: all come from different shape families and differences between them can be detected preattentively (including the difference between open and closed shapes [130][5]). In contrast, the Gane and Sarson dialect uses rectangle variants for all constructs (Fig. 16).

4.2.3 Redundant Coding

Redundancy is an important technique in communication theory to reduce errors and counteract noise [40], [121]. The visual distance between symbols can be increased by **redundant coding**: using multiple visual variables to distinguish between them [73]. As an example, color could be used to improve discriminability between entities and relationships in ER diagrams (Fig. 17). Most SE diagramming notations rely on only a single variable to distinguish between symbols, which is less robust to misinterpretation.

4.2.4 Perceptual Popout

According to **feature integration theory**, visual elements with unique values for at least one visual variable can be detected preattentively and in parallel across the visual field [109], [131]. Such elements appear to "**pop out**" from a display without conscious effort. On the other hand, visual elements that are differentiated by unique combinations of values (**conjunctions**) require serial search, which is much slower and error-prone.

The clear implication of this for visual notation design is that each graphical symbol should have a unique value on at least one visual variable. This requirement is violated in UML Class Diagrams, where three visual variables (shape, brightness, and texture) are used in combinatorial fashion to distinguish between 20 different types of relationships (Table 1). Discrimination between relationship types relies on unique combinations of values, sometimes in combination with labels or context. Only two of the relationship types have unique values on any variable, which precludes **perceptual popout** in most cases.

Note that redundant coding is different to using conjunctions to distinguish between symbols. Conjunctions use visual variables in a **multiplicative** (**combinatorial**) manner: Each variable is *necessary* but not sufficient to distinguish between symbols (only combinations of values are unique). In redundant coding, variables are used in an **additive** (**reinforcing**) manner: Each variable is *sufficient* on its own to distinguish between symbols. For example, in Fig. 17, each symbol has a unique value for each variable (shape and color), so in principle, either could be used to distinguish between them.

4.2.5 Textual Differentiation

SE notations sometimes rely on text to distinguish between symbols. For example, UML frequently uses text and typographic characteristics (bold, italics, and underlining) to distinguish between element and relationship types (Fig. 18). ***i**** [150], one of the leading requirements engineering notations, uses labels to differentiate between most of its relationship types [88]. Symbols that differ only on textual characteristics are technically homographs, as they have zero visual distance (Semiotic Clarity).

Textual differentiation of symbols is a common but cognitively ineffective way of dealing with excessive graphic complexity (Graphic Economy) as text processing relies on less efficient cognitive processes. To maximize discriminability, symbols should be differentiated using visual variables so that differences can be detected automatically and in parallel by the perceptual system. Textual differentiation of symbols also confounds the role of text in diagrams. Labels play a critical role at the **sentence**

5. This may have been more a matter of good luck than good design as the phenomenon of perceptual popout was only discovered after DFDs were designed.

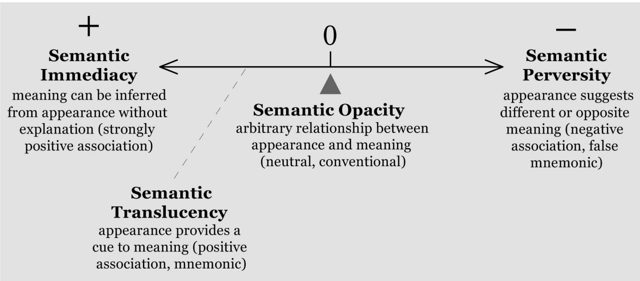


Fig. 19. Semantic transparency defines the degree of association between a symbol's form and content.

(diagram) level in distinguishing between **symbol instances (tokens)** and defining their correspondence to the real world. For example, with the labels in Fig. 17, the two entity types would be indistinguishable and the diagram would lack real-world meaning. Using labels to distinguish between **symbol types** (at the language level) confounds their role. Also, when labels are used to distinguish between relationship types (Fig. 18), it precludes the use of domain-relevant names for relationships. Text is an effective way to distinguish between symbol instances but not between symbol types.

4.3 Principle of Semantic Transparency: Use Visual Representations Whose Appearance Suggests Their Meaning

Semantic transparency is defined as the extent to which the meaning of a symbol can be inferred from its appearance. While Perceptual Discriminability simply requires that symbols should be *different* from each other, this principle requires that they provide cues to their meaning (form *implies* content). 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.[6] Semantically transparent representations reduce cognitive load because they have built-in mnemonics: Their meaning can be either be perceived directly or easily learned [104]. Semantic transparency is not a binary state but a continuum (Fig. 19).

- A symbol is **semantically immediate** if a novice reader would be able to infer its meaning from its appearance alone (e.g., a stick figure to represent a person).
- A symbol is **semantically opaque** (or **conventional**) if there is a purely arbitrary relationship between its appearance and its meaning (e.g., rectangles in ER diagrams): This represents the zero point on the scale.
- A symbol is **semantically perverse** (or a **false mnemonic)** if a novice reader would be likely to infer a different (or even opposite) meaning from its appearance: This represents a negative value on the scale. Some UML conventions (e.g., package merge) have this unfortunate property.
- In between semantic immediacy and opacity, there are varying degrees of **semantic translucency** (**mnemonicity**), where symbols provide a cue to their meaning (and therefore, an aid to memory) but require some initial explanation.

The most obvious form of association between symbols and their referent concepts is *perceptual resemblance*, but many others are also possible: *common logical properties* (e.g., spatial inclusion to show subsets in Venn diagrams), *functional similarities* (e.g., a trash can for deleted items), *metaphor* (e.g., crossed swords for conflict), and *cultural associations* (e.g., red for danger). Semantic transparency corresponds to Norman's [95] concept of **natural mappings**: the use of physical analogies, visual metaphors, and cultural associations to design physical objects. It also corresponds to Gurr's concept of systematicity: matching semantic properties of represented objects to visual properties of symbols [47].

4.3.1 Icons (Perceptual Resemblance)

Icons are symbols that perceptually resemble the concepts they represent [103]. This reflects a basic distinction in semiotics between **symbolic** and **iconic** signs [103]. Iconic representations speed up recognition and recall and improve intelligibility of diagrams to naive users [15], [80]. They also make diagrams more accessible to novices: A representation composed of pictures appears less daunting than one composed of abstract symbols [104]. Finally, they make diagrams more visually appealing: people prefer real objects to abstract shapes [5], [104].

Icons are pervasively used in HCI (e.g., graphical user interfaces) [93] and cartography [112] but surprisingly rarely in SE visual notations. Most SE notations rely exclusively on abstract geometrical shapes to represent constructs [104]. Such symbols don't convey anything about their referent concepts: Their meaning is purely conventional and must be learned. **Rich pictures** [17] are a rare example of an SE visual notation that is almost exclusively iconic: The resulting diagrams are visually appealing and cartoon-like, unlike the dull, technical-looking diagrams typically used in SE practice (Fig. 20).

4.3.2 Semantically Transparent Relationships

Semantic transparency also applies to representing relationships. Certain spatial arrangements of visual elements predispose people toward a particular interpretation of the relationship among them even before the meaning of the elements is known [47], [148]. In a set of experiments designed to discover how diagrams are spontaneously interpreted, Winn [148] used diagram elements with nonsense labels arranged in different spatial configurations and asked subjects to describe the relationships among them. The results are summarized in Fig. 21.

Most SE visual notations make only limited use of such relationships and rely mainly on different types of connecting lines to represent relationships (e.g., Table 1). Such links are very versatile but provide few clues to their meaning as they can be used to represent almost any type of relationship. Fig. 22 shows how spatial relationships (spatial enclosure and overlap) could be used to represent overlapping subtypes in ER models. This conveys the relationship among the entities

6. The "naturalness" of notations is a contentious issue in SE research, with authors often arguing that one representation is more "natural" or "intuitive" than another. In most cases, such claims are based on opinion or conjecture. However, at least for visual representations, it is possible to make such claims based on empirical evidence.

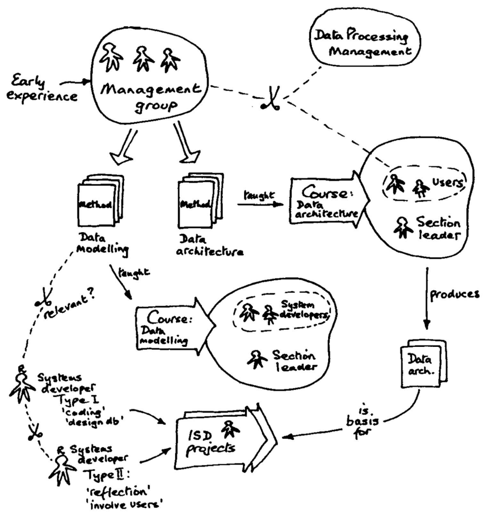


Fig. 20. Rich pictures: a rare but highly effective example of the use of iconic representations in SE [17].

in a more semantically transparent way than using arrows, so is more likely to be interpreted correctly and more easily remembered.

The representation on the right of Fig. 22 obeys the principle of systematicity [47] as spatial enclosure has the same logical properties as the IS-A (subtype) relationship: transitivity, irreflexivity, and asymmetry. The extent to which diagrams exploit such mappings can greatly improve their effectiveness for problem solving [19], [47], [69], [115]. It also prevents errors in using the notation, as the geometric properties of the spatial relationship enforce the logical constraints on the relationship [20], [115]. It also makes subtype relationships more discriminable from all other types of relationships (Perceptual Discriminability).

Note that spatial enclosure could *not* be used to represent generalization in UML. First, spatial enclosure is already used for many other purposes in UML and so would lead to symbol overload (Semiotic Clarity). Second, it would violate the principle of systematicity. UML supports **multiple inheritance**, which allows many-to-many relationships between superclasses and subclasses. This conflicts with the geometric properties of spatial enclosure, which allows only one-to-many relationships.

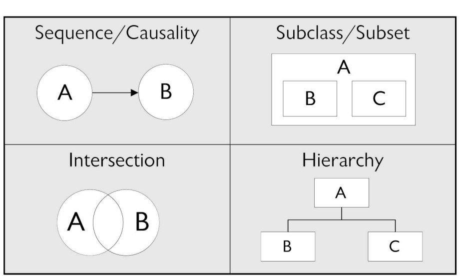
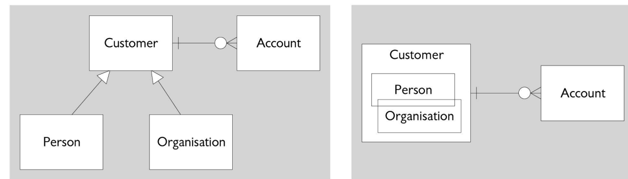


Fig. 21. Semantically transparent relationships: these spatial relationships are interpreted in a spontaneous or natural way [148].



(a) (b)

Fig. 22. Spatial enclosure and overlap (b) convey the concept of overlapping subtypes in a more semantically transparent way than arrows (a). Both representations convey the same semantics: that a customer can be a person, an organization, or both.

4.4 Principle of Complexity Management: Include Explicit Mechanisms for Dealing with Complexity

Complexity management refers to the ability of a visual notation to represent information without overloading the human mind. Complexity is also one of the defining characteristics of the SE field, where complexity levels exceed those in any other discipline [28], [32]. It is also one of the most intractable problems in visual notation design: a well-known problem with visual representations is that they do not scale well [22]. Currently, complexity management is incorporated into SE visual notations in notation-specific ways (**point solutions** [16]) or not at all: this principle defines general requirements for a solution to complexity management in visual notations.

In this context, "complexity" refers to **diagrammatic complexity**,[7] which is measured by the number of elements (symbol instances or tokens) on a diagram. While this is ostensibly a diagram (sentence) level issue, it requires notational features to solve it. Complexity has a major effect on cognitive effectiveness as the amount of information that can be effectively conveyed by a single diagram is limited by human perceptual and cognitive abilities:

- Perceptual limits: The ability to discriminate between diagram elements increases with diagram size [102].
- Cognitive limits: The number of diagram elements that can be comprehended at a time is limited by working-memory capacity. When this is exceeded, a state of **cognitive overload** ensues and comprehension degrades rapidly [83].

Effective complexity management is especially important when dealing with novices, who are less equipped to cope with complexity [128]. Excessive complexity is one of the major barriers to end-user understanding of SE diagrams [84], [119].

Surprisingly, some of the leading SE visual notations lack mechanisms for managing complexity. In the absence of these, problems must be represented as single "monolithic" diagrams, no matter how complex they become. For example, the ER model has been in use for over three decades and still lacks such mechanisms. As a result, "absurdly complex diagrams" are often produced, in practice, that overwhelm end users [62] (see Fig. 23 for a real-world example). ***i****, a language developed more

7. This should be clearly distinguished from **graphic complexity**, which measures the number of symbol types in a notation (Graphic Economy).

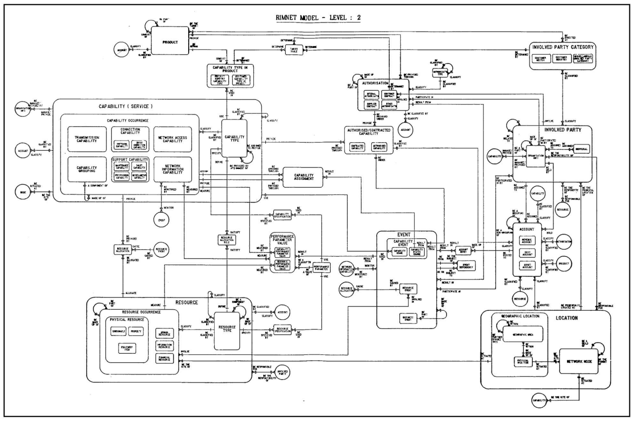


Fig. 23. In the absence of complexity management mechanisms, ER models must be shown as single monolithic diagrams.

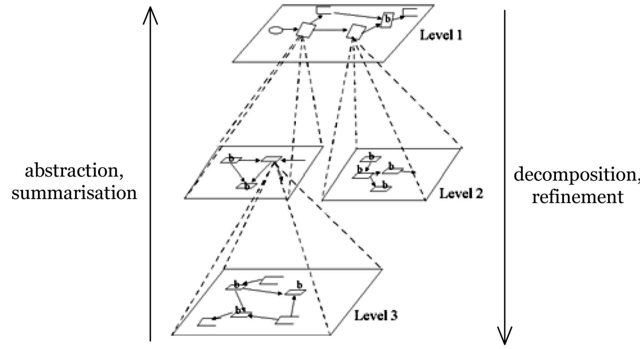


Fig. 24. Hierarchical organization allows a system to be represented at multiple levels of abstraction, with complexity manageable at each level.

recently and specifically for communication with end users, also lacks such mechanisms [88], showing that this lesson has still not been learned over time.

In the absence of explicit complexity management mechanisms, practitioners often develop informal solutions to the problem, meaning that it is solved by **secondary notation**. However, this is undesirable as it results in highly idiosyncratic solutions and proliferation of **colloquial forms**. According to ontological theory [143], complexity management mechanisms are essential elements of all SE notations, which means that they should be included in the **primary notation**.

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 [143]. However, while ontological theory defines the semantic constructs required to support complexity management, it does not define a syntactic solution to the problem.

4.4.1 Modularization

The most common way of reducing complexity of large systems is to divide them into smaller parts or **subsystems**: This is called modularization. Baldwin and Clark [4] have proposed this as a unifying paradigm for the IT industry, which helps cope with the mind-boggling levels of complexity encountered. To avoid overloading the human mind, notations must provide the ability to divide large diagrams into perceptually and cognitively manageable “chunks.” **Cognitive load theory** shows that reducing the amount of information presented at a time to within the limitations of working memory improves speed and accuracy of understanding and facilitates deep understanding of information content [81], [128]. Empirical studies show that modularizing SE diagrams can improve end-user understanding and verification by more than 50 percent [84]. Modularization can take place in a top-down (e.g., decomposition *à la* DFDs) or bottom-up manner (e.g., packages in UML).

Modularization requires certain semantic constructs to be included in the notation: either a subsystem construct (e.g., UML packages) or decomposable constructs (e.g., UML activities). But including such constructs is not enough: For this to be effective at the syntactic level, diagrammatic conventions for decomposing diagrams need to be defined. In particular, UML packages provide a *semantic* but not a *syntactic* solution to the problem of complexity management (which requires more than defining a graphical symbol to represent the construct).

4.4.2 Hierarchy (Levels of Abstraction)

Hierarchy is one of the most effective ways of organizing complexity for human comprehension as it allows systems to be represented at different levels of detail, with complexity manageable at each level [30]. This supports **top down understanding**, which has been shown to improve understanding of SE diagrams [94]. Simon [125] proposed hierarchy as a general architecture for structuring complex systems.

Hierarchical organization is a natural result of top-down decomposition: When a system is decomposed to multiple levels, the result will usually be a hierarchy of diagrams at different levels of abstraction. When modularization takes place in a bottom-up manner, higher level diagrams need to be explicitly created by a process of **summarization** (**abstraction**). Elements on higher level diagrams “explode” to complete diagrams at the next level, following the principle of **recursive decomposition** [27] (Fig. 24). This simple mechanism supports both modularization and hierarchical structuring and is the common denominator among visual notations that effectively manage complexity (e.g., UML Activity Diagrams, Statecharts). Visual languages that support recursive decomposition are called **hierarchical visual languages** [24].

An example of excellence in managing diagrammatic complexity in the SE field (and probably one of the best examples in any field) are DFDs. These incorporate modularity, hierarchical structuring, and an explicit complexity limit: 7 ± 2 “bubbles” per diagram (consistent with the known limits of working memory). In this respect, they were ahead of their time and still are: Many more recent SE notations could learn from their example. In particular, UML lacks a consistent approach to complexity management: different diagram types have different ways of dealing with complexity, while some diagram types (e.g., class diagrams) have none at all.

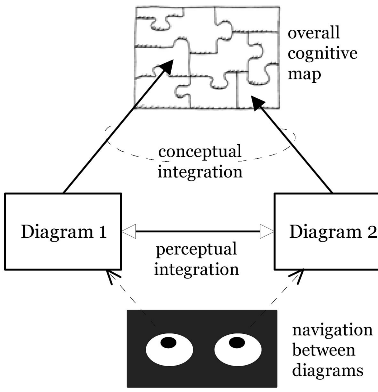


Fig. 25. Cognitive integration: When multiple diagrams are used to represent a domain, explicit mechanisms are needed to support perceptual and conceptual integration.

4.5 Principle of Cognitive Integration: Include Explicit Mechanisms to Support Integration of Information from Different Diagrams

Cognitive integration only applies when multiple diagrams are used to represent a system. This is a critical issue in SE, where problems are typically represented by systems of diagrams rather than single diagrams. It applies equally to diagrams of the same type (**homogeneous integration**)—for example, a set of leveled DFDs—or diagrams of different types (**heterogeneous integration**)—for example, a suite of UML diagrams or ArchiMate views. This principle is closely related to Complexity Management, which leads to multiple diagrams as a result of modularization, but applies even when modularity is not used (due to heterogeneous integration).

Representing systems using multiple diagrams places additional cognitive demands on the reader to mentally integrate information from different diagrams and keep track of where they are [123]. Kim et al. [48], [61] have proposed a theory to address this issue, called the **cognitive integration of diagrams**. This is an important contribution as most previous research in diagrammatic reasoning has focused on single diagrams. According to their theory (which has been validated in an SE context), for multi-diagram representations to be cognitively effective, they must include explicit mechanisms to support (Fig. 25):

- **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.

4.5.1 Conceptual Integration

One important mechanism to support conceptual integration is a **summary** (**long shot**) **diagram**, which provides a view of the system as a whole. This acts as an overall cognitive map into which information from individual diagrams can be assembled [61], [110]. Examples of such diagrams are **rich pictures** in Soft System Methodology and **context diagrams** in DFDs. In homogeneous integration, such a diagram is a natural result of hierarchical structuring: It is the "root" of the hierarchy. However, in heterogeneous integration, a new diagram will need to be created for this purpose.

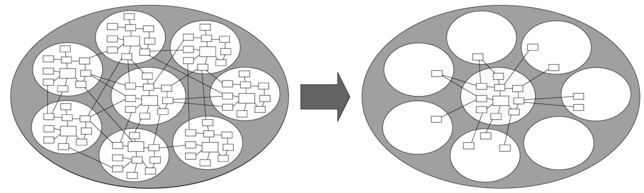


Fig. 26. Contextualization: Each diagram should include its surrounding context to show how it fits into the system as a whole.

Contextualization (or **focus** + **context**) is a technique used in information visualization where the part of a system of current interest (**focus**) is displayed in the **context** of the system as a whole [67], [135]. In a diagramming context, this means including contextual information on each diagram showing its relationships to elements on other diagrams. The simplest and most effective way to do this is to include all directly related elements from other diagrams (its "immediate neighborhood") as **foreign elements** (Fig. 26). Including overlap between diagrams in this way allows each element in the system of diagrams to be understood in terms of its relationships to all other elements, which supports conceptual integration. It also supports perceptual integration by simplifying transitions between related diagrams, through the mechanism of **visual momentum** [141].

4.5.2 Perceptual Integration

There are a range of mechanisms that can be used to support perceptual integration, which draw on the design of physical spaces (urban planning), virtual spaces (HCI), and graphical spaces (cartography and information visualization). Whether navigating around a city, a Web site, an atlas, or a set of diagrams, **wayfinding** follows the same four stages [75]:

- Orientation: Where am I?
- Route choice: Where can I go?
- Route monitoring: Am I on the right path?
- Destination recognition: Am I there yet?

Clear labeling of diagrams (**identification**) supports orientation and destination recognition. **Level numbering** (as used to show structure of documents) supports orientation by showing the user where they are in the system of diagrams. Including **navigational cues** on diagrams (**signposting**) supports route choice. A **navigational map**, showing all diagrams and the navigation paths between them, supports orientation, route monitoring, and route choice: Readers can use this to navigate through the information space and keep track of where they are.

No existing notations fully satisfy this principle (which is not surprising as the theory is relatively new), but DFDs come closest: They include a long-shot diagram (**context diagram**), orientation information (**level numbering**), and contextualization (**balancing**). However, they lack a navigational map, don't support horizontal (lateral) navigation, and downward navigation is ambiguous. UML is a counter-exemplar as it lacks a long-shot diagram and the relationships between different diagram types are unclear.

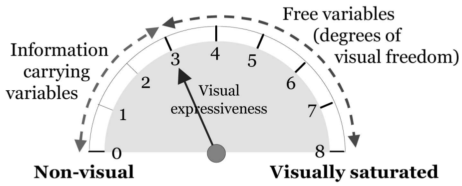


Fig. 27. Visual expressiveness.

4.6 Principle of Visual Expressiveness: Use the Full Range and Capacities of Visual Variables

Visual expressiveness is defined as the number of visual variables used in a notation. This measures utilization of the graphic design space. While **visual distance** (Perceptual Discriminability) measures pairwise visual variation between symbols, visual expressiveness measures visual variation across the entire visual vocabulary. Using a range of visual variables results in a perceptually enriched representation that exploits multiple visual communication channels and maximizes computational offloading.

Visual expressiveness partitions the set of visual variables into two subsets (Fig. 27):

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

The number of free variables is called the **degrees of visual freedom** and is the inverse of visual expressiveness. A notation with no information-carrying visual variables (visual expressiveness = zero; eight degrees of visual freedom) is called **nonvisual** (or textual), while a notation that uses all visual variables (visual expressiveness = eight}; zero degrees of visual freedom) is **visually saturated**.

Most SE notations use only a single visual variable to encode information: shape (see Fig. 28). Such notations are **visually one-dimensional**: they use only one of the eight available visual communication channels, and ironically, the one with the lowest bandwidth. Shape is one of the least powerful visual variables as it can only be used to encode nominal data and is one of the least cognitively efficient [74].

In contrast to the visually impoverished forms of SE notations, Fig. 29 shows an example from cartography, a discipline with over 5,000 years experience in graphically encoding information. This defines 38 different graphical conventions using six visual variables (shape, texture, brightness, size, color, and orientation). This is an order of magnitude more visually expressive than most SE notations and represents the point of visual saturation in cartography

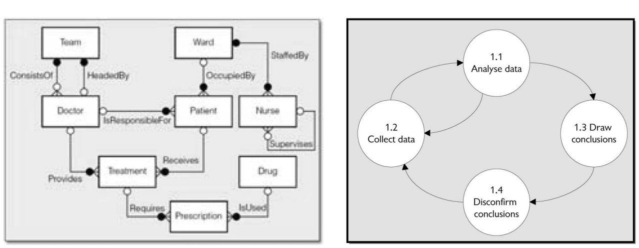


Fig. 28. Visual monosyllabism: ER diagrams and DFDs use only a single visual variable to encode information (shape).

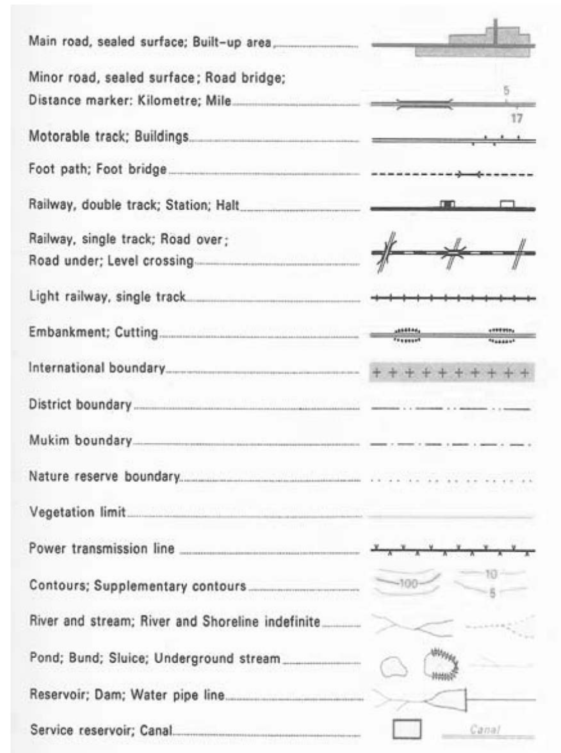


Fig. 29. Visual saturation: This cartographic legend uses six visual variables to define 38 distinct graphical conventions [134]. A color version of this figure may be viewed at http://doi.ieeecomputersociety.org/10.1109/TSE.2009.67.

(as the planar variables are reserved for encoding geographical location).

A second point to note about this example is that five visual variables (shape, texture, brightness, size, and color) are used to distinguish between 19 types of lines *with no conjunctions*. Compare this to Table 1, which uses three visual variables to distinguish between a similar number of lines with mostly conjunctions. This shows how visual expressiveness can be used to improve discriminability (Perceptual Discriminability). Cartographers are masters of graphical representation and notation designers can learn much from them.

4.6.1 Use of Color

Color is one of the most cognitively effective of all visual variables: the human visual system is highly sensitive to variations in color and can quickly and accurately distinguish between them [77], [149]. Differences in color are detected three times faster than shape and are also more easily remembered [72], [129]. Yet, surprisingly, color is rarely used in SE notations and is specifically prohibited in UML:

> "UML avoids the use of graphic markers, such as color, that present challenges for certain persons (the color blind) and for important kinds of equipment (such as printers, copiers, and fax machines)" [98].

ArchiMate is an example of graphic excellence in this respect, as it uses color to distinguish between concepts in different architectural layers (Fig. 30). This enables information in different architectural layers to be separated in the mind [19].

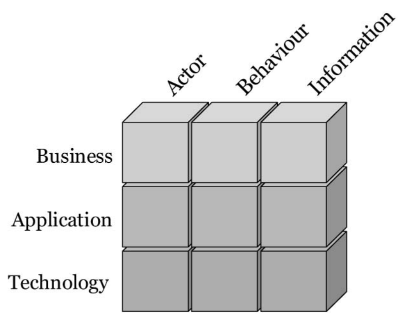


Fig. 30. Use of color in ArchiMate: Color is used to distinguish between constructs in different architectural layers. A color version of this figure may be viewed at http://doi.ieeecomputersociety.org/10.1109/TSE.2009.67.

However, color should never be used as the sole basis for distinguishing between symbols as it is sensitive to variations in visual perception (e.g., color blindness) and screen/printer characteristics (e.g., black-and-white printers). To avoid loss of information (**robust design**), color should only be used for redundant coding. Event-driven Process Chains (EPCs) [116] and ArchiMate are two of the few SE notations to use color to encode information, but both make the mistake of using it in a nonredundant way. When diagrams are reproduced in black and white, differences between some symbols disappear.

4.6.2 Choice of Visual Variables: Form Follows Content

The choice of visual variables should not be arbitrary but should be based on the nature of the information to be conveyed [8]. Different visual variables have properties that make them suitable for encoding different types of information. For example, color can only be used for nominal data as it is not psychologically ordered [65]. Also, different visual variables have different **capacities** (number of perceptible steps) [8], [127]. The properties of each visual variable have been established by research in psychophysics (summarized in Table 2).

The aim should be to match properties of visual variables to the properties of the information to be represented (i.e., form follows content):

TABLE 2
Different Visual Variables Have Different Capabilities for Encoding Information: Power = Highest Level of Measurement That Can Be Encoded; Capacity = Number of Perceptible Steps

Variable	Power	Capacity
Horizontal position (x)	Interval	10-15
Vertical position (y)	Interval	10-15
Size	Interval	20
Brightness	Ordinal	6-7
Colour	Nominal	7-10
Texture	Nominal	2-5
Shape	Nominal	Unlimited
Orientation	Nominal	4

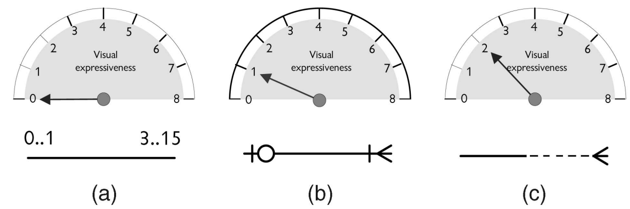


Fig. 31. Levels of visual expressiveness: (a) UML = 0, (b) IE = 1 (shape), and (c) Oracle = 2 (shape, brightness).

- Power: The power of the visual variable should be greater than or equal to the measurement level of the information.
- Capacity: The capacity of the visual variable should be greater than or equal to the number of values required.

4.6.3 SE Notations Use Only a Limited Range of Values of Visual Variables

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 [104]. These are the least effective shapes for human visual processing and empirical studies show that curved, 3D, and iconic shapes should be preferred [5], [56], [148]. Other visual variables are used in binary fashion: for example, UML uses brightness and texture in addition to shape but only two values of each: black and white for value, solid and dashed for texture. The combination of using only a small subset of visual variables and a small subset of the capacity of each variable means that SE notations utilize only a tiny fraction of the graphic design space.

4.6.4 Textual versus Graphical Encoding

To maximize visual expressiveness, graphical encoding should be preferred to textual encoding (subject to constraints on graphic complexity: see Graphic Economy). The more work that can be done by visual variables, the greater the role of perceptual processing and computational offloading. To illustrate possible trade-offs between visual and textual encoding in notation design, Fig. 31 shows three alternative encodings of relationship cardinalities (or multiplicities). These represent three different levels of visual expressiveness:

- 0: the UML convention, which is nonvisual (i.e., textual),
- 1: the IE convention, which uses shape, and
- 2: the Oracle convention, which uses shape (to encode the maximum) and texture (to encode the minimum).

The UML convention is the most **semantically expressive** as it can define exact values for minimum and maximum cardinalities (rather than 0, 1, or many like the other two). However, it is the least cognitively effective as it relies entirely on text processing [57]. The IE and Oracle representations are also more semantically transparent as the multiple claws of the "crow's foot" suggest a many relationship: The effectiveness of this has been confirmed by experimental studies, e.g., [57], [108].

The UML convention is, therefore, less discriminable, less semantically transparent, and less visually expressive than the other two, which represents a poor compromise between semantic expressiveness and cognitive effectiveness (though a fairly typical one, as SE notation designers tend to prioritize semantic expressiveness over everything else). Surprisingly, UML was developed more recently than the other two techniques, which represents a later "evolutionary form." However, without sound theory and principles for choosing between alternative representations, notations can get worse rather than better over time.

4.7 Principle of Dual Coding: Use Text to Complement Graphics

Perceptual Discriminability and Visual Expressiveness both advise against using text to encode information in visual notations. However, this does not mean that text has no place in visual notation design. Pictures and words are not enemies and should not be mutually exclusive [39], [132]. According to **dual coding theory** [100], 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. In particularly, text should never be used as the sole basis for distinguishing between symbols (as discussed in Perceptual Discriminability), but can be usefully used as a form of redundant coding, to reinforce and clarify meaning. Another argument for dual coding is that people differ widely in their spatial and verbal processing abilities. Including graphics and text is likely to improve understanding by people across the full spectrum of spatial and verbal abilities [144].

4.7.1 Annotations

Including textual explanations (annotations) can improve understanding of diagrams in the same way that comments can improve understanding of programs. According to the principle of **spatial contiguity** [81], including these on the diagram itself is much more effective than including them separately (e.g., in a separate document, as is commonly done in practice). An example of design excellence here is UML, which explicitly includes an annotation construct, though representing it using a graphical symbol was not a good representational choice (as discussed in Semiotic Clarity).

4.7.2 Hybrid (Graphics+Text) Symbols

Textual encoding can be used to reinforce and expand the meaning of graphical symbols. The rightmost representation in Fig. 32 shows a hybrid (graphics + text) representation of the same relationship as shown in Fig. 31. This combines the semantic expressiveness of the UML convention with the visual expressiveness of the Oracle convention, while also taking advantage of dual coding.

In the hybrid representation, the text both expands and reinforces the meaning of the graphics.

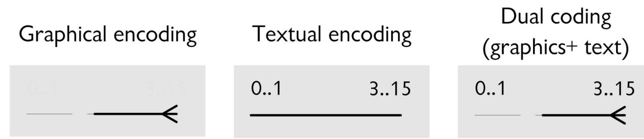


Fig. 32. Dual coding: the best of both worlds?

- It expands its meaning by enabling exact minimum and maximum cardinalities to be specified.
- It reinforces its meaning by providing an additional cue to what it means. Empirical studies show that novices sometimes have difficulty remembering what cardinality symbols mean on ER diagrams [35], [52]: Redundantly, including textual cardinalities, increases the likelihood they will be accurately interpreted. The text thus adds value even if the cardinalities are all 0, 1, or many.

Dual coding does not affect discriminability as visual distance is not affected by the addition of text. However, it aids interpretation by providing textual cues to the meaning of symbols when they are not semantically transparent and improves retention through interlinked visual and verbal encoding in memory.

4.8 Principle of Graphic Economy: The Number of Different Graphical Symbols Should Be Cognitively Manageable

Graphic complexity is defined by the number of graphical symbols in a notation: the size of its visual vocabulary [94]. This differs from diagrammatic complexity (Complexity Management), as it relates to complexity at the type (language) level rather than the token (sentence) level. A notation designer can create an unlimited number of symbols by combining visual variables together in different ways. However, compared to textual languages, this strategy is only effective up to a certain point, as there are cognitive limits on the number of visual categories that can be effectively recognized [139]. Beyond this, each new symbol introduced reduces cognitive effectiveness.

Graphic complexity affects novices much more than experts, as they need to consciously maintain meanings of symbols in working memory. If the symbols are not mnemonic, the user must remember what the symbols mean or else a legend must be supplied and frequently referenced, which all adds to the effort of processing diagrams. Empirical studies show that graphic complexity significantly reduces understanding of SE diagrams by novices [94].

The human ability to discriminate between perceptually distinct alternatives (**span of absolute judgment**) is around six categories [83]: 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. Each new construct normally requires a new symbol and while new constructs are often

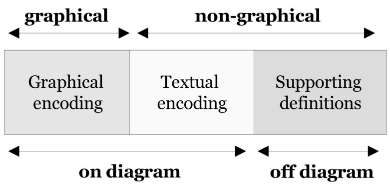


Fig. 33. A balancing act: To keep graphic (and diagram) complexity manageable, notation designers need to make decisions about what information to encode graphically, what to encode textually, and what to include in supporting definitions.

added, old ones are rarely removed. For example, the first SE notation (program flowcharts) started off with five symbols [36] but expanded by a factor of 8 by the time the final ISO standard was published [58]. There are three main strategies for dealing with excessive graphic complexity:

- Reduce semantic complexity.
- Introduce symbol deficit.
- Increase visual expressiveness.

4.8.1 Reduce (Or Partition) Semantic Complexity

The number of semantic constructs in a notation is a major determinant of graphic complexity as different constructs are usually represented by different symbols.[8] Simplifying the semantics of a notation thus provides an obvious way of reducing graphic complexity. In more complex languages (e.g., UML and ArchiMate), the semantic constructs may need to be partitioned to reduce graphic and diagrammatic complexity.

4.8.2 Introduce Symbol Deficit

Graphic complexity can also be reduced directly (without affecting semantics) by introducing symbol deficit (Semiotic Clarity). This means choosing *not* to show some constructs graphically.[9] This changes the balance between graphical and textual encoding in the notation, as the constructs removed from the visual notation will usually need to be defined in text. Most visual notations rely at least to some extent on text to keep their visual vocabularies manageable: It is the general purpose tool of last resort [97]. The interpretation of any SE diagram almost always depends on a division of labor between graphics and text [44], [97]. Where this line is drawn (the **graphics-text boundary**) is a critical design decision. Another important design decision is how much information to include on the diagram itself and how much in supporting definitions (**on-diagram** versus **off-diagram**).

Diagrams work best as abstractions or summaries rather than stand-alone specifications and showing too much information graphically can be counterproductive [44], [97]. Also, some information is more effectively encoded in textual form: Diagrams are better for representing some types of information but worse for others (e.g., business rules and detailed procedural logic) [44]. Part of the secret to using visual notation effectively may be knowing when *not* to use it [22], [104]: to find the right balance between graphical, textual, and off-diagram encoding that maximizes computational offloading while avoiding excessive graphic and diagrammatic complexity (Fig. 33).

8. This is not a direct relationship as symbol excess, symbol deficit, symbol overload, and symbol redundancy can increase or decrease graphic complexity independently of the number of constructs (Semiotic Clarity).

9. This conflicts with Visual Expressiveness, which advocates encoding as much information as possible graphically. However, this should only be done subject to limits on graphic complexity.

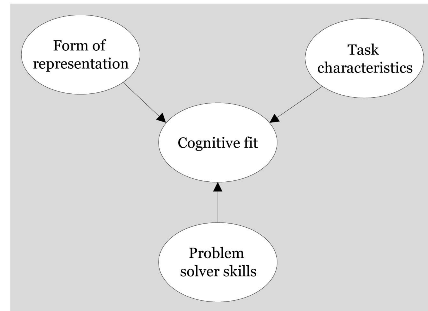


Fig. 34. Cognitive fit is the result of a three-way interaction between the representation, task, and problem solver [137].

Many SE notations make the mistake of trying to encode too much information in graphical form. For example, ORM uses 21 different graphical symbols to represent different types of constraints [49], even though graphics is poorly suited for encoding such information (as shown by the largely unsuccessful attempts to develop visual notations for modeling business rules, e.g., [113]).

4.8.3 Increase Visual Expressiveness

This is an approach to dealing with excessive graphic complexity that works not by reducing the number of symbols but by increasing human discrimination ability. The span of absolute judgment can be expanded by increasing the number of perceptual dimensions on which stimuli differ [83]. The six-symbol limit thus only applies if a single visual variable is used (which is true for most SE notations, which use shape as the sole information-carrying variable). Using multiple visual variables to differentiate between symbols (Visual Expressiveness) can increase human discrimination ability in an almost additive manner. Fig. 29 shows how multiple visual variables can be used to effectively distinguish between a large number of categories.

4.9 Principle of Cognitive Fit: Use Different Visual Dialects for Different Tasks and Audiences

Cognitive fit theory is a widely accepted theory in the information systems (ISs) field that has been validated in a wide range of domains, from managerial decision making to program maintenance [117], [137], [138]. The theory states that different representations 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 (Fig. 34).

Most SE notations exhibit **visual monolinguism**: They use a single visual representation for all purposes. However, cognitive fit theory suggests that this "one size fits all" assumption may be inappropriate and different visual dialects may be required for different tasks and/or audiences ("representational horses for cognitive courses" [104]). These represent *complementary* rather than *competing* visual dialects as discussed earlier. There are at least two reasons for creating multiple visual dialects in an SE context:

- Expert-novice differences (problem solver skills).
- Representational medium (task characteristics).

4.9.1 Expert-Novice Differences

One of the major challenges in designing SE notations is the need to develop representations that are understandable by both business and technical experts. This adds to the difficulty of the task as in most engineering contexts, diagrams are only used to communicate among experts. There are well-known differences in the way experts and novices process diagrams [21], [63], [74], [96], [136], [149]. While we all have the same perceptual and cognitive hardware, notation experts develop **diagram schemas** in long-term memory which largely automates the process of diagram interpretation [19], [105]. For nonexperts, interpretation is slower, more error-prone, and requires conscious effort. The most important expert-novice differences are:

- Novices have more difficulty discriminating between symbols [15], [63].
- Novices are more affected by complexity as they lack "chunking" strategies [12].
- Novices have to consciously remember what symbols mean [149].

These differences are rarely taken into account in the design of SE visual notations, even though they are routinely used to communicate with business stakeholders (who are domain experts but notational novices). While it might seem reasonable to design a notation to be understandable to novices and use this also for experts (following the "lowest common denominator" principle), **cognitive load theory** (not related to cognitive fit theory) suggests that this may be incorrect. Optimizing representations for novices can reduce their effectiveness for experts and vice versa: this is called the **expertise reversal effect** [60].

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. Notations designed for communication with novices will need to use more discriminable symbols (Perceptual Discriminability), reduced complexity (Complexity Management), more mnemonic conventions (Semantic Transparency), explanatory text (Dual Coding), and simplified visual vocabularies (Graphic Economy).

Practitioners sometimes develop their own, informal notations for communicating with users (usually simplified versions of the standard notation), but this capability should be provided by the primary notation rather than leaving it to secondary notation. Two examples of design excellence in this regard are ORM [49] and Oracle data modeling [6], which define specialized notations for communicating with end users. ORM uses a tabular representation, while Oracle defines a natural-language-based normative language [53] (which supports Dual Coding).

4.9.2 Representational Medium

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

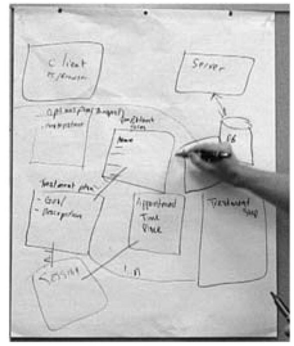


Fig. 35. Notational requirements for hand sketching are different from those for drawing tools and tend to limit visual expressiveness.

(see Fig. 35). Hand drawing presents special challenges for visual notation design because of the limited drawing abilities of most software engineers (as drawing is typically not a skill included in SE curricula). Some of the important notational requirements are:

- Perceptual Discriminability: Discriminability requirements are higher due to variations in how symbols are drawn by different people. As within-symbol variations increase, between-symbol differences need to be more pronounced.
- Semantic Transparency: Pictures and icons are more difficult to draw than simple geometric shapes, especially for the artistically challenged.
- Visual Expressiveness: Some visual variables (color, value, and texture) are more difficult to use (due to drawing ability and availability of equipment, e.g., color pens).

The need to use notations for sketching provides a possible explanation for why SE visual notations have such limited visual vocabularies. Requirements for hand drawing may have acted to constrain their visual expressiveness, and would explain why highly effective techniques such as colour, icons and 3D shapes are so rarely used. 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. Cognitive fit allows the best of both worlds: a simplified visual dialect for sketching and an enriched notation for final diagrams.

4.10 Interactions among Principles

Fig. 36 summarizes the interactions among the principles (note that effects are not necessarily symmetrical). Knowledge of interactions can be used to make **trade-offs** (where principles conflict with one another) and exploit **synergies** (where principles support each other).

The most important interactions are:

- Semiotic Clarity can affect Graphic Economy either positively or negatively: Symbol excess and symbol redundancy increase graphic complexity, while symbol overload and symbol deficit reduce it.
- Perceptual Discriminability increases Visual Expressiveness as it involves using more visual variables and a wider range of values (a side effect of

	Semiotic Clarity	Perceptual Discriminability	Semantic Transparency	Complexity Management	Cognitive Integration	Visual Expressiveness	Dual Coding	Graphic Economy	Cognitive Fit
Semiotic Clarity								±	
Perceptual Discriminability						+			+
Semantic Transparency		+							±
Complexity Management								−	+
Cognitive Integration	−			+				−	
Visual Expressiveness		+						+	±
Dual Coding									+
Graphic Economy		+		+		−			+
Cognitive Fit									

Fig. 36. Interactions between principles: + indicates a positive effect, − indicates a negative effect, and ± indicates a positive or negative effect depending on the situation. A color version of this figure may be viewed at http://doi.ieeecomputersociety.org/10.1109/TSE.2009.67.

increasing visual distance); similarly, Visual Expressiveness is one of the primary ways of improving Perceptual Discriminability.

- Increasing Visual Expressiveness reduces the effects of graphic complexity, while Graphic Economy defines limits on Visual Expressiveness (how much information can be effectively encoded graphically).
- Increasing the number of symbols (Graphic Economy) makes it more difficult to discriminate between them (Perceptual Discriminability).
- Perceptual Discriminability, Complexity Management, Semantic Transparency, Graphic Economy, and Dual Coding improve effectiveness for novices, though Semantic Transparency can reduce effectiveness for experts (Cognitive Fit).
- Semantic Transparency and Visual Expressiveness can make hand drawing more difficult (Cognitive Fit)

5 Conclusion

Historically, issues of visual syntax have been ignored or undervalued in SE research. One aim of this paper is to raise awareness about the importance of such issues in notation design. Visual representation decisions have a profound effect on the usability and effectiveness of SE notations, equal to (if not greater than) than decisions about semantics. For this reason, visual syntax deserves at least equal effort and attention in the notation design process.

Visual notation design currently exists as a "dark art," an unselfconscious process that resists explanation even by those who practice it [53]. The goal of this paper is to establish the foundations for a science of visual notation design: to help it progress from a craft to a design discipline (self-conscious process) based on explicit principles. Having sound principles for designing visual syntax (distinct from those for designing semantics) will enable notation designers to design both syntax and semantics of notations in a systematic manner. It will also help them to clearly separate syntactic and semantic issues, which are frequently confounded: This supports **separation of concerns**, one of the basic tenets of SE.

SE visual notations are currently designed without explicit design rationale. In the same way that reasons for design decisions should be provided when designing software systems, they should also be provided when designing visual notations. We need to be able to defend our graphic designs and provide sound justification for visual representation choices [133]. Ideally, such justifications should be based on scientific evidence rather than subjective criteria, as is currently the case.

A surprising result of our analysis of existing SE notations is that some older (even obsolete) visual notations such as DFDs are better designed than more recent ones, contrary to expectations of "notational Darwinism." Without sound principles for visual notation design, practice can just as easily go backward as forward (like any unselfconscious culture). Naive theories of graphic design (like naive theories of physics [82] or psychology [95]) are as likely to be wrong as they are to be right.

5.1 The Physics of Notations: A Theory for Visual Notation Design

The Physics of Notations consists of three key components: a design goal, a descriptive theory, and a prescriptive theory.

5.1.1 The Dependent Variable (Design Goal)

Cognitive effectiveness is defined as the primary-dependent variable for evaluating and comparing visual notations and the primary design goal in constructing them. This variable is operationally defined and can, therefore, be empirically evaluated.

5.1.2 Descriptive (Type IV) Theory: How Visual Notations Communicate

Section 3 defines a theory of *how* and *why* visual notations communicate, based on extant theories from communication, semiotics, graphic design, visual perception, and cognition. This provides a basis for explaining and predicting why some visual representations will be more effective than others.

5.1.3 Prescriptive (Type V) Theory: Principles for Designing Cognitively Effective Visual Notations

Section 4 defines a set of principles for designing cognitively effective visual notations. These provide a scientific basis for comparing, evaluating, improving, and constructing visual notations, which has previously been lacking in the SE field. Importantly, these principles are not based on common sense, experience, or observations of "best practices" but on theory and empirical evidence about cognitive effectiveness of visual representations. They synthesize the best available research evidence from a wide range of fields, including communication, semiotics, graphic design, visual perception, psychophysics, cognitive psychology, HCI, information visualization, information systems, education, cartography, and diagrammatic reasoning.

Together, the principles form a design (Type V) theory. Gregor and Jones [46] have defined a template for specifying such theories: Table 3 shows how the Physics of Notations fits into this. All components are specified except mutability of artifacts (which relates to evolution of notations), which suggests a possible direction for future research. In comparison, the CDs framework only "ticks" two of the boxes (Constructs and Expository Instantiation), confirming that it is a Type I theory.

Gregor and Jones's template is an important contribution for two reasons:

TABLE 3
Design Theory Components [46]

Component	The Physics of Notations
Scope and Purpose	To design cognitively effective visual notations
Constructs	Visual notation (language), visual sentence (diagram), visual dialect, visual vocabulary (symbol set), visual grammar (composition rules), graphical symbol (type), symbol instance (token), visual variable, cognitive effectiveness
Principles of form and function	Semiotic Clarity, Perceptual Discriminability, Semantic Transparency, Complexity Management, Cognitive Integration, Visual Expressiveness, Dual Coding, Graphic Economy, Cognitive Fit (all principles of form)
Artifact mutability	[Not addressed]
Testable propositions	Visual notations that satisfy the principles will be more cognitively effective than those that do not. The theory effectively defines a causal model with the principles as independent variables (causes) and cognitive effectiveness as the dependent (outcome) variable.
Justificatory knowledge	Information theory, graphic design theory, theory of symbols, feature integration theory, object recognition theory, psychophysics theory, semiotic theory, cognitive load theory, cognitive integration theory, working memory theory, gestalt theory, cognitive fit theory, dual coding theory, schema theory, human information processing theory, multimedia learning theory, wayfinding theory, cartographic abstraction theory, modularity theory, ontological theory
Principles of implementation (optional)	Methods for evaluating notations (e.g. perceptual popout analysis for evaluating Perceptual Discriminability) Methods for improving notations (e.g. redundant coding for improving PD).
Expository instantiation (optional)	Expository examples: UML, ER, DFDs, ***i****, ArchiMate, EPCs, ORM, rich pictures, program flowcharts, cartography.

- It recognizes design theories as a separate and legitimate class of scientific theories. This is important for researchers in **artificial sciences** (like SE), where they represent the majority of theories proposed [34].
- It helps clarify and formalize the structure of design theories. Design (artificial science) theories are fundamentally different to traditional (natural science) theories, but are typically presented in a wide variety of formats.

5.2 Practical Significance

A significant proportion of SE research and practice is devoted to **method engineering**. In most cases, visual notations form an integral part of the methods proposed. The principles in this paper can be used by notation designers to:

- Design notations: Visual notations can be constructed in a systematic way, based on the semantics to be expressed. Existing SE notations use a limited variety of visual forms, a well-known characteristic of unselfconscious design cultures [1]. Having explicit design principles supports more extensive exploration of the design space and development of new and innovative visual forms.
- Compare notations: The principles support comparison of notations, where syntactic and semantic issues are often confused. They also provide the basis for resolving some long-standing disputes about the relative merits of competing visual dialects (e.g., De Marco versus Gane & Sarson DFDs and IE versus Chen ER diagrams).[10]
- Evaluate and improve notations: The principles can be used to identify potential problems in existing visual notations and to resolve them. This paper has identified serious design flaws in some of the leading SE notations, together with some suggestions for how they could be improved: However, these represent only the "tip of the iceberg" of improvements that are possible (e.g., see [86], [88], [90]).

5.2.1 Impact on SE Practice

A major focus in this paper has been on improving understanding of visual notations by business stakeholders (end users and customers). This is particularly important in an SE context, as effective user-developer communication is critical for successful development of software systems. It is also one of the major weaknesses in existing SE notations: Expert-novice differences are rarely taken into account in their design, even though they are routinely used to communicate with novices.

However, visual notations are also used for communication among technical experts (e.g., members of the development team) and as **cognitive externalizations** [151] to support design and problem solving. Improving their cognitive effectiveness will also improve their utility for these purpose: Optimizing visual notations for human information processing will optimize them for use by both software engineers and their customers, who share the same perceptual and cognitive hardware and software (subject to considerations of Cognitive Fit).

5.3 Theoretical Significance

This paper complements previous research in SE on classification and implementation of visual languages [23], [24]. It also complements research on semantic analysis of notations. Ontological analysis is a widely accepted approach for analyzing semantics of notations, which supports rigorous, construct-by-construct analysis (e.g., [99]). The principles defined in this paper support similarly rigorous, symbol-by-symbol analysis of visual syntax (e.g., [90]). Used together, these approaches allow both syntax and semantics of notations to be evaluated in a theoretically sound manner.

The CDs framework is currently the predominant approach for analyzing visual languages in the IT field. The Physics of Notations represents a significant advance on this framework for evaluating and designing visual notations:

10. Space does not permit a full analysis here, but De Marco DFDs should be preferred on grounds of Perceptual Discriminability and Visual Expressiveness, while IE ER diagrams should be preferred based on Perceptual Discriminability, Semantic Transparency, and Visual Expressiveness.

- It was specifically developed for visual notations rather than being adapted for this purpose. This reduces its generality, but supports detailed predictions and prescriptions.
- It supports detailed, symbol-by-symbol analysis of notations as opposed to only "broad brush" analysis.
- The principles are explicitly justified using theory and empirical evidence.
- The principles are clearly defined and operationalized using evaluation procedures and/or metrics.
- The principles define desirable properties of visual notations, which can be used for evaluation and comparison.
- The principles provide prescriptive guidelines for designing and improving visual notations.
- The theory can be used to generate predictions that can be empirically tested, which is falsifiable.

The Physics of Notations incorporates both a Type IV theory (Section 3) and a Type V theory (Section 4), which are higher evolutionary forms than the CDs framework (Type I). However, it should not be seen as a direct competitor for the CDs framework as its scope is much more modest (visual notations rather than cognitive artifacts). Instead, it should be seen as *complementary*: It provides exactly the type of detailed, domain-specific analysis that the authors of the CDs framework argued was necessary to supplement the "broad brush" analysis provided by CDs [44]. It also focuses exclusively on visual representation aspects, which the CDs framework excludes.

5.4 Limitations and Further Research

5.4.1 Encoding Side Design Goals

This paper defines cognitive effectiveness as the primary goal in visual notation design. This is a **decoding-side** (processing) goal: Effectiveness is defined from the receiver's rather than the sender's viewpoint (cf. Fig. 6). However, ease of expression, an **encoding-side** goal, is also an important consideration in design of visual notations: For example, one of the barriers to the adoption of UML is that many programmers find it easier to write code directly than produce UML diagrams first. Encoding-side and decoding-side goals can often be in conflict, like the **R-principle** (minimizing speaker's effort) and the **Q-principle** (minimizing listener's effort) in linguistics [55]. The Physics of Notations does not consider encoding-side issues as it focuses only on the effectiveness of the resulting representation. Further research could expand the theory to include such considerations.

5.4.2 Validation of Principles: Truth versus Utility

The principles proposed in this paper represent an initial starting point for establishing a science of visual notation design. However, further research is required to test, refine, and extend the principles.

Empirical validation (truth). The principles can be used to generate predictions that can be empirically tested. Positive relationships are hypothesized between all principles and cognitive effectiveness: representations that satisfy the principles are predicted to be more effective than those that do not. This defines a **causal model** that can be tested by comparing the cognitive effectiveness of notations that satisfy each principle with those that violate it (i.e., in a similar way to how the predictions of ontological theory have been tested, e.g., [13]).

However, because of the way in which the principles were developed (by synthesizing the best available research evidence), they are prevalidated to at least some extent. There is empirical evidence from the SE field for most of the principles: <u>Semantic Transparency</u>, <u>Complexity Management</u>, <u>Cognitive Integration</u>, <u>Graphic Economy</u>, and <u>Cognitive Fit</u>; and from other fields for all principles except <u>Semiotic Clarity</u>. This is the only principle that is based only on theory, as semiotics is not a highly empirical discipline. Where principles have not been validated in a SE context, empirical testing may be needed to confirm that the results generalize to the SE domain (though some principles have been validated in so many different domains that further replication would be of marginal benefit). Another role of empirical testing would be to measure the practical impact of each principle (**effect size**), which would provide valuable information to notation designers in making design decisions.

Pragmatic validation (utility). Another aspect of validation is whether the principles provide a useful basis for evaluating and designing visual notations. This relates to their practical value (**utility**) rather than their scientific (**truth**) value, which is an important consideration in validating **methodological knowledge** as opposed to **propositional knowledge**) [111]. The principles have so far been successfully used to evaluate and improve three leading SE notations; ArchiMate [86]: an international standard language for enterprise architecture modeling, UML [90]: the industry standard language for modeling software systems; and ***i**** [88]: one of the leading requirements engineering notations. It has also been used to evaluate a visual notation from another field: ORES, a proprietary cognitive mapping technique [89]). The principles have so far not been used to design a visual notation from first principles, but research is in progress to do this: one in SE and one in another domain.

5.5 Wider Significance

The principles defined in this paper also provide a potential contribution outside the SE field. Visual notations are used in a wide range of scientific fields, e.g., physics (Feynman diagrams), biology (cladograms), chemistry (molecular structure diagrams), mathematics (Venn diagrams), education (knowledge maps), engineering (circuit diagrams), linguistics (parse trees), and scientific research (causal graphs). While the principles were specifically developed for designing SE notations, they are based on general principles of visual perception and cognition which, in principle, are applicable to visual notations in any domain.

ACKNOWLEDGMENTS

This work originally grew out of a project commissioned by Henry Franken, CEO of BiZZdesign B.V., and supported by the Dutch Research Innovation Agency (SenterNovem). The author would also like to thank Jos van Hillegersberg (Head, Department of Information & Change Management, University of Twente) for creating such a stimulating environment in which to conduct research.

REFERENCES

[1] C.W. Alexander, *Notes on the Synthesis of Form.* Harvard Univ. Press, 1970.

[2] D.E. Avison and G. Fitzgerald, *Information Systems Development: Methodologies, Techniques and Tools,* third ed. Blackwell Scientific, 2003.

[3] C.W. Bachman, "Data Structure Diagrams," *DataBase: A Quarterly Newsletter of SIGBDP,* vol. 1, no. 2, 1969.

[4] C.Y. Baldwin and K.B. Clark, *"Design Rules Volume 1: The Power of Modularity.* MIT Press, 2000.

[5] M. Bar and M. Neta, "Humans Prefer Curved Visual Objects," *Psychological Science,* vol. 17, no. 8, pp. 645-648, 2006.

[6] R. Barker, *CASE*Method: Entity Relationship Modelling.* Addison-Wesley Professional, 1990.

[7] D. Batra, J.A. Hoffer, and R.P. Bostrom, "Comparing Representations with Relational and EER Models," *Comm. ACM,* vol. 33, no. 2, pp. 126-139, 1990.

[8] J. Bertin, *Semiology of Graphics: Diagrams, Networks, Maps.* Univ. of Wisconsin Press, 1983.

[9] I. Biederman, "Recognition-by-Components: A Theory of Human Image Understanding," *Psychological Rev.,* vol. 94, no. 2, pp. 115-147, 1987.

[10] A.F. Blackwell, *Cognitive Dimensions of Notations Resource Site,* http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/index.html, 2009.

[11] A.F. Blackwell and T.R.G. Green, "Notational Systems—The Cognitive Dimensions of Notations Framework," *HCI Models, Theories, and Frameworks: Towards a Multi-Disciplinary Science,* J.M. Carroll, ed., pp. 328-365, Morgan Kaufmann, 2003.

[12] J. Blankenship and D.F. Dansereau, "The Effect of Animated Node-Link Displays on Information Recall," *J. Experimental Education,* vol. 68, no. 4, pp. 293-308, 2000.

[13] F. Bodart, A. Patel, M. Sim, and R. Weber, "Should the Optional Property Construct Be Used in Conceptual Modelling: A Theory and Three Empirical Tests," *Information Systems Research,* vol. 12, no. 4, pp. 384-405, 2001.

[14] G. Booch, *Object-Oriented Analysis and Design with Applications,* second ed. Benjamin Cummings Publishing, 1994.

[15] C. Britton and S. Jones, "The Untrained Eye: How Languages for Software Specification Support Understanding by Untrained Users," *Human Computer Interaction,* vol. 14, pp. 191-244, 1999.

[16] S.K. Card and J. Mackinlay, "The Structure of the Information Visualization Design Space," *Proc. 1997 IEEE Symp. Information Visualization,* 1997.

[17] P.B. Checkland and J. Scholes, *Soft Systems Methodology in Action.* Wiley, 1990.

[18] P.P. Chen, "The Entity Relationship Model: Towards An Integrated View of Data," *ACM Trans. Database Systems,* vol. 1, no. 1, pp. 9-36, 1976.

[19] P.C.H. Cheng, "Why Diagrams Are (Sometimes) Six Times Easier than Words: Benefits beyond Locational Indexing," *Proc. Int'l Conf. Theory and Application of Diagrams,* 2004.

[20] P.C.-H. Cheng and R. Barone, "Representing Complex Problems: A Representational Epistemic Approach," *Learning to Solve Complex Scientific Problems,* D.H. Jonassen, ed., pp. 97-130, Lawrence Erlbaum Assoc., 2007.

[21] P.C.-H. Cheng, R.K. Lowe, and M. Scaife, "Cognitive Science Approaches to Understanding Diagrammatic Representations," *Artificial Intelligence Rev.,* vol. 15, nos. 1/2, pp. 79-94, 2001.

[22] W. Citrin, "Strategic Directions in Visual Languages Research," *ACM Computing Surveys,* vol. 24, no. 4, 1996.

[23] G. Costagliola, A. Delucia, S. Orefice, and G. Polese, "A Classification Framework to Support the Design of Visual Languages," *J. Visual Languages and Computing,* vol. 13, pp. 573-600, 2002.

[24] G. Costagliola, V. Deufemia, and G. Polese, "A Framework for Modeling and Implementing Visual Notations with Applications to Software Engineering," *ACM Trans. Software Eng. and Methodology,* vol. 13, no. 4, pp. 431-487, 2004.

[25] J. Dagit, J. Lawrance, C. Neumann, M. Burnett, R. Metoyer, and S. Adams, "Using Cognitive Dimensions: Advice from the Trenches," *J. Visual Languages and Computing,* vol. 17, pp. 302-327, 2006.

[26] I. Davies, P. Green, M. Rosemann, M. Indulska, and S. Gallo, "How Do Practitioners Use Conceptual Modelling in Practice?" *Data and Knowledge Eng.,* vol. 58, pp. 358-380, 2006.

[27] T. De Marco, *Structured Analysis and System Specification.* Yourdon Press, 1978.

[28] E.W. Dijkstra, "The Humble Programmer (1972 Turing Award Lecture)," *Comm. ACM,* vol. 15, no. 10, pp. 859-866, 1972.

[29] C. Ding and P. Meteti, "A Framework for the Automated Drawing of Data Structure Diagrams," *IEEE Trans. Software Eng.,* vol. 16, no. 5, pp. 543-557, May 1990.

[30] R.L. Flood and E.R. Carson, *Dealing with Complexity: An Introduction to the Theory and Application of Systems Science.* Plenum Press, 1993.

[31] C. Gane and T. Sarson, *Structured Systems Analysis.* Prentice-Hall, 1979.

[32] A.G. Ganek and T.A. Corbi, "The Dawning of the Autonomic Computing Era," *IBM Systems J.,* vol. 42, no. 1, pp. 5-19, 2003.

[33] A. Gehlert and W. Esswein, "Towards a Formal Research Framework for Ontological Analyses," *Advanced Eng. Informatics,* vol. 21, pp. 119-131, 2007.

[34] R.L. Glass, V. Ramesh, and I. Vessey, "An Analysis of Research in Computing Disciplines," *Comm. ACM,* vol. 47, no. 6, pp. 89-94, 2004.

[35] R.C. Goldstein and V.C. Storey, "Some Findings on the Intuitiveness of Entity Relationship Constructs," *Entity Relationship Approach to Database Design and Querying,* Elsevier Science, 1990.

[36] H.H. Goldstine and J. von Neumann, *Planning and Coding of Problems for an Electronic Computing Instrument.* Inst. for Advanced Study, 1947.

[37] N. Goodman, *Languages of Art: An Approach to a Theory of Symbols.* Bobbs-Merrill Co., 1968.

[38] P. Goolkasian, "Pictures, Words and Sounds: From Which Format Are We Best Able to Reason?" *J. General Psychology,* vol. 127, no. 4, pp. 439-459, 2000.

[39] R.S. Goonetilleke, H.M. Shih, H.K. On, and J. Fritsch, "Effects of Training and Representational Characteristics in Icon Design," *Int'l J. Human-Computer Studies,* vol. 55, pp. 741-760, 2001.

[40] D.M. Green and J.A. Swets, *Signal Detection Theory and Psychophysics.* Wiley, 1966.

[41] P. Green and M. Rosemann, "Integrated Process Modelling: An Ontological Evaluation," *Information Systems,* vol. 25, no. 2, pp. 73-87, 2000.

[42] T.R.G. Green, "Cognitive Dimensions of Notations," *People and Computers V,* A. Sutcliffe and L. Macaulay, eds., pp. 443-460, Cambridge Univ. Press, 1989.

[43] T.R.G. Green, A.E. Blandford, L. Church, C.R. Roast, and S. Clarke, "Cognitive Dimensions: Achievements, New Directions and Open Questions," *J. Visual Languages and Computing,* vol. 17, pp. 328-365, 2006.

[44] T.R.G. Green and M. Petre, "Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework," *J. Visual Languages and Computing,* vol. 7, pp. 131-174, 1996.

[45] S. Gregor, "The Nature of Theory in Information Systems," *MIS Quarterly,* vol. 30, no. 3, pp. 611-642, 2006.

[46] S. Gregor and D. Jones, "The Anatomy of a Design Theory," *J. Assoc. Information Systems,* vol. 8, no. 5, pp. 312-335, 2007.

[47] C.A. Gurr, "Effective Diagrammatic Communication: Syntactic, Semantic and Pragmatic Issues," *J. Visual Languages and Computing,* vol. 10, pp. 317-342, 1999.

[48] J. Hahn and J. Kim, "Why Are Some Diagrams Easier to Work With?: Effects of Diagrammatic Representation on the Cognitive Integration Process of Systems Analysis and Design," *ACM Trans. Computer-Human Interaction,* vol. 6, no. 3, pp. 181-213, 1999.

[49] T.A. Halpin, "ORM 2 Graphical Notation," Technical Report ORM2-01, Neumont Univ., 2005.

[50] D. Harel, "On Visual Formalisms," *Comm. ACM,* vol. 31, no. 5, pp. 514-530, 1988.

[51] D. Harel and B. Rumpe, "Meaningful Modeling: What's the Semantics of 'Semantics'," *Computer,* vol. 37, no. 10, pp. 64-72, Oct. 2004.

[52] S. Hitchman, "Practitioner Perceptions on the Use of Some Semantic Concepts in the Entity Relationship Model," *European J. Information Systems,* vol. 4, no. 1, pp. 31-40, 1995.

[53] S. Hitchman, "The Details of Conceptual Modelling Notations Are Important—A Comparison of Relationship Normative Language," *Comm. AIS,* vol. 9, no. 10, 2002.

[54] S. Hitchman, "The Entity Relationship Model and Practical Data Modelling," *J. Conceptual Modelling,* vol. 31, 2004.

[55] L. Horn, "Toward a New Taxonomy for Pragmatic Inference: Q-Based and R-Based Implicature," *Meaning, Form and Use in Context,* D. Schiffrin, ed., pp. 11-42, Georgetown Univ. Press, 1984.

[56] P. Irani and C. Ware, "Diagramming Information Structures Using 3D Perceptual Primitives," *ACM Trans. Computer-Human Interaction,* vol. 10, no. 1, pp. 1-19, 2003.

[57] P. Irani, C. Ware, and M. Tingley, "Using Perceptual Syntax to Enhance Semantic Content in Diagrams," *IEEE Computer Graphics and Applications,* vol. 21, no. 5, pp. 76-85, Sept./Oct. 2001.

[58] ISO, *Information Processing—Documentation Symbols for Data, Program and System Flowcharts, Program Network, Network Charts and System Resource Charts (ISO 5807-1985),* 1985.

[59] ISO/IEC, *ISO/IEC Standard 24744: Software Engineering—Metamodel for Development Methodologies,* 2007.

[60] S. Kalyuga, P. Ayres, S.K. Chang, and J. Sweller, "The Expertise Reversal Effect," *Educational Psychologist,* vol. 38, no. 1, pp. 23-31, 2003.

[61] J. Kim, J. Hahn, and H. Hahn, "How Do We Understand a System with (So) Many Diagrams? Cognitive Integration Processes in Diagrammatic Reasoning," *Information Systems Research,* vol. 11, no. 3, pp. 284-303, 2000.

[62] R. Kimball, "Is ER Modeling Hazardous to DSS?" *DBMS Magazine,* 1995.

[63] K. Koedinger and J. Anderson, "Abstract Planning and Conceptual Chunks: Elements of Expertise in Geometry," *Cognitive Science,* vol. 14, pp. 511-550, 1990.

[64] S.M. Kosslyn, "Graphics and Human Information Processing," *J. Am. Statistical Assoc.,* vol. 80, no. 391, pp. 499-512, 1985.

[65] S.M. Kosslyn, "Understanding Charts and Graphs," *Applied Cognitive Psychology,* vol. 3, pp. 185-226, 1989.

[66] S.M. Kosslyn, *Graph Design for the Eye and Mind.* Oxford Univ. Press, 2006.

[67] J. Lamping and R. Rao, "The Hyperbolic Browser: A Focus + Context Technique for Visualizing Large Hierarchies," *J. Visual Languages and Computing,* vol. 7, pp. 33-55, 1996.

[68] *Enterprise Architecture at Work: Modelling, Communication and Analysis,* M. Lankhorst, ed. Springer-Verlag, 2005.

[69] J.H. Larkin and H.A. Simon, "Why a Diagram Is (Sometimes) Worth Ten Thousand Words," *Cognitive Science,* vol. 11, no. 1, pp. 65-100, 1987.

[70] J. Lee, "Design Rationale Systems: Understanding the Issues," *IEEE Expert,* vol. 12, no. 3, pp. 78-85, May/June 1997.

[71] W. Lidwell, K. Holden, and J. Butler, *Universal Principles of Design: A Cross-Disciplinary Reference.* Rockport Publishers, 2003.

[72] G.L. Lohse, "A Cognitive Model for Understanding Graphical Perception," *Human-Computer Interaction,* vol. 8, no. 4, pp. 353-388, 1993.

[73] G.L. Lohse, "The Role of Working Memory in Graphical Information Processing," *Behaviour and Information Technology,* vol. 16, no. 6, pp. 297-308, 1997.

[74] G.L. Lohse, D. Min, and J.R. Olson, "Cognitive Evaluation of System Representation Diagrams," *Information and Management,* vol. 29, pp. 79-94, 1995.

[75] K. Lynch, *The Image of the City.* MIT Press, 1960.

[76] M. Lynch, "The Externalized Retina: Selection and Mathematization in the Visual Documentation of Objects in the Life Sciences," *Human Studies,* vol. 11, pp. 201-234, 1988.

[77] J. Mackinlay, "Automating the Design of Graphical Presentations of Relational Information," *ACM Trans. Graphics,* vol. 5, no. 2, pp. 110-141, 1986.

[78] D.C. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* W.H. Freeman and Company, 1982.

[79] J. Martin and C. Finkelstein, *Information Engineering.* Savant Inst., 1981.

[80] K. Masri, D. Parker, and A. Gemino, "Using Iconic Graphics in Entity Relationship Diagrams: The Impact on Understanding," *J. Database Management,* vol. 19, no. 3, pp. 22-41, 2008.

[81] R.E. Mayer and R. Moreno, "Nine Ways to Reduce Cognitive Load in Multimedia Learning," *Educational Psychologist,* vol. 38, no. 1, pp. 43-52, 2003.

[82] M. McCloskey, "Intuitive Physics," *Scientific Am.,* vol. 248, no. 4, pp. 122-130, 1983.

[83] G.A. Miller, "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information," *The Psychological Rev.,* vol. 63, pp. 81-97, 1956.

[84] D.L. Moody, "Complexity Effects on End User Understanding of Data Models: An Experimental Comparison of Large Data Model Representation Methods," *Proc. 10th European Conf. Information Systems,* 2002.

[85] D.L. Moody, "What Makes a Good Diagram? Improving the Cognitive Effectiveness of Diagrams in IS Development," *Proc. 15th Int'l Conf. Information Systems Development,* 2006.

[86] D.L. Moody, "Review of ArchiMate: The Road to International Standardisation," technical report, ArchiMate Foundation and BiZZDesign B.V., 2007.

[87] D.L. Moody, "Theory Development in Visual Language Research: Beyond the Cognitive Dimensions of Notations," *Proc. IEEE Symp. Visual Languages and Human-Centric Computing,* 2009.

[88] D.L. Moody, P. Heymans, and R. Matulevicius, "Improving the Effectiveness of Visual Representations in Requirements Engineering: An Evaluation of the i* Visual Notation," *Proc. 17th IEEE Int'l Conf. Requirements Eng.,* 2009.

[89] D.L. Moody, R.M. Mueller, and C. Amrit, "Review of ORES Methodology, Notation and Toolset," technical report, Egon-Sparenberg B.V., 2007.

[90] D.L. Moody and J. van Hillegersberg, "Evaluating the Visual Syntax of UML: An Analysis of the Cognitive Effectiveness of the UML Suite of Diagrams," *Proc. First Int'l Conf. Software Language Eng.,* 2008.

[91] S.J. Morris and O.C.Z. Gotel, "Flow Diagrams: Rise and Fall of the First Software Engineering Notation," *Proc. Int'l Conf. Theory and Application of Diagrams,* 2006.

[92] A.A. Newell and H.A. Simon, *Human Problem Solving.* Prentice-Hall, 1972.

[93] M. Niemela and J. Saarinen, "Visual Search for Grouped versus Ungrouped Icons in a Computer Interface," *Human Factors,* vol. 42, no. 4, pp. 630-635, 2000.

[94] J.C. Nordbotten and M.E. Crosby, "The Effect of Graphic Style on Data Model Interpretation," *Information Systems J.,* vol. 9, no. 2, pp. 139-156, 1999.

[95] D. Norman, *The Design of Everyday Things.* MIT Press, 1988.

[96] L.P. Novick, "The Importance of Both Diagrammatic Conventions and Domain-Specific Knowledge for Diagram Literacy in Science: The Hierarchy as an Illustrative Case," *Diagrammatic Representation and Inference,* D. Barker-Plummer, R. Cox, and N. Swoboda, eds., Springer, 2006.

[97] J. Oberlander, "Grice for Graphics: Pragmatic Implicature in Network Diagrams," *Information Design J.,* vol. 8, no. 2, pp. 163-179, 1996.

[98] Object Management Group (OMG), *Unified Modeling Language Version 2.0: Superstructure,* 2005.

[99] A.L. Opdahl and B. Henderson-Sellers, "Ontological Evaluation of the UML Using the Bunge-Wand-Weber Model," *Software and Systems Modelling,* vol. 1, no. 1, pp. 43-67, 2002.

[100] A. Paivio, *Mental Representations: A Dual Coding Approach.* Oxford Univ. Press, 1986.

[101] S. Palmer and I. Rock, "Rethinking Perceptual Organisation: The Role of Uniform Connectedness," *Psychonomic Bull. and Rev.,* vol. 1, no. 1, pp. 29-55, 1994.

[102] M. Patrignani, "Visualization of Large Graphs," Dottorato di Ricerca (doctoral dissertation), Ingegneria Informatica, Univ. degli Studi di Roma, 2003.

[103] C.S. Peirce, *Charles S. Peirce: The Essential Writings (Great Books in Philosophy),* E.C. Moore, ed. Prometheus Books, 1998.

[104] M. Petre, "Why Looking Isn't Always Seeing: Readership Skills and Graphical Programming," *Comm. ACM,* vol. 38, no. 6, pp. 33-44, 1995.

[105] S. Pinker, "A Theory of Graph Comprehension," *Artificial Intelligence and the Future of Testing,* R. Freedle, ed., pp. 73-126, Lawrence Erlbaum and Assoc., 1990.

[106] K.R. Popper, "Science as Falsification," *Conjectures and Refutations,* pp. 30-39, Routledge and Keagan Paul, 1963.

[107] H.C. Purchase, D. Carrington, and J.-A. Allder, "Empirical Evaluation of Aesthetics-Based Graph Layout," *Empirical Software Eng.,* vol. 7, no. 3, pp. 233-255, 2002.

[108] H.C. Purchase, R. Welland, M. McGill, and L. Colpoys, "Comprehension of Diagram Syntax: An Empirical Study of Entity Relationship Notations," *Int'l J. Human-Computer Studies,* vol. 61, pp. 187-203, 2004.

[109] P.T. Quinlan, "Visual Feature Integration Theory: Past, Present and Future," *Psychological Bull.,* vol. 129, no. 5, pp. 643-673, 2003.

[110] A. Renkl and R.K. Atkinson, "Structuring the Transition from Example Study to Problem Solving in Cognitive Skill Acquisition: A Cognitive Load Perspective," *Educational Psychologist,* vol. 38, no. 1, pp. 15-23, 2003.
[111] N. Rescher, *Methodological Pragmatism: Systems-Theoretic Approach to the Theory of Knowledge.* Basil Blackwell, 1977.
[112] A.H. Robinson, J.L. Morrison, P.C. Muehrcke, A.J. Kimerling, and S.C. Guptill, *Elements of Cartography,* sixth ed. Wiley, 1995.
[113] R.G. Ross, *The Business Rule Book,* second ed. Database Research Group, 1996.
[114] B. Rossion and G. Pourtois, "Revisiting Snodgrass and Vanderwart's Object Pictorial Set: The Role of Surface Detail in Basic-Level Object Recognition," *Perception,* vol. 33, pp. 217-236, 2004.
[115] M. Scaife and Y. Rogers, "External Cognition: How Do Graphical Representations Work?" *Int'l J. Human-Computer Studies,* vol. 45, pp. 185-213, 1996.
[116] A.-W. Scheer, *ARIS—Business Process Modelling,* third ed. Springer, 2000.
[117] T.M. Shaft and I. Vessey, "The Role of Cognitive Fit in the Relationship between Software Comprehension and Modification," *MIS Quarterly,* vol. 30, no. 1, pp. 29-55, 2006.
[118] G. Shanks, E. Tansley, J. Nuredini, D. Tobin, and R. Weber, "Representing Part-Whole Relations in Conceptual Modeling: An Empirical Evaluation," *MIS Quarterly,* vol. 32, no. 4, pp. 553-571, 2008.
[119] G.G. Shanks and P. Darke, "Understanding Corporate Data Models," *Information and Management,* vol. 35, pp. 19-30, 1998.
[120] G.G. Shanks, E. Tansley, and R.A. Weber, "Using Ontology to Validate Conceptual Models," *Comm. ACM,* vol. 46, no. 10, pp. 85-89, 2003.
[121] C.E. Shannon and W. Weaver, *The Mathematical Theory of Communication.* Univ. of Illinois Press, 1963.
[122] P. Shoval and S. Shiran, "Entity-Relationship and Object-Oriented Data Modeling: An Experimental Comparison of Design Quality," *Data and Knowledge Eng.,* vol. 21, pp. 297-315, 1997.
[123] K. Siau, "Informational and Computational Equivalence in Comparing Information Modelling Methods," *J. Database Management,* vol. 15, no. 1, pp. 73-86, 2004.
[124] K. Siau and Q. Cao, "Unified Modeling Language: A Complexity Analysis," *J. Database Management,* vol. 12, no. 1, pp. 26-34, 2001.
[125] H.A. Simon, *Sciences of the Artificial,* third ed. MIT Press, 1996.
[126] I. Spence, "Visual Psychophysics of Simple Graphical Elements," *J. Experimental Psychology: Human Perception and Performance,* vol. 16, no. 4, pp. 683-692, 1990.
[127] S.S. Stevens, *Psychophysics.* John Wiley & Sons, 1975.
[128] J. Sweller, "Cognitive Load Theory, Learning Difficulty and Instructional Design," *Learning and Cognition,* vol. 4, pp. 295-312, 1994.
[129] A. Treisman, "Perceptual Grouping and Attention in Visual Search for Features and for Objects," *J. Experimental Psychology: Human Perception and Performance,* vol. 8, pp. 194-214, 1982.
[130] A. Treisman, "Search, Similarity and Integration of Features between and within Dimensions," *J. Experimental Psychology: Human Perception and Performance,* vol. 17, no. 3, pp. 652-676, 1991.
[131] A. Treisman and G.A. Gelade, "A Feature Integration Theory of Attention," *Cognitive Psychology,* vol. 12, pp. 97-136, 1980.
[132] E.R. Tufte, *Envisioning Information.* Graphics Press, 1990.
[133] E.R. Tufte, *The Visual Display of Quantitative Information,* second ed. Graphics Press, 2001.
[134] E.R. Tufte, *Beautiful Evidence.* Graphics Press, 2006.
[135] O. Turetken, D. Schuff, R. Sharda, and T.T. Ow, "Supporting Systems Analysis and Design through Fisheye Views," *Comm. ACM,* vol. 47, no. 9, pp. 72-77, 2004.
[136] M.P. Verdi, S.M. Crooks, and D.R. White, "Learning Effects of Print and Digital Geographic Maps," *J. Research and Technology in Education,* vol. 35, no. 2, pp. 290-302, 2003.
[137] I. Vessey, "Cognitive Fit: A Theory-Based Analysis of the Graphs versus Tables Literature," *Decision Sciences,* vol. 22, pp. 219-240, 1991.
[138] I. Vessey and Galletta, "Cognitive Fit: An Empirical Study of Information Acquisition," *Information Systems Research,* vol. 2, pp. 63-84, 1992.
[139] A. von Klopp Lemon and O. von Klopp Lemon, "Constraint Matching for Diagram Design: Qualitative Visual Languages," *Proc. First Int'l Conf. Theory and Application of Diagrams,* 2000.
[140] Y. Wand and R.A. Weber, "An Ontological Model of an Information System," *IEEE Trans. Software Eng.,* vol. 16, no. 11, pp. 1282-1292, Nov. 1990.
[141] J. Watts-Perotti and D.D. Woods, "How Experienced Users Avoid Getting Lost in Large Display Networks," *Int'l J. Human-Computer Interaction,* vol. 11, no. 4, pp. 169-299, 1999.
[142] R.A. Weber, "Are Attributes Entities? A Study of Database Designers' Memory Structures," *Information Systems Research,* vol. 7, no. 2, pp. 137-162, 1996.
[143] R.A. Weber, *Ontological Foundations of Information Systems (Coopers and Lybrand Accounting Research Methodology Monograph No. 4),* p. 212, Coopers and Lybrand, 1997.
[144] D.A. Weigmann, D.F. Dansereau, E.C. McCagg, K.L. Rewey, and U. Pitre, "Effects of Knowledge Map Characteristics on Information Processing," *Contemporary Educational Psychology,* vol. 17, pp. 136-155, 1992.
[145] M. Wertheimer, "Laws of Organization in Perceptual Forms," *A Source Book of Gestalt Psychology,* W. Ellis, ed., pp. 71-88, Routledge and Kegan Paul, 1938.
[146] C. Wheildon, *Type and Layout: Are You Communicating or Just Making Pretty Shapes?* Worsley Press, 2005.
[147] A.W. White, *The Elements of Graphic Design: Space, Unity, Page Architecture and Type.* Allworth Press, 2002.
[148] W.D. Winn, "Encoding and Retrieval of Information in Maps and Diagrams," *IEEE Trans. Professional Comm.,* vol. 33, no. 3, pp. 103-107, Sept. 1990.
[149] W.D. Winn, "An Account of How Readers Search for Information in Diagrams," *Contemporary Educational Psychology,* vol. 18, pp. 162-185, 1993.
[150] E. Yu, "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering," *Proc. Third IEEE Int'l Conf. Requirements Eng.,* 1997.
[151] J. Zhang, "The Nature of External Representations in Problem Solving," *Cognitive Science,* vol. 21, no. 2, pp. 179-217, 1997.
[152] J. Zhang and D.A. Norman, "Representations in Distributed Cognitive Tasks," *Cognitive Science,* vol. 18, no. 1, pp. 87-122, 1994.

Daniel L. Moody received the PhD degree in Information Systems from the University of Melbourne (Australia), but his career spans research and practice equally. He has held academic positions at universities in Australia, Brazil, Czech Republic, Iceland, Netherlands, Norway, Slovenia, and Spain, has published more than 100 scientific papers, and chaired several international conferences. He has also held senior IT positions in some of Australia's largest private organizations and has conducted consulting assignments in 12 different countries (in Asia, Europe, and North America). He has consulted in a wide range of industries, including banking, law enforcement, television, pharmaceuticals, biotechnology, airlines, emergency services, healthcare, education and the environment. He is a member of the IEEE, the IEEE Computer Society, and the Association of Information Systems (AIS), is the current President of the Australian Data Management Association (DAMA), and is listed in *Who's Who in Science and Engineering*.