MIS Quarterly

THEORY & REVIEW

COGNITIVE DIAGRAM UNDERSTANDING AND TASK PERFORMANCE IN SYSTEMS ANALYSIS AND DESIGN[1]

Monika Malinova
Vienna University of Economics and Business, Welthandelsplatz 1,
1020 Vienna, AUSTRIA {monika.malinova@wu.ac.at}

Jan Mendling
Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, AUSTRIA and
University of Ljubljana, Kongresni trg 12, 1000 Ljubljana, SLOVENIA {jan.mendling@wu.ac.at}

Models play an important role in systems analysis and design (SAD). A diagrammatic model is defined as a mapping from a domain to a visual representation in such a way that relevant information is preserved to meet a specific goal. So far, cognitive research on diagram criteria in relation to task performance has been fragmented. The aim of this paper is to (1) consolidate research on the cognitive processing steps involved during understanding and task performance with diagrams, (2) consolidate corresponding criteria for such diagrams to best support cognitive processing, and (3) demonstrate the support effective diagrams provide for performing SAD tasks. Addressing the first aim, we develop a theoretical cognitive framework of task performance with diagrams called CogniDia. It integrates different cognitive theories from research on diagrams in software engineering and information systems. Regarding the second aim, we review the literature to organize criteria for effective cognitive processing of diagrams. We identify research gaps on verbal and task processing. Regarding the third aim, we use the theoretical cognitive framework to investigate how diagrams support the SAD process effectively.

Keywords: Models, diagrams, systems analysis and design, cognitive processing, criteria, guidelines, task analysis, CogniDia

Introduction

Models continue to play an important role in systems analysis and design (SAD) (Fernández-Sáez et al. 2018; Recker et al. 2021). In the specification phase of classical development approaches, models help designers to understand the static and dynamic aspects of a to-be-designed information system (Wand and Weber 2002), or more generally, a domain, in which a specific system design problem is located. Models are also used for model-driven development (Gray and Rumpe 2018; Zhang and Patel 2010), enterprise architecture management (Lange et al. 2016), and business process improvement (Dijkman et al. 2012). The usage of models provides several benefits. First, developing models forces designers to make their domain understanding explicit. This helps them to establish a shared understanding of the domain with different stakeholders (Arias et al. 2000). Second, the explication of knowledge as models stimulates discussion and critique. Models can be validated by designers or verified by analysis tools. If errors are discovered already in models, they can still be easily corrected (Moody 1998). Third, the usage of models supports problem solving. Efficiency and effective-

[1]Jeffrey Parsons was the accepting senior editor for this paper. Ofer Arazy as the associate editor.

ness of models in this context has been studied in prior research (Gemino and Wand 2004). Fourth, models provide traceability from requirements to implementation decisions. In many cases, they directly serve as documentation (Davies et al. 2006).

The concept of a *model* (or script) in the context of SAD is often defined as an abstract mapping from a domain, in such way that it maintains relevant information in order to serve specific goals (Mendling 2008; Stachowiak 1973; Wand and Weber 2002), ranging from *ad hoc* information needs (Staples 2014) to supporting a process improvement initiative (Dumas et al. 2018). The models resulting from such a mapping can be represented in different ways. Most of the models created for specification use a diagrammatic representation by capturing information in a visual way. In this paper, we focus on those models in SAD that make use of such a diagrammatic representation. We refer to these diagrammatic models as *diagrams*, for brevity. Note that there are models that are not diagrammatic, such as textual use case descriptions, and diagrammatic representations that are not models, such as abstract visual graphs without a concrete domain reference. Both characteristics of a diagram (i.e., the semantic anchoring in a domain with its link to a goal and its diagrammatic representation) are equally important for research into their overall effectiveness. For this reason, we will use the term diagram in a narrow sense to refer to the intersection of models and diagrammatic representations.

The information systems (IS) discipline has largely benefited from representation theory for researching models (Burton-Jones et al. 2017; Recker and Green 2019; Wand and Weber 1990, 1993, 1995), providing criteria for ontological fidelity such as clarity and completeness, faithful tracking of real-world phenomena, and good decomposition. Burton-Jones et al. (2017) emphasize that representation theory is not "a psychological theory that explains how humans perceive or learn" (p. 1309). Also Wand and Weber (2002) identify the need for a theory that explains how humans use diagrams to accomplish various tasks. We are not aware that such an integrated theoretical framework has been developed meeting their call. Next to ontological research, there are various cognitive insights that should be reflected in such a theoretical framework. A challenge is that cognitive research in this area is fragmented in the following way. First, in terms of cognitive processing, prior research has developed several theories that take different perspectives on the subject matter. The list of theories to build upon is long, but not integrated with regard to how humans understand diagrams and use them to perform tasks. For instance, cognitive theories such as cognitive fit by Vessey (1991) or multimedia learning by Mayer (2002) hardly discuss the semantic connection of diagrams with modeling languages (or grammars), which is a crucial feature of diagrams in SAD as emphasized by Wand and Weber (2002). This means that many of the conjectures that cognitive theories make might be overly generic for diagrams. Second, there is a growing body of research contributions that investigate criteria grounded in diverse perspectives for evaluating diagrams, ranging from semiotics (Lindland et al. 1994), visual notations (Moody 2009), or ontology (Guizzardi 2005; Wand and Weber 1990). Many of these evaluations focus on specific types of diagrams. An example is the review by Figl (2017), who identifies 279 publications related to the comprehension of procedural visual process models alone. There are also relevant studies that focus on UML class diagrams and entity–relationship diagrams, on use case descriptions and flow charts, on different types of graphs, to name but a few. These studies on specific criteria are only partially integrated with cognitive theories, such that the respective findings are fragmented. For these reasons, a theoretical perspective is required for a better integration. Such a theoretical perspective will help to identify areas, in which research has been missing so far and where cognitive research and, for example, representation theory might complement each other.

This paper develops a theoretical framework for **cogni**tive understanding and task performance with **dia**grams (CogniDia). Our CogniDia framework integrates different theoretical lenses that have been used in research on diagrams in software engineering and IS research. The CogniDia framework aims to answer our first research question:

> *Which cognitive processing steps are involved during understanding and task performance with diagrams?*

We use the CogniDia framework as a classification device in a literature review to consolidate prior research and identify opportunities for future research. With this review, we aim to answer our second research question:

> *Which criteria have been identified by prior research to best support cognitive processing of diagrams?*

Our findings point to less explored areas, which motivate a shift of focus from artefact-centric research on diagrams towards a task-centric view. For this reason, we aim to answer our third research question:

> *How can we demonstrate the support effective diagrams provide for performing SAD tasks?*

We believe that a task-centric view on diagrams has the potential to stimulate research on how users benefit from diagrams to solve specific SAD tasks.

This paper is structured as follows. First, we establish the SAD process as a context for diagram usage and highlight the information needs of complex SAD tasks. We then discuss understanding and task performance with different external representations. The subsequent section introduces our CogniDia framework. We then describe the literature review method used and summarize the identified criteria in relation to the cognitive processing steps of the CogniDia framework. We proceed to illustrate the use of CogniDia for investigating how diagrams support SAD tasks, followed by a discussion of the implications and a presentation of our conclusions.

Diagrams in Systems Analysis and Design

In this section, we discuss the task context for cognitive processing of diagrams in SAD. To this end, we first describe the SAD process as a reference framework for diagram usage. Upon this foundation, we explain the connection between tasks and information needs served by diagrams. We will call those persons working on SAD tasks *designers* in a larger sense including analysts and developers, and those that are involved with SAD tasks (e.g., by providing information) *stakeholders*, which includes among others domain experts.

Systems Analysis and Design Process

Diagrams are extensively used in SAD. On GitHub alone, there are more than 93,000 UML diagrams (Robles et al. 2017). Recent surveys report an extensive usage of diagrams for problem solving (95%) and documentation (91%) (Hutchinson et al. 2014) using visual use case models (39%), data models (33%), and business process models (23%) as the most popular ones (Wagner et al. 2019). Big corporations like ABB use models at large scale to support development (Anda et al. 2006). A recent study by Sabegh and Recker (2017) found models to be used in *all* participating organizations.

This usage of diagrams relates to different tasks of the SAD process. We capture this process using the model of engineering theories by Staples (2014). We selected this model for several reasons. First, this model builds on the three worlds that Popper and Eccles (1977) describes for clarifying knowledge creation, which provides it with a solid grounding in the philosophy of science. Second, the model focuses on the consensus of design activities upon which frameworks for requirements engineering and software engineering agree (Bourque and Fairley 2014; Sommerville 2011; Wagner et al. 2019). Third, the model emphasizes the design process and abstracts from management tasks and from tasks that follow after the implementation, such as maintenance or configuration management.

Figure 1 illustrates the model by Staples (2014) with five abstract categories of engineering tasks, which we consider here for SAD. These tasks define a process from *elicit requirements* based on the anticipated usage situation, *refine requirements*, *specify design*, *decompose design* toward *implement system*. The usage situation (and also the information system as implemented) reside in World 1, the world of physical entities (Popper and Eccles 1977), which in SAD is also often referred to as the "real world" (Bourque and Fairley 2014). Both requirements and design specification belong to World 3, the world of abstract knowledge. The five task categories transition through World 2, the world of the cognitive tasks. Tasks produce different outputs. These include the requirements and design specification as well as the software code of the information system. The information system resides in World 1. The requirements and design specification both belong to World 3, which have various World 1 representations, often distributed over several dozens of information artifacts in practice (Fernández-Sáez et al. 2018).

The first task, elicit requirements, describes practices for discovering requirements from stakeholders, the application domain and the organizational environment (Cox et al. 2009). Its goal is to explicate and document all relevant requirements. These can be organized in a document or a system as a requirements specification. The second task, refine requirements, is the practice of resolving incompatibilities, missing information (Cox et al. 2009), and requirement conflicts (Sommerville 2011). The goal of this task is to refine the requirements specification until it is correct, complete and consistent. The third task, specify design, is the practice of identifying a design of the system that meets the requirements (Firesmith and Henderson-Sellers 2002). The goal of this task is to specify a high-level architecture of the system. The fourth task, decompose design, describes the practice of defining structure and behavior in detail (Firesmith and Henderson-Sellers 2002). The goal of this task is the specification of an object model, a database, and other components. The fifth task, implement system, is the practice of developing working software (Bourque and Fairley 2014). The goal of this task is having a running information system available that meets both the requirements and the design specification.

Systems Analysis and Design Tasks

SAD tasks differ in terms of their complexity, ranging from the development of a new enterprise system to incrementing

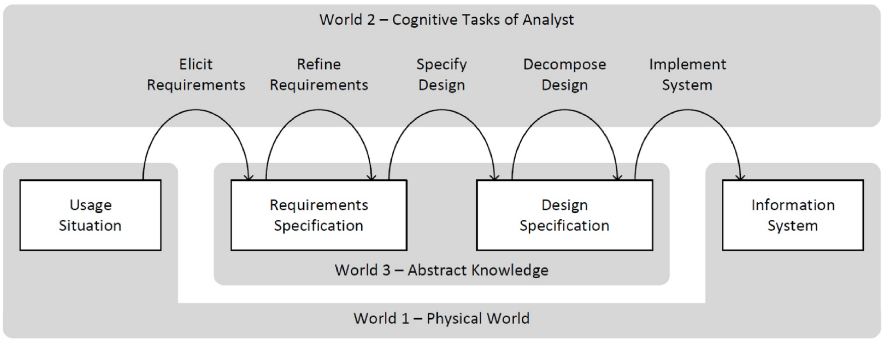

Figure 1. Categories of Tasks in System Analysis and Design (adapted from M. Staples, "Critical Rationalism and Engineering: Ontology," *Synthese* (191:10), © 2014)

the value of a variable by 1. The SAD tasks that we described above have in common that they are complex. Complex tasks can be characterized by unknown alternatives and uncertain consequences of action, inexact means–ends relationships and numerous subtasks (Campbell 1988; Simon and March 1958). These characteristics have different implications for task decomposition, task control, and information needs.

Complex tasks require *decomposition*; they cannot be processed in one single step. Humans decompose tasks using a goal hierarchy. The goal of a task can be achieved by meeting its subordinate goals, which in turn define subordinate tasks. In this way, designers apply means–ends analysis in order to construct a solution step by step (Newell and Simon 1972). Some of these subordinate goals match recurring tasks that are well defined and that can be addressed by designers using routine strategies. Table 1 shows the hierarchy of typical tasks for each of the five SAD tasks as described in Bourque and Fairley (2014), Graham et al. (1997) and Sommerville (2011). Other tasks are specific to the individual circumstances of a SAD project and require *ad hoc* problem solving by the designer. Though SAD tasks are diverse in what they aim to achieve, they are processed by humans using the same cognitive *control* mechanisms. Research on task analysis has studied these mechanisms in the fields of engineering psychology (Wickens et al. 2015), ergonomics (Stanton 2006), and human–computer interaction (John and Kieras 1996a, 1996b). No matter at which level a SAD task is anchored in the hierarchy, it is performed using a recurring sequence of *selection, action,* and *evaluation*. This sequence is described by Recker and Green (2019) and equally supported by prior research on the control of action processes (Beach and Mitchell 1978; Locke and Latham 1990) and design practice (Cross 2011; Gedenryd 1998; Rittel 1972).

The need for selection is connected with the fact that tasks have information needs. This required information can be available in the mind of the designer, for example if an algorithm has to be developed that sorts a list. The subsequent action can then be the implementation of the well-known quick sort algorithm. This is just one instance of a diverse spectrum of actions ranging from creating and editing as well as reading and analyzing information artifacts, acquiring information by looking it up or talking to stakeholders, challenging and updating working hypotheses, developing problem-solving strategies, etc. Evaluation is concerned with checking if the implemented code meets the goal of the task and if it is correct and complete. Evaluation might also lead to new insights, for example, that a design assumption does not work. This can trigger the designer to jump back to the requirements specification. In her seminal work, Guindon (1990) found that designers show a general tendency to progress from requirements elicitation to system implementation, but they also jumped back and forth opportunistically, such that requirements can still be added or changed at a late stage. Complex SAD tasks often have information needs that cannot be addressed by information present in the mind of the designer. In such a case, it has to be acquired from information artifacts or stakeholders. Sillito et al. (2008) observed that designers constantly (1) search for focus points for the next design task, (2) explore the boundaries of the relevant context of this focus point, (3) try to understand its structure, and (4) try to understand its connections with related structures. This means that such complex tasks constantly define information needs that have to be met by consulting information artifacts created by previous tasks of the SAD process.

Diagrams are information artifacts that can be used to support designers working on SAD tasks. They serve as external representations of task-relevant information outside the brain (Zhang 1997; Zhang and Norman 1994; Zhang and Patel 2006). In this way, they extend the capacity of working memory and knowledge in long-term memory. There are different

Table 1. Tasks

SAD Task	SAD Task (Level 2)	SAD Task (Level 3)	Source		
			1	2	3
Elicit requirements			x		
	Identify client's vision			x	
		Describe application		x	
		Elicit vision		x	
		Document vision		x	
	Define problem			x	
	Establish mission and objectives			x	
	Evaluate existing systems			x	
	Identify requirement' sources			x	x
	Identify elicitation techniques				x
	Identify requirements		x		
		Identify reusable requirements		x	
	Document requirements		x		
Refine requirements			x	x	x
	Analyze requirements		x	x	x
		Model application context		x	
		Model operational requirements		x	
		Model domain concepts		x	
	Understand requirements		x		
	Classify requirements		x		x
	Organize requirements		x		
	Prioritize requirements		x		
	Negotiate requirements		x		x
	Specify requirements		x	x	x
		Define system requirements			x
		Specify system requirements			x
		Specify software requirements			x
		Establish requirements for users		x	
		Establish user requirements for distributed systems		x	
		Establish user database requirements		x	
	Formalize requirements				x
	Ensure requirement-design fit			x	
	Analyze customer organizations			x	
	Analyze market			x	
	Analyze technology			x	
	Analyze users			x	
	Manage requirements change		x		
	Validate requirements		x		x
		Review requirements			x
		Prototype requirements			x
		Validate models			x
		Test acceptance			x

SAD Task	SAD Task (Level 2)	SAD Task (Level 3)	Source		
			1	2	3
Specify design			x		x
	Understand system context		x		
	Understand interactions		x		
	Design architecture		x	x	x
	Create system architecture			x	
		Choose architectural patterns		x	
		Apply architectural patterns		x	
		Reuse architecture		x	
		Determine major components		x	
		Determine reusable system components		x	
		Determine major mechanisms		x	
	Create software architecture			x	
		Apply architectural patterns		x	
		Develop layer design		x	
		Trace software architecture		x	
	Document architecture			x	
	Prototype architecture			x	
	Enforce architecture			x	
	Develop capacity plan			x	
	Establish data migration strategy			x	
	Manage subsystems			x	
Decompose design			x	x	x
	Identify objects		x		
	Specify interfaces		x		
	Develop design models		x		
	Engineer components			x	
		Screen candidate list of components		x	
		Evaluate potential components		x	
		Choose appropriate components		x	
		Establish policy on components acquisition		x	
		Screen candidate list of components		x	
		Identify reusable components		x	
		Integrate components		x	
	Design database			x	
		Design database model		x	
		Design and implement physical database		x	
		Design data centers		x	
	Design application			x	
		Prototype human architecture		x	x
		Design user interface		x	x
		Capture design		x	
		Document design		x	
		Optimize design		x	
		Refactor		x	
		Undertake usability design		x	
	Evaluate software design quality				x

SAD Task	SAD Task (Level 2)	SAD Task (Level 3)	Source		
			1	2	3
Implement system			x	x	x
	Plan implementation				x
	Manage implementation				x
	Design implementation				x
	Code			x	x
	Test implementation				x
		Test unit			x
		Test integration			x
		Test components	x		
		Test system	x		
		Test customers	x		
	Evaluate implementation quality				x
	Improve software components			x	
	Integrate			x	x
		Plan integration		x	
		Execute integration plans		x	
		Report on status of integration		x	
		Integrate with existing systems		x	
		Integrate content with user interface		x	
	Test software		x	x	x
		Plan software testing		x	x
		Validate software	x		x
		Verify software	x		x
		Design test suite		x	x
		Code test suite		x	
		Document test suite		x	
		Develop test environment			x
		Execute software testing		x	x
		Evaluate test results			x
		Report test results		x	x
		Track defects			x
	Deploy software			x	
		Deliver product to customer		x	
		Train users		x	
	Maintain software		x		x

Sources: (1) Sommerville 2011; (2) Graham et al. 1997; (3) Bourque and Fairley 2014

types of external representations. Benefits of using diagrams along the SAD process have been investigated in various empirical studies. They report, for example, more effective requirements elicitation with business process diagrams than with text (Trkman et al. 2016), better judgment of functional requirements with a UML-like notation as compared to text (Schlauderer and Overhage 2018), improved traceability from requirements to implementation by using UML diagrams (Anda et al. 2006), improved systematic refinement and verifiability by using UML diagrams (Bunse 2006), and improved source code comprehension by using UML diagrams (Dzidek et al. 2008; Fernández-Sáez et al. 2018; Scanniello et al. 2018). Also connections between types of tasks and types of diagrams have been reported in the literature. Appendix A gives an overview of diagram types used for tasks as according to textbooks. There is some empirical evidence that use case narratives and diagrams are more frequently used for requirements-related tasks and class diagrams and sequence diagrams for specification and system-related tasks, but there is no strict matching (Dobing and Parsons 2006).

The reported benefits of diagrams relate to understanding. Recker and Green (2019) define diagram understanding as a specific type of recurring task that is frequently used to support superordinate SAD tasks. The selection component of diagram understanding tasks is concerned with information needs of the task and taking decisions about which diagrams to read. The action component represents the performance gains based on understanding a diagram that was read. The evaluation involves reflections upon the usefulness of the diagram and an update about corresponding expectations.

The cognitive processing of the diagram understanding task holds the key for appraising the benefits of diagrams for superordinate SAD tasks in comparison to other types of external representations. But even though the mentioned empirical studies provide important pieces of evidence, they are hardly concerned with the development of a theory that explains how diagrams establish the observed benefits. Next we discuss characteristics of diagrams that help us to assess the benefits they provide in comparison to other representations.

Understanding and Task Performance with Different External Representations

The benefits that diagrams provide for SAD tasks are associated with information needs of the designer and their function as a specific type of external representation of this information. In this section, we define what diagrams are. We also explain how they differ from other types of external representations including text and text with images. We then discuss the benefits associated with diagrams for understanding and task performance.

Diagrammatic Models

Diagrams are a specific type of representation. Larkin and Simon (1987) distinguish two classes of representation: sentential and visual ones. A pure sentential representation is text. Its key characteristic is that its information can only be accessed in a sequential way. A pure form of visual representation are images. Their two-dimensional nature permits the rapid association of pieces of information based on their location and spatial proximity. We position diagrammatic representations in between text and images on a spectrum of different representation types.

Different types of diagrams are extensively used in SAD. A diagram can be understood as a product of a *mapping* from some original (Stachowiak 1973). This original can be in the real world or imagined, in this way embracing as-is and to-be diagrams as well as descriptive and normative diagrams. The mapping from a domain of interest to a diagram is an abstraction. It preserves relevant properties of that domain and abstracts from irrelevant details (Kühne 2006). The appropriateness of a specific mapping and its classification of relevant matters can only be established with reference to a goal. For this reason, the same domain of interest will certainly be mapped to different diagrams when the goals are different. Creating a diagram of a police car in a drawing book for children will differ from a police car diagram for a mechanical engineer who has to design the assembly line for its construction.

Diagrams are often created and used by different people. The designer creating a diagram is concerned with understanding and obtaining knowledge of the domain of interest. Developing this understanding requires the designer to consult documentation or to communicate with stakeholders (Frederiks and Van der Weide 2006). The creation of a diagram depends upon the understanding of the designer as emphasized by Krogstie et al. (2006) who state that "diagrams are explicit representations of some portions of reality as perceived by some actor" (p. 91). In order to overcome limitations or biases of subjective perceptions, there are different methods of validation and verification for establishing correctness and completeness (Dumas et al. 2018). While the creator of a diagram is mostly concerned with describing a domain for a given goal, the reader is more concerned with understanding a domain using the diagram. To this end, readers rely on prior knowledge of the domain and about how information is represented in the diagram. Note that designers are usually both creators and readers of diagrams along the SAD process.

Diagrams in SAD capture symbolic and linguistic information. In this way, they differ for example from models in engineering that are geometric (Clayton et al. 2002; Kyriakou et al. 2017), in geography that are spatial (Dent et al. 1999), or econometrics that are statistical (Granger 1981). A diagram in SAD is typically a visual representation that is constructed from lines, shapes, ideograms, glyphs, and terms. Its syntax is defined by a modeling language, whereas the diagram semantics are a combination of the modeling language and the natural language semantics (Leopold 2013). The different types of elements from which a diagram is built inherit properties from text (glyphs and terms) and from images (lines, shapes, and ideograms). For this reason, a diagram is best understood as a combination of verbal and visual elements. Figure 2 shows an example of a business process model and notation (BPMN) diagram of a simple order process, which includes verbal and visual elements. The process is triggered by the receipt of an order, which is then checked for completeness. If incomplete, the order is

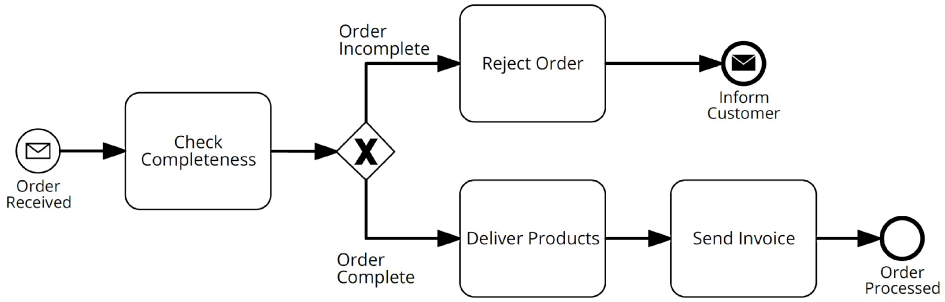

Figure 2. Example of a Business Process Diagram in BPMN Notation

rejected and the customer informed. If complete, the products are delivered and an invoice is sent. The diagram contains a *glyph* in the diamond-shaped element that describes a choice. The glyph, a capital X, alludes to this choice being "eXclusive." *Terms* are placed inside all activity elements and next to round event elements and the two arcs after the choice. Envelops as *ideograms* are used to indicate that the start and end event relate to sending and receiving of information. The rectangles with rounded corners are *shapes* that represent activities, the circles represent events and the diamond a choice gateway. Finally, these elements are connected with arcs as special types of *lines*. Predefined visual elements are often called *symbols*. The semantics associated with a diagram and its elements are partially inherited and partially idiosyncratic. The textual elements of diagrams inherit their semantics from natural language and a domain while ideograms point to abstracted entities that can often be associated with concrete images. The envelop in the BPMN diagram is such a case. Lines, shapes, and partially ideograms obtain idiosyncratic semantics from their connection with the modeling languages. For instance, the interpretation of circles as events is defined by the modeling language described in the BPMN specification.

Languages consist of a predefined set of elements and rules for combining them (syntax) as well as a mapping from the elements to a domain (semantics). Natural language consists of words (syntax) that have a meaning (semantics) and grammatical rules for building phrases and sentences (Larkin and Simon 1987). *Modeling languages* form a specific type of language. They specify, visualize, construct, describe and document some domain of the real world (Paige et al. 2000). Modeling languages are often associated with diagrammatic representations, as emphasized by definitions of Wand and Weber (2002) and Moody (2009). Compared to a sentential language, a modeling language contains both an abstract and concrete syntax (Harel and Rumpe 2000; Moody 2009). *Abstract syntax* includes all concepts a modeling language offers and rules that are used to combine them to form valid expressions. Those types of lines, shapes, and ideograms that are predefined by the modeling language define the visual vocabulary (Harel and Rumpe 2000). They are also referred to as *concrete syntax* (i.e., the visual representation of the concepts that comprise the abstract syntax). The predefined concrete syntax is also called *primary notation*, as opposed to orthogonal visual highlighting, which is called *secondary notation* (Green and Petre 1996). The set of underlying concepts that are associated with concrete syntax elements together with the rules to combine them are referred to as *abstract syntax*, whereas the meaning of the concepts is called *semantics* (Harel and Rumpe 2000).

The choice of representation has an impact on how easily the reader can access the information (Kintsch and Van Dijk 1978; Schnotz 2005). Next, we summarize research on understanding and task performance using different representation types.

Understanding and Task Performance with Only Text or Only Image

Text is a type of representation that consists of words combined into phrases and full sentences. The most foundational property of a text in relation to understanding is its *readability*, that is, how easily its content can be accessed by a reader. Readability of text differs according to the way words and sentences are constructed, which impacts a reader's understanding, reading speed, and level of interest in the material (Collins-Thompson 2014). The construction–integration model (Kintsch 1988) describes that the cognitive processes associated with text reading refer to three levels: constructing the surface form, the text base, and the situation model. First, the reading of a text yields a representation of its linguistic structure in the sensory memory of the reader, called the surface form (Pearson and Cervetti 2015). Second, the connection of this surface form with the reader's syntactic knowledge shapes the text base, which is maintained in the

working memory (Kintsch 1988). Third, the active combination of the text base with semantic knowledge yields the situation model. Text understanding is only achieved at this level (McNamara et al. 1996).

Various factors play a role for the successful processing at each level of text reading, most notably who is reading, what is read, and why it is read (Just and Carpenter 1980; Kendeou et al. 2016; Kintsch 1988; Pearson and Cervetti 2015). An important reader characteristic is expertise. It has a strong influence on how difficult a text appears to be (Benjamin 2012). Searching for a specific piece of information in a text is often overwhelming, because it requires a linear search down the text (Larkin and Simon 1987).

Performing tasks require both searching and understanding (Newell and Simon 1972; Simon 1978). Readers address them by continuously searching through the text and understanding its relevance for the solution until a satisfactory solution has been found.

Chi et al. (1981) find that the speed with which a task is completed depends heavily on the reader's skills. Their study reports that the absence of prior knowledge typically results in a linear search down the entire text. According to Chi et al., experts often make use of selective search by choosing good moves and then conducting a limited search to test the choice of moves. In essence, understanding and task performance supported by *images* works fairly similar along the three memory stages (Schnotz 2005). To this end, visual entities have to be recognized in the image based on gestalt laws (Wertheimer 1923), semantically interpreted using prior knowledge, and organized to match the task at hand (Hochpöchler et al. 2013). Here, the working memory of the visual channel is the bottleneck, while text is typically processed via the verbal channel. The visual channel can process information in parallel, which is in contrast to text where reading is done sequentially (Clark and Paivio 1991). For these reasons, certain tasks like search can be better supported by images and diagrammatic representations than by text only (Larkin and Simon 1987).

Understanding and Task Performance with Complementary Text and Image

Text and images have different strengths that are grounded in the characteristics of the visual and verbal channels. The combination of both has been investigated in the area of instructional design (Mayer 2002; Schnotz 2005). The theoretical argument for the benefit of this combination is provided by the dual coding theory. This theory posits two parallel processing channels for verbal information (text) and for nonverbal information (images) (Paivio 1991). In this way, the limited capacity of the working memory can be used more efficiently (Paivio 1991), and can result in better recall and respective task performance (Clark and Paivio 1991).

The parallel processing of dual-coded information also has disadvantages. It requires cognitive coordination and extra effort to construct a coherent mental model of the meaning captured in both text and image (Schnotz 2005). The construction of a joint mental model is only possible if text and image are semantically related and simultaneously available in working memory (Mayer 2002; Schnotz 2005). If this prerequisite is fulfilled, there are two types of connections that have to be constructed to link both (Paivio 1991). First, *associative connections* have to be established by linking words within the verbal cognitive system and by linking images within the visual cognitive system. Second, *referential connections* have to be established to link words with images (Paivio 1991).

The construction of these referential connections represents an extra effort of dual coding that can lead to inefficient cognitive processing. First, we might observe a *split-attention effect*, when the working memory is overwhelmed by constructing these connections (Chandler and Sweller 1991; Sweller 1988). Second, the full benefits of dual coding can only materialize when images are presented via the visual channel and words via the verbal channel. The *modality effect* occurs when words have to be recoded from the visual to the verbal channel (Chandler and Sweller 1991; Sweller 1988). Third, there will be likely a *redundancy effect*, because text and image have to overlap at least partially before any connections can be made (Chandler and Sweller 1991; Sweller 1988). The principles of multimedia learning established by Mayer (2002) give directions for preparing information in such a way that these disadvantages of dual coding do not occur.

Various benefits of a combined usage of text and image have been demonstrated in the studies by Mayer. It was also observed that tasks with multimedia input combining text and image are approached differently depending on expertise (Hochpöchler et al. 2013). Experts tend to follow a task-specific information-selection approach, in which only the required verbal and nonverbal information is selected to complete the task at hand. Novices tend to use a coherence-formation approach, in which the entire text is read and the accompanying image inspected (Hochpöchler et al. 2013).

Understanding and Task Performance with Diagrams

Research on multimedia learning demonstrates the benefits of cognitive processing by combining text with images. The aim

of using diagrams is to build on these benefits by directly integrating verbal and visual information. The theoretical argument for diagrammatic formats is developed by Larkin and Simon (1987), who distinguish sentential and visual representation formats. Diagrams are meant to avoid some of the problems that have been identified for the combination of text and image. Both verbal and visual content of diagrams is tightly integrated in order to avoid the split-attention and redundancy effect. Still, diagrams inherit the extra effort of integrating verbal and visual information from the combination of text and image.

The components of a diagram afford for different task characteristics. *Glyphs* and *terms* provide the richness of natural language, often in abbreviated form, which is effectively processed using the verbal channel. *Lines* and *shapes* partition the canvas of the diagram. In this way, they facilitate spatial search (Larkin and Simon 1987). *Ideograms* offer immediate visual processing along the visual channel and intuitive interpretation (Gurr 1999).

Lines, shapes, and ideograms have clear semantics defined by the *modeling language*. Prior research identifies various factors that are important for task performance with diagrams. Also in this context, diagram, language, reader and task characteristics are distinguished (see Figl 2017). Diagram characteristics in essence refer to the combined complexity of verbal and visual elements (Chidamber and Kemerer 1994; Mendling, Reijers, and Recker 2010; Mendling et al. 2012), which can be eased by effective use of secondary notation (Moody 2009; Petre 1995; Petrusel et al. 2017). Several characteristics of the modeling language affect diagram usage. Research has found that deficiencies at the syntactic and semantic level of the language compromise task performance (Figl et al. 2012; Recker 2011). Expertise both at the level of language knowledge and domain knowledge is required to interpret a diagram well (Chi et al. 1981; Curtis et al. 1992; Gemino and Wand 2004; Petre 1995). And all these aspects interact with task characteristics, with understanding and problem solving being the essential categories (Khatri et al. 2006).

The advantages of the spatial search affordances provided by diagrams have been studied by Vessey (1991) and Gurr (1999), among others. The cognitive fit theory formulated by Vessey posits that the problem representation and the problem solving task shape the mental representation that eventually leads to the solution. For this reason, cognitive fit theory would propose a diagrammatic representation to meet the requirements of a search task. The theory also emphasizes the need to match the skills of the person performing a task with the representation and the task at hand (Vessey 1991).

We observe that there is a rich spectrum of research on both the cognitive processes of understanding text and understanding text complemented by image. However, a diagram is neither pure text, pure image, nor text complemented by an image, but rather an integrated representation consisting of text (glyphs and terms) and images (lines, shapes, and ideograms). First, diagrams combine the benefits of rich textual information with spatial search. This spatial-search affordance is specifically important for complex SAD tasks that heavily require searching context information (Sillito et al. 2008). Second, the interpretation of various diagram elements requires knowledge of the modeling language. Third, also diagram, reader and task characteristics influence understanding and task performance. Fourth, there should be a cognitive fit between the task at hand and the diagram. We observe that prior research discusses various strengths of diagrams. Next, we develop the CogniDia framework in order to explain the cognitive processes involved during task performance with diagrams from an integrated perspective.

Cognitive Processing of Diagrams

The previous section has shown that a specific framework for performing tasks with diagrams is missing. In this section, we build on established cognitive theories that cover important aspects of such a framework. First, we summarize these theories. Second, we present our cognitive framework of understanding and task performance with diagrams (CogniDia).

Relevant Cognitive Theories

Various theories from prior research describe aspects that are relevant for the cognitive processing of diagrams. We found the following theories to provide important building blocks towards an integrated framework: the integrated theory of the mind (ITM) (Anderson et al. 2004), dual coding theory (DCT) (Paivio 1991), cognitive theory of multimedia learning (CTML) (Mayer 2002), model of working memory (MoWM) (Baddeley 1992), cognitive load theory (CLT) (Sweller 1988), cognitive fit theory (CFT) (Vessey 1991), and human problem solving (HPS) (Newell and Simon 1972). We use the core constructs of these theories for describing the cognitive processes involved during understanding and task performance with diagrams. Specifically, we identify the following aspects that have to be reflected in an integrated framework.[2]

[2]A brief description of the main constructs from the seven theories our cognitive framework is built on can be found in Appendix B.

First, the integrated theory of the mind by Anderson et al. (2004) defines an architectural model of how the mind is organized. The mental operations involved in task performance with diagrams are coordinated and executed by a production system, also called *central executive* by Baddeley (1992). Several specialized modules exchange information with the central executive via buffers. Buffers hold the currently most relevant chunk for further processing, such as the verbal and visual elements of diagrams required for performing tasks. In one mental processing cycle, the central executive reads from the buffers, matches the items with available *production rules*, selects the most appropriate one, and executes it (Newell and Simon 1972). Production rules are low-level mental operations to achieve low-level tasks like adding a box to a diagram (Card et al. 1980). As a result, new items can be written to one or more of the buffers. This might include output to the intentional buffer, which is used to add new subgoals to the goal hierarchy or to mark subgoals as completed. The operation of the mind is conceptualized by theories such as GOMS (acronym for goals, operators, methods, and selection rules) (John and Kieras 1996a, 1996b), ACT-R (acronym for adaptive control of thought–rational) (Anderson et al. 2004; Anderson et al. 1997), and hierarchical task analysis (Stanton 2006). In essence, the mind performs a complex task by the help of goal decomposition. If a task is too complex to be directly achieved, it is decomposed into subtasks. This procedure cascades until subtasks are tractable and results are popped back to superordinate tasks.

Second, the dual-channel assumption states that humans process verbal and nonverbal information in separate, related systems. This assumption is associated with Paivio's dual coding theory (Clark and Paivio 1991) and Baddeley's model of working memory (Baddeley 1992). Diagrams include verbal elements as printed words. According to the sensory-modality approach by Baddeley, humans initially process printed words via their eyes and spoken words via their ears. In contrast, the presentation-mode approach of dual coding theory stresses that both spoken and printed words are considered as verbal, while illustrations, video or background sounds as nonverbal elements (Paivio 1991). The theory of multimedia learning by Mayer (2002) integrates both approaches by postulating that printed words are initially sensed by the eyes and then enter the verbal system. This reflects the fact that humans can mentally create sounds corresponding to the image of a word, such that they are further processed in the verbal system (Mayer 2002).

Third, according to the dual coding theory and the theory of multimedia learning, associative and referential connections are constructed to combine verbal and visual elements (Mayer 2002; Paivio 1991). For example, for the business process diagram from Figure 2 associative connections are established between the terms *order* and *received* which links them together in a phrase. Furthermore, there is a referential connection of the phrase *order received* with the circle filled with a white envelope, offering the interpretation that a message notifies when an order has been received.

Fourth, the central executive integrates elements of both verbal and nonverbal mental models (Baddeley 1992). At this stage, humans take advantage of prior knowledge stored in their long-term memory in order to facilitate the understanding of the diagram. This is in line with cognitive load theory, which states that prior knowledge facilitates automated information processing (Sweller 1988).

Fifth, according to cognitive fit theory, effective processing requires a fit between the problem representation and the task at hand (Vessey 1991). This argument is also valid for diagrams. Performing a task is supposedly easy if the task makes use of that type of information that is readily accessible from the diagram. This is echoed in recommendations that a modeling language (Curtis et al. 1992) and a diagram itself should be appropriate for a given objective (Krogstie et al. 1995b; Kühne 2006; Lankhorst 2013; Yadav et al. 1988). Humans perform a task by constantly searching and integrating solution fragments from the diagram until a satisfactory solution is reached. According to the human problem solving theory by Newell and Simon (1972), searching and integrating are the two types of cognitive processes involved in problem solving. Task performance follows the principle of hierarchical decomposition. To this end, a goal hierarchy is dynamically constructed in the intentional buffer, which controls search and integrate operations. In line with the cognitive load theory by Sweller (1988), it has to be highlighted that knowledge plays an important role at various stages of cognitive processing. The understanding aspects of information input builds on prior knowledge of different types.

A Cognitive Framework of Understanding and Task Performance with Diagrams

Here, we integrate the above mentioned theories and propose a cognitive framework of understanding and task performance with diagrams. The framework is shown in Figure 3. It describes the cognitive processes involved when humans read and understand diagrams, and solve tasks using them. Next, we discuss how this framework reflects prior theories.

Mental architecture: The cognitive framework of understanding and task performance with diagrams assumes a mental architecture as it is described by the integrated theory

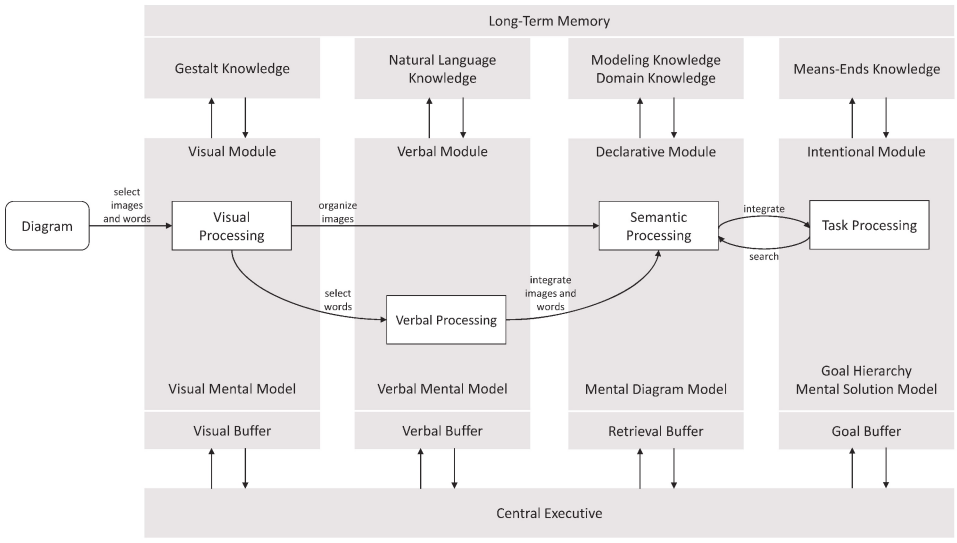

Figure 3. Cognitive Framework of Understanding and Task Performance with Diagrams (CogniDia)

of the mind (Anderson et al. 2004). Figure 3 shows the architectural components in grey. The *central executive* coordinates specialized modules via buffers and integrates knowledge from long-term memory. At a certain point in time, humans would not be paying attention to all the elements of a diagram, but only to a specific chunk (Anderson et al. 2004). This chunk is selected and held by the appropriate buffer. During processing in the declarative module, the mental diagram model is held by the retrieval buffer as the visual and verbal elements of the diagram are semantically processed. Similarly, humans would only attend to those facts that were recently retrieved from long-term memory necessary for processing (Anderson et al. 2004). Four modules described in Anderson et al. (2004) and Baddeley (1992) are of specific importance for task performance with diagrams. The *visual module* coordinates eye motion and focus, and is responsible for processing the nonverbal elements in diagrams. It utilizes gestalt knowledge. The *verbal module* processes words. It builds on natural language knowledge for parsing words and grammar. The *declarative module* holds explicit facts. It interacts with those parts of the long-term memory that keep domain and modeling knowledge. The *intentional module* maintains and manages a hierarchy of goals. It is connected with means– ends knowledge in the long-term memory.

This architectural model provides a functional view of the different modules and their interaction. Task performance with diagrams defines a logical sequence how these modules collaborate. Mind that cognitive processing is complex. Tasks that are identified by humans as elementary require more than a thousand different production rules and trigger tens of thousands of singular productions (Altmann and John 1999). Also, production rules often involve several of the different kinds of buffers both as input and output (Anderson et al. 2004). This means that the stages of the CogniDia framework are an abstraction of a typical sequence of numerous related productions. To this end, the CogniDia framework describes four stages of cognitive processing: visual processing, verbal processing, semantic processing and task processing.

Visual, verbal and semantic processing of diagrams: The diagram first enters the *visual module*. During the step of *visual processing*, the contents of the provided diagram are sensed by the eyes. The CogniDia framework considers diagrams as a specific input of cognitive processing, including both verbal and visual elements. Both types of elements are selected by the *visual buffer* for further processing. The two types reflect dual coding theory (Paivio 1991) and the model of working memory (Baddeley 1992).

First, glyphs and terms enter the nonverbal cognitive system. Humans mentally create corresponding sounds for the presented words, and as such they are transmitted by the central executive to the *verbal module* for processing via the *verbal buffer* in the verbal cognitive system (Anderson et al. 2004). Natural language knowledge is a prerequisite for understanding verbal elements. Ideograms, shapes and lines are processed by the nonverbal cognitive system. Gestalt knowledge is required for this step. These observations imply that reading a diagram involves both *visual processing* and *verbal*

processing (Mayer 2002). In the working memory, words are then linked with other words, and symbols with other symbols via associative connections, and words are linked with symbols via referential connections (Mayer 2002; Paivio 1991). The result is a *visual mental model* and a *verbal mental model* of the diagram with referential connections.

Both visual and verbal mental models are transmitted to the *declarative module* by the central executive (Anderson et al. 2004) for *semantic processing*. Semantic processing is concerned with the interpretation of the diagram in the working memory by integrating prior knowledge processed via the *retrieval buffer* (Anderson et al. 2004; Baddeley 1992). For reading diagrams, prior knowledge is required on the syntax and semantics of the *modeling language* and knowledge of the *domain* that the diagram describes. Domain knowledge is knowledge of a specific area to which a set of theoretical concepts is applied, while modeling knowledge of concepts and languages is domain-independent (Khatri et al. 2006). This knowledge is formally described as a domain ontology or a foundational ontology, respectively (Guizzardi 2005). According to the cognitive load theory, prior knowledge facilitates automated information processing, with the effect that cognitive capacity is freed for performing tasks (Sweller 1988). Semantic processing yields a *mental diagram model* representing the interpretation of the diagram in the working memory. At this stage, the cognitive framework reflects the theory of multimedia learning (Mayer 2002) and its active processing assumption during the integration of verbal and visual elements with prior knowledge.

The reading and understanding of diagrams is usually not done sequentially. At any stage of visual, verbal and semantic processing, the central executive system can loop back to the input diagram to iterate, refresh or confirm the understanding. In Figure 3, this is represented by the arcs connecting the different modules and the central executive.

Task processing: The next step of *task processing* largely builds on the interaction of the central executive with the goal hierarchy in the *intentional module* and the diagram in the declarative module. At the level of *task processing*, the cognitive framework reflects cognitive fit theory (Vessey 1991) and human problem solving theory (Newell and Simon 1972). According to cognitive fit theory, effective processing requires a fit between the problem representation and the task at hand (Vessey 1991). This argument is also relevant for diagrams as a problem representation. Diagrams are created to accomplish goals, which means that they anticipate (explicitly or implicitly) a specific category of tasks they aim to support. For example, the Level 2 and 3 tasks of the SAD task *elicit requirements* in Table 1 contribute to the goal of all requirements being documented.

Humans become aware of tasks either by being instructed by others via the visual and verbal channel or by retrieval from long-term memory. *Task processing* builds on the mental model of the diagram, *means–end knowledge* from long-term memory, and the goal hierarchy. The goal hierarchy represents a control structure for performing tasks. It is recursively constructed if a goal is too complex to be directly achieved. First, when performing a task, means–ends analysis is applied (Newell and Simon 1972) to find useful production rules that produce subtasks. In this way, a task like *validate requirements* from Table 1 can be decomposed into several subtasks, such as *review requirements*, *prototype requirements*, *validate models*, etc., which are further decomposed into checking if customer needs are captured, and so forth. This results in a dynamically constructed goal hierarchy, which is bottom-up resolved, as discussed in research on GOMS (Card et al. 1980; John and Kieras 1996a, 1996b) and ACT-R (Anderson et al. 2004; Anderson et al. 1997).

Second, search and integration processes are employed to build solution fragments based on the mental model of the diagram (Newell and Simon 1972). Put differently, the reader continuously searches the diagram for a part of the solution and integrates the new insights into an overall solution, while the *goal buffer* keeps track of the state of the task (Anderson et al. 2004). Task processing continues until a satisfactory solution to the provided task is constructed and a *mental solution model* is created. The reader could at any stage corroborate both the diagram and goal hierarchy.

Comparison with Prior Cognitive Theories

We observe that the seven mentioned theories provide important building blocks for describing the cognitive processes of performing tasks based on diagrams. As Table 2 highlights, none of the theories can by itself explain all cognitive processes involved during task performance with diagrams. The main cognitive processes of an integrated cognitive framework are concerned with diagrams as input and output. First, the cognitive processes involved with regard to the provided diagram are the separate processing of the visual and verbal elements shown in the diagram, and the semantic processing of the combination of the visual and verbal elements. Second, both inputs are integrated with different types of prior knowledge by the central executive. Third, task processing is controlled by the goal hierarchy. Last, the interconnected processing of diagram and goal hierarchy facilitates the construction of a mental solution model.

Each of these cognitive processes is covered by one or more of the seven theories specifically. For example, the dual coding theory supports the processing of visual and verbal information, however it does not mention the other types of

Table 2. Comparison of CogniDia with Focus of Other Cognitive Models

Concerns of an Integrated Cognitive Framework	ITM	DCT	CToML	MoWM	CLT	CFT	HPS	CogniDia
Processing of visual information	x	x	x	x				x
Processing of verbal information		x	x	x				x
Processing of semantic information	x		x	x				x
Processing of task	x					x	x	x
Integration of gestalt knowledge								x
Integration of natural language knowledge								x
Integration of modeling knowledge								x
Integration of domain knowledge			x	x	x			x
Integration of means–ends knowledge	x		x	x	x		x	x

processing. Similarly, the multimedia learning theory and the model of working memory do not explain how humans solve tasks. Table 2 shows that the CogniDia framework supports all concerns of an integrated cognitive framework of understanding and task performance with diagrams.

The CogniDia framework and prior theories highlight the potential benefits of using diagrams for performing tasks. Effective design of diagrams is a key challenge for reaping the benefits of a combined verbal and visual representation of information in diagrams (Larkin and Simon 1987). We address this point next by reviewing criteria for the effective cognitive processing of diagrams.

Criteria for Effective Cognitive Processing of Diagrams

Our second research question focuses on criteria that prior research formulates for supporting effective cognitive processing of diagrams. Next, we explain how we proceeded to systematically identify these criteria. Then, we present the criteria grouped by the processing steps of the cognitive framework of understanding and task performance with diagrams.

Review of Criteria for Effective Cognitive Processing of Diagrams

For identifying diagram criteria, we conducted a systematic literature review following the guidelines proposed by Kitchenham et al. (2009). As we are specifically interested in diagrammatic models (i.e., diagrams that serve as a model with corresponding modeling languages), we exclude general works on diagrams that do not refer to models. Among others, this excludes work on reasoning about diagrams in artificial intelligence (Anderson and McCartney 2003) and research on graph drawing (Hu and Nöllenburg 2016).

Our objective is to identify which characteristics a diagram should exhibit for being effective in terms of cognitive processing toward a correct mental solution model. Since diagrams refer to a modeling language, there are also criteria at the language level. We prepared a review protocol of our literature search that specifies the following: electronic libraries, keywords, type of literature, domain, and time frame of the search. For example, our search terms reflect alternative traditions in terminology including the closely related terms *language*, *grammar*, *notation*, and *technique*. We are also interested in relevant papers from a wide spectrum of IS domains where diagrams are often used, such as conceptual modeling, data modeling, information modeling, among others. We observed that criteria are discussed from three related angles: guidelines, principles, and quality criteria. We used these terms as our main keywords and the aforementioned domains as a second set of keywords to construct the search queries.

Paper Selection and Data Extraction

After iteratively reviewing the search results (e.g., title inspection, abstract reading, entire paper reading, additional search and backward search), we ended up with a total of 82 primary sources that we use for identifying criteria for modeling languages and diagrams. From each primary source we extracted the name, description, domain and source of the criteria. In addition, we noted which criteria are empirically supported. For this, we kept track of those sources that presented some type of empirical evaluation of the proposed criteria (e.g., experiment, case study, action research).

The data extraction resulted in a large set of data that we clustered into criteria for languages and diagrams. Next, we merged redundant criteria. We also ensured that criteria are consistently named and described. For instance, we always added a verb to a criterion label, such that it indicates a direction of what should be done to potentially improve the

diagram. Removing the redundant criteria left us with a total of 279 criteria. We went through these 279 criteria and assigned them to the four cognitive processing steps of our CogniDia framework (visual, verbal, semantic, and task processing). For instance, the criteria concerned with the visual aspects of diagrams are clustered in the category "visual processing." We found 59 criteria that did not relate to any of the cognitive processing steps and excluded them from further analysis. The remaining 220 criteria were then arranged into more specific categories. Each category includes criteria that pertain to a similar matter. For instance, many criteria are concerned with the modeling language fit with a real-world domain, which we put into the category "ontological fidelity." This procedure yielded 22 categories.[3]

Criteria for Effective Cognitive Processing of Diagrams

The categories of criteria that we identified are presented in Tables 4–7. Each table focuses on one of the cognitive processing steps of the CogniDia framework and its associated criteria. Each criterion category is briefly described and for each category we provide two criteria as examples. The numbers next to each category indicate its number of criteria.[4] In the following, we outline our findings.

General Observations on the Criteria

Reviewing the 82 research articles, we found that criteria belonging to visual, verbal and task processing directly support cognitive effectiveness of a specific diagram. Those criteria that facilitate semantic processing have a broader scope. More than half of them relate to properties of the modeling language and the rest to specific diagrams. For this reason, we distinguish between diagram quality and language quality criteria in the context of semantic processing.

Table 3 shows the number of articles published and criteria proposed for each cognitive processing step in intervals of five years (1988–2020), with the exception of Wertheimer (1923). Notably, the first article that proposes one criterion on verbal processing was published only in 2000, followed by 11 articles and an additional 20 criteria. Criteria in all other categories have been discussed over the whole observation period. Criteria for semantic and visual processing gained the most attention, in terms of both number of published articles and proposed criteria. The number of articles published since the 1990s has been stable with a decrease in the last decade.[5]

Some comments are warranted on the analysis of criteria.[6] First, we found that not all criteria are empirically supported. Some criteria are introduced with common sense arguments (e.g., Britton and Jones 1999), others are based on empirical research (e.g., Mendling, Reijers, and van der Aalst 2010), empirically evaluated by means of an experiment (e.g., Lloyd and Jankowski 1999) or a case study (e.g., Heravizadeh et al. 2008). Second, many criteria are also presented for a specific type of diagram. We kept track of the diagram type, but tried to generalize if possible. Third, there are also trade-offs between criteria. For example, ensuring a diagram is as informative as possible might undermine the criterion of using as few elements as possible. Fourth, criteria differ in their level of granularity. For instance, there are very abstract criteria, for example ensuring the visual aesthetics of a diagram, and more specific criteria, such as minimizing crossings of edges and nodes, which is a criterion that would, in fact, enhance the visual aesthetics of a diagram. In the remainder of this section, we lift the discussion to the level of criterion categories.

Visual Processing Criteria

Visual processing is concerned with processing of the visual elements in a diagram via the visual cognitive system. There are 92 criteria related to this category, which highlights the focus of prior research on the visual aspects of diagrams. Table 4 shows the seven categories of visual criteria with a brief description and examples.

Almost 40% of the criteria are concerned with the *aesthetics* of individual elements or combination of elements. Several criteria are concerned with the visual appearance of diagrams in general and aim to ensure a pleasant and satisfying interaction for the reader (Sánchez-González et al. 2013). Many criteria are concerned with the way elements are drawn within a diagram. For instance, prior research suggests appropriate positioning of elements (Eichelberger and Schmid 2009; Tamassia et al. 1988) and ensuring that elements and edges do not cross (Ding and Mateti 1990). Often recommended is the usage of secondary notation to provide visual cues to the readers (Green and Petre 1996; Guentert et al. 2012; Tsironis

[3]Details on the literature review method we employed and the list of the primary sources are shown in Appendix C.

[4]A full list of the criteria along with additional details (e.g., the domain where the criterion was applied and the source where we found the criterion) can be found in Appendix D.

[5]A chart illustrating the evolution of papers proposing criteria over time can be found in Appendix D.

[6] For detailed analysis see Appendix D.

Table 3. Number of Articles (Criteria) per Cognitive Step over Time

	Verbal Processing	Visual Processing	Semantic Processing–Diagram Quality	Semantic Processing–Language Quality	Task Processing
1923		1 (6)			
1988–89		2 (11)	2 (7)	1 (1)	1 (3)
1990–94		5 (23)	8 (25)	9 (28)	5 (8)
1995–99		9 (44)	14 (41)	9 (30)	13 (24)
2000–04	2 (4)	9 (47)	12 (39)	11 (21)	12 (18)
2005–09	3 (4)	6 (39)	15 (39)	13 (29)	9 (20)
2010–14	4 (10)	6 (23)	9 (27)	9 (32)	4 (9)
2015–18	2 (2)	1 (7)	1 (10)	2 (7)	2 (7)

Table 4. Visual Processing Criteria Categories

Category	Description	Example Criteria
Aesthetics (36)	Criteria concerned with the aesthetic features of individual elements and the combination of elements within diagrams (e.g., layout features, element usage).	• Ensure a diagram provides users with pleasing visual interaction. • Place disconnected elements at diagram border.
Visual consistency (23)	Criteria that provide users with an impression of consistency, logical coherence and uniformity of elements in diagrams.	• Repeat aspects throughout a diagram. • Ensure nothing is placed randomly in a diagram.
Visual simplicity (6)	Criteria concerned with decreasing the diagram complexity.	• Minimize the use of elements that do not add to language expressiveness. • Ensure symbols included in diagram are cognitively manageable.
Pop-out (6)	Criteria that ensure elements are emphasized in diagrams.	• Choose element size according to their importance. • Apply contrast in symbols.
Grouping (13)	Criteria concerned with grouping elements together in diagrams.	• Group elements that appear to move in the same direction. • Ensure reader can identify how each diagram element relates to the whole.
Symbol distinctiveness (2)	Criteria that ensure symbols can be distinguished from each other.	• Assign clear boundaries to the symbol. • Make symbols easy to recognize.
Symbol clarity (6)	Criteria that ensure symbols would be familiar and clear to the users.	• Make symbols familiar to its users. • Ensure symbols offer feedback to users.

et al. 2009). Although such syntactic modifications tend to improve the processing, its uncontrolled usage might potentially distract users (Karsai et al. 2014).

Most of the criteria on visual *grouping* emphasize the importance of clustering similar elements (ter Hofstede et al. 1993), either according to symbol familiarity (Eichelberger and Schmid 2009), visual similarity of elements (Britton and Jones 1999), or element connectivity (Eichelberger and Schmid 2009). *Consistency* is another matter discussed in prior research. Corresponding criteria such as repeating some aspect of the design throughout the diagram (Britton and Jones 1999) aim to provide users with an impression of consistency, logical coherence and uniformity of elements in diagrams.

Pop-out criteria ensure elements are easy to recognize in diagrams. For instance, symbols can be best processed when they can be clearly distinguished from the diagram background (Huang et al. 2002). Several works emphasize the benefits of using mnemonic symbols that allude to the intended semantics (Huang et al. 2002; Kesh 1995) and maximizing the familiarity of symbols (Britton et al. 2000; Huang et al. 2002; McDougall et al. 1999). These types of criteria lead to *symbol clarity* (i.e., ensuring symbols look familiar and clear to the users).

Table 5. Verbal Processing Criteria Categories

Category	Description	Example Criteria
Clarity of text (7)	Criteria that increase the clarity of text included in diagrams.	• Make information content explicit, not implicit. • Ensure users can easily discern all diagram abbreviations.
Effective presentation of text (7)	Criteria with regard to the effective presentation of text in diagrams.	• Use verb-object activity labels. • Make all text fonts the same.

Verbal Processing Criteria

Verbal processing is concerned with the processing of the verbal elements in diagrams via the verbal cognitive system. 14 criteria belong to this category. Table 5 shows that half of them are meant to increase the *clarity of text* included in diagrams (Moody 2009), while the other half ensures the effective presentation of text in diagrams. Most of them refer to style of labels (de Oca et al. 2014; Mendling, Reijers, and van der Aalst 2010).

Semantic Processing Criteria

Semantic processing involves the interpretation of diagram contents in the working memory by integrating prior knowledge. More than one third of all criteria we found relate to this step. The corresponding criteria do not address visual or verbal aspects in isolation, but rather consider their combined usage and their effect on semantic processing. Table 6 organizes these criteria in two blocks: *diagram quality* and *language quality*.

There are four categories of criteria concerned with diagram quality. Criteria in the category *semantic complexity* ensure that a diagram contains the minimal possible constructs (Cherfi et al. 2007) by, for instance, using appropriate concepts to describe corresponding elements from the real world (Blackwell et al. 2001). The quality of syntax, semantics, and pragmatics of diagrams is also prominently covered in prior research. *Semantic quality* criteria ensure that all components of a diagram are clearly defined (Wand and Wang 1996) by achieving a correct representation of real-world aspects (Burton-Jones et al. 2009) and ensuring it is appropriate for the domain being represented (Nelson et al. 2012). *Syntactic quality* criteria emphasize the importance of predefined rules imposed by the modeling language (Overhage et al. 2012). *Pragmatic quality* criteria are concerned with usability. To make a diagram usable means to ensure it can be readily understood (Krogstie and Sølvberg 2003) and that users can faithfully follow the steps the diagram specifies (Davis et al. 1993).

Many studies highlight the importance of a modeling language being faithful to a real-world domain (Blackwell et al. 2001; Gemino and Wand 2004; Green and Petre 1996; Lindland et al. 1994). Criteria that ensure a faithful representation of a real-world domain using a modeling language belong to the category *ontological fidelity*. In essence, these criteria ensure that there is a direct correspondence between the language constructs and ontological concepts. Prior research has shown that ontological fidelity, among others, is one of the key antecedents to cognitive effectiveness of diagrams (Recker et al. 2011). Research by Wand and Weber (1990, 1993) on representation theory and the corresponding Bunge–Wand–Weber ontology (BWW) has established standards for evaluating modeling languages (Burton-Jones et al. 2017). There are extensions for combined ontological quality of several models (Recker and Green 2019) and on the Unified Foundational Ontology (UFO) (Guizzardi 2005; Guizzardi et al. 2015).

A modeling language should also provide certain *language features* to its users. Armenise et al. (1993) state that a language should represent both technical and nontechnical activities. Other criteria refer to the way how elements are grouped or separated (Sánchez-González et al. 2013). Various aspects of *simplicity* of a modeling language are important for facilitating the semantic processing of diagrams. For example, a language should neither contain redundant elements (Parsons and Wand 2008; Shanks et al. 2003) nor include elements that lead to inefficient diagrams due to language design (Karsai et al. 2014). *Consistency* is also a major concern a language should reflect. In particular, it is vital to ensure that the specification of elements is consistent within the language (Bielkowicz and Tun 2001). Some of the criteria we found inform the design of a modeling language before the design has commenced. These criteria belong to the *fit with modeling knowledge* category and emphasize the importance of reusing existing language concepts, language definitions and notational elements (Karsai et al. 2014).

Table 6. Semantic Processing Criteria Categories

Category	Description	Example Criteria
Diagram Quality		
Semantic complexity (9)	Criteria that increase the simplicity of diagrams.	• Ensure diagram contains the minimal possible constructs. • Split up larger diagrams into smaller diagrams.
Syntactic quality (4)	Criteria about the importance of conforming to pre-defined rules.	• Ensure diagram conforms to language rules. • Impose necessary restrictions for well-formed statements in diagrams.
Semantic quality (8)	Criteria concerned with the validity and completeness of a diagram with regard to a domain.	• Ensure diagram is correct representation of a real-world aspect. • Do not model what you believe to be false.
Pragmatic quality (19)	Criteria with regard to the usability of diagrams by its users.	• Ensure feasible diagram comprehension is achieved. • Ensure diagram specifies all steps needed to produce accurate results.
Language Quality		
Fit with modeling knowledge (7)	Criteria about reusing knowledge from existing languages and ensuring the language is appropriate with the knowledge of its users.	• Reuse language concepts from existing languages. • Ensure users are able to master the language basics.
Ontological fidelity (13)	Criteria concerned with the modeling language being complete with regard to and fit with a real-world domain.	• Define language that fits a domain. • Ensure language achieves ontological completeness and clarity.
Simplicity (6)	Criteria that decrease the complexity of a modeling language.	• Ensure language offers no redundant elements. • Avoid elements that are not desired by users.
Consistency (3)	Criteria concerned with the language's consistency.	• Ensure element specifications are consistent within the language. • Ensure the various elements within a language are consistent.
Clarity (4)	Criteria that increase the language clarity and produce diagrams clear to the users.	• Ensure semantics assigned to the syntax is context independent. • Define language that avoids making mistakes.
Language features (22)	Criteria about the features and functionality a modeling language should provide to its users.	• Define language that enables representing technical and nontechnical activities. • Offer composition and decomposition in hierarchical fashion.

Table 7. Task Processing Criteria Categories

Category	Description	Example Criteria
Diagram-user fit (3)	Criteria concerning the amount of cognitive effort the user needs to effectively solve a problem.	• Ensure a diagram can reduce the user's cognitive effort needed to solve a problem. • Ensure cognitive fit.
Diagram-goal fit (11)	Criteria that ensure that the diagram helps users to achieve their goals.	• Ensure a diagram is specific enough to satisfy all user needs. • Define the appropriate scope and focus of diagram in relation to tasks.
Language appropriateness (5)	Criteria regarding the appropriateness of the language to its potential uses.	• Ensure a language is generally applicable. • Ensure a language is usable.

Task Processing Criteria

Task processing ties the mental models of the diagram and the task together to infer a solution. The three categories holding 19 criteria that relate to this cognitive step are shown in Table 7. Criteria in the category *diagram–goal fit* emphasize that performing a task requires all statements in a diagram to be correct and relevant to it (Levitin and Redman 1995; Lindland et al. 1994). If so, it serves the information needs of the reader. Diagram completeness is achieved only when the diagram contains all possible statements about the problem domain (Parsons and Wand 2013; Wand and Wang 1996). It is also important to ensure that the diagram will help users to achieve their goals (Simsion and Witt 2004). Whether a diagram satisfies user needs depends upon different criteria. There are criteria of *language appropriateness* that are imposed upon the diagram by the corresponding modeling language. For instance, a language that is not restricted to a specific domain might be preferable (McGinnes and Kapros 2015; Tsironis et al. 2009). Similarly, *diagram–user fit* can be achieved when a language exploits the output medium capabilities and the human visual channel to reduce the problem-solving effort (Britton et al. 2000).

Discussion

The systematic literature review yielded 220 criteria for effective cognitive processing of diagrams. The criteria were clustered into the four cognitive processing steps of our CogniDia framework. Many criteria were found for visual and semantic processing. Several criteria are discussed in the literature on task processing and only few criteria have been formulated for verbal processing.

It is important to emphasize that criteria do not represent independent design dimensions. Some criteria are explicitly conflicting. For instance, *tp10*[7] advocates to achieve diagram completeness, while several other criteria call for minimizing the diagram size (*vp46*) and complexity (*vp60, vp61, sp1, sp2*). Also *vp62* "Minimize syntactic sugar" and *vp3* "Use secondary notation," as well as *vp64* "Minimize symbol variety" and *vp85* "Maximize perceptual discriminability" directly conflict. Much of the discussion of trade-offs in the literature is on the conceptual level. Blackwell et al. (2001) state that it "is important to bear in mind that … these trade-off relationships do exist" (p. 104). Also Moody (2009) observes that "knowledge of interactions can be used to make trade-offs" (p. 772).

Empirical evidence on trade-offs is scarce so far. Eichelberger and Schmid (2009) test different aesthetic guidelines for UML class diagrams, but do not find significant differences. Mendling (2008) builds a logistic regression for explaining control flow errors in process models. High complexity measured by connectivity is the strongest coefficient (β = +4.008) contributing to error probability while structuredness is an even stronger coefficient (β = −9.957) reducing error probability. An experiment following up on this work yielded mixed results (Dumas et al. 2012). Another experiment by Mendling et al. (2012) finds that richer text labels of business process diagrams negatively affects understanding of control flow. We subscribe to an earlier statement by Blackwell et al. (2001) that "far more analysis of trade-off relationships needs to be done" (p. 104).

Another research challenge is the difference in level of granularity of the criteria. Several criteria are so general that they need to be operationalized by more specific criteria. These general criteria resemble principles "that can be boiled down to modeling guidelines" (Lankhorst 2013, p. 124). For example, a pleasing visual aesthetics of diagrams (*vp1*) (Purchase, Carrington, and Allder 2002) can be tied to criteria on how edges and nodes should be represented (*vp8–vp16, vp28–vp33*). Similarly, in order to maximize the visual structuredness of a diagram (*vp38*), criteria on where each element should be placed (*vp49–vp57*) need to be considered. Furthermore, criteria that suggest to minimize syntax complexity (*sp62*), avoid inefficient language elements (*sp63*) and ensure minimality (*sp64*) would also avoid hard mental operations (*sp60*). And clearly, for a modeling language to be ontologically complete and clear (*sp50, sp54*), it should not exhibit cases of construct deficit, overload, redundancy and excess (*sp53, sp55, sp56, sp57*). While criteria on diagrams can often be objectively measured, matters like ontological deficiencies require the judgment of experts. Despite the different levels of granularity and potential trade-offs, the criteria are useful for the analysis of diagrams. In the following, we discuss how SAD tasks can be analyzed using CogniDia criteria.

Studying Systems Analysis and Design Tasks with CogniDia

The aim of this section is to demonstrate how SAD tasks can be studied using CogniDia. To this end, we selected three examples from different stages of the SAD process: the first one anchored in requirements elicitation, the second in software architecture design specification, and the third one on system implementation. Focusing on different stages of the SAD process, which are supported with different types of diagrams, offers insights into the applicability of CogniDia in

[7]The IDs correspond to the criteria presented in Appendix D.

various settings. We discuss each of these three application examples using a common scheme including (1) task goal, (2) task description, (3) information needs, (4) desired outcomes, (5) task decomposition, (6) available diagram, and (7) CogniDia criteria. All examples are based on empirical studies on diagrams.

Requirements Elicitation Task Supported by Ontology Diagram

The task *elicit requirements* is typically performed at the early stage of the SAD process. The *goal* of this task is to explicate and document all relevant requirements based on the anticipated usage situation of the to-be-developed information system. The task can be *described* as the practice for discovering requirements from stakeholders, the application domain and the organizational environment (Cox et al. 2009). The elicitation task defines various *information needs* that the designer has to address. Nuseibeh and Easterbrook (2000) summarize them as the needs to understand the boundaries of the system, identify its stakeholders with their goals, tasks and use cases. Its *desired outcomes* are documents describing the organization which the system is meant to operate, the data it will process, the behavior it will support, the domain it will operationalize, and its nonfunctional properties (Hess et al. 2019; Nuseibeh and Easterbrook 2000).

The *elicit requirements* task is complex as it is characterized by unknown alternatives and uncertain consequences of action, inexact means–ends relationships, and numerous subtasks (Campbell 1988; Simon and March 1958). Table 1 shows various subtasks that are described in the literature, such as *identify client's vision*, *define problem*, *identify requirements*, and *document requirements*. The designer faces the challenge of breaking down the overall goal of this task into a hierarchy of subgoals, which can be addressed by separate tasks at a lower level.

Let us assume that requirements have to be elicited for a travel booking system. This task can be broken down into subgoals for identifying various stakeholders, information objects and functions that the system is meant to support. Let us also assume that elicitation has so far yielded the *available diagram* shown in Figure 4, which is taken from an experiment by Bera et al. (2011). Before interviewing the next stakeholder, one of the tasks is to identify questions that can be asked about the rules for cancellation of a customer's reservation. Mind that Bera et al. designed the diagram intentionally to have ontological deficiencies.

We observe that the diagram provides reasonable support for visual and verbal processing. Processing the *visual* elements of the ontology diagram could be improved by adding secondary notation, beyond the official semantics of the language *(vp3)*, such that important information would be emphasized. Regarding *verbal processing*, the text labels consistently use upper camel case. Furthermore, the diagram entails high semantic processing and task processing effort and violates several *CogniDia criteria*. Regarding *semantic processing*, the diagram has ontological deficiencies in terms of missing prefixes, missing interacting classes and missing connections, which makes it difficult to mentally reconstruct the domain and to formulate questions. Most strikingly, the class *Travel Agent* is omitted and merely shown as a property of the *InitialItinerary* and *FinalItinery* classes (Bera et al. 2011). This violates criterion *sp50* on ontological completeness. Diagram readers of ontologically incomplete diagrams are unable to interpret all real-world phenomena that might interest them (Wand and Weber 1993). As a result, the diagram is not ontologically expressive (*sp51*), because it does not capture the main aspects of reality. By omitting the travel agent class, the criterion *sp94* "Ensure class inclusion" is violated, which is required for the correct semantic processing of the diagram. Due to this issue, important interactions between this class and other classes are not shown, which affects the overall ontological quality of the diagram (*sp49*).

Regarding *task processing*, we observe that these issues affect also diagram–goal fit and criterion *tp10* on completeness in relation to the task at hand. This means that information needs of the task are not fully served. In particular, this makes it difficult to perform tasks that require the properties of the class *TravelAgent*. Prior domain and modeling knowledge also have an effect on the task performance with the ontology diagram. A person with prior knowledge of the travel domain might use this knowledge as a compensation for the diagram deficiencies (Pretz et al. 2003). Likewise, a person with knowledge of data modeling concepts (entities, classes, properties) may find the task easier than a person without.

System Specification Task Supported by UML Diagram

The task *specify system* is typically performed when transitioning from requirements specification to design specification in the SAD process. The *goal* of this task is to specify a high-level architecture of the system. The task can be *described* as the practice of identifying a design of the system that meets the requirements (Firesmith and Henderson-Sellers 2002) and documenting it (Capilla et al. 2016). The specification task defines various *information needs* that the designer has to take into account. Designers need to understand the architectural compliance with requirements, the system context and associated forces and implications of design decisions (Kruchten et al. 2005). Its *desired outcome* is a docu-

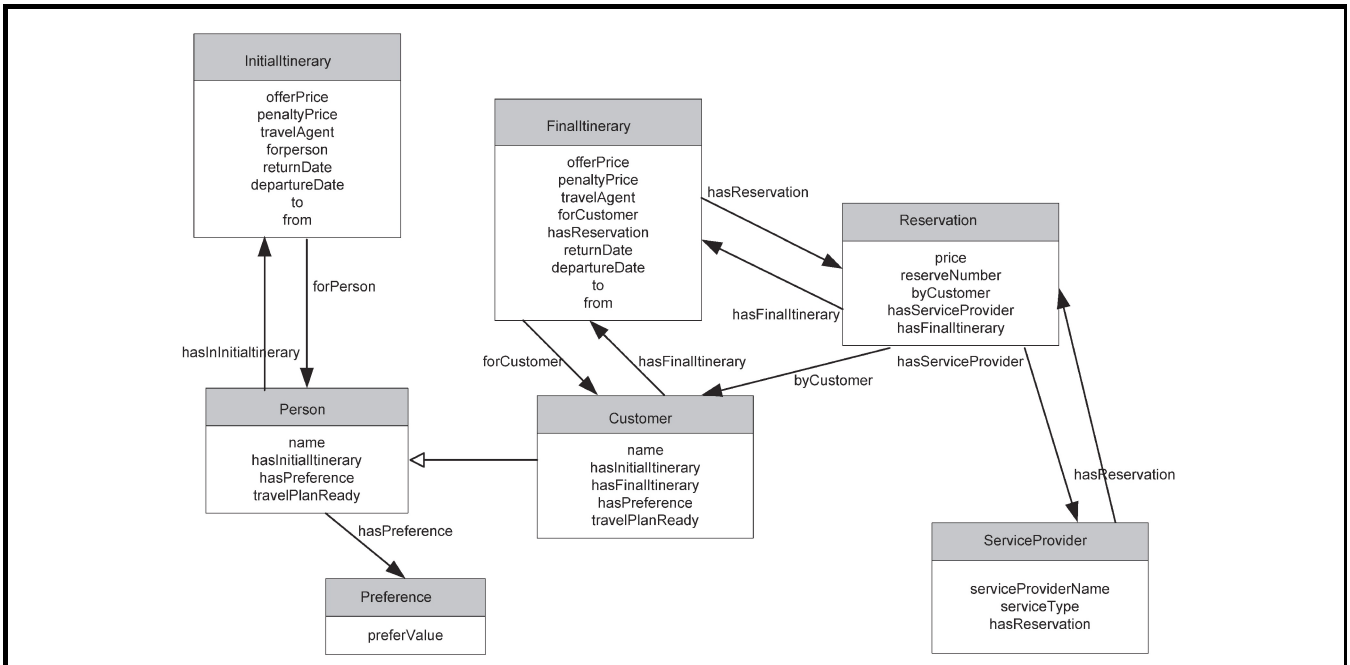

Figure 4. Ontology Diagram of Travel Booking (from P. Bera, A. Burton-Jones, and Y. Wand, "Guidelines for Designing Visual Ontologies to Support Knowledge Identification," *MIS Quarterly* (35:4), © 2011. Used with permission.)

mentation of design elements and their connections, along with an explication of architectural knowledge including design intentions and their underlying rationale (e.g., with reference to patterns) (Capilla et al. 2016).

The *specify system* task is complex as it is characterized by unknown alternatives and uncertain consequences of action. Capilla et al. (2016) emphasize its connection with the definition by Rittel and Webber (1973) of wicked problems. Table 1 shows various subtasks that are described in the literature including, *Understand system context*, *Design architecture*, *Document architecture*, and *Manage subsystems*. The designer faces the challenge of breaking down the overall goal of this task into a hierarchy of subgoals, which can be addressed by separate tasks at a lower level.

Let us assume that a system architecture has to be designed for a financial application system. This task can be broken down into subgoals for choosing architectural patterns, determining major components, and understanding their interactions. Let us also assume that the architecture design has so far yielded the *available diagram* shown in Figure 5, which is taken from an experiment by Heijstek et al. (2011). The designer is now facing the challenge to understand the interactions of this first version of the design. In this context, Heijstek et al. consider questions like "*Is system x the only component that may modify attribute y?*" and "*Through what node does system x connect to system y?*"

We observe that the diagram offers reasonable support for CogniDia criteria of visual, verbal and semantic processing. *Visual processing* could be slightly improved by avoiding the bends of the interaction connections (*vp30*). Also *verbal processing* has a minor inconsistency (*ap9*) of camel case labels (e.g., ClientData) and spaced labels (e.g., Mortgage Action). Overall, the diagram provides clarity for *semantic processing* (*sp29*). Heijstek et al. focus in their experiment on task processing and the fact that not all questions relevant for architecture design can be answered based on the diagram. A challenge is language appropriateness and the identification of language uses (*tp19*). Capilla et al. find that UML is used for architecture design, but its gaps in terms of capturing architectural knowledge including patterns, intentions, and rationale are compensated by deviating from the language standard or by providing textual documentation. A consequence of these limitations of UML also implies weaknesses in terms of diagram–goal fit, and specifically achieving diagram completeness with respect to the problem domain (*tp10*).

Implement System Task Supported by BPMN Diagram

The SAD task *implement system* is the final task of the system analysis and design process. Its *ultimate goal* is to have a running information system available that meets both the re-

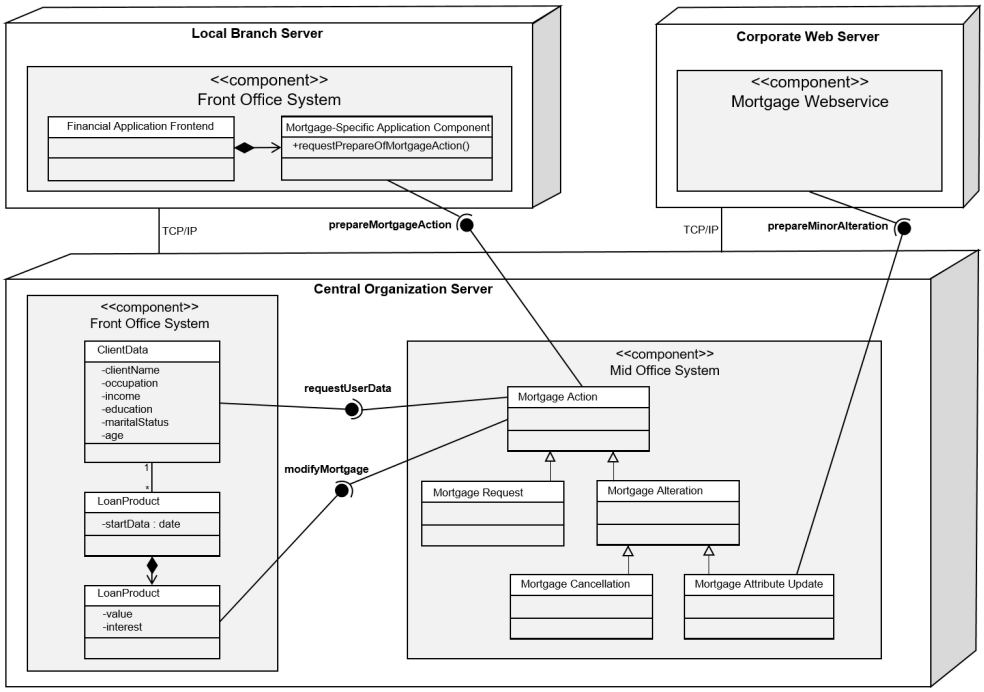

Figure 5. UML Diagram of a Financial System Architecture (© 2011 IEEE. Reprinted, with permission, from W. Heijstek, T. Kühne, and M. R. Chaudron, "Experimental Analysis of Textual and Graphical Representations for Software Architecture Design," in *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement*)

quirements and the design specification. The task can be *described* as the practice of developing working software (Bourque and Fairley 2014). The *implement system* task defines various information needs. Sillito et al. (2008) identified an extensive list of 44 question types that designers ask while implementing, many of them related to the context of the current focus point of work. Several of these questions are concerned with the behavior of system entities and how it varies across cases. The *desired outcome* of implementation is a correctly operating system.

The *iplement sstem* task is a complex task due to the richness of contextual dependencies that a designer has to understand (Sillito et al. 2008). Table 1 shows various subtasks that are described in the literature. In addition to the implementation planning and coding, a major subtask is to ensure the software quality, such as *evaluate implementation quality*, *improve software components*, *test software*, and *maintain software*. One of the aims of testing the software is making sure the system's behavior is correct.

Let us assume a system is implemented for processing customer complaints. Let us also assume that the requirements specification includes the *available* BPMN diagram shown in Figure 6, which is taken from Dumas et al. (2018) and used in an experiment by Petrusel et al. (2017). Among others, one of the checks the software tester needs to do is to ensure that the activities "Telephone Confirmation to external party (G)" and "Incident Agenda (J)" are mutually exclusive.

We observe that the task to test the correct behavior of the system is impeded by a high effort for visual processing and verbal processing and violations of various related *CogniDia criteria*. The diagram violates multiple *visual processing* criteria. First, the criterion *vp47* "Maximize visual structuredness" states that humans need less effort to understand a diagram if it is clearly structured (e.g., Britton et al. 2000). A business process diagram is structured if every split connector matches the respective join connector of the same type (Mendling, Reijers, and van der Aalst 2010). We can observe that the diagram shown in Figure 6 is not structured, because the XOR-join gateway before the activity "Archiving system" in the middle of the diagram does not have a corresponding XOR-split gateway. This also makes it more difficult to find the point where the two paths originate from. Second, the diagram exhibits two end events *case closed*, which violates the criterion *vp25* "Minimize start/end events". There is evidence that unstructured diagrams and diagrams which include more than one start and end events are more likely to include errors (Mendling et al. 2007). Third, the readability of dia-

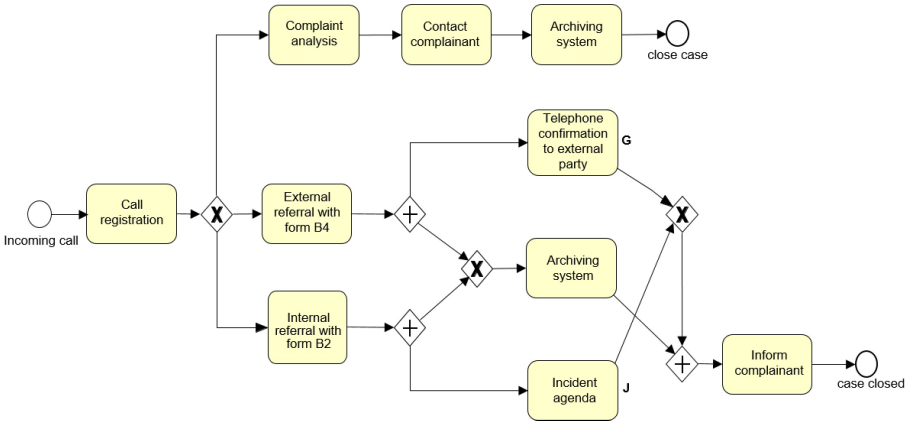

Figure 6. BPMN Process Diagram of a Service Process (Reprinted by permission from Springer, *Fundamentals of Business Process Management* (2nd ed.), M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, © 2018.)

grams is also compromised when elements are overlapping (Eichelberger and Schmid 2009). The diagram exhibits crossing arcs which violates the criterion *vp45* "Minimize element overlap."

The way how labels are formulated is important for the effective *verbal processing* of diagrams. From the process diagram we can observe that, first, some labels are much longer than others. de Oca et al. (2014) advises to use short activity labels, because they are easier to read and understand (*ap8*). Second, the style of the activity labels is not consistent (*ap9*). It has been shown that, especially for process diagrams, the verb–object style (*ap10*) (e.g., *Inform complainant*) is significantly less ambiguous and more useful than action–noun style (e.g., *Complaint analysis*) (Mendling, Reijers, and van der Aalst 2010).

Understanding the process diagram and performing the task using the diagram also entails a high *semantic processing* effort. The contents of the process diagram can be represented in a more compact way. If redundant elements are eliminated (e.g., activity *Archiving system*), the simplicity of the diagram can be improved (*sp1*). The combination of violated visual and verbal processing criteria also leads to a decreased diagram clarity (*sp29*), which is important for the diagram to be easily read (Becker et al. 2000; Schuette and Rotthowe 1998). Furthermore, the back tracking to find the common predecessor highlights the role of BPMN modeling knowledge for this task. This is also relevant for *task processing*: the task can be reformulated as "Find common predecessor and check if it is an XOR-gateway." Finally, this means that *task processing* requires a diagram–user fit with respect to prior knowledge of XOR-gateway semantics in BPMN.

Task Comparison

These three examples of task-diagram pairs clarify that tasks can put stress on different aspects of cognitive processing and that diagrams can have different deficiencies. First, we observe that the diagram supporting the requirements elicitation task shows weaknesses in terms of semantic processing. Second, the diagram supporting the architecture design task exhibits explicit weaknesses of addressing the information needs of task processing. Third, the diagram supporting implementation has deficiencies in its support of visual and verbal processing. All of these weaknesses can be systematically discussed using the criteria defined by CogniDia. This observation highlights the potential of using CogniDia to help clarify some of the inconsistent findings of prior research as reported in Mendling et al. (2019) and Ritchi et al. (2020). Next, we will discuss implications of our research.

Implications

In this section, we discuss implications for research and practice. We also highlight limitations. Overall, we found research to be fragmented, both on cognitive theories and on criteria in relation to CogniDia. The integrated theory of mind by Anderson et al. (2004) appeared to be the broadest cognitive theory, even though it does not discuss verbal processing explicitly. Hardly any paper discusses criteria across the several cognitive processing steps. Few papers take a broader perspective (e.g., Davis et al. 1993; Krogstie et al. 2006; Lindland et al. 1994; Paige et al. 2000). There is much emphasis on visual and semantic processing and not much on verbal and task processing. This fragmentation emphasizes the need to approach diagrams in IS research from a more

holistic and more pragmatic angle. Our proposal of task types and studying SAD tasks with the CogniDia framework is a first step. Decomposing tasks and linking their information needs to CogniDia provides a foundation for formulating empirical hypotheses. It is the merit of our CogniDia framework that it structures the field from a perspective of cognitive processing while other frameworks start from a foundation in semiotics (Krogstie et al. 2006; Lindland et al. 1994), ontology (Guizzardi 2005; Recker et al. 2011; Wand and Weber 1990), or a combination of these two (Nelson et al. 2012). With this approach, we integrate SAD tasks with diagram research.

Next, we highlight implications for developing future research on each cognitive processing step of the CogniDia framework. Table 8 gives a corresponding overview highlighting open research questions and potential theoretical pillars.

Implications for Visual Processing Research

The results of our review highlights that several aspects of visual processing are well established and intensively researched. However, there are several matters that we currently do not know including the degree to which diagrams can be misleading or demotivating.

The literature on diagrams in IS design appears to implicitly assume a rational usage by both designers and stakeholders. This assumption might be too strong to hold in practice, as emphasized by Browne and Parsons (2012). Bresciani and Eppler (2015) compile an extensive set of potential pitfalls of visual representations, which might in parts be highly relevant for diagram usage. We give a few examples. First, visual information can be *misleading* (Tufte 1985; Wainer 1984) and lead to incorrect conclusions (Bresciani and Eppler 2015). Some modeling tools, for instance, use the colors red and green for different types of symbols, even though these colors might trigger associations with traffic light color codes (**RQ1** and **RQ2** from Table 8).

Second, a visual representation might look *ugly* to some users (Nicolini 2007) and reduce the motivation to explore its content even if it is informative. Aesthetic preferences might lead to discarding a modeling language even though it might be highly effective for a specific application (**RQ3**). Third, frameworks like the physics of notations by Moody (2009) provide guidance for the effective usage of the visual variables identified by Bertin (1983); however, without an explicit discussion of potential biases. This matter is complicated, because the appropriate level and style of secondary notation depends also on contextual and cultural factors (**RQ4**).

The three examples point to the necessity to understand how diagrams might potentially distort specific design tasks. Research in this area might build on theories of dual information processing (Slovic et al. 2004). These theories assume that human cognition builds on an analytic system for rational thinking and an experiential system guided by affect and heuristics. The analytic system works slowly and requires high cognitive effort, the experiential system is fast and immediate (Kahneman 2011). Examples of IS research that establish a connection between cognitive biases and diagram characteristics is scarce. A notable example is Parsons and Saunders (2004) on anchoring and adjustment, which demonstrates that developers stick more to reusable code than it would be rationally justified.

A good summary on cognitive IS research on modeling is provided by Browne and Parsons (2012). Research on dual-process theories as summarized by Slovic et al. (2004) or Kahneman (2011) provide a rich repertoire for future research. There are also opportunities for utilizing eye-tracking, electroencephalography or other tools from neuroIS (Riedl and Léger 2016) as exemplified in recent studies by Petrusel et al. (2017) and Bera et al. (2019).

Implications for Verbal Processing Research

The results of our review point to concerns that are critical for effective verbal processing. Several important research questions are not yet answered regarding the amount of text, the usage of terms, and the integration of text and symbols.

The literature on diagrams in IS research largely focuses on their visual elements and often neglects textual content. Arguably, diagrams can hardly capture domain semantics without including text. Research on conceptual models has partially used the cognitive theory of multimedia learning by Mayer (2002) that emphasizes the equal importance of text and images. Among others, his continuity principle suggests to place text close to the corresponding image. However, this theory is less concerned with characteristics of text elements. An experiment by Mendling, Reijers, and Recker (2010) highlights ambiguity as a potential problem of text in diagrams. Ambiguity might be relevant at different levels. First, terms can be terminologically ambiguous or difficult. A diagram on cataract (clouding of the lens in the eye) will be more difficult to understand for many people than a diagram on online shopping. A diagram might also be confusing when it uses the term application both as software and as a treatment of the skin (**RQ5, RQ6**). Second, diagrams make use of text at different levels of granularity. For example, entity–relationship diagrams use terms to label visual elements, while process

Table 8. Future Research Directions and Potential Theoretical Pillars

Cognitive Step	Direction	Open Research Questions	Theories
Visual processing	Affects, Heuristics, Biases	**RQ1** Upon which conditions are diagrams misleading? **RQ2** How can we design diagrams that are not misleading? **RQ3** Which characteristics of a diagram reduce the motivation of a person to read it? **RQ4** How can biases be avoided during the selection of visual variables and secondary notation elements?	Dual-process theories of cognition
Verbal processing	Positions of words	**RQ5** What makes a term in a diagram difficult or ambiguous? **RQ6** What are the consequences of term difficulty and ambiguity for diagrams? **RQ7** What is an appropriate level of granularity for text in diagrams? **RQ8** What is an appropriate amount of text in diagrams? **RQ9** To what extent can text compensate a lack of understanding of the modeling language?	Theories of reading
Semantic processing	Diagram cohesion	**RQ10** What are the building blocks of prior domain knowledge that are typically integrated into the mental model of a diagram? **RQ11** What are the consequences of this integration for understanding the benefits of diagrams? **RQ12** To which extent are characteristics of a domain ignored when the modeling language does not cover them? **RQ13** What are potentially negative effects for domain understanding? **RQ14** To which extent are formal ontologies consistent with corresponding mental models?	Theories of sense-making, Representation theory, Formal Ontology
Task processing	Process of task solving	**RQ15** What are factors that make tasks difficult? **RQ16** How can tasks be designed in such a way that they guide users to good results? **RQ17** In which way do certain diagram characteristics influence search and integration processes during task solving? **RQ18** How can task performance be systematically analyzed? **RQ19** How does a team make sense of a diagram in order to solve a task? **RQ20** Which characteristics of a diagram help to develop a shared understanding of the task? **RQ21** To which extent are solutions developed by the help of diagrams better or worse as compared to other means of support?	External cognition, Task taxonomies

diagrams often use verb phrases. Full sentences are hardly used in diagrams (**RQ7, RQ8, RQ9**).

Research on text in diagrams can build on theories of reading. Among others, these theories formulate the immediacy assumption: the reader tries to interpret each word of a text as it is encountered and relates it to its antecedent as soon as possible (Just and Carpenter 1980). This operation is typically triggered for a sequence of words. In a diagram, the reader would likely relate not only words to words, but also words to symbols. In this context, research emphasizes the benefits of using frequent words (Just and Carpenter 1980), because they offer fast access in memory. Such access can be hampered. Linguistics describes the problem of interpreting a word that has several meanings as homonymy. Homonyms like "application" are inherently ambiguous and require disambiguation during look-up. Homonymy and polysemy is a known problem of terms in diagrams (Pittke et al. 2015), though its consequences are less understood. Theories of reading also point to text coherence as an important factor of text understanding. How this notion can be applied to diagrams has not yet been discussed.

Implications for Semantic Processing Research

Our review summarizes a diverse set of factors that influence semantic processing, both at the level of the modeling language and the individual diagram. Still, there are questions in relation to prior knowledge and semantic processing that are not yet fully understood. The literature mostly studies diagrams with the implicit assumption that their content is self-contained and reconstructed in the brain of the reader. For this reason, prior knowledge is often only seen as a factor that improves speed and accuracy of semantically processing a diagram. Therefore, it is often a controlled factor in experiments (Mendling et al. 2019). This limiting view focuses on the share of the diagram's content that is covered by the mental model, but ignores in how far the mental model might be richer than the external representation. This observation raises important questions on the relationship between diagrams and both domain knowledge and language knowledge. First, prior domain knowledge enriches mental models during the semantic processing of a diagram. An entity relationship

diagram of cataract treatment has a different meaning to a business student than to an ophthalmologist (**RQ10, RQ11**).

Second, modeling languages offer predefined schemata for looking at a specific domain. These schemata help to spot certain domain phenomena quicker and to understand their interrelations more clearly. However, it is not clear to which extent this creates blindness to the phenomena that are not part of the schemata provided by the modeling language or induce learning processes such as conceptual change (see Kang et al. 2004). Presumably, a phenomenon looks different to a person that is trained in system dynamics or in entity–relationship diagrams. Glaser (1992) refers to this threat as *forcing* in the context of the grounded theory method, and recommends to approach phenomena without any predefined schemata (**RQ12, RQ13**).

Third, cognitive research and research on representation and ontology can mutually inform each other. Some work using cognitive theories for explaining how ontological deficiencies negatively affect understanding, acceptance and performance are summarized in Burton-Jones et al. (2017). In contrast, a cognitive perspective offers also the chance to investigate the consistency between mental models and formal ontologies. Indeed, here are chances for fundamental discoveries given the fact that formal models of reasoning such as propositional logic and Bayesian networks appear to be inconsistent with the way how humans reason about causality (**RQ14**) (Goldvarg and Johnson Laird 2001; Rutter 2007; Sloman and Lagnado 2015).

These examples stress the importance of prior knowledge for working with diagrams. Research in this area can build on theories of sense-making. Various sense-making theories have been described with some commonalities (Zhang and Soergel 2014). One of the premises of sense-making is that there is an inherent connection between how one looks at a situation and what sense one is able to construct of it (Dervin 1998). Sense-making builds on prior knowledge and corresponding schemata. When new information is acquired, the reader needs to actively construct a revised or entirely new knowledge structure. When new information does not fit available schemata, readers feel internal conflict. They can either filter the data or refine the structure. In case of a great mismatch between data and structure, readers may experience cognitive dissonance (Cooper and Carlsmith 2015; Zhang and Soergel 2014). The specific role of diagrams as input to sense-making processes can be approached from the perspective of structural knowledge (Jonassen and Wang 1992). Applications of corresponding cognitive principles have largely focused on notational alternatives (see Browne and Parsons 2012; Parsons and Wand 2008). The sense-making model of Russell et al. (1993), for instance, allows us to discuss benefits of a suitable modeling language, since the sense-making steps of searching, instantiating, and modifying a schema can potentially be bypassed. Zhang and Soergel (2014) also posit with reference to Piaget that readers might be less inclined to change conceptual schemata. In our context, this would imply that diagram readers might indeed force phenomena onto the schemata of the modeling language in order to avoid dissonance with their language knowledge. This could potentially explain that readers can sometimes still draw correct conclusions from diagrams with ontological deficiencies.

Implications for Task Processing

The results of our review show that several aspects of task processing are well understood. However, the respective criteria mostly formulate goals, and it is not yet well understood how these goals can be best achieved. The literature on diagrams in IS design seems to have treated task performance largely as a black box and from a static perspective. This means that a diagram and a task are thrown into that black box yielding a solution. Clearly, this is an abstraction.

The literature appears to implicitly assume that in the relationship between the task and diagram, it is the task that is given and the diagram is designed to support it. However, also tasks can be shaped in different ways. This has the following consequences. First, tasks can at least be differentiated in terms of their type, complexity and environment (Campbell 1988; Kelton et al. 2010). Second, tasks may differ in complexity and clarity for various reasons. In practice, many IS design tasks remain underspecified requiring the interpretation of the analyst. Vitalari (1985) found that much work of IS analysts is to shape and sharpen tasks based on working hypotheses (**RQ15, RQ16**). This implies that information needs of tasks are actively identified and decided upon by the designer; they are not external or predefined.

The visual characteristics of diagrams make certain search tasks easier (Larkin and Simon 1987). While most experimental studies on diagrams investigate the impact of some factor on performance, hardly any of them looks at the actual search and integration process (**RQ17, RQ18**). RQ18 could be partially answered by combining eye-tracking with process mining (Van Der Aalst 2016). Furthermore, task performance is mostly assumed to be a concern of a single person. However, one of the key features of a diagram is its characteristic of representing some externalized knowledge that can be shared by different task stakeholders. Indeed, much design work is a team effort that requires coordination (**RQ19, RQ20, RQ21**).

The examples stress the need to better understand tasks in IS design at the domain level and the impact of their characteristics on performance. Applications of cognitive fit theory have often studied ways of tailoring a representation to the needs of a task. But also tasks can be designed in more or less effective ways. Theories of cognitive task design (Hollnagel 2003) discuss this matter, among others with applications in safety-critical sociotechnical systems. The analysis perspective of cognitive task design can also inform the development of taxonomies of IS design tasks. Work by Nickerson et al. (2013) can serve as a conceptual foundation. Such work might further develop cognitive fit theory to a theory that informs the co-design of representations and tasks.

These examples highlight the benefits of investigating processes of diagram usage and task performance, both of individuals and teams. A perspective for discussing these matters is offered by cognitive research on external representations. As summarized by Zhang (1997), this stream of research distinguishes different types of external representations with different degrees of efficiency. According to this perspective, task performance is regarded as a product of the relationship between the diagram and the task at hand (Davern et al. 2012). The consequences of such external representations are not only related to intellectual performance, but also impact decision making strategies and, consequentially, action (Kleinmuntz and Schkade 1993). For this reason, research on external representations has been extended toward external cognition and connected with the concept of affordance (Zhang and Patel 2006). Based on these concepts, diagrams can be studied as a specific type of technology, potentially embedded in a digital tool, that facilitates processes of team cognition (Fiore and Wiltshire 2016). The affordance concept also offers a direction to investigate opportunities for nudging team members (i.e., to instruct their task performance into a good direction) (Weinmann et al. 2016).

Implications for Practice

Our research also offers insights into the fundamental question: *Should we use diagrams in practice?* The CogniDia framework provides several facets of an answer. First, SAD tasks are complex, which means that they require an external representation of information (Zhang and Norman 1994). The CogniDia framework emphasizes the benefits of diagrams for SAD tasks where external information is interactively used. Diagrams in contrast to text facilitate fast visual search and verbal information reduced to key terms, which both supports semantic processing and task decomposition. Second, a diagram can still be highly ineffective when major criteria are violated. Our consolidated list of criteria is a valuable resource that practitioners can use as guidelines. To that end, we reformulated the criterion categories of the four CogniDia cognitive processing steps as 22 practitioner guidelines in Table 9. For example, a practitioner should *ensure the diagram is aesthetically pleasing*, a guideline that is grounded in the visual processing category *aesthetics*. The guidelines shown in Table 9 can immediately be used by practitioners as guidance for creating cognitively effective diagrams and modeling languages.

Third, the focus on tasks highlights another potential. Information in diagrams with a formally defined modeling language is structured. This offers opportunities for automation of smaller tasks like syntax check, formal verification or interactive querying. In this way, not only information can be offloaded from working memory to diagrams, but also analysis tasks from humans to software tools.

Limitations

Our review also has limitations. First, different strategies are possible for reviewing the cognitive processing of diagrams of understanding and task performance. We highlight two examples: Figl (2017) reports on a review of the understanding of procedural visual business process models. Her focus is on categories of factors that influence understanding performance. Saghafi and Wand (2014) conduct a meta-analysis on the effect of ontological guidelines on conceptual model understanding. Their focus is on the effect sizes of treatments that relate to ontological guidelines. Beyond these examples, other strategies are possible to provide insights complementary to our review.

Second, both the search strategy and consolidation strategy of our research had to reflect what has been called the vocabulary problem by Furnas et al. (1987). For our search strategy, we had to cover a plethora of synonyms for key concepts, as for example modeling language, notation, technique, grammar, etc. The size of our initial set of 18,819 papers reflects the breadth of our search. For the consolidation of heterogeneous formulations of criteria we utilized coding strategies inspired from Strauss and Corbin (1998). Both researchers involved with this study continuously discussed the naming of criteria and options to integrate separate criteria into more abstract ones. We designed a template for naming the criteria in a consistent way. The result is difficult to judge from a perspective of correctness. We resorted to the informed agreement of both authors. We carefully documented our study in the appendix, such that other researchers can discuss it.

Third, our review can also be criticized from the perspective of its scope. There is more general research on diagrams, for example, in the graph drawing community (Battista et al.

Table 9. Twenty-two CogniDia Guidelines for Practitioners

Cognitive Step	Practitioner Guidelines	Cognitive Step	Practitioner Guidelines
Visual Processing	Ensure diagram is aesthetically pleasing	*Semantic Processing*	***Diagram Quality***
			Decompose diagram to minimize complexity
	Harmonize elements in diagram		Ensure diagram satisfies modeling language rules
	Ensure diagram is simple		
	Emphasize key elements		Ensure diagram correctly represents a real-world domain
	Group similar elements		
	Ensure different elements can be distinguished		Ensure readers can easily understand the diagram
	Use mnemonic symbols		
Verbal Processing	Use text to complement diagram elements		***Language Quality***
			Ensure language is appropriate for its users
	Use short and uniform element descriptions		Ensure language faithfully represents a real-world domain
Task Processing	Ensure diagram satisfies user expectations		Ensure language is simple
			Ensure language elements are considered
	Ensure diagram is correct and relevant about a problem domain		Ensure language produces reliable diagrams
	Anticipate diagram tasks before diagram creation		Ensure language includes all relevant features that leads to reliable diagrams

1998; Hu and Nöllenburg 2016). We note that major works from that community show up in our paper list even though we excluded generic diagram research from our search strategy (e.g., Purchase, Allder, and Carrington 2002; Tamassia et al. 1988). Our scoping is justified by the assumption that diagrams and SAD tasks have specific characteristics, and we find support for this assumption. An extensive set of criteria refers to the modeling languages that are associated with diagrams and their connection with cognitive processing. We believe that also types of tasks related to SAD will have domain-specific interactions with diagrams. However, this is not yet evidenced by the literature.

Conclusion

In this paper we discussed effective cognitive processing of diagrams. We provide the following contributions. First, we developed an overview of tasks in SAD and essential characteristics of a diagram. Second, we developed a cognitive framework of understanding and task performance with diagrams (CogniDia). This framework integrates insights from established cognitive theories and relates them to the specifics of diagrams. Third, we consolidated criteria for effective cognitive processing of diagrams. We find a plethora of criteria that relate to visual and semantic processing, but less on verbal and task processing. Fourth, we analyzed SAD tasks with diagrams utilizing the CogniDia framework.

Our discussion highlights research gaps and open research questions for each of the cognitive processing steps. We specifically point to the potential of building on dual process theories of cognition, theories of reading, sense-making, cognitive task design and external cognition in order to address these research questions. Future research in these areas will help us develop new theories on how diagrams and tasks are intertwined in systems analysis and design.

References

Altmann, E. M., and John, B. E. 1999. "Episodic Indexing: A Model of Memory for Attention Events," *Cognitive Science* (23:2), pp. 117-156.

Anda, B., Hansen, K., Gullesen, I., and Thorsen, H. K. 2006. "Experiences from Introducing UML-Based Development in a Large Safety-Critical Project," *Empirical Software Engineering* (11:4), pp. 555-581.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. 2004. "An Integrated Theory of the Mind," *Psychological Rview* (111:4), pp. 1036-1060.

Anderson, J. R., Matessa, M., and Lebiere, C. 1997. "Act-R: A Theory of Higher Level Cognition and its Relation to Visual Attention," *Human–Computer Interaction* (12:4), pp. 439-462.

Anderson, M., and McCartney, R. 2003. "Diagram Processing: Computing with Diagrams," *Artificial Intelligence* (145:1-2), pp. 181-226.

Arias, E., Eden, H., Fischer, G., Gorman, A., and Scharff, E. 2000. "Transcending the Individual Human Mind—Creating Shared Understanding through Collaborative Design," *ACM Transactions on Computer–Human Interaction* (7:1), pp. 84-113.

Armenise, P., Bandinelli, S., Ghezzi, C., and Morzenti, A. 1993. "A Survey and Assessment of Software Process Representation Formalisms," *International Journal of Software Engineering and Knowledge Engineering* (3:03), pp. 401-426.

Baddeley, A. 1992. "Working Memory," *Science* (255:5044), pp. 556-559.

Battista, G. D., Eades, P., Tamassia, R., and Tollis, I. G. 1998. *Graph Drawing: Algorithms for the Visualization of Graphs*, Hoboken, NJ: Prentice Hall.

Beach, L. R., and Mitchell, T. R. 1978. "A Contingency Model for the Selection of Decision Strategies," *Academy of Management Review* (3:3), pp. 439-449.

Becker, J., Rosemann, M., and von Uthmann, C. 2000. "Guidelines of Business Process Modeling," in *Business Process Management*, W. van der Aalst, J. Desel, and A. Oberweis (eds.), Berlin: Springer, pp. 30-49.

Bendraou, R., Jézéquel, J.-M., Gervais, M.-P., and Blanc, X. 2010. "A Comparison of Six UML-Based Languages for Software Process Modeling," *IEEE Transactions on Software Engineering* (36:5), pp. 662-675.

Benjamin, R. G. 2012. "Reconstructing Readability: Recent Developments and Recommendations in the Analysis of Text Difficulty," *Educational Psychology Review* (24:1), pp. 63-88.

Bera, P., Burton-Jones, A., and Wand, Y. 2011. "Guidelines for Designing Visual Ontologies to Support Knowledge Identification," *MIS Quarterly* (35:4), pp. 883-908.

Bera, P., Soffer, P., and Parsons, J. 2019. "Using Eye Tracking to Expose Cognitive Processes in Understanding Conceptual Models," *MIS Quarterly* (43:4), pp. 1105-1126.

Bertin, J. 1983. *Semiology of Graphics: Diagrams, Networks, Maps* (W. j. Berg, trans.), Madison, WI: The University of Wisconsin Press.

Bielkowicz, P., and Tun, T. T. 2001. "A Comparison and Evaluation of Data Requirement Specification Techniques in SSADM the Unified Process," in *Proceedings of the International Conference on Advanced Information Systems Engineering*, Berlin: Springer, pp. 46-59.

Blackwell, A. F., Whitley, K. N., Good, J., and Petre, M. 2001. "Cognitive Factors in Programming with Diagrams," *Artificial Intelligence Review* (15:1-2), pp. 95-114.

Bourque, P., and Fairley, R. E. 2014. *Guide to the Software Engineering Body of Knowledge (SWEBOK) (R)): Version 3.0*, Los Alamitos, CA: IEEE Computer Society Press.

Bresciani, S., and Eppler, M. J. 2015. "The Pitfalls of Visual Representations: A Review and Classification of Common Errors Made While Designing and Interpreting Visualizations," *Sage Open* (5:4), pp. 1-14.

Britton, C., and Jones, S. 1999. "The Untrained Eye: How Languages for Software Specification Support Understanding in Untrained Users," *Human–Computer Interaction* (14:1-2), pp. 191-244.

Britton, C., Jones, S., Kutar, M., Loomes, M., and Robinson, B. 2000. "Evaluating the Intelligibility of Diagrammatic Languages Used in the Specification of Software," in *Proceedings of the International Conference on Theory and Application of Diagrams*, M. Anderson, P. Cheng, and V. Haarslev (eds.), Berlin: Springer, pp. 376-391.

Browne, G. J., and Parsons, J. 2012. "More Enduring Questions in Cognitive IS Research," *Journal of the Association for Information Systems* (13:12), Article 2.

Bunse, C. 2006. "Using Patterns for the Refinement and Translation of UML Models: A Controlled Experiment," *Empirical Software Engineering* (11:2), pp. 227-267.

Burton-Jones, A., and Meso, P. N. 2006. "Conceptualizing Systems for Understanding: An Empirical Test of Decomposition Principles in Object-Oriented Analysis," *Information Systems Research* (17:1), pp. 38-60.

Burton-Jones, A., and Meso, P. N. 2008. "The Effects of Decomposition Quality and Multiple Forms of Information on Novices' Understanding of a Domain from a Conceptual Model," *Journal of the Association for Information Systems* (9:12), pp. 748-748.

Burton-Jones, A., Recker, J., Indulska, M., Green, P. F., and Weber, R. 2017. "Assessing Representation Theory with a Framework for Pursuing Success and Failure," *MIS Quarterly* (41:4), pp. 1307-1333.

Burton-Jones, A., Wand, Y., and Weber, R. 2009. "Guidelines for Empirical Evaluations of Conceptual Modeling Grammars," *Journal of the Association for Information Systems* (10:6).

Campbell, D. J. 1988. "Task Complexity: A Review and Analysis," *Academy of Management Review* (13:1), pp. 40-52.

Capilla, R., Jansen, A., Tang, A., Avgeriou, P., and Babar, M. A. 2016. "10 Years of Software Architecture Knowledge Management: Practice and Future," *Journal of Systems and Software* (116), pp. 191-205.

Card, S. K., Moran, T. P., and Newell, A. 1980. "The Keystroke-Level Model for User Performance Time with Interactive Systems," *Communications of the ACM* (23:7), pp. 396-410.

Chandler, P., and Sweller, J. 1991. "Cognitive Load Theory and the Format of Instruction," *Cognition and Instruction* (8:4), pp. 293-332.

Cherfi, S. S.-S., Akoka, J., and Comyn-Wattiau, I. 2002. "Conceptual Modeling Quality—from EER to UML: Schemas Evaluation," in *Conceptual Modeling—ER2002*, Berlin: Springer, pp. 414-428.

Cherfi, S. S.-S., Akoka, J., and Comyn-Wattiau, I. 2007. "Perceived Vs. Measured Quality of Conceptual Schemas: An Experimental Comparison," in *ER2007: Proceedings of the 26th International Conference on Conceptual Modeling*, Australian Computer Society, pp. 185-190.

Chi, M. T., Glaser, R., and Rees, E. 1981. *Expertise in Problem Solving*, Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Chidamber, S. R., and Kemerer, C. F. 1994. "A Metrics Suite for Object Oriented Design," *IEEE Transactions on Software Engineering* (20:6), pp. 476-493.

Clark, J. M., and Paivio, A. 1991. "Dual Coding Theory and Education," *Educational Psychology Review* (3:3), pp. 149-210.

Clayton, M. J., Warden, R. B., and Parker, T. W. 2002. "Virtual Construction of Architecture Using 3D CAD and Simulation," *Automation in Construction* (11:2), pp. 227-235.

Collins-Thompson, K. 2014. "Computational Assessment of Text Readability: A Survey of Current and Future Research," *International Journal of Applied Linguistics* (165:2), pp. 97-135.

Cooper, J., and Carlsmith, K. M. 2015. "Cognitive Dissonance," in *International Encyclopedia of the Social & Behavioral Sciences* (2nd ed.), Amsterdam: Elsevier, pp. 76-78.

Cox, K., Niazi, M., and Verner, J. 2009. "Empirical Study of Sommerville and Sawyer's Requirements Engineering Practices," *IET Software* (3:5), pp. 339-355.

Cross, N. 2011. *Design Thinking: Understanding How Designers Think and Work*, Oxford, UK: Berg.

Curtis, B., Kellner, M. I., and Over, J. 1992. "Process Modeling," *Communications of the ACM* (35:9), pp. 75-90.

Davern, M., Shaft, T., and Te'eni, D. 2012. "Cognition Matters: Enduring Questions in Cognitive IS Research," *Journal of the Association for Information Systems* (13:4), Article 1.

Davies, I., Green, P., Rosemann, M., Indulska, M., and Gallo, S. 2006. "How Do Practitioners Use Conceptual Modeling in Practice?," *Data & Knowledge Engineering* (58:3), pp. 358-380.

Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., Kincaid, G., Ledeboer, G., Reynolds, P., and Sitaram, P. 1993. "Identifying and Measuring Quality in a Software Requirements Specification," in *Proceedings of the 1st International Software Metrics Symposium*, Los Alamitos, CA: IEEE Computer Society Press, pp. 141-152.

de Oca, I. M.-M., Snoeck, M., and Casas-Cardoso, G. 2014. "A Look into Business Process Modeling Guidelines through the Lens of the Technology Acceptance Model," in *Proceedings of the 7th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling*, U. Frank, Ó. Pastor, P. Loucopoulos, and I. Petrounias (eds.), Berlin: Springer, pp. 73-86.

Dent, B. D., Torguson, J. S., and Hodler, T. W. 1999. *Cartography: Thematic Map Design*, New York: WCB McGraw-Hill.

Dervin, B. 1998. "Sense-Making Theory and Practice: An Overview of User Interests in Knowledge Seeking and Use," *Journal of Knowledge Management* (2:2), pp. 36-46.

Dijkman, R. M., La Rosa, M., and Reijers, H. A. 2012. "Managing Large Collections of Business Process Models—Current Techniques and Challenges," *Computers in Industry* (63:2), pp. 91-97.

Ding, C., and Mateti, P. 1990. "A Framework for the Automated Drawing of Data Structure Diagrams," *IEEE Transactions on Software Engineering* (16:5), pp. 543-557.

Dobing, B., and Parsons, J. 2006. "How UML Is Used," *Communications of the ACM* (49:5), pp. 109-113.

Dromey, G. R. 1995. "A Model for Software Product Quality," *IEEE Transactions on Software Engineering* (21:2), pp. 146-162.

Dumas, M., La Rosa, M., Mendling, J., Mäesalu, R., Reijers, H. A., and Semenenko, N. 2012. "Understanding Business Process Models: The Costs and Benefits of Structuredness," in *International Conference on Advanced Information Systems Engineering*, J. Ralyté, X. Franch, S. Brinkkemper, and S. Wrycza (eds.), Berlin: Springer, pp. 31-46.

Dumas, M., La Rosa, M., Mendling, J., and Reijers, H. A. 2018. *Fundamentals of Business Process Management* (2nd ed.), Berlin: Springer.

Dzidek, W. J., Arisholm, E., and Briand, L. C. 2008. "A Realistic Empirical Evaluation of the Costs and Benefits of UML in Software Maintenance," *IEEE Transactions on Software Engineering* (34:3), pp. 407-432.

Eichelberger, H., and Schmid, K. 2009. "Guidelines on the Aesthetic Quality of UML Class Diagrams," *Information and Software Technology* (51:12), pp. 1686-1698.

Fernández-Sáez, A. M., Chaudron, M. R., and Genero, M. 2018. "An Industrial Case Study on the Use of UML in Software Maintenance and its Perceived Benefits and Hurdles," *Empirical Software Engineering* (23:6), pp. 3281-3345.

Figl, K. 2017. "Comprehension of Procedural Visual Business Process Models," *Business & Information Systems Engineering* (59:1), pp. 41-67.

Figl, K., Derntl, M., Rodriguez, M. C., and Botturi, L. 2010. "Cognitive Effectiveness of Visual Instructional Design Languages," *Journal of Visual Languages & Computing* (21:6), pp. 359-373.

Figl, K., Mendling, J., and Strembeck, M. 2012. "The Influence of Notational Deficiencies on Process Model Comprehension," *Journal of the Association for Information Systems* (14:6), pp. 312-338.

Fiore, S. M., and Wiltshire, T. J. 2016. "Technology as Teammate: Examining the Role of External Cognition in Support of Team Cognitive Processes," *Frontiers in Psychology* (7), Article 1531.

Firesmith, D. G., and Henderson-Sellers, B. 2002. *The Open Process Framework: An Introduction*, London: Pearson Education.

Frederiks, P. J., and Van der Weide, T. P. 2006. "Information Modeling: The Process and the Required Competencies of its Participants," *Data & Knowledge Engineering* (58:1), pp. 4-20.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. 1987. "The Vocabulary Problem in Human‑System Communication," *Communications of the ACM* (30:11), pp. 964-971.

Gedenryd, H. 1998. "How Designers Work—Making Sense of Authentic Cognitive Activities," Cognitive Science, Lund University.

Gemino, A., and Wand, Y. 2004. "A Framework for Empirical Evaluation of Conceptual Modeling Techniques," *Requirements Engineering* (9:4), pp. 248-260.

Giraldo, F. D., Espana, S., Pastor, O., and Giraldo, W. J. 2018. "Considerations About Quality in Model-Driven Engineering," *Software Quality Journal* (26:2), pp. 685-750.

Glaser, B. G. 1992. *Basics of Grounded Theory Analysis*: *Emergence Vs. Forcing*, Mill Valley, CA: Sociology Press.

Goldvarg, E., and Johnson Laird, P. N. 2001. "Naive Causality: A Mental Model Theory of Causal Meaning and Reasoning," *Cognitive science* (25:4), pp. 565-610.

Graham, I., Henderson-Sellers, B., and Younessi, H. 1997. *The Open Process Specification*, Boston: Addison-Wesley Publishing Co.

Granger, C. W. 1981. "Some Properties of Time Series Data and Their Use in Econometric Model Specification," *Journal of Econometrics* (16:1), pp. 121-130.

Gray, J., and Rumpe, B. 2018. "Agile Model-Based System Development," *Software & Systems Modeling* (17), pp. 1053-1054.

Green, T. R. G. 1989. "Cognitive Dimensions of Notations," in *People and Computers V*, A. Sutcliffe and L. Macaulay (eds.), Cambridge, UK: Cambridge University Press, pp. 443-460.

Green, T. R. G., and Petre, M. 1996. "Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework," *Journal of Visual Languages & Computing* (7:2), pp. 131-174.

Guentert, M., Kunze, M., and Weske, M. 2012. "Evaluation Measures for Similarity Search Results in Process Model Repositories," in *Proceedings of the International Conference on Conceptual Modeling*, P. Atzeni, D. Cheung, and S. Ram (eds.), Berlin: Springer, pp. 214-227.

Guindon, R. 1990. "Designing the Design Process: Exploiting Opportunistic Thoughts," *Human–Computer Interaction* (5:2-3), pp. 305-344.

Guizzardi, G. 2005. *Ontological Foundations for Structural Conceptual Models*, CTIT Ph.D. Thesis Series, No. 05-74, Centre for Telematics and Information Technology, University of Twente.

Guizzardi, G., Wagner, G., Almeida, J. P. A., and Guizzardi, R. S. 2015. "Towards Ontological Foundations for Conceptual Modeling: The Unified Foundational Ontology (UFO) Story," *Applied Ontology* (10:3-4), pp. 259-271.

Günther, S. 2011. "Development of Internal Domain-Specific Languages: Design Principles and Design Patterns," in *Proceedings of the 18th Conference on Pattern Languages of Programs*, New York: ACM, pp. 1-25.

Gurr, C. A. 1999. "Effective Diagrammatic Communication: Syntactic, Semantic and Pragmatic Issues," *Journal of Visual Languages & Computing* (10:4), pp. 317-342.

Harel, D., and Rumpe, B. 2000. *Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff*, 2000 Technical Report, Jerusalem: Weizmann Science Press of Israel.

Heidari, F., and Loucopoulos, P. 2014. "Quality Evaluation Framework (QEF): Modeling and Evaluating Quality of Business Processes," *International Journal of Accounting Information Systems* (15:3), pp. 193-223.

Heijstek, W., Kühne, T., and Chaudron, M. R. 2011. "Experimental Analysis of Textual and Graphical Representations for Software Architecture Design," in *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement*, Los Alamitos, CA: IEEE Computer Society Press, pp. 167-176.

Heravizadeh, M., Mendling, J., and Rosemann, M. 2008. "Dimensions of Business Processes Quality (QoBP)," in *Proceedings of the International Conference on Business Process Management*, D. Ardagna, M. Mecella, and J. Yang (eds.), Berlin: Springer, pp. 80-91.

Hess, A., Diebold, P., and Seyff, N. 2019. "Understanding Information Needs of Agile Teams to Improve Requirements Communication," *Journal of Industrial Information Integration* (14), pp. 3-15.

Heymans, P., Schobbens, P.-Y., Trigaux, J.-C., Bontemps, Y., Matulevicius, R., and Classen, A. 2008. "Evaluating Formal Properties of Feature Diagram Languages," *IET Software* (2:3), pp. 281-302.

Hochpöchler, U., Schnotz, W., Rasch, T., Ullrich, M., Horz, H., McElvany, N., and Baumert, J. 2013. "Dynamics of Mental Model Construction from Text and Graphics," *European Journal of Psychology of Education* (28:4), pp. 1105-1126.

Hollnagel, E. 2003. "Prolegomenon to Cognitive Task Design," in *Handbook of Cognitive Task Design*, E. Hollnagel (ed.), Mahwah, NJ: Erlbaum, pp. 3-15.

Hu, Y., and Nöllenburg, M. (eds.). 2016. *Graph Drawing and Network Visualization: 24th International Symposium on Graph Drawing and Network Visualization,* Athens, Greece, September 19-21, 2016.

Huang, S.-M., Shieh, K.-K., and Chi, C.-F. 2002. "Factors Affecting the Design of Computer Icons," *International Journal of Industrial Ergonomics* (29:4), pp. 211-218.

Hutchinson, J., Whittle, J., and Rouncefield, M. 2014. "Model-Driven Engineering Practices in Industry: Social, Organizational and Managerial Factors That Lead to Success or Failure," *Science of Computer Programming* (89), pp. 144-161.

John, B. E., and Kieras, D. E. 1996a. "The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast," *ACM Transactions on Computer–Human Interaction* (3:4), pp. 320-351.

John, B. E., and Kieras, D. E. 1996b. "Using GOMS for User Interface Design and Evaluation: Which Technique?," *ACM Transactions on Computer–Human Interaction* (3:4), pp. 287-319.

Jonassen, D. H., and Wang, S. 1992. "Acquiring Structural Knowledge from Semantically Structured Hypertext," *Journal of Computer-Based Interaction* (20), pp. 1-8.

Just, M. A., and Carpenter, P. A. 1980. "A Theory of Reading: From Eye Fixations to Comprehension," *Psychological Review* (87:4), pp. 329-354.

Kahneman, D. 2011. *Thinking, Fast and Slow*, New York: Macmillan.

Kang, S., Scharmann, L. C., and Noh, T. 2004. "Reexamining the Role of Cognitive Conflict in Science Concept Learning," *Research in Science Education* (34:1), pp. 71-96.

Karsai, G., Krahn, H., Pinkernell, C., Rumpe, B., Schindler, M., and Völkel, S. 2014. "Design Guidelines for DomainSpecific Languages," in *Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling*, M. Rossi, J. Sprinkle, J. Gray, and J. P. Tolvanen (eds.), Helsinki School of Economics, pp. 7-13.

Kelton, A. S., Pennington, R. R., and Tuttle, B. M. 2010. "The Effects of Information Presentation Format on Judgment and Decision Making: A Review of the Information Systems Research," *Journal of Information Systems* (24:2), pp. 79-105.

Kendeou, P., McMaster, K. L., and Christ, T. J. 2016. "Reading Comprehension: Core Components and Processes," *Policy Insights from the Behavioral and Brain Sciences* (3:1), pp. 62-69.

Kesh, S. 1995. "Evaluating the Quality of Entity Relationship Models," *Information and Software Technology* (37:12), pp. 681-689.

Khatri, V., Vessey, I., Ramesh, V., Clay, P., and Park, S.-J. 2006. "Understanding Conceptual Schemas: Exploring the Role of Application and Is Domain Knowledge," *Information Systems Research* (17:1), pp. 81-99.

Kintsch, W. 1988. "The Role of Knowledge in Discourse Comprehension: A Construction-Integration Model," *Psychological Review* (95:2), pp. 163-182.

Kintsch, W., and Van Dijk, T. A. 1978. "Toward a Model of Text Comprehension and Production," *Psychological Review* (85:5), pp. 363-394.

Kiper, J. D., Howard, E., and Ames, C. 1997. "Criteria for Evaluation of Visual Programming Languages," *Journal of Visual Languages & Computing* (8:2), pp. 175-192.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. 2009. "Systematic Literature Reviews in Software Engineering—A Systematic Literature Review," *Information and Software Technology* (51:1), pp. 7-15.

Kleinmuntz, D. N., and Schkade, D. A. 1993. "Information Displays and Decision Processes," *Psychological Science* (4:4), pp. 221-227.

Kolovos, D. S., Paige, R. F., Kelly, T., and Polack, F. A. 2006. "Requirements for Domain-Specific Languages," in *Proceedings of the ECOOP Workshop on Domain-Specific Program Development*, Nantes, France.

Krogstie, J. 1998. "Integrating the Understanding of Quality in Requirements Specification and Conceptual Modeling," *ACM SIGSOFT Software Engineering Notes* (23:1), pp. 86-91.

Krogstie, J. 2003. "Evaluating UML Using a Generic Quality Framework," Chapter 1 in *UML and the Unified Process*, L. Favre (ed.), Hershey, PA: IRM Press, pp. 1-22.

Krogstie, J., Lindland, O. I., and Sindre, G. 1995a. "Defining Quality Aspects for Conceptual Models," in *Proceedings of the IFIP WG 8.1 Working Conference on Information System Concepts*, U. Frank, Ó. Pastor, P. Loucopoulos, and I. Petrounias (eds.), Berlin: Springer, pp. 216-231.

Krogstie, J., Lindland, O. I., and Sindre, G. 1995b. "Towards a Deeper Understanding of Quality in Requirements Engineering," *International Conference on Advanced Information Systems Engineering*: Springer, pp. 82-95.

Krogstie, J., Sindre, G., and Jorgensen, H. 2006. "Process Models Representing Knowledge for Action: A Revised Quality Framework," *European Journal of Information Systems* (15:1), pp. 91-102.

Krogstie, J., and Sølvberg, A. 2003. *Information Systems Engineering: Conceptual Modeling in a Quality Perspective*, Trondheim, Norway: Kompendiumforlaget.

Kruchten, P., Lago, P., Van Vliet, H., and Wolf, T. 2005. "Building Up and Exploiting Architectural Knowledge," in *Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture*, R. Nord, N. Medvidovic, R. Krikhaar, J. Stafford, and J. Bosch (eds.), Washington, DC: IEEE Computer Society, pp. 291-292.

Kühne, T. 2006. "Matters of (Meta-)Modeling," *Software & Systems Modeling* (5:4), pp. 369-385.

Kyriakou, H., Nickerson, J. V., and Sabnis, G. 2017. "Knowledge Reuse for Customization: Metamodels in an Open Design Community for 3D Printing," *MIS Quarterly* (41:1), pp. 315-332.

Lange, C. F., and Chaudron, M. R. 2005. "Managing Model Quality in UML-Based Software Development," in *Proceedings of the 13th IEEE International Workshop on Software Technology and Engineering Practice*, Washington, DC: IEEE Computer Society, pp. 7-16.

Lange, M., Mendling, J., and Recker, J. 2016. "An Empirical Analysis of the Factors and Measures of Enterprise Architecture Management Success," *European Journal of Information Systems* (25:5), pp. 411-431.

Lankhorst, M. (ed.). 2013. *Enterprise Architecture at Work. Modelling, Communication, and Analysis* (4th ed.), Berlin Springer.

Larkin, J. H., and Simon, H. A. 1987. "Why a Diagram Is (Sometimes) Worth Ten Thousand Words," *Cognitive Science* (11:1), pp. 65-100.

Leopold, H. 2013. *Natural Language in Business Process Models*, Berlin: Springer.

Levitin, A., and Redman, T. 1995. "Quality Dimensions of a Conceptual View," *Information Processing & Management* (31:1), pp. 81-88.

Lindland, O. I., Sindre, G., and Solvberg, A. 1994. "Understanding Quality in Conceptual Modeling," *IEEE Software* (11:2), pp. 42-49.

Lloyd, K. B., and Jankowski, D. J. 1999. "A Cognitive Information Processing and Information Theory Approach to Diagram Clarity: A Synthesis and Experimental Investigation," *Journal of Systems and Software* (45:3), pp. 203-214.

Locke, E. A., and Latham, G. P. 1990. *A Theory of Goal Setting & Task Performance*, Hoboken, NJ: Prentice-Hall.

Maes, A., and Poels, G. 2007. "Evaluating Quality of Conceptual Modelling Scripts Based on User Perceptions," *Data & Knowledge Engineering* (63:3), pp. 701-724.

Mayer, R. E. 2002. "Multimedia Learning," in *Psychology of Learning and Motivation* (41), pp. 85-139.

McDougall, S. J. P., Curry, M. B., and de Bruijn, O. 1999. "Measuring Symbol and Icon Characteristics: Norms for Concreteness, Complexity, Meaningfulness, Familiarity, and Semantic Distance for 239 Symbols," *Behavior Research Methods, Instruments, & Computers* (31:3), pp. 487-519.

McDougall, S. J. P., de Bruijn, O., and Curry, M. B. 2000. "Exploring the Effects of Icon Characteristics on User Performance: The Role of Icon Concreteness, Complexity, and Distinctiveness," *Journal of Experimental Psychology: Applied* (6:4), pp. 291-291.

McGinnes, S., and Kapros, E. 2015. "Conceptual Independence: A Design Principle for the Construction of Adaptive Information Systems," *Information Systems* (47), pp. 33-50.

McNamara, D. S., Kintsch, E., Songer, N. B., and Kintsch, W. 1996. "Are Good Texts Always Better? Interactions of Text Coherence, Background Knowledge, and Levels of Understanding in Learning from Text," *Cognition and Instruction* (14:1), pp. 1-43.

Mendling, J. 2008. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*, Berlin: Springer.

Mendling, J., Neumann, G., and van der Aalst, W. 2007. "Understanding the Occurrence of Errors in Process Models Based on Metrics," in *On the Move to Meaningful Internet Systems: OTM 2019 Conferences*, H. Panetto, C. Debruyne, M. Preuß, D. Lewis, C. A. Ardagna, and R. Meersman (eds.), Berlin: Springer, pp. 113-130.

Mendling, J., Recker, J., Reijers, H. A., and Leopold, H. 2019. "An Empirical Review of the Connection between Model Viewer Characteristics and the Comprehension of Conceptual Process Models," *Information Systems Frontiers* (21:5), pp. 1111-1135.

Mendling, J., Reijers, H. A., and Recker, J. 2010. "Activity Labeling in Process Modeling: Empirical Insights and Recommendations," *Information Systems* (35:4), pp. 467-482.

Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. 2010. "Seven Process Modeling Guidelines (7PMG)," *Information and Software Technology* (52:2), pp. 127-136.

Mendling, J., Strembeck, M., and Recker, J. 2012. "Factors of Process Model Comprehension—Findings from a Series of Experiments," *Decision Support Systems* (53:1), pp. 195-206.

Merriam-Webster. 2015. *Merriam-Webster*.

Mohagheghi, P., Dehlen, V., and Neple, T. 2009. "Definitions and Approaches to Model Quality in Model-Based Software Development—A Review of Literature," *Information and Software Technology* (51:12), pp. 1646-1669.

Moody, D. L. 1998. "Metrics for Evaluating the Quality of Entity Relationship Models," in *Conceptual Modeling–ER'98*, T. W. Ling, S. Ram, and M. L. Lee (eds.), Berlin: Springer, pp. 211-225.

Moody, D. L. 2009. "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering," *Software Engineering, IEEE Transactions on* (35:6), pp. 756-779.

Moody, D. L., and Shanks, G. G. 1994. "What Makes a Good Data Model? Evaluating the Quality of Entity Relationship Models," in *Entity-Relationship Approach—ER'94 Business Modeling and Re-engineering,* P. Loucopoulos (ed.), Berlin: Springer, pp. 94-111..

Moody, D. L., and Shanks, G. G. 2003. "Improving the Quality of Data Models: Empirical Validation of a Quality Management Framework," *Information Systems* (28:6), pp. 619-650.

Moody, D. L., Shanks, G. G., and Darke, P. 1998. "Improving the Quality of Entity Relationship Models—Experience in Research and Practice," in *Conceptual Modeling—ER'98*, T. W. Ling, S. Ram, and M. L. Lee (eds.), Berlin: Springer, pp. 255-276.

Nelson, H. J., Poels, G., Genero, M., and Piattini, M. 2012. "A Conceptual Modeling Quality Framework," *Software Quality Journal* (20:1), pp. 201-228.

Newell, A., and Simon, H. A. 1972. *Human Problem Solving*, Englewood Cliffs, NJ: Prentice-Hall.

Nickerson, R. C., Varshney, U., and Muntermann, J. 2013. "A Method for Taxonomy Development and its Application in Information Systems," *European Journal of Information Systems* (22:3), pp. 336-359.

Nicolini, D. 2007. "Studying Visual Practices in Construction," *Building Research & Information* (35:5), pp. 576-580.

Nuseibeh, B., and Easterbrook, S. 2000. "Requirements Engineering: A Roadmap," in *Proceedings of the Conference on the Future of Software Engineering*, New York: ACM, pp. 35-46.

Overhage, S., Birkmeier, D. Q., and Schlauderer, S. 2012. "Quality Marks, Metrics, and Measurement Procedures for Business Process Models," *Business & Information Systems Engineering* (4:5), pp. 229-246.

Paige, R. F., Ostroff, J. S., and Brooke, P. J. 2000. "Principles for Modeling Language Design," *Information and Software Technology* (42:10), pp. 665-675.

Paivio, A. 1991. "Dual Coding Theory: Retrospect and Current Status," *Canadian Journal of Psychology* (45:3), pp. 255-287.

Parsons, J. 1996. "An Information Model Based on Classification Theory," *Management Science* (42:10), pp. 1437-1453.

Parsons, J., and Saunders, C. 2004. "Cognitive Heuristics in Software Engineering Applying and Extending Anchoring and Adjustment to Artifact Reuse," *IEEE Transactions on Software Engineering* (30:12), pp. 873-888.

Parsons, J., and Wand, Y. 1997. "Choosing Classes in Conceptual Modeling," *Communications of the ACM* (40:6), pp. 63-69.

Parsons, J., and Wand, Y. 2008. "Using Cognitive Principles to Guide Classification in Information Systems Modeling," *MIS Quarterly* (32:4), pp. 839-868.

Parsons, J., and Wand, Y. 2013. "Extending Classification Principles from Information Modeling to Other Disciplines," *Journal of the Association for Information Systems* (14:5), pp. 245-245.

Pearson, P. D., and Cervetti, G. N. 2015. "Fifty Years of Reading Comprehension Theory and Practice," *Research-Based Practices for Teaching Common Core Literacy*, P. D. Pearson and E. H. Hiebert (eds.), New York: Teacher College Press, pp. 1-24.

Petre, M. 1995. "Why Looking Isn't Always Seeing: Readership Skills and Graphical Programming," *Communications of the ACM* (38:6), pp. 33-44.

Petrusel, R., Mendling, J., and Reijers, H. A. 2017. "How Visual Cognition Influences Process Model Comprehension," *Decision Support Systems* (96), pp. 1-16.

Pittke, F., Leopold, H., and Mendling, J. 2015. "Automatic Detection and Resolution of Lexical Ambiguity in Process Models," *IEEE Transactions on Software Engineering* (41:6), pp. 526-544.

Popper, K. R., and Eccles, J. C. 1977. "The Three Worlds 1, 2 and 3," in *Th Self and its Brain*, K. Popper (ed.), Berlin: Springer, pp. 36-50.

Pretz, J. E., Naples, A. J., and Sternberg, R. J. 2003. "Recognizing, Defining, and Representing Problems," *The Psychology of Problem Solving* (30:3), pp. 9-26.

Purchase, H. C., Allder, J.-A., and Carrington, D. 2000. "User Preference of Graph Layout Aesthetics: A UML Study," in *Proceedings of the 8th International Symposium on Graph Drawing*, J. Marks (ed.), Berlin: Springer, pp. 5-18.

Purchase, H. C., Allder, J.-A., and Carrington, D. A. 2002. "Graph Layout Aesthetics in UML Diagrams: User Preferences," *Journal of Graph Algorithms and Applications* (6:3), pp. 255-279.

Purchase, H. C., Carrington, D., and Allder, J.-A. 2002. "Empirical Evaluation of Aesthetics-Based Graph Layout," *Empirical Software Engineering* (7:3), pp. 233-255.

Recker, J. 2011. *Evaluations of Process Modeling Grammars: Ontological, Qualitative and Quantitative Analyses Using the Example of BPMN*, Berlin: Springer.

Recker, J., and Green, P. 2019. "How Do Individuals Interpret Multiple Conceptual Models? A Theory of Combined Ontological Completeness and Overlap," *Journal of the Association for Information Systems* (20:8), pp. 1210-1241.

Recker, J., Lukyanenko, R., Jabbari, M., Samuel, B. M., and Castellanos, A. 2021. "From Representation to Mediation: A New Agenda for Conceptual Modeling Research in a Digital World," *MIS Quarterly* (45:1), pp. 269-300.

Recker, J., Rosemann, M., Green, P. F., and Indulska, M. 2011. "Do Ontological Deficiencies in Modeling Grammars Matter?," *MIS Quarterly* (35:1), pp. 57-79.

Recker, J., Rosemann, M., and Krogstie, J. 2007. "Ontology Versus Pattern-Based Evaluation of Process Modeling Languages: A Comparison," *Communications of the Association for Information Systems* (20:1), pp. 48-48.

Riedl, R., and Léger, P.-M. 2016. *Fundamentals of NeuroIS: Information Systems and the Brain*, Berlin: Springer.

Ritchi, H., Jans, M., Mendling, J., and Reijers, H. A. 2020. "The Influence of Business Process Representation on Performance of Different Task Types," *Journal of Information Systems* (34:1), pp. 167-194.

Rittel, H. W. J. 1972. "Second Generation Design Methods," interview in *Design Methods Group 5th Anniversary Report*, DMG Occasional Paper (1), pp. 5-10. Reprinted in N. Cross (ed.), *Developments in Design Methodology*, Chichester, UK: John Wiley & Sons, 1984, pp. 317-327.

Rittel, H. W., and Webber, M. M. 1973. "Dilemmas in a General Theory of Planning," *Policy Sciences* (4:2), pp. 155-169.

Robles, G., Ho-Quang, T., Hebig, R., Chaudron, M. R., and Fernandez, M. A. 2017. "An Extensive Dataset of UML Models in GitHub," in *Proceedings of the 2017 IEEE/ACM International Conference on Mining Software Repositories*, New York: IEEE, pp. 519-522.

Russell, D. M., Stefik, M. J., Pirolli, P., and Card, S. K. 1993. "The Cost Structure of Sensemaking," in *Proceedings of the INTERACT'93 and CHI'93 Conference on Human Factors in Computing Systems*, pp. 269-276.

Rutter, M. 2007. "Proceeding from Observed Correlation to Causal Inference: The Use of Natural Experiments," *Perspectives on Psychological Science* (2:4), pp. 377-395.

Sabegh, M. A. J., and Recker, J. 2017. "Combined Use of Conceptual Models in Practice: An Exploratory Study," *Journal of Database Management* (28:2), pp. 56-88.

Saghafi, A., and Wand, Y. 2014. "Do Ontological Guidelines Improve Understandability of Conceptual Models? A Meta-Analysis of Empirical Work," in *Proceedings of the 47th Hawaii International Conference on System Sciences*, New York: IEEE, pp. 4609-4618.

Sánchez-González, L., García, F., Ruiz, F., and Piattini, M. 2013. "Toward a Quality Framework for Business Process Models," *International Journal of Cooperative Information Systems* (22:01).

Scanniello, G., Gravino, C., Genero, M., Cruz-Lemus, J. A., Tortora, G., Risi, M., and Dodero, G. 2018. "Do Software Models Based on the UML Aid in Source-Code Comprehensibility? Aggregating Evidence from 12 Controlled Experiments," *Empirical Software Engineering* (23:5), pp. 2695-2733.

Schlauderer, S., and Overhage, S. 2018. "BOSDL: An Approach to Describe the Business Logic of Software Services in Domain-Specific Terms," *Business & Information Systems Engineering* (60:5), pp. 393-413.

Schnotz, W. 2005. "An Integrated Model of Text and Picture Comprehension," *The Cambridge Handbook of Multimedia Learning*, R. E. Mayer (ed.), New York: Cambridge University Press, pp. 49-69.

Schuette, R., and Rotthowe, T. 1998. "The Guidelines of Modeling—An Approach to Enhance the Quality in Information Models," in *Proceedings of the International Conference on Conceptual Modeling (ER'98)*, Berlin: Springer, pp. 240-254.

Shanks, G., Tansley, E., and Weber, R. 2003. "Using Ontology to Validate Conceptual Models," *Communications of the ACM* (46:10), pp. 85-89.

Siau, K., and Tan, X. 2005. "Improving the Quality of Conceptual Modeling Using Cognitive Mapping Techniques," *Data & Knowledge Engineering* (55:3), pp. 343-365.

Sillito, J., Murphy, G. C., and De Volder, K. 2008. "Asking and Answering Questions During a Programming Change Task," *IEEE Transactions on Software Engineering* (34:4), pp. 434-451.

Simon, H. A. 1978. "On the Forms of Mental Representation," *Perception and Cognition: Issues in the Foundations of Psychology* (9), pp. 3-18.

Simon, J. G., and March, H. A. 1958. *Organizations* (2nd ed.), Hoboken, NJ: Wiley.

Simsion, G., and Witt, G. 2004. *Data Modeling Essentials*, San Francisco: Morgan Kaufmann.

Sindiy, O., Litomisky, K., Davidoff, S., and Dekens, F. 2013. "Introduction to Information Visualization (InfoVis) Techniques for Model-Based Systems Engineering," *Procedia Computer Science* (16), pp. 49-58.

Sloman, S. A., and Lagnado, D. 2015. "Causality in Thought," *Annual Review of Psychology* (66), pp. 223-247.

Slovic, P., Finucane, M. L., Peters, E., and MacGregor, D. G. 2004. "Risk as Analysis and Risk as Feelings: Some Thoughts About Affect, Reason, Risk, and Rationality," *Risk Analysis* (24:2), pp. 311-322.

Sommerville, I. 2011. *Software Engineering* (9th ed.), London: Pearson Education

Stachowiak, H. 1973. *Allgemeine Modelltheorie*, Berlin: Springer-Verlag.

Stanton, N. A. 2006. "Hierarchical Task Analysis: Developments, Applications, and Extensions," *Applied Ergonomics* (37:1), pp. 55-79.

Staples, M. 2014. "Critical Rationalism and Engineering: Ontology," *Synthese* (191:10), pp. 2255-2279.

Stapleton, G., and Delaney, A. 2008. "Evaluating and Generalizing Constraint Diagrams," *Journal of Visual Languages & Computing* (19:4), pp. 499-521.

Strauss, A., and Corbin, J. 1998. *Basics of Qualitative Research: Procedures and Techniques for Developing Grounded Theory*, Thousand Oaks, CA: SAGE Publications.

Sweller, J. 1988. "Cognitive Load During Problem Solving: Effects on Learning," *Cognitive Science* (12:2), pp. 257-285.

Tamassia, R., Di Battista, G., and Batini, C. 1988. "Automatic Graph Drawing and Readability of Diagrams," *IEEE Transactions on Systems, Man, and Cybernetics* (18:1), pp. 61-79.

ter Hofstede, A. H. M., Proper, H. A., and Van Der Weide, T. P. 1993. "Formal Definition of a Conceptual Language for the Description and Manipulation of Information Models," *Information Systems* (18:7), pp. 489-523.

Trkman, M., Mendling, J., and Krisper, M. 2016. "Using Business Process Models to Better Understand the Dependencies among User Stories," *Information and Software Technology* (71), pp. 58-76.

Tsironis, L., Anastasiou, K., and Moustakis, V. 2009. "A Framework for BPML Assessment and Improvement: A Case Study Using IDEF0 and eEPC," *Business Process Management Journal* (15:3), pp. 430-461.

Tufte, E. R. 1985. "The Visual Display of Quantitative Information," *The Journal for Healthcare Quality* (7:3), p. 15.

van der Aalst, W. 2016. "Data Science in Action," Chapter 1 in *Process Mining: Data Science in Action* (2nd. ed.), , Berlin: Springer, pp. 3-23.

Vessey, I. 1991. "Cognitive Fit: A Theory-Based Analysis of the Graphs Versus Tables Literature," *Decision Sciences* (22:2), pp. 219-240.

Vitalari, N. P. 1985. "Knowledge as a Basis for Expertise in Systems Analysis: An Empirical Study," *MIS Quarterly* (9:3), pp. 221-241.

Wagner, S., Fernández, D. M., Felderer, M., Vetrò, A., Kalinowski, M., Wieringa, R. J., Pfahl, D., Conte, T. U., Christiansson, M. T., Greer, D., Lassenius, C., Männistö, R., Nayebi, M., Oivo, M., Penzenstadler, B., Prikladnicki, R., Ruhe, G., Schekelmann, A., Sen, S., Spínola, R. O., Tuzcu, A., de la Vara, J. L., and Winkler, D. 2019. "Status Quo in Requirements Engineering: A Theory and a Global Family of Surveys," *ACM Transactions on Software Engineering and Methodology* (28:2), Article 9.

Wainer, H. 1984. "How to Display Data Badly," *The American Statistician* (38:2), pp. 137-147.

Wand, Y., and Wang, R. Y. 1996. "Anchoring Data Quality Dimensions in Ontological Foundations," *Communications of the ACM* (39:11), pp. 86-95.

Wand, Y., and Weber, R. 1990. "An Ontological Model of an Information System," *IEEE Transactions on Software Engineering* (16:11), pp. 1282-1292.

Wand, Y., and Weber, R. 1993. "On the Ontological Expressiveness of Information Systems Analysis and Design Grammars," *Information Systems Journal* (3:4), pp. 217-237.

Wand, Y., and Weber, R. 1995. "On the Deep Structure of Information Systems," *Information Systems Journal* (5:3), pp. 203-223.

Wand, Y., and Weber, R. 2002. "Research Commentary: Information Systems and Conceptual Modeling—A Research Agenda," *Information Systems Research* (13:4), pp. 363-376.

Weinmann, M., Schneider, C., and Vom Brocke, J. 2016. "Digital Nudging," *Business & Information Systems Engineering* (58:6), pp. 433-436.

Wertheimer, M. 1923. "Untersuchungen zur Lehre von der Gestalt II," *Psychologische Forschung* (4:1), pp. 301-350.

Wickens, C. D., Hollands, J. G., Banbury, S., and Parasuraman, R. 2015. *Engineering Psychology and Human Performance*, Hove, UK: Psychology Press.

Winn, W. 1993. "An Account of How Readers Search for Information in Diagrams," *Contemporary Educational Psychology* (18:2), pp. 162-185.

Yadav, S., Bravoco, R., Chatfield, A., ahd Rajkumar, T. M. 1988. "Comparison of Analysis Techniques for Information Requirement Determination," *Communications of the ACM* (31:9), pp. 1090-1097.

Zhang, J. 1997. "The Nature of External Representations in Problem Solving," *Cognitive Science* (21:2), pp. 179-217.

Zhang, J., and Norman, D. A. 1994. "Representations in Distributed Cognitive Tasks," *Cognitive Science* (18:1), pp. 87-122.

Zhang, J., and Patel, V. L. 2006. "Distributed Cognition, Representation, and Affordance," *Pragmatics & Cognition* (14:2), pp. 333-341.

Zhang, P., and Soergel, D. 2014. "Towards a Comprehensive Model of the Cognitive Process and Mechanisms of Individual Sensemaking," *Journal of the Association for Information Science and Technology* (65:9), pp. 1733-1756.

Zhang, Y., and Patel, S. 2010. "Agile Model-Driven Development in Practice," *IEEE Software* (28:2), pp. 84-91.

About the Authors

Monika Malinova is a postdoctoral researcher with the Institute for Data, Process and Knowledge Management at the Vienna University of Economics and Business (WU Vienna). She obtained her Ph.D. in 2016 from WU Vienna. Her thesis was awarded the Dr. Maria Schaumayer Excellent Dissertation Award in 2016. In 2015, Monika was awarded the Internationalization Program Scholarship from the Dr. Maria Schaumayer Foundation. Since January 2018, she is a Hertha Firnberg Fellow funded by the FWF Austrian Science Fund. The Hertha Firnberg is a program offered to highly qualified female scientists who are working towards a career in universities. Monika's research interests include business process management, process innovation, and conceptual modeling.

Jan Mendling is a Full Professor with the Institute for Data, Process and Knowledge Management at the Vienna University of Economics and Business (WU Vienna) and a Visiting Professor with the School of Economics and Business at the University of Ljubljana, Slovenia. His research interests include business process management and information systems. He has published more than 400 research papers and articles, among others in the *MIS Quarterly, Journal of the Association for Information Systems, ACM Transactions on Software Engineering and Methodology, IEEE Transaction on Software Engineering, European Journal of Information Systems,* and *Decision Support Systems*. He is a board member of the Austrian Society for Process Management, one of the founders of the Berliner BPM-Offensive, and member of the IEEE Task Force on Process Mining. He is a co-author of the textbooks *Fundamentals of Business Process Management and Wirtschaftsinformatik* (in German).

Appendix A

Diagrams and SAD Tasks

Table A1 lists examples of diagrams that Bourque and Fairley (2014), Graham et al. (1997), and Sommerville (2011) associate with the five tasks of the systems analysis and design (SAD) process. Note that Dobing and Parsons (2006) show in an empirical study that the major UML diagrams are used throughout the entire SAD process with a tendency that use cases and activity diagrams are more frequently used at the early stages and that class and sequence diagrams are more frequently used for design specification.

Table A1. Diagrams Produced/Used During SAD Tasks

SAD Task	Diagrams
Elicit Requirements	Activity Diagram, Data-Flow Diagram, Use Case Diagram, Class Diagram, Business Process Diagram, Object Diagram
Refine Requirements	Sequence Diagram, Class Diagram, Semantic Data Diagram
Specify Design	Class Diagram, Sequence Diagram, Collaboration Diagram, State Diagram, Data-Flow Diagram, Component Diagram, Deployment Diagram, Communication Diagram, State Transition Diagram, Activity Diagram, Decision Diagram, Flowchart, Structure Chart, Entity Relationship Diagram, Object Diagram
Decompose Design	Object Diagram, Role Diagram, Business Process Diagram, Content Diagram
Implement System	Data-Flow Diagram, Control-Flow Diagram, Workflow Diagram, Executable Diagram

Appendix B

Theoretical Foundation

A brief description of the cognitive theories that our cognitive framework of understanding and task performance with diagrams (CogniDia) is built on is shown in Table B1, while Table B2 briefly describes the main constructs that underlay the seven theories.

Table B1. Cognitive Theories

Theory ID	Theory Name	Theory Description
ITM	Integrated Theory of Mind	Concerned with different modules which explain how the mind is organized by exchanging information with the central executive via buffers (Anderson et al. 2004).
DCT	Dual Coding Theory	Concerned with how humans process and represent verbal and nonverbal information in separate, related systems (Clark and Paivio 1991; Paivio 1991).
CTML	Cognitive Theory of Multimedia Learning	Concerned with how humans learn more deeply from words and pictures than from words alone (Mayer 2002).
MoWM	Model of Working Memory	Concerned with the components of working memory that play a role during a wide range of cognitive activities (Baddeley 1992).
CLT	Cognitive Load Theory	Concerned with the way in which cognitive resources are used during learning and problem solving (Chandler and Sweller 1991).
CFT	Cognitive Fit Theory	Concerned with the effectiveness and efficiency of problem solving when both the problem representation and the task at hand emphasize the same type of information (Vessey 1991).
HPS	Human Problem Solving	Concerned with how humans solve problems by searching and integrating information in a problem space (Newell and Simon 1972).

Table B2. Key Constructs in Cognitive Theories

Construct	Construct Used in	Construct Description
Visual module	ITM	The visual module identifies visual elements from a representation by coordinating eye motion and focus.
Declarative module	ITM	The declarative module retrieves information from long-term memory.
Intentional module	ITM	The intentional module keeps track of a hierarchy of goals.
Manual module	ITM	The manual module controls the hand.
Visual buffer	ITM	The visual buffer keeps track of locations and visual objects.
Retrieval buffer	ITM	The retrieval buffer holds information retrieved from long-term declarative memory.
Goal buffer	ITM	The goal buffer keeps track of one's internal state in solving a problem.
Manual buffer	ITM	The manual buffer is responsible for controlling and monitoring hand movement.
Production rule	ITM	A production rule updates the buffers in the mental architecture. The conditions of the production rule specify a pattern of activity in the buffers that the rule will match, and the action specifies changes to be made to buffers.
Verbal stimuli	DCT, CToML, CFT	The verbal representation (words) which is being apprehended by the senses.
Nonverbal stimuli	DCT, CToML, CFT	The nonverbal representation (images) which is being apprehended by the senses.
Problem solving task	CFT	A problem solving task is the representation of the task that needs to be solved.
Prior knowledge	CToML, CLT, MoWM	Prior knowledge is learned knowledge saved as schemes in long-term memory.
Problem solving skills	CFT	The procedure for dealing with situations as they arise. A skill exists only in the context of some objective or task.
Verbal system	DCT, CToML, MoWM	The verbal system is specialized for the representation and processing of verbal stimuli (e.g., words). The verbal system processes verbal stimuli in a sequential manner.
Nonverbal system	DCT, CToML, MoWM	The nonverbal system is specialized for the representation and processing of nonverbal stimuli (e.g., images). The nonverbal system processes nonverbal stimuli in parallel.
Central executive	MoWM, ITM	The central executive controls and executes mental operations. It incorporates information from the verbal system, nonverbal system and long-term memory.
Sensory memory	CToML, CLT	The sensory memory receives the stimuli and stores it for a very short time.
Working memory	CToML, CLT, MoWM	The working memory actively processes information to create mental constructs.
Long-term memory	CToML, CLT, MoWM, ITM	The repository of all things learned.
Associative connections	DCT, CToML	Associative connections joins representations within both verbal and nonverbal systems.
Referential connections	DCT, CToML	Referential connections makes links between the verbal and nonverbal systems.
Selecting material	CToML	Selecting material entails bringing material from the represented stimuli into the working memory.
Organizing material	CToML	Organizing material entails building structural relations among the elements from the stimuli.
Integrating material	CToML	Integrating material entails building connections between the verbal and nonverbal models with prior knowledge.
Intrinsic cognitive load	CLT	Intrinsic cognitive load is intrinsic to the material being dealt with. High-element interactivity imposes high intrinsic load.
Extraneous cognitive load	CLT	Extraneous cognitive load is the effort required to process poorly designed instructional material.
Germane cognitive load	CLT	Germane cognitive load is the effort that contributes to the construction of schemes.
Mental verbal model	DCT, CToML	The mental verbal model is a coherent representation of the learner's working memory of the selected words or phrases.
Mental nonverbal model	DCT, CToML	The mental nonverbal model is a coherent representation in the learner's working memory of the selected images.
Mental problem representation	CFT	The way the problem solver represents the problem in working memory.
Problem solution	CFT, HPS	The solution of the problem being posed.
Searching solution	HPS	Searching and integrating information from the representation until a situation which corresponds to the solution is found.

Appendix C

Systematic Literature Review on Diagram Criteria

Here we explain how we proceeded to identify papers that include diagram criteria in relation to IS design by means of the systematic literature review. Our objective was to identify criteria on diagrams such that it can be easily cognitively processed by a person reading it. Since diagrams are based on a specific modeling language, there are also criteria that relate to the language.

We prepared a review protocol of our search for literature on criteria for both modeling languages and diagrams. We searched the following six electronic libraries: ScienceDirect, IEEExplore, ACM Digital Library, Springer, EBSCO, and AISeL (AIS electronic Library). We focus on these libraries because they include articles in the fields of computer science and information systems. Additionally, we also consider the eight IS basket journals: *European Journal of Information Systems* (EJIS), *Information Systems Journal* (ISJ), *Information Systems Research* (ISR), *Journal of the Association of Information Systems* (JAIS), *Journal of Information Technology* (JIT), *Journal of Management Information Systems* (JMIS), *Journal of Strategic Information Systems* (JSIS) and *MIS Quarterly* (MISQ). We do this because some of the IS basket journals are not indexed in any of the aforementioned libraries. Once we had derived a list of relevant primary sources, we additionally used the "snowball" technique by inspecting the references of the papers we identified as result of our search in order to find more relevant references. Our search terms reflect alternative traditions in terminology including the closely related terms *language*, *grammar*, *notation*, and *technique*. We are also interested in relevant papers from a wide spectrum of IS domains where diagrams are often used, such as conceptual modeling, data modeling, information modeling, process modeling, software requirements specification, domain-specific languages and programming languages. Since a modeling language is a type of visual language, we also include papers from the domains of visual languages and visual notations. Furthermore, we search for papers from the domain of icon design, since modeling languages consist of ideograms, shapes and lines, and icon design is the process of designing them.

Table C1. Search Expressions

Keyword	Conjunction	Keyword	Type	Field	Time Period
modeling	AND	criteria	journals	computer science	1980–2020
modeling language	OR	principle	magazines	decision science	
modeling grammar		guideline,	proceedings	business & management	
modeling technique modeling notation modeling script process model* data model* conceptual model* information model* requirements specification visual language programming language domain-specific icon design		evalu*	chapter	business information systems	

We observed that criteria are discussed from three related angles: guidelines, principles and quality criteria. Although being related, they serve a different purpose and are used at different points in time, either during the diagram or language development, or after a diagram or a language has been developed. Some are used only for diagrams, while others are applied during language design. To be precise, we use the following definitions.

A *guideline* is "a rule of instruction that shows or tells how something should be done" (Merriam-Webster 2015). It gives specific recommendations on how to create a diagram from scratch (Mendling, Reijers, and van der Aalst 2010). A *principle* is "a rule that helps you know what is right and wrong and that influences your actions" (Merriam-Webster 2015). It is a fundamental standard accepted as true and used to improve the quality of diagrams and languages. In terms of languages, it helps language developers meet their goals (Paige et al. 2000).

Similarly, principles provide modelers with directions that will primarily lead to the model satisfying the reader's expectations. A *quality criterion* is a benchmark for judging a language or a diagram (Krogstie et al. 2006; Lindland et al. 1994). Thus, quality criteria are used to evaluate the goodness and appropriateness of an existing modeling language or a diagram created by a modeling language.

We used these terms as our main keywords (guideline, principle and criteria) and the aforementioned domains as a second set of keywords to construct the expressions and queries for automated searches. We added the term "evaluation" (specifically, "evalu*") to the set of main keywords that we used in combination with the domains. Finally, we also used the diagram type in the search term. For example, while we searched for guidelines for process modeling, we also searched for guidelines for process models. Where applicable, we did this for all domains. Similar to "evalu*," we searched for "process model*," which as result will return papers about both process modeling and process models. Additionally, we used the logical conjunctions "AND" and "OR" to connect both the main keywords and the domains. We used these conjunctions within the same search expression in order to connect one or all domains with all main keywords (e.g., "modeling language" AND (guideline OR principle OR criteria OR evalu*)). We also composed separate search expressions for each of the diverse keyword combinations (e.g., "modeling language" AND guideline, "modeling language" AND principle, etc.). Depending on the metadata stored in the library, we searched for four main types of publications, namely journals, magazines, proceedings and chapters. We are aware that diagrams are used in most likely all fields we could imagine. However, as we are interested in only diagrammatic models used during the SAD process we searched for papers within five main fields, namely computer science, decision science, business & management, business information systems, and information systems. All search expressions can be seen in Table C1.

Paper Selection and Data Extraction

The six libraries provided us with a total of 18,819 papers, each including presumably one or more of the keywords we included in the search expressions. As a first step we reviewed the titles of each paper that was identified by the initial search. By inspecting the title, we identified many papers that were outside the scope of our research. For those papers that passed the inspection of the title, we reviewed the abstract. As a result, we selected 140 papers, which is less than 1% of the initial search. The 140 papers went to the second round of reviews, which includes reading the entire paper in order to ensure that the study is indeed about guidelines, principles or quality criteria for developing or evaluating modeling languages and diagrams produced by languages. After reading the 140 papers we ended up with 64 papers we included in our final set of primary sources. The search we did using Google Scholar, in order to ensure papers from the eight basket IS journals are not left out, led us to four additional papers we included.

Table C2. Sources

ID	Source	ID	Source	ID	Source
1	Armenise et al. 1993	29	Karsai et al. 2014	57	Overhage et al. 2012*
2	Becker et al. 2000	30	Kesh 1995*	58	Paige et al. 2000
3	Bendraou et al. 2010	31	Kiper et al. 1997	59	Parsons 1996
4	Bielkowicz and Tun 2001	32	Kolovos et al. 2006	60	Parsons and Wand 1997
5	Blackwell et al. 2001	33	Krogstie et al. 2006*	61	Parsons and Wand 2008*
6	Britton and Jones 1999	34	Krogstie et al. 1995a	62	Parsons and Wand 2013
7	Britton et al. 2000*	35	Krogstie et al. 1995b	63	Purchase, Carrington, & Allder 2000*
8	Burton-Jones and Meso 2006*	36	Krogstie 1998	64	Purchase, Allder, and Carrington 2002
9	Burton-Jones and Meso 2008*	37	Krogstie 2003	65	Recker et al. 2007
10	Cherfi et al. 2002*	38	Krogstie and Sølvberg 2003	66	Recker 2011*
11	Cherfi et al. 2007*	39	Kühne 2006	67	Sánchez-González et al. 2013
12	Curtis et al. 1992	40	Lange and Chaudron 2005*	68	Schuette and Rotthowe 1998*
13	Davis et al. 1993	41	Lankhorst 2013	69	Shanks et al. 2003
14	de Oca et al. 2014*	42	Levitin and Redman 1995	70	Siau and Tan 2005
15	Ding and Mateti 1990	43	Lindland et al. 1994	71	Simsion and Witt 2004
16	Dromey 1995	44	Lloyd and Jankowski 1999*	72	Sindiy et al. 2013
17	Eichelberger and Schmid 2009*	45	Maes and Poels 2007*	73	Stapleton and Delaney 2008
18	Figl et al. 2010*	46	McDougall et al. 1999*	74	Tamassia et al. 1988*
19	Gemino and Wand 2004	47	McDougall et al. 2000*	75	ter Hofstede et al. 1993
20	Giraldo et al. 2018	48	McGinnes and Kapros 2015	76	Tsironis et al. 2009*
21	Green 1989	49	Mendling, Reijers, and van der Aalst 2010*	77	Wand and Weber 1990
22	Green and Petre 1996	50	Mohagheghi et al. 2009	78	Wand and Weber 1993
23	Günther 2011	51	Moody and Shanks 1994	79	Wand and Wang 1996
24	Heidari and Loucopoulos 2014	52	Moody et al. 1998	80	Wertheimer 1923
25	Heravizadeh et al. 2008*	53	Moody 1998*	81	Winn 1993
26	Heymans et al. 2008*	54	Moody and Shanks 2003	82	Yadav et al. 1988*
27	Huang et al. 2002*	55	Moody 2009		
28	Burton-Jones et al. 2009	56	Nelson et al. 2012		

Finally, we skimmed through the references of our final selection of papers, which yielded another 18 relevant papers. These mostly include books, technical reports or articles that were not indexed in any of the libraries we used. We ended up with a total of 82 primary sources that we use to identify criteria for modeling languages and diagrams. The list of the primary sources is shown in Table C2.

The final step of the literature review is extracting and synthesizing the required data from the 82 primary sources. We read each of the 82 papers carefully. Some observations from reading informed our further classification. First, some papers partially referred to diagrams and languages. Second, most of them discussed specific types of diagrams, such as conceptual models or data models. Third, some criteria appeared to be more related to evaluating an entire language (e.g., Unified Modeling Language (UML)), a diagram of a certain language domain (e.g., data model) or element instances of a single diagram (e.g., ideograms, shapes, lines). We kept track whether the proposed criteria are applied on a language level or diagram level (at the level of a modeling language or the level of a diagram). However, differentiating between the different types of levels was not as obvious because different papers discuss the same criteria at language-level or at diagram-level, depending on the focus of the paper. If not explicit, we interpreted the intention of the criteria and classified them accordingly.

We extracted the relevant information from all sources. This includes the name, description, domain and source of the criterion. In addition, we recorded which criteria are empirically supported. For this, we kept track of those sources that conducted some type of empirical evaluation

of the criteria they propose (e.g., experiment, case study, action research). Those sources which include empirical evaluation of the criteria are marked with an asterisk (*) in Table C2. The data extraction resulted in a large set of data clustered into criteria used during language development or diagram creation. Although many of the criteria were repeating, we kept track of all of them. Thus, we ended up with a total of 866 criteria. We used Microsoft Excel to keep track of the data.

The next step was concerned with the identification of redundant criteria that could be merged. Both researchers discussed merge options until agreement was achieved. Merging was guided by the following considerations. First, those criteria that share the same name and refer to the same meaning were merged. Second, we also merged those that were assigned a different name, but shared the same meaning. Third, we ensured that the final set of criteria are consistently named and described. For instance, we always add a verb to a criterion label, such that it indicates a direction of what should be done. Additionally, since the 82 papers come from various domains, we made sure that their labels and descriptions are generically formulated. For example, when the definition of a criterion is referring to an information model, we replace the term *information model* with the term *diagram*. Removing the redundant criteria left us with a total of 279 criteria.

Then, we went through the 279 criteria and clustered them depending on which of the four cognitive processing steps (visual processing, verbal processing, semantic processing, task processing) from our CogniDia framework they pertain to. For instance, criteria concerned with the visual aspects of diagrams are clustered in the category "visual processing". Similarly, those criteria that are concerned with solving tasks were clustered in the category "task processing," respectively. Both researchers were involved in this clustering and reached an agreement on the classification.

We found 59 criteria that do not relate to any of the cognitive processing steps. These criteria apply to aspects not directly within the scope of human cognitive processing (e.g., pre-usage perception, post-usage perception, machine processing, resources, diagram utility and externalizing knowledge). For that reason, they are not included in this paper. Furthermore, we clustered the remaining 220 criteria into categories. Each category includes criteria that pertain to a similar matter. For instance, many criteria are concerned with the expressiveness of a modeling language in terms of the real world, which we clustered into a corresponding category "ontological fidelity." The 220 criteria were clustered into a total of 22 categories.

Appendix D

Criteria for Effective Cognitive Processing of Diagrams

Figure D1 illustrates the evolution of criteria over time. It is measured in terms of the number of papers published in a time interval of five years that introduces at least one criterion from the respective cognitive processing step.

Tables D2–D5 show an elaborate description of each criterion that belong to each cognitive processing step (visual processing, verbal processing, semantic processing, task processing). Each table includes the following: unique ID, name, description, the domain where the criterion has been applied according to prior research, and the source where we found the criterion. The domain ID's can be seen in Table D1. The domain illustrates where the respective criterion has been used. For example, criterion VP1 (Maximize visual aesthetics) is assigned to the domain PM which stands for process modeling. This means that criterion VP1 has been applied in the process modeling domain. We assigned a number to each primary source we analyzed. These are shown in Table C2. Additionally, criteria are clustered into further categories, these can also be seen in the respective Tables D2–D5.

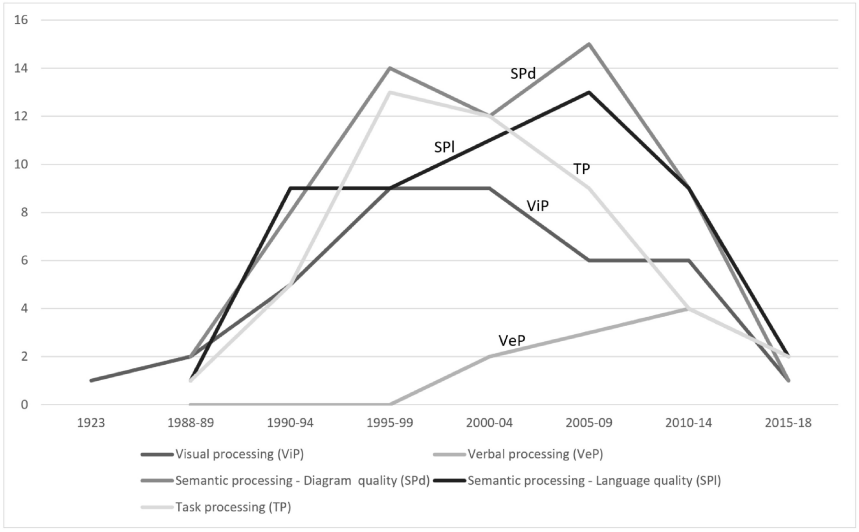

Figure D1. Evolution of Criteria Over Time

Table D1. IDs for Domains

Domain	ID
Information Systems	IS
Meta-Modeling	MM
Enterprise Modeling	EM
Process Modeling	PM
Conceptual Modeling	CM
Data Modeling	DM
Information Modeling	IM
Software Requirements Specification	SRS
Domain-Specific Language	DSL
Diagrams	D
Visual Language	VL
Visual Programming Language	VPL
Information Visualization	IV
Graphic User Interfaces	GUI

Table D2. Visual Processing

ID	Source	Description	Domain	Paper
Aesthetics				
vp1	Maximize visual aesthetics	Make sure the diagram provides users with a pleasing and satisfying interaction.	CM, PM	20, 40, 67
vp2	Differentiate between a diagram and its visualization	Make a clear distinction between a diagram and its visualizations in order to avoid considering a visualization of a diagram as the diagram itself, rather than focusing on the diagram contents. Do not use visual terms while describing a diagram. For example, avoid using terms such as "above" to denote that some concept is more important than another one.	EM	41
vp3	Use secondary notation	Make use of the various visual variables to convey extra meaning, beyond the official semantics of the language.	PM, CM, DSL, VPL, VL	5, 20, 22, 23, 31, 55, 76
vp4	Ensure easy location of contents	Organize the concepts in a diagram such that readers can easily locate information, and logical relationships among adjacent sections is apparent.	SRS, EM	13, 41
vp5	Align comment position	Keep comments, which may connect to multiple other diagram elements, as close as possible to the related elements. Thus, place comments connected to multiple diagram elements at the center if possible.	VL	14
vp6	Adjust diagram to presentation media	Choose the visual presentation properties of the generated views based on whether the presentation media will be electronic and/or hardcopy, or is for reports versus presentation, or stand-alone versus part of a larger package. The presentation media will drive some of the visual presentation properties of the generated views (example of properties: font size and type, aspect ratio of the diagram (e.g., landscape vs. portrait orientation), amount of text versus symbols and images).	IV	72
vp7	Minimize crossings of edges and nodes	Do not cross edges with nodes other than the origin and target ones.	DM	15
vp8	Align adjacent arrows direction	Make all arcs undirected which have an adjacent arrow indicating the direction of the message sent, rather than all arcs being directed.	SRS	63, 64
vp9	Maximize visual distance between lines	Clearly show line separation in order to influence the degree of ease with which two lines are differentiated.	DM	15
vp10	Enhance angles of edges	Do not make angles between edges too small.	IM, Di	68, 74
vp11	Align edge centers	If the number of edges between two nodes is even and the origin and target nodes are circles or ellipses, then draw edges such that all arc centers are on the line which is perpendicular and passes the center of the line connecting the centers of the two nodes.	DM	15
vp12	Harmonize edge length	Make the arrows adjacent to the arcs the same length as the arcs, rather than shorter than the arcs.	SRS	63, 64
vp13	Align edge direction	Highlight a uniform flow of the edges and have similar edge directions.	VL	17
vp14	Minimize hyper edge length	Draw as direct as possible the edges between edges such as generalizations of associations or XOR-constraints.	VL	17
vp15	Align visual distance of edges	Keep related edges close together and unrelated edges separated from each other.	DM	15
vp16	Maximize edge contours	Depict edges clearly in order to ease the perception of the figure ground.	VL	17
vp17	Minimize distance of high degree elements to center	Place elements with high degree in the center of the drawing.	IM, Di	68, 74
vp18	Maximize distance to center of disconnected elements	Place disconnected elements at the border of the drawing.	VL	17
vp19	Highlight coupling in diagrams	Spatially separate the classes connected to diagram elements outside a package from the classes connected to elements inside only. A package is an explicitly drawn cluster of similar elements.	VI	17

ID	Source	Description	Domain	Paper
vp20	Position elements in hierarchy	Apply median positions to elements in hierarchies (place elements being higher in hierarchy as close as possible to the median positions of the related elements on the next lower hierarchy level) in order to have them visually close to the median position.	VL	17
vp21	Align association rhomb position	Center the rhomb in n-ary associations between the connected classes to expose the special nature of this relation in order to have it centered.	VL	17
vp22	Align association classes position	Center association classes with respect to its dashed line and place them next to the center of the association path it connects to.	VL	17
vp23	Maximize symbol order	Consider the sequence of displaying the series of symbols when a symbol is dynamic.	GUI, DM, EM, Di	27
vp24	Use joined target style for generalization	Apply the joined target style instead of the separate target style for generalizations in order to improve readability. Both target styles are defined in UML as presentation options.	VL	17
vp25	Minimize start/end events	Use one start and one end event. The number of start and end events is positively connected with an increase in error probability.	PM	2
vp26	Minimize OR routing paths	Avoid OR routing elements. Diagrams that have AND and XOR connectors are less error-prone. There are some ambiguities in the semantics of the OR join leading to paradoxes and implementation problems.	PM	2
vp27	Minimize routing paths	Minimize the routing paths per element. The higher the degree of an element in the diagram (i.e., the number of input and output arcs together), the harder it becomes to understand the diagram.	PM	2
vp28	Use edge crossing symbol	Apply the symbol for edge crossings at associations to improve readability when relations cross an association.	VL	17
vp29	Minimize edge crossings	Minimize the number of edge intersections.	DM, IM, SRS, VL, PM, Di	14, 15, 17, 63, 68, 74
vp30	Minimize edge bends	Make sure the edge bend is as small as possible if the distance between two adjacent turns is small and the sides other than the one between the two turn points are long.	DM	15
vp31	Minimize total edge length	Minimize the global length of the edges.	IM, Di	68, 74
vp32	Minimize longest edge length	Minimize length of the longest edge.	IM, Di	68, 74
vp33	Minimize the number edge bends	Minimize the total number of bends in polyline edges.	SRS, VL, PM	14, 17, 63
vp34	Join inheritance lines	Join inheritance lines prior to reaching the superclass, rather than being represented as separate arcs.	SRS	63, 64
vp35	Maximize orthogonality	Maximize the number of orthogonally drawn connecting elements.	PM	14
vp36	Maximize face number	Maximize the number of faces drawn as convex polygons.	IM, Di	68, 74
Visual Consistency				
vp37	Harmonize visual design	Repeat some aspect of the design throughout the diagram such that people would perceive the diagram contents as coherent and organized.	SRS	6, 7
vp38	Harmonize visual appearance	Provide a uniform appearance of the drawing to support similarity and homogeneity.	VL	17
vp39	Ensure element consistency	Represent elements of a diagram in exactly the same way if they are the same; if they are not the same, then represent them differently.	SRS	6, 7
vp40	Use consistent symbols	Ensure that all symbols which have the same semantics are drawn in the same way and the meaning of the symbol should be consistent with user's mental models. Inconsistent symbols are difficult to learn and recall.	DM, IM, SRS, GUI	6, 7, 15, 27, 68
vp41	Harmonize symbol features	Arrange the features in a symbol carefully. Their relative direction, location, etc., could affect symbol quality.	GUI	27

ID	Source	Description	Domain	Paper
vp42	Harmonize shape and thickness of edges	Use the same shape and thickness for edges of one kind in a diagram. Use different shapes and/or thickness for different kind of edges in the same diagram. Use the same shape and thickness for different kinds of edges in different diagrams.	DM	15
vp43	Adapt element size	Adapt the size of elements in order for all elements to have enough space.	PM	14
vp44	Harmonize diagram density	Have uniform density of elements in the drawing.	IM, Di	68, 74
vp45	Minimize element overlap	Avoid overlaps between elements as well as between elements and edges.	VL, PM	14, 17
vp46	Minimize diagram size	Minimize the area occupied by the drawing to support a homogeneous node and edge distribution and to reduce the need of scrolling the final drawing.	IM, Di, PM, VL	14, 17, 68, 74
vp47	Maximize visual structuredness	Clearly structure a diagram, such that it involves less effort on the part of readers to find, decompose, and abstract information, and thus be easier to understand.	DM, SRS, PM	2, 6, 7, 16, 44, 71
vp48	Harmonize syntax and semantics in diagram layout	Prefer a layout which does not have any impact on the meaning of the diagram, and thus, does not affect the translation of the concrete to the abstract syntax and the semantics.	DSL	29
vp49	Harmonize elements in diagram	Ensure that nothing is placed randomly in a diagram. Ensure that each element has a strong visual connection with something else in the diagram and the overall structure should appear to be a cohesive unit.	SRS	6, 7
vp50	Align element orientation	Use horizontal or vertical orientations as they are more likely perceived as a figure than other orientations.	VL	17
vp51	Use grid and gutters for alignment	Layout technical content because they provide visually appealing organization and/or symmetry. Use space, gutters and grid to align elements. Gutters separate and organize content spaces. The grid allows to align each of the graphical elements on the page.	IV, VL	17, 72
vp52	Maximize regularity	Systematically arrange elements by certain patterns such that the regularity of the underlying structure can be formulated geometrically.	DM	15
vp53	Maximize symmetry	Balance the diagram with respect to the vertical axis or horizontal axis by arranging elements in diagrams such that the size, form, shape, and arrangement of figure elements on opposite sides of a plane, line, or point correspond to each other.	Di, DM, VL, IM	15, 17, 68, 74
vp54	Maximize proportionality	Apply visual order that complements the structured functional decomposition of the diagram.	IV	72
vp55	Align to an orthogonal grid	Align nodes and edges to an orthogonal grid.	SRS	63, 64
vp56	Align hierarchy structure	Represent hierarchical structures vertically, rather than horizontally.	IM, Di	68, 74
vp57	Use visual partition	Consider the use of partitions, e.g., pools and swimlanes.	PM	14
vp58	Minimize symbols' typeface	Limit the symbols' typeface to one or two type families only.	GUI	27
vp59	Test symbols before use	Test the symbols before use.	GUI	27
Visual Simplicity				
vp60	Minimize diagram complexity	Create simple diagrams. The simpler it looks, the simpler its components are geometrically, the easier it can be understood.	DM	15
vp61	Minimize visual complexity	Use as few elements in the diagram as possible. Diagrams should take up as little space on the printed page as possible.	CM, CM, VL, PM, DSL	2, 6, 7, 14, 20, 29, 55, 58
vp62	Minimize syntactic sugar	Minimize the use of elements which do not contribute to the expressiveness of the language. Syntactic sugar mainly serves to improve readability. An overuse of the addition of syntactic sugar distracts, because verbosity hinders to see the important content directly.	DSL	29
vp63	Support complexity management	Make sure the language has the ability to represent information without overloading the human mind.	CM, VL	20, 55

ID	Source	Description	Domain	Paper
vp64	Minimize symbol variety	Ensure the number of different graphical symbols the language provides and the diagram includes is cognitively manageable.	VI, SRS, CM, VPL	5, 6, 20, 22, 55
vp65	Maximize symbol simplicity	Make the symbol features as simple as possible, consistent with the inclusion of features that are necessary.	GUI, VL, DM	15, 17, 27, 46
Pop-out				
vp66	Maximize symbol prominence	Apply contrast in symbols by using color, shape, and size which can cause them to pop out perceptually and call attention to themselves.	VL	47
vp67	Align element size	Chose the size of the elements according to the importance of the individual elements. Keep element sizes consistent.	DM, VL	15, 17
vp68	Minimize distance of important elements to center or diagram boundary	Place some elements at the center or at the boundaries of the diagram in order to emphasize the elements.	VL	17
vp69	Maximize discrimin-ability between sym-bols and background	Make sure the symbol can be clearly differentiated from the background.	GUI	27
vp70	Use symbol boundaries	Assign clear boundaries to a symbol.	GUI	27
vp71	Use color in symbols	Use color in symbol design. It is recommended to use color selectively, so that it brings key information into focus.	VL, IV, GUI	17, 27, 72
Grouping				
vp72	Maximize role-expressiveness	Make sure the reader can identify how each component of a diagram relates to the whole.	VPL	5, 21, 22
vp73	Maximize flow direction	Maximize the number of connecting elements respecting workflow direction.	PM	14
vp74	Maximize cohesiveness	Closely relate the attributes of the entities to each other, because each entity represents a certain theme.	CM, DM, SRS	8, 9, 16, 30, 42, 78
vp75	Cluster similar elements	Cluster similar elements and consider a spatial distribution according to these clusters.	CM, DM, VL	15, 17, 75
vp76	Represent elements in a continuation	Place elements in straight or smoothly curved lines if they belong together.	SRS, DM, VL	6, 17, 44, 80
vp77	Group proximal elements	Group elements physically close to each other.	DM, VL	6, 7, 17, 44, 80
vp78	Group visually similar elements	Group symbols that are similar to each other in size, shape, color, etc.	SRS, DM, VL	6, 17, 44, 80
vp79	Group elements with common fate	Group symbols that move or appear to move in the same direction and/or at the same speed.	SRS, DM	6, 44, 80
vp80	Unite connected elements	Unite (group) elements which are connected to each other.	VL	17
vp81	Group familiar symbols	Group elements that will be familiar or (semantically) meaningful to the user.	VL	17, 80
vp82	Group elements as a straight line	Group elements so that they are displayed as a straight line or on a shape established by an external semantic feature.	VL	17
vp83	Group elements forming a closure	Group elements together if they form a closed figure.	SRS, DM	6, 44, 80
vp84	Unify symbols	Unify symbols as much as possible. For example, when solid and outline figures occur together, the solid figure should be within the outline figure.	GUI	27
Symbol Distinctinctiveness				
vp85	Maximize perceptual discriminability	Make different symbols in the language easy to distinguish from one another. The ease with which different symbols in a language can be distinguished from one another depends on how physically distinct each symbol is from others in the language.	CM, Di, SRS, GUI, VL, VPL, DSL	6, 7, 27, 20, 29, 47, 55, 81
vp86	Maximize symbol recognizability	Make symbols easy to recognize by giving them emergent properties that facilitate easier recognition. Two symbols do not share emergent properties.	GUI, DM, EM, Di	15, 27, 41, 81

ID	Source	Description	Domain	Paper
Symbol Clarity				
vp87	Maximize symbol familiarity	Make the symbols of the language familiar to its users. If symbols are familiar it means that the symbols in the language will be closely related to the concepts that they represent and therefore that their meaning will be clear, even to untrained users.	DM, SRS, GUI, VL	6, 7, 15, 27, 44, 46
vp88	Use standard shapes for symbols	Use standard elements such as regular polygons, circles, ellipses, trapezoids, and diamonds unless special elements are required. Element shapes should be kept consistent.	DM	15
vp89	Use mnemonic symbols	Construct symbols such that they express their intended messages clearly. Ensure symbols are concrete such that it allows people to use their knowledge of the everyday world in order to interpret them. Ensure that an unambiguous relationship exists between each symbol in the diagram and the object from the real world to which it refers. When we look at a diagram, even if it represents something we do not understand, we can nonetheless detect the objects it contains and then discriminate among them and configure them into groups.	Di, DM, GUI, VL	27, 30, 46, 81
vp90	Maximize semantic transparency	Use symbols whose appearance suggests their meaning. The symbol's implicit meanings should be close to the intended ones.	VL, CM, GUI	20, 27, 55
vp91	Minimize symbol ambiguity	Make the symbols' meanings stable, thus not a matter of ambiguity.	GUI	27
vp92	Provide symbol feedback	Make sure the symbol offers some kind of feedback to users.	GUI	27

Table D3. Verbal Processing

ID	Source	Description	Domain	Paper
Clarity of Text				
ap1	Maximize explicitness	Make the informational content explicit, not implicit.	Di	73
ap2	Apply dual coding	Use text to complement graphics.	CM, VL	20, 55
ap3	Provide comments on elements	Permit comments on diagram elements, because they are essential for explaining design decisions.	DSL	29
ap4	Emphasize diagram contents	Draw attention to the information, not the form. One way to do this is by excluding meta-data of represented constructs.	IV	72
ap5	Include diagram information	Include diagram information such as title, type, authorship, revision, status, dates in abstracting viewpoints as templates for views.	IV	72
ap6	Provide abbreviation details	Ensure that the audience can easily discern all abbreviations and acronyms within a diagram.	IV	72
ap7	Label data appropriately	Label the stored data relating to each entity such that the germane models can be determined.	IS	48
Effective Presentation of Text				
ap8	Minimize label size	Use short activity labels.	PM	14
ap9	Use uniform label style	Use a uniform style for names and flow descriptions.	PM	14
ap10	Use verb-object labels	Use verb-object activity labels.	PM	49, 14
ap11	Use directional indicators	Label arcs with two relationship labels and two directional indicators, rather than one.	SRS	63, 64
ap12	Adjust font size based on content significance	Use hierarchy of type, where the size of the font is proportional to the significance of the content. Have no fonts smaller than 7 pt. Consider a uniform orientation of text labels and if not restricted use similar sizes for semantic groups of elements.	IV, VL	17, 72
ap13	Use uniform font type	Make all text fonts the same, rather than using different fonts for different types of labels. Use the same font type within and across all diagrams. Use the same font for headlines. Match diagram font type to the products they are employed in.	SRS	17, 63, 64, 72
ap14	Align text horizontally	Make all text labels horizontal, rather than a mixture of horizontal and vertical.	SRS	63, 64

Table D.4. Semantic Processing

ID	Name	Description	Domain	Source
Diagram Quality				
Semantic Complexity				
sp1	Maximize simplicity	Ensure that the diagram contains the minimal possible constructs. Ensure that the language is as simple as possible in order to express the concepts of interest and to support its users in their preferred ways of working. Simplicity is a well-known criterion which enhances the understandability of a language. A short notation is more preferable for frequently used elements rather than for rarely used elements. Reducing the amount of expressions or simplifying their appearance while the semantics is not changed leads to better understanding of the application language and the domain.	CM, DM, DSL, SRS, VDL	10, 11, 13, 23, 26, 30, 32, 51, 52, 53, 54, 58
sp2	Minimize complexity	Achieve the lowest possible complexity in diagrams. There is a relationship between the complexity of a diagram and its understanding and error probability: more complex diagrams tend to be more difficult to understand and more prone to errors.	CM, IS, DM, PM	14, 20, 40, 44, 82
sp3	Treat different concerns orthogonally	Address different concerns in different parts of the diagram, or in different, related diagrams.	EM	41
sp4	Decompose diagram with more than 50 elements	Split up larger diagrams into smaller diagrams. Large subcomponents with a single entry and a single exit can be replaced by one activity that points to the original subcomponent as a separate diagram.	PM	49
sp5	Ensure appropriate level of abstraction	Make sure that an abstraction (decomposition) is appropriate for a particular level.	IS	17
sp6	Maximize consistency	Apply the same type of concepts to denote the same type of elements from the real world. Model similar relations in a similar manner. Use the same terms to denote the same concepts, also in related diagrams.	CM, DM, IS, SRS, DSL, IV, VPL, PM, EM, MM	5, 6, 7, 16, 20, 22, 29, 30, 41, 42, 50, 57, 58, 69, 72, 82
sp7	Use hidden dependencies	Include hidden dependencies in a diagram. A hidden dependency is a relationship between two components such that one of them is dependent on the other, but that the dependency is not fully visible. For example, when abstraction mechanisms are used and the abstraction levels are not seen, but the elements in each level are dependent on each other.	CM, SRS, DSL, VPL	5, 6, 21, 22, 29, 75
sp8	Minimize weak coupling	Avoid including unnecessary interaction among objects in a diagram.	CM, SRS	8,9, 16, 78
sp9	Use cross-referencing	Use cross-references in the diagram to relate sections containing requirements to other sections containing: identical (i.e., redundant requirements, more abstract or more detailed descriptions of the same requirements and requirements that depend on them or on which they depend.	CM, SRS	15, 20, 40
Syntactic Quality				
sp10	Maximize diagram conformance to rules	Create a diagram such that it conforms to the rules of the language which govern the combination of sentences to form complex expressions.	PM	57
sp11	Maximize syntactic quality	Create a diagram such that it corresponds with the language extension of the language in which the diagram is written.	CM, IS, PM	20, 28, 34, 35, 36, 37, 38, 40, 43, 50, 56, 57, 70, 76, 82
sp12	Impose restrictions	Impose only the necessary restrictions on what constitutes a well-formed statement.	DM	73
sp13	Maximize determinism	Make every event at every level in the level structure of the system either an external event or a well-defined internal event. Determinism will be violated in diagrams if an analyst does not specify the condition causing a split in an activity.	CM	8, 9, 77

ID	Name	Description	Domain	Source
Semantic Quality				
sp14	Maximize intentional quality	Make sure the diagram remains true to the mindset and the meanings defined by the real world.	CM	56
sp15	Maximize semantic quality	Create a diagram such that it is a correct representation of a domain from the real world, focusing on specific aspects. All statements in the diagram are according to the syntax and vocabulary of the modeling language. Semantic correctness postulates that the structure and the behavior of the model is consistent with the real world.	CM, IS, DM, SRS, PM, EM	2, 4, 10, 13, 16, 20, 25, 28, 30, 33, 34, 35, 36, 37, 38, 41, 43, 45, 50, 51, 52, 53, 54, 56, 57, 70, 69, 76, 82
sp16	Maximize representational fidelity	Make sure the diagram faithfully represents someone's perception, or some group's negotiated perception, of the semantics of the domain.	CM	28, 56
sp17	Ensure maxim of quality	Do not model what you believe to be false. Do not model that for which you lack adequate evidence.	EM	41
sp18	Minimize ambiguity	Clearly define all components of a view. Formulate the elements of a diagram with respect to content. Every requirement stated in a diagram has only one possible interpretation.	IS, DM, SRS, PM	13, 15, 42, 57, 79
sp19	Maximize domain precision	Provide details when specifying an attribute's domain.	DM	42
sp20	Maximize losslessness	Make sure that every inherited state variable and every emergent state variable in a system is preserved in the decomposition. Losslessness will be violated in class diagrams if an analyst decomposes classes in such a way that important associations among classes are lost or part-whole relationships are not correctly specified and the attributes or class names of the wholes or parts are lost.	CM	8, 9, 77
sp21	Maximize systematic design	Create a diagram such that it postulates well-defined relationships between diagrams which belong to different views.	IM, PM	2, 68
Pragmatic Quality				
sp22	Maximize social pragmatic quality	Make sure people understand the diagrams.	CM, DM, DSL, VL, PM	37
sp23	Maximize agreement	Ensure that the various projections due to the different people reading the diagram are consistent. Agreement covers agreement in knowledge, agreement in interpretation, and both relative and absolute agreement.	CM, PM	20, 33, 34, 35, 37, 38
sp24	Ensure diagram fitness	Make sure that the users can faithfully follow the steps the diagram specifies.	SRS	15
sp25	Maximize internalizability	Make sure that the externalized diagram is persistent and available enabling users to make sense of it. It can be achieved by two means: persistency and availability.	CM	20, 34, 35, 37, 38
sp26	Maximize comprehensibility appropriateness	Make sure that the diagrams from a language can be easily understood. A diagram is understandable iff readers can easily comprehend the meaning of all requirements with less mental effort.	CM, DM, SRS, IS, VPL, PM	10, 13, 20, 31, 35, 36, 37, 40, 51, 52, 53, 54, 57, 70, 82
sp27	Balance compactness and comprehensibility	Ensure that the diagram is comprehensible by avoiding too much verbosity, and compact by focusing on frequently used elements rather than rarely used elements.	DSL	29, 40
sp28	Maximize communicability	Make sure the diagram produced by the use of the language is readable and understandable.	DM, IS	71, 82
sp29	Maximize clarity	Create a diagram such that it is easy to read.	IS, CM, IM, PM	2, 10, 11, 68, 78

ID	Name	Description	Domain	Source
sp30	Maximize pragmatic quality	Make sure that the diagram and the audience's interpretation of the diagram correspond (i.e., the statements that the audience think that the diagram consists of).	CM, PM	20, 34, 35, 37, 38, 42, 43, 45, 50, 56, 57, 70
sp31	Maximize user-domain appropriateness	Make sure that the audience is already familiar with or is able to get familiar with the domain of both the language and the diagram.	CM, PM	20, 33, 34, 35, 38, 43, 56, 70
sp32	Maximize expressiveness	Make sure the diagram represents user requirements in a natural way.	CM, D	10, 11, 73
sp33	Maximize maxim of manner	Avoid obscurity of expression, avoid ambiguity, be brief, avoid unnecessary concepts and relations, and be orderly.	EM	41
sp34	Maximize schema expressiveness	Make sure the diagram is expressive as a whole and as informative as necessary.	CM, EM	10, 11, 41
sp35	Annotate versions	Make sure the readers can easily determine which requirements will be satisfied in which version of the product.	SRS	13
sp36	Annotate relative importance	Make sure that readers can easily determine which requirements are of most importance to customers, which are next most important, etc.	SRS	13
sp37	Annotate relative stability	Make sure that the readers can easily determine which requirements are most likely to change, which are next most likely, etc.	SRS	13
sp38	Articulate domain changes	Make sure that changes in the domain become known to all users of the diagram. Known changes should become articulated into the diagram.	PM	33
sp39	Maximize diagram precision	Create a diagram such that numeric quantities are used whenever possible and the appropriate levels of precision are used for all numeric quantities. A variable or constant is imprecisely typed when its precision is not sufficient to meet the required accuracy of the computation.	SRS, PM	12, 13, 16
sp40	Maximize diagram verifiability	Ensure that there exist finite, cost effective techniques that can be used to verify that every requirement stated in the diagram is satisfied by the system to be built.	SRS	13
Language Quality				
Fit with Modeling Knowledge				
sp41	Reuse existing language concepts	Reuse language concepts from previously existing languages, only if the concepts of the languages to be composed fit together.	DSL	29
sp42	Reuse existing language definitions	Take the definition of a language as a starter to develop a new language, rather than creating a language from scratch. The new language might retain a look-and-feel of the original.	DSL	29
sp43	Reuse existing notational elements	Adopt formal notation the domain experts already have, rather than inventing a new one. This will allow the user to easily identify familiar notational elements.	DSL	29
sp44	Reuse existing type systems	Reuse existing type systems to improve comprehensibility and to avoid errors that are caused by misinterpretations in an implementation.	DSL	29
sp45	Maximize language-user appropriateness	Make sure that the audience is able to learn, understand, and use the language. The goal is to ensure that there are no statements in the explicit knowledge of the user that cannot be expressed in the language.	CM, PM	20, 34, 37, 43, 56, 76
sp46	Maximize linguistic quality	Make sure that the users of the representation are able to master the basics of the employed modeling language in order to understand the conceptual representation.	CM	56
sp47	Maximize applied domain-language appropriateness	Develop a language such that it is appropriate to the modeler's knowledge of the real-world domain.	CM	56
Ontological Fidelity				
sp48	Maximize language-domain appropriateness	Develop a language such that it fits the domain and it enables the making of the kind of statements needed in the domain. The expressiveness of a language is the ability to generate diagrams that capture information about a modeled domain.	CM, IM, SRS, DSL, PM, VPM	5, 6, 19, 20, 22, 29, 32, 33, 34, 36, 37, 43, 56, 68

ID	Name	Description	Domain	Source
sp49	Increase ontological quality	Make sure the language (syntax and semantics of the language) is appropriate for expressing the concepts of the diagram and for ultimately encoding the concepts in the real world.	CM	56
sp50	Maximize ontological completeness	Make sure the language represents the same information about the real world that can be represented in the diagram. Users of ontological incompleteness are unable to represent all real world phenomena that might interest them.	IS, PM	66, 78
sp51	Maximize concept expressiveness	Define a language such that it provides concepts which are expressive enough to capture the main aspects of the reality.	CM, D	10, 11, 73
sp52	Maximize expressive power	Make sure the language achieves ontological completeness and ontological clarity.	IS, PM	1, 78
sp53	Minimize construct deficit	Avoid an ontological construct that has no mapping from any modeling construct 1:0.	IS, PM, VL	20, 55, 65, 66, 77
sp54	Maximize ontological clarity	Make sure each construct in the language clearly represents an ontological construct.	IS, PM	66, 77
sp55	Minimize construct overload	Avoid single modeling construct mapping to two or more ontological constructs m:1.	IS, PM, VL	55, 65, 66, 77
sp56	Minimize construct redundancy	Avoid two or more modeling constructs mapping to a single ontological construct 1:m.	CM, DM, DSL, IM, IS, SRS, PM, VL	4, 13, 20, 29, 55, 58, 59, 60, 61, 62, 65, 66, 69, 71, 77
sp57	Minimize construct excess	Avoid one modeling construct that does not map onto any ontological construct 0:1.	IS, PM, VL	20, 55, 65, 66, 77
sp58	Maximize construct equivalence	Define a language as a 1:1 mapping to a reference framework. The construct prescribed by the reference framework can unequivocally be mapped to one and only one construct of the modeling language (1:1 mapping).	PM, DSL	32, 65
sp59	Maximize semantic richness	Develop a language such that it is able to express what is actually performed during software development processes.	PM	3
sp60	Maximize semiotic clarity	Ensure a 1:1 correspondence between semantic constructs and graphical symbols.	VL	20, 55
Simplicity				
sp61	Minimize hard mental operations	Define a language such that it decreases the mental operations of users. Hard mental operations are a further barrier for readers of a diagram who are unfamiliar with the language in which it is written.	SRS, VPL	5, 6, 21, 22
sp62	Minimize syntax complexity	Make sure that the syntax rules of a language are understandable.	IS, DM	82
sp63	Avoid inefficient language elements	Avoid elements which would lead to inefficient diagram already during language design so that only the language user is able to introduce inefficiency.	DSL	29
sp64	Ensure minimality	Avoid elements that are not desired by the user and every aspect of the requirements appears only once.	CM, DM	4, 8, 9, 10, 78
sp65	Harmonize element types	Use as many elements as required to express a meaning. Some languages use a lot of symbols or a lot of space to achieve the results that other languages achieve more compactly.	VPL	5, 22
sp66	Support user-based symbol selection	Make sure the language give users the option to use the symbols they prefer.	GUI	27
Consistency				
sp67	Maximize external consistency	Make sure the elements within a diagram are consistent across various diagrams.	DM, SRS	4, 13
sp68	Maximize internal consistency	Ensure that the various elements within a language and a diagram are consistent.	DM, SRS	4, 13
sp69	Ensure consistent level of abstraction and detail	Consistently decompose one function into a set of subfunctions at the next lower level of refinement. All of the decomposed subfunctions should be at the same abstraction at a particular level.	IS	82

ID	Name	Description	Domain	Source
Clarity				
sp70	Support reliable diagrams	Define a language such that it supports the production of reliable diagrams. A reliable diagram is one that has the capability to maintain a specified level of performance when used under specified conditions.	CM, PM	24, 25, 58
sp71	Minimize error-proneness	Define a language such that no mistakes occur. When the language is used, "careless mistakes" should not be induced.	VPL	5, 22
sp72	Maximize semantic standardization	Make sure the language's defined semantics is standardized.	OM	76
sp73	Maximize interpretation	Ensure that the semantics assigned to each piece of syntax are independent of context.	Di	73
Language Features				
sp74	Enable redundant recoding	Define a language such that it enables expressing information in a diagram in more than one way.	IS, SRS	6, 77
sp75	Enable coexistence of several views	Define a language such that it enables the coexistence of several views of a domain and assure the various views are mutually consistent. Each view should support cognitive economy and inference.	CM, PM	3, 60
sp76	Enable multiple perspectives	Provide heterogeneous diagram types e.g., for representing and visualizing different perspectives. Multiple perspective languages offer different tools for representing more than one view on the same set of entities.	VL	18
sp77	Enable different representation	Define a language such that it would enable representing both technical and non-technical activities.	PM	1
sp78	Include visual routines	Ensure that the language provides the ability to shift attention from one location in the diagram to another. Users first find the most easily detected, simplest, and most prominent symbol in a diagram.	Di, SRS, VPL	6, 7, 22, 81
sp79	Enable sequence representation	Define a language such that it offers the opportunity to represent sequences.	CM	75
sp80	Enable parallelism	Provide mechanisms for specifying concurrent activities and synchronizing their evolution, because software processes are intrinsically concurrent.	PM	1
sp81	Include generalization	Allow for the creation of new object types by uniting existing object types. Generalization is to be applied when different object types play identical roles in fact types.	CM, SRS, DSL, PM	1, 18, 23, 75
sp82	Include abstraction	Provide types of abstraction and structuring mechanisms. An abstraction mechanism allows one to focus on the important aspects of a system while irrelevant details remain hidden.	SRS, DSL, VPL, PM	1, 5, 6, 16, 22, 23
sp83	Use modularity	Provide the possibility of structuring a specification or program in many logically independent units, called modules. The construction of sub-diagrams using some standard pattern or plan, thus reducing the number of linkages that a reader needs to consider.	DM, SRS, DSL, VL, PM	1, 3, 14, 15, 16, 18, 29, 70
sp84	Include granularity	Define a language and create a diagram such that different granularity can be efficiently managed.	DM, PM	1, 12, 42
sp85	Include decomposition	Define a language such that it would enable to break down complex scenarios into smaller, manageable diagrams.	IS, DM	15, 82
sp86	Include hierarchy	Define a language such that it offers composition and decomposition in hierarchical fashion. A diagram should express a clear structure in terms of semantic and visual hierarchy. A diagram is traceable iff it is written in a manner that facilitates the referencing of each individual statement.	SRS, DSL, VL, PM	15, 17, 18, 29, 32, 76
sp87	Use specialization	Represent one or more possibly overlapping subtypes of an object type.	CM	75
sp88	Include relevant properties	Include every relevant property of every instance in at least one class. Include each relevant property possessed by all instances of a class in the class definition.	CM	59, 60, 62
sp89	Maximize covering	Ensure that every instance is a member of at least one class at every time. Include every inference about properties of instances in at least one class.	CM	60, 62
sp90	Invoke data categories	Associate each entity type with one or more predefined generic categories. Category-specific functionality is invoked at run time for each entity type.	IS	48

ID	Name	Description	Domain	Source
sp91	Ensure distinct behavior for subclasses	Define a subclass when instances have the same properties as those of the superclass, but different behavior. When distinct behavior is used to define subclasses, instances of the subclass must possess additional structural or relational properties of interest which are not shared by other instances of the superclass.	IM	59
sp92	Use value restriction for subclasses	Use value restriction to define subclasses only when instances of the subclass possess additional properties of interest which are not shared by other instances of the superclass.	IS, IM	59, 61
sp93	Intersect extensions of existing classes	Probe for additional properties associated with the subclass when a user identifies a subclass with more than one superclass. If none can be identified, the class is unnecessary.	IS, IM	59, 61
sp94	Ensure class inclusion	Include every potential class in some class structure. Inclusion ensures that every concept of interest can, in principle, be included in a diagram that supports cognitive principles.	CM	60
sp95	Abstract from instances	Define a class only if there are instances in the relevant universe possessing all properties defining the class. For every proper class, a subset of the properties is sufficient to identify class membership.	IS, CM, IM, PM	59, 60, 61

Table D5. Task Processing

ID	Name	Description	Domain	Source
Diagram-User Fit				
tp1	Clarify user objectives	Ensure user objectives are defined in relation to a diagram. Ensure a diagram is appropriate for specified user objectives. A diagram needs to be usable in place of an original with respect to some purpose.	IS, PM, EM, MM	35, 39, 41, 82
tp2	Maximize computational offloading	Ensure a diagram reduces the amount of cognitive effort needed to solve the problem by providing the means for direct perceptual recognition of important elements in it.	SRS	6, 7
tp3	Ensure cognitive fit	Use different visual dialects for different tasks and audiences.	CM, VL	20, 55
Diagram-Goal Fit				
tp4	Define right level of diagram detail	Ensure a diagram is specific enough so that any system built that satisfies the requirements in the diagram satisfies all user needs, and abstract enough so that all systems that satisfy all user needs also satisfy all requirements. This is a function of how the diagram is being used.	SRS	13
tp5	Maximize seamlessness	Map abstractions in the problem space to implementations in the solution space without changing notation, thereby avoid the impedance mismatches that often arise throughout the development process.	CM	58
tp6	Balance diagram objectives	Ensure a diagram provides the best balance among the possibly conflicting objectives.	DM	71
tp7	Maximize diagram productivity	Ensure a diagram enables users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.	PM	33
tp8	Maximize learnability	Make sure a diagram can be used by specified users to achieve specified goals when learning to use it.	PM	25, 70
tp9	Define diagram scope	Find the appropriate scope and focus while creating a diagram because modeling in itself is not an objective. A diagram serves a purpose to answer some particular questions. A diagram is pertinent to satisfy intended applications.	CM, DM, DSL, EM	20, 29, 41, 42, 50
tp10	Achieve diagram completeness	Ensure a diagram contains all the statements which would be correct and relevant about the problem domain.	IS, CM, DM, IM, SRS, PM, EM	5, 10, 13, 16, 20, 30, 33, 34, 35, 37, 38, 40, 41, 43, 50, 51, 52, 53, 54, 57, 59, 60, 62, 70, 68, 69, 77, 79, 82
tp11	Achieve diagram feasibility	Stop modeling when the diagram has reached a state where further modeling is regarded less beneficial than that by accepting the diagram in its current state. Make diagrams as correct and complete as needed.	CM, DM, IM, PM, EM	2, 20, 33, 34, 35, 38, 40, 41, 47, 51, 52, 68
tp12	Specify information objects	Consider which specific information objects need to be included in the diagram such that it will be appropriate.	IM	68
tp13	Use relevant elements	Ensure a diagram does not include elements without relevance. Irrelevant elements can be eliminated without loss of meaning for the diagram user.	CM, PM	2, 20, 40
tp14	Maximize diagram validity	Define all statements in the diagram such that they are correct and relevant to the problem.	CM, DM, PM, EM, MM	20, 30, 33, 34, 35, 37, 38, 39, 41, 42, 43
Language Appropriateness				
tp15	Define objectives	Ensure a language helps to satisfy the goals and objectives for the resulting diagrams. A given language type will be better suited for achieving some modeling objectives than others.	PM	12
tp16	Maximize language functionality	Ensure a language is generally applicable rather than restricted to a specific field of application. This is a function of how well a language performs its objective.	IS, VPL, PM	31, 48, 76

ID	Name	Description	Domain	Source
tp17	Maximize ease of use	Make sure the users can easily learn and use the language and conveniently use the modeling procedure.	PM	76
tp18	Maximize usability	Ensure a language is usable. It relates to the degree of difficulty faced by an analyst applying a language. Create a diagram such that it can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use.	IS, DM, DSL, PM	30, 32, 70, 76, 82
tp19	Identify language uses	Identify language uses early. The language defined will be used for at least one task. An early identification of the language uses (before its development) has strong influence on the concepts of under specification.	DSL	29