Abstract. The Business Process Model and Notation (BPMN) has recently emerged as an international standard for modelling business processes. This is one of the most important developments in the history of the IS field, which has the potential to unify the business process management (BPM) field in the same way UML has unified software engineering. Like most business process modelling notations, BPMN is a visual language: it uses diagrams as the primary means for documenting, analysing, communicating and designing business processes. This paper conducts a systematic analysis of the BPMN 2.0 visual notation using a theory for visual notation design (the Physics of Notations). The analysis reveals some serious flaws in the BPMN visual notation, which represent potential barriers to its usability and effectiveness in practice, particularly for communicating with end users. The conclusion from our analysis is that radical surgery is required for BPMN 2.0 to achieve its goal of providing a common language for communicating between business and technical experts. A broader goal of this paper is to raise awareness about the importance of visual representation in IS modelling, which has historically received little scientific attention. Finally, the paper demonstrates how a design theory can be used to improve IS practice.

1. Introduction

1.1 Visual Representation: An Important but Neglected Issue

Visual notations\(^1\) play a critical role in the IS field, and have dominated both research and practice from its earliest beginnings: they are used in all areas and all levels of IS practice, from strategic planning to integrated circuit design. They play a particularly critical role in communicating with business stakeholders, as diagrams are believed to convey information more effectively to non-technical people than text [1].

Yet historically, visual representation issues have received so little attention in IS research. IS modelling notations are typically evaluated exclusively on their semantics, with issues of visual syntax rarely mentioned [e.g. 2, 3]. When notations are empirically compared, differences are generally attributed to semantic rather than syntactic differences, even though differences in visual representation are often just as great. Visual representation thus acts as a significant (but unrecognised) confounding in such studies.

\(^1\) Also called graphical notations, diagramming notations or visual languages
One possible reason for the lack of attention to visual representation is that methods for analysing visual syntax are less mature than those available for analysing semantics [4-6]. However, another explanation is that researchers consider visual syntax to be unimportant: a matter of “aesthetics” rather than effectiveness [7]. This view is contradicted by research in diagrammatic reasoning, which shows that the form of problem representations has a comparatively greater effect on human understanding and problem solving performance than their content [8-11]. Empirical studies in IS contexts confirm this: the graphical form of notations significantly affects understanding, especially by novices [7, 12-18]. This suggests that issues of visual representation (form) should be treated with as much care, if not more, as issues of semantics (content).

1.2 BPMN: An International Standard for Business Process Modelling

The Business Process Model and Notation (BPMN) has recently emerged as an international standard language for business process modelling. It was developed by an industry consortium (BPMI) in response to a call for standardisation in the business process management (BPM) field. Its aim was to provide a common language for modelling business processes, replacing the multiple competing notations that existed. The first version of BPMN (1.0) was developed in 2004 and adopted as an international standard by the Object Management Group (OMG) in 2006. By the end of 2008, it had reached a “tipping point” of mass adoption [19]. Minor updates to the language were made in 2007 (1.1) and 2008 (1.2), with the first major update (2.0) in 2011 [20]. BPMN 2.0 represents a significant advance on previous versions in both complexity and power: the specification has almost doubled in size (from 308 to 538 pages) since the first version.

BPMN is one of the most important developments in the history of the BPM field for at least 3 reasons:

- Standardisation: It is the first ever international standard for business process modelling, so has the potential to unify the field in the same way UML has done for software engineering.
- Adoption in practice: It is backed by some of the leading players in the IT industry (e.g. IBM, SAP, Oracle) and has achieved unprecedented levels of adoption in practice since its release: no other notation has had such an uptake in such a short time [21].
- Executability: One of its unique features (new to BPMN 2.0) is that it has formal execution semantics [19], meaning that models can be automatically executed by process engines: even UML has not achieved this after more than a decade of trying. This fulfils the vision of model driven development (MDD)² and has the potential to revolutionise development practices in the BPM field.

Like most business process modelling notations, BPMN is a visual notation: it uses diagrams as the primary means for documenting, analysing, designing and communicating business processes. However, it is even more visual than most: almost everything in the language can be expressed in graphical form, which seems to be a unifying design principle. BPMN tries to make the semantics of process models unambiguous from diagrams alone [19].

BPMN consists of 4 diagram types (called FLOW ELEMENT CONTAINERS in the BPMN metamodel):

---

² Model driven development (MDD) is a development paradigm in which the system implementation is generated directly from models.
1. **Process Diagrams**: these define internal workflows that occur within business processes (called *orchestration* in BPMN).

2. **Collaboration Diagrams**: these define communication (message flows) among participants: how business processes interact with their environment.

3. **Choreography Diagrams**: these define communication protocols among participants and consist of sequences of message exchanges.

4. **Conversation Diagram**: this is a simplified version of a Collaboration Diagram, with individual messages grouped into conversations.

The last two are new diagram types introduced in BPMN 2.0.

---

**Figure 1. BPMN consists of 4 diagram types**

---

1.3 **How BPMN was developed**

The development of BPMN was a monumental effort, involving more than 120 people, 140 meetings over 6 years [21]. Like UML, it was developed as a result of synthesising existing notations:

“The membership of the OMG has brought forth expertise and experience with many existing notations and has sought to consolidate the best ideas from these divergent notations into a single standard notation. Examples of other notations or methodologies that were reviewed are UML Activity Diagram, UML EDOC Business Processes, IDEF, ebXML BPSS, Activity-Decision Flow (ADF) Diagram, RosettaNet, LOVeM, and Event-Process Chains (EPCs).”

There is no explanation of how the “best” ideas were determined, but seems to have been based on **social consensus** (i.e. majority vote) [22]. While it might be democratic to choose graphical conventions in this way, it is not scientific: choices should be made based on weight of evidence about the efficacy of alternative representations rather than weight of opinions.

**Design rationale** is the process of documenting design decisions made and the reasons *why* they were made. This provides traceability in the design process and helps justify the
final design [23]. Such rationale is conspicuously absent in the design of most IS visual notations: symbols are typically defined without any explanation of why they were chosen, making the decision making process opaque. As Hitchman [7] says:

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

The BPMN 2.0 specification follows the same pattern. A striking thing when reading through the voluminous document (538 pages) is the complete absence of design rationale, even when the choice of symbols seems counterintuitive.

1.4 Objectives of this paper

This paper conducts a systematic, symbol by symbol analysis of the BPMN 2.0 visual notation, the first so far conducted. It is timely for such an analysis to take place, with the BPMN 2.0 specification only finalised earlier this year. The scope of the analysis is limited to the visual syntax of the notation: the graphical representation of constructs. Issues relating to the metamodel (e.g. choice of semantic constructs and relationships among them) or formal semantics are specifically excluded. This paper complements semantic analyses of BPMN [e.g. 24] that have been previously conducted.

Given the widespread adoption of BPMN in practice, this research has high practical relevance: visual representation is a major determinant of the usability and effectiveness of any IS modelling notation, but most of the debate about BPMN has so far focused on semantic rather than syntactic issues. Independent analyses of this kind perform an important role in informing notation designers of the strengths and weaknesses of notations and how they can be improved. In BPMN (like most IS visual notations), visual representation decisions are typically made in an arbitrary and subjective manner, yet are rarely questioned.

Our aim in conducting this analysis is constructive. We are not trying to undermine BPMN or to propose a rival notation but to improve its effectiveness in practice, especially for communicating with business stakeholders. We believe that BPMN is an important development in the BPM field, which has the potential to unify the field in the same way UML has unified software engineering. In this spirit, rather than simply pointing out problems, where possible, we suggest ways of resolving them.

The broader research goal is to raise awareness about the importance of visual representation issues in IS research, which have historically been ignored or undervalued [7]. Visual syntax has a profound effect on the usability and effectiveness of IS notations, equal to (if not greater than) semantic aspects. For this reason, it deserves at least equal attention in analysing, comparing and designing IS notations.

IS visual notation design currently corresponds to what Alexander [25] calls an unselfconscious design culture: one that is not based on explicit design principles but on instinct, imitation and tradition. The broader practical goal of this paper is to change the way IS visual notations are designed to a selfconscious design culture, which is one based on explicit and evidence-based principles.
2. Research Approach

2.1 Ontological analysis: a theory for evaluating semantics of notations

The IS field has developed mature methods for evaluating semantics of notations. In particular, **ontological analysis** has become widely accepted as a way of evaluating IS notation semantics and has been used for this purpose for more than two decades [26-27]. Ontological analyses have been conducted on a wide range of IS notations [e.g. 2, 28-29]. This involves a two-way mapping between the modelling notation and an ontology. According to the theory, there should be a one-to-one correspondence between the concepts in the ontology and constructs in the notation. If not, one or more of the following anomalies will occur (Figure 2):

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

![Figure 2. Ontological Analysis: there should be a 1:1 mapping between ontological concepts and notation constructs](image)

Ontological analysis specifically excludes visual representation aspects: it focuses on content rather than form. Clearly, it would be desirable to have a comparable theory to this at the syntactic level so that visual syntax can also be evaluated in a sound manner.

2.2 The Physics of Notations: A theory for evaluating and designing visual notations

One possible reason for the lack of attention to visual syntax in IS research is the lack of accepted theories and principles for evaluating and designing visual notations. In the absence of such principles, evaluations can only be carried out in a subjective manner. The analysis in this paper is based on a recently-proposed theory of visual notations, called the **Physics of Notations** [30]. This provides a scientific [prescriptive] basis for comparing, evaluating, improving, and constructing visual notations, which has previously been lacking in the IS field. The theory consists of three primary components:

- Dependent variable or design goal
- Descriptive (Type IV) theory
- Prescriptive (Type V) theory
**Dependent Variable or Design Goal**

Visual notations are uniquely human-oriented representations: their primary purpose is to facilitate human communication and problem solving [31]. To be most effective in doing this, they need to be optimised for processing by the human mind. **Cognitive effectiveness** is defined as the speed, ease and accuracy with which a representation can be processed by the human mind [8]. This provides an operational definition of visual notation “goodness” that can be empirically evaluated. This is defined as the primary **dependent variable** for evaluating and comparing visual notations and the primary **design goal** in constructing them.

The cognitive effectiveness of visual notations is one of the most widely accepted and infrequently challenged assumptions in the IS field. However, as Larkin and Simon [8] showed in their seminal paper, “Why a Diagram is (Sometimes) Worth 10,000 Words”, cognitive effectiveness is not an inherent property of visual notations but something that must be designed into them. There can be huge differences between effective and ineffective visual notations and poorly designed notations can be far less effective than text.

**Descriptive (Type IV) Theory: How visual notations communicate**

This defines a theory of how visual notations communicate based on theories from communication, semiotics, graphic design, visual perception and cognition. This represents a **descriptive (positive) theory** of visual notations, or in Gregor’s [32] terminology, a **Type IV theory** (a theory for explaining and predicting). This provides a basis for explaining and predicting why some visual notations will be more effective than others.

**The Design Space (encoding side).** There are 8 elementary **visual variables** that can be used to graphically encode information (Figure 3) [33]. These define the dimensions of the graphic design space. They also define a set of primitives – a **visual alphabet** – for constructing visual notations: any graphical symbol can be defined by specifying particular values for visual variables (e.g. shape = rectangle, colour = green) [34]. Notation designers can create an unlimited number of symbols using different combinations of values of these variables.

**The Solution Space (decoding side).** Designing cognitively effective visual notations is a problem of optimising them for processing by the human mind, in the same way that software systems are optimised for particular hardware. Principles of visual perception and cognition provide the basis for evaluating cognitive effectiveness and making informed choices among the infinite possibilities in the graphic design space.
Prescriptive (Type V) Theory: Principles for designing cognitively effective visual notations

The prescriptive component of the Physics of Notations is a set of nine principles for designing cognitively effective visual notations:

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

The principles were synthesised from theory and empirical evidence about cognitive effectiveness of visual representations from a wide range of fields. All principles represent desirable properties of notations. This means improving a visual notation with respect to any of the principles will increase its cognitive effectiveness, subject to tradeoffs among the principles. The Physics of Notations thus defines a **causal theory**, which posits (positive) causal relationships between each principle and cognitive effectiveness (Figure 4): the principles represent **independent (causal) variables**, while cognitive effectiveness is the sole **dependent (outcome) variable**. All principles are operationalised using evaluation procedures and/or metrics.

![Figure 4. Causal structure of the Physics of Notations (using standard notation for representing scientific theories [35])](image-url)
Together the principles form a **prescriptive (normative) theory** for visual notation design, or in Gregor’s [32] terminology, a **Type V theory**: a theory for design and action. Table 1 shows how the Physics of Notations fits into Gregor and Jones’ [36] template for specifying Type V theories:

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

### 2.3 Research Methodology

The 9 principles for visual notation design were used to conduct a systematic, symbol-by-symbol analysis of the BPMN visual notation. The findings for each principle are described in a separate section (Sections 3-11). The analysis for each principle is structured as follows:

- Definition of principle: what is it, why is it important and how it is evaluated
- Results of evaluation: how well BPMN satisfies this principle
- Recommendations for improvement: how deficiencies can be resolved and the notation improved on this principle

Cross-references between principles are indicated by underlining; new concepts or terms by **bolding**; and BPMN constructs by **SMALL CAPITALS**.

### 3. Semiotic Clarity

#### 3.1 Definition of Principle

The Principle of Semiotic Clarity states that there should be a one-to-one correspondence between semantic constructs and graphical symbols. This is necessary to satisfy the requirements of a **notational system**, as defined by Goodman’s **theory of symbols** [37]. When there is not a 1:1 correspondence, one or more of the following anomalies can occur (using analogous terms to those used in **ontological analysis** [38]) (Figure 5):
• **Symbol deficit**: when a construct is not represented by any symbol
• **Symbol redundancy**: when a single construct is represented by multiple symbols
• **Symbol overload**: when the same symbol is used to represent multiple constructs
• **Symbol excess**: when a symbol does not represent any construct.

This principle represents the extension of ontological analysis to the syntactic level.

![Figure 5. Principle of Semiotic Clarity: there should be a 1:1 correspondence between semantic constructs and graphical symbols](image)

### 3.2 Results of Evaluation

Evaluating semiotic clarity involves a two-way mapping between a notation’s metamodel (which defines its semantic constructs) and its symbol set (visual vocabulary). There are 98 semantic constructs in BPMN, which defines the **semantic complexity** of the notation. There are 177 symbols in the BPMN visual vocabulary, which defines the **graphic complexity** of the notation. This results in a **net symbol balance** of +79. The results of the semiotic clarity analysis are summarised in Table 2.

<table>
<thead>
<tr>
<th>Table 2. Semiotic Clarity Analysis Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructs = 98</td>
</tr>
<tr>
<td>Net symbol balance = +79</td>
</tr>
<tr>
<td><strong>Anomaly type</strong></td>
</tr>
<tr>
<td>Symbol redundancy</td>
</tr>
<tr>
<td>Symbol overload</td>
</tr>
<tr>
<td>Symbol excess</td>
</tr>
<tr>
<td>Symbol deficit</td>
</tr>
</tbody>
</table>

**Symbol redundancy (synographs)**

There are 7 instances of symbol redundancy in BPMN, summarised in Figure 6. These are called **synographs**, as they are the graphical equivalent of synonyms in textual languages. Synographs place a burden of choice on the notation user to decide which symbol to use and additional cognitive load on the reader to remember multiple representations of the same construct. They also unnecessarily add to graphic complexity (**Graphic Economy**).

• EXCLUSIVE (XOR) GATEWAYS may be represented using two alternative symbols. No explanation is given for why a choice is provided and may be the result of **decision paralysis**, where the designers were unable to agree on which symbol to use. As we will see, the Physics of Notations provides clear (and evidence-based) rules for choosing between these alternatives (see **Perceptual Discriminability**, **Semantic Transparency** and **Dual Coding**).
• MESSAGE FLOWS may be shown with or without a message icon. Including the icon allows additional information to be graphically expressed (shading indicates that it is a reply to a previous message).
• SUBPROCESSES and SUBCHOREOGRAPHIES may be shown as collapsed (hiding their internal details) or expanded (showing their internal details).
• POOLS and LANES may be oriented either vertically or horizontally. This changes one of the visual variables (orientation), resulting in visually distinct symbols. This could result in misinterpretation, as changing the orientation of other BPMN symbols changes their meaning (e.g. MULTI-INSTANCE LOOPS, CANCEL EVENTS, EXCLUSIVE GATEWAYS).

<table>
<thead>
<tr>
<th>Construct</th>
<th>Symbol</th>
<th>Alternative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclusive (XOR) Gateway</td>
<td>![Symbol]</td>
<td>![Alternative]</td>
</tr>
<tr>
<td>Message Flow</td>
<td>![Message Flow]</td>
<td>![Message Flow]</td>
</tr>
<tr>
<td>Subprocess</td>
<td>![Collapsed Subprocess]</td>
<td>![Expanded Subprocess]</td>
</tr>
<tr>
<td>Subchoreography</td>
<td>![Collapsed Subchoreography]</td>
<td>![Expanded Subchoreography]</td>
</tr>
<tr>
<td>Pools and lanes</td>
<td>![Pools and lanes]</td>
<td>![Pools and lanes]</td>
</tr>
</tbody>
</table>

Figure 6. Symbol redundancy (synographs) in BPMN

In all cases, the choice of which symbol to use in all cases is left up to the modeller: there is no “default” representation. There is also no design rationale for providing alternative symbols or guidelines for choosing between them based on the context.

Symbol overload (homographs)
Symbol overload is the worst type of semiotic clarity violation as it results in ambiguity and the potential for misinterpretation [37]. Symbol overload is a common response to graphic complexity: when designers run out of ideas for new symbols, they often try to get existing symbols to do extra work. There are 5 instances of symbol overload in BPMN, summarised in Figure 7. These are called homographs, as they are the graphical equivalent of homonyms in textual languages in BPMN.
An Analysis of the BPMN 2.0 Visual Notation

### Table 1: Graphical Symbol Overload in BPMN

<table>
<thead>
<tr>
<th>Graphical Symbol</th>
<th>Alternative Meanings</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Data Association" /></td>
<td>Data Association</td>
</tr>
<tr>
<td><img src="image" alt="Association" /></td>
<td>Association</td>
</tr>
</tbody>
</table>

| ![Parallel Multi-Instance Loop (Activity)](image) | Parallel Multi-Instance Loop (Activity) |
| ![Collection (Data Object)](image) | Collection (Data Object) |
| ![ParticipantMultiplicity](image) | Participant Multiplicity (Pool) |

| ![Spatial enclosure (A encloses B)](image) | Spatial enclosure (A encloses B) |
| ![Responsibility (Pools)](image) | Responsibility (Pools) |
| ![Categorisation (Lanes, Groups)](image) | Categorisation (Lanes, Groups) |
| ![Composition (Subprocesses, Subchoreographies)](image) | Composition (Subprocesses, Subchoreographies) |

Figure 7. Symbol overload (homographs) in BPMN (maybe add resolution refs?)

All of these represent **contextual differentiation**: where the meaning of the symbol differs depending on where it appears. For example, the multi-instance marker means different things depending on whether it appears inside an **ACTIVITY**, **CHOREOGRAPHY ACTIVITY**, **PARTICIPANT** or **DATA OBJECT**. Contextual differentiation violates one of the fundamental properties of the symbol system of graphics: **monosemey**, which means that all symbols should have a single meaning, defined in advance and independent of context [33].

A more subtle case of symbol overload occurs at the “subatomic” level. BPMN is unusual among IS modelling notations in that it builds composite symbols from smaller sub-elements (**graphemes**). There are several cases where symbols contain the same grapheme, even though their meanings are different or unrelated. These represent **partial homographs**, and are shown in Figure 8:

<table>
<thead>
<tr>
<th>Grapheme</th>
<th>Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Cross" /></td>
<td><img src="image" alt="Cancel" />, <img src="image" alt="Exclusive (XOR) Gateway" /></td>
</tr>
<tr>
<td><img src="image" alt="Plus sign" /></td>
<td><img src="image" alt="Parallel (AND) Gateway" />, <img src="image" alt="Collapsed Marker" /></td>
</tr>
<tr>
<td><img src="image" alt="Bold circle" /></td>
<td><img src="image" alt="None" />, <img src="image" alt="Inclusive (OR) Gateway" /></td>
</tr>
</tbody>
</table>

Figure 8. Partial homographs in BPMN
Partial homographs can lead to misinterpretation as people may read meaning into the similarity/overlap/intersection between symbols. In testing comprehension of public information symbols, Zwaga and Boersema [39] found that symbols that shared common elements were frequently confused.

**Symbol excess (visual noise)**

Symbol excess refers to symbols that have no semantic content: these correspond to “noise” or “filler” words in natural language (e.g. “um”, “ah”). There are two instances of symbol excess in BPMN 2.0: CATCHING and THROWING LINK EVENTS, which function as intra-page and inter-page connectors. These are modelled in the BPMN metamodel as INTERMEDIATE EVENTS but this is a misconceptualisation as they have no semantic content and act purely as navigational aids within and between diagrams.

![Image](image-url)

**Figure 9. Link events are page connectors and should not be modelled as events**

Also, using the same symbol for on-page and off-page connectors is perceptually ambiguous: when the reader sees a THROWING LINK EVENT, they must visually scan for a matching CATCHING LINK EVENT, which may be on the same or a different page. Traditional flowcharts avoid such confusion by using clearly different symbols for on-page connectors, off-page connectors and events (Figure 10).

![Image](image-url)

**Figure 10. Inter-page and intra-page connectors are clearly differentiated from each other and from events in traditional flowcharts**

**Symbol deficit (visual silence)**

In most IS modelling notations, a certain level of symbol deficit is desirable to limit diagrammatic complexity (Complexity Management) and graphic complexity (Graphic Economy) [30]. This represents a clear difference with ontological analysis where all deviations from a 1:1 mapping are considered harmful [27]. There is a relatively low level of symbol deficit in BPMN, given its semantic complexity.

### 3.3 Recommendations for Improving Semiotic Clarity

To improve semiotic clarity of BPMN, all instances of symbol redundancy, overload and excess should be removed. Symbol deficit needs to be increased as it is one of the most effective ways of reducing graphic complexity (which is one of BPMN’s most serious problems: see Graphic Economy).

**Remove synographs**

Symbol redundancy can be resolved by choosing one of the alternative symbols used to represent the construct as the standard (and only) representation and removing the other symbol(s) from the notation. Later principles (e.g. Perceptual Discriminability, Semantic Trans-
parency, Dual Coding) provide clear guidelines for resolving synographs (e.g. see Figure 20, Figure 29 and Figure 65 for resolution of the first synograph).

**Resolve homographs**
Symbol overload can be resolved by creating visually distinct symbols to represent each of the possible meanings of the homograph. Later principles (e.g. Perceptual Discriminability, Semantic Transparency) provide guidance for resolving homographs (e.g. see Figure 60 for resolution of the first synograph).

**Remove symbol excess**
Symbol excess can be resolved by simply removing the unnecessary symbols. Removal of LINK INTERMEDIATE EVENTS is discussed in Complexity Management.

**Increase symbol deficit**
Possible ways of increasing symbol deficit and reducing graphic complexity are discussed in Graphic Economy.

4. **Perceptual Discriminability**

4.1 **Definition of Principle**

Perceptual Discriminability refers to the ease and accuracy with which symbols can be differentiated from one another other. Accurate discrimination between symbols is a prerequisite for accurate interpretation of diagrams [6]. Discriminability requirements are much higher for novices than for experts as we are able to make much finer distinctions with practice [40].

**Perceptual Discriminability Metric: Visual Distance**

Discriminability is primarily determined by the visual distance between symbols, which is measured by the number of visual variables on which they differ and the size of these differences. In general, the greater the visual distance between symbols, the faster and more accurately they will be recognised [41]. If differences between symbols are too subtle, errors in interpretation are likely, especially by novices.

**Perceptual Popout**
According to feature integration theory, visual elements with unique values for at least one visual variable can be detected pre-attentively and in parallel across the visual field [42-43]. Such elements appear to “pop out” from a display without conscious effort. In contrast, visual elements that are differentiated by unique combinations of values (conjunctions) require serial search, which is much slower and error-prone. The implication of this for visual notation design is that each graphical symbol should have a unique value on at least one visual variable.

**Visual-Semantic Congruence**
In general, the visual distance between symbols should be consistent with the semantic distance between the constructs they represent: constructs with divergent meanings should be represented using clearly different symbols, while similar constructs should have similar symbols [5].
4.2 Results of Evaluation

On the positive side, BPMN relies exclusively on graphical means to discriminate between symbols. Many modern visual notations (e.g. UML [44], i* [45]) use text to differentiate between symbols, which is cognitively ineffective and reduces suitability for communication across international boundaries (as different symbol sets must be defined for different languages).

**Graphic complexity effects**

Many of the discriminability problems in BPMN stem from the sheer number of symbols in the notation: its **graphic complexity** (see Graphic Economy). BPMN has an order of magnitude more symbols than any existing process modelling notation, which makes the task of designing discriminable symbols much more difficult (which is why Graphic Economy has a positive relationship on Perceptual Discriminability: see Error! Reference source not found.). The more symbols there are in total, the more closely they will be packed together in the graphic design space and the mean visual distance between symbols will be reduced.

**Perceptual Popout Analysis: Events**

**EVENTS** present serious discriminability problems due to their size (they are the smallest symbols on BPMN diagrams) and numerosity. They also illustrate the difficulty of achieving perceptual popout in the face of graphic complexity: 3 visual variables (**shape**, **brightness**, and **texture**) are used in combinatorial fashion to distinguish between 51 different types of events [63 with duplicates]. Discrimination between symbols relies on unique combinations of values with only one exception: the **TERMINATE END EVENT** is the only symbol with a unique value for any variable (icon = circle). However it is questionable whether even this exhibits perceptual popout as it is virtually indistinguishable from the **PARALLEL MULTIPLE END EVENT** due to its size (Figure 11).

**Visual Semantic Congruence: Groups and Event Subprocesses**

An example of a visual-semantic congruence problem in BPMN is the similarity between **GROUPS** and **EVENT SUBPROCESSES**. These symbols are the same shape (rountangles), are used
to enclose other diagram elements and are distinguished by a very subtle difference in border style (dotted vs dashed line). While not identical, they are visually very similar. Their visual distance is 1 (a single step on a single visual variable: texture) while their semantic distance is 6 (as measured by the shortest path between them in the metaclass inheritance hierarchy [46]), which represents a problem of visual-semantic congruence (Figure 12).

<table>
<thead>
<tr>
<th>Visual distance = 1: symbols differ by one perceptible step on a single visual variable (texture)</th>
<th>Semantic distance = 6: extract from metamodel showing shortest path between constructs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Subprocess (expanded)</td>
<td>Base element</td>
</tr>
<tr>
<td>Group</td>
<td>1</td>
</tr>
<tr>
<td>Subprocess</td>
<td>2</td>
</tr>
<tr>
<td>Activity</td>
<td>3</td>
</tr>
<tr>
<td>Flow node</td>
<td>4</td>
</tr>
<tr>
<td>Flow element</td>
<td>5</td>
</tr>
<tr>
<td>Artifact</td>
<td>6</td>
</tr>
<tr>
<td>Event Subprocess = Subprocess where Triggered</td>
<td></td>
</tr>
</tbody>
</table>

**Parallel and Exclusive Gateways**

The symbols used for exclusive (XOR) and parallel (AND) gateways are very similar even though they have very different, even diametrically opposed, semantics:

- An exclusive (XOR) gateway defines **conditional flows** and corresponds to a decision: only one of the outgoing sequence flows are taken, depending on the result of the decision. These represent dependent (alternative) paths.
- A parallel (AND) gateway defines **unconditional flows** and corresponds to parallel processing: all of the outgoing sequence flows are taken at the same time. These represent independent paths.

There is only a very small visual distance between the symbols used to represent these constructs (Figure 13): they use the same basic shape (diamond) and the same internal marker (cross) with a different orientation (45°). This represents a single perceptible step on a single variable, as 45° is the minimum perceptible difference in orientation [33]. Again, the symbols are not identical but are very similar and could easily be confused.

**Figure 13. The symbols for Exclusive and Parallel Gateways are very similar, even though they have very different semantics**

**Cancel and Parallel Multiple Events**

As in the previous case, the cancel event and parallel multiple event use the same configuration of symbols but with a different orientation (45°) of the internal marker (Figure 14). The similarity between the symbols is misleading (and most likely coincidental) as their semantics is unrelated. As in the previous case, these symbols could be easily confused.
**Pools and Lanes**

The symbols used for POOLS and LANES are almost identical: the only difference is that POOLS have lines separating their labels from the rest of the shape, which is barely perceptible. However, they have very different business semantics: POOLS represent PARTICIPANTS while LANES can be any way of categorising diagram elements. They also have very different grammatical rules: POOLS can only appear on COLLABORATION DIAGRAMS, while LANES can appear on any diagram except CHOREOGRAPHY DIAGRAMS. Also, SEQUENCE FLOWS can cross LANE boundaries but not POOL boundaries, while MESSAGE FLOWS must cross POOL boundaries.

**Discriminability of icons**

A lot of effort in BPMN has been spent on designing icons to distinguish between different constructs. In general, using icons is an excellent notation design strategy, as they improve understanding by naïve users (see Semantic Transparency). However, beyond a certain point, it becomes counterproductive: when there are too many icons, there is a swamping effect and discriminability decreases sharply [18]. This is a well-known problem in designing graphical user interfaces: if there are too many icons to easily remember, learnability and usability are reduced [47]. There are a total of 41 icons in BPMN (Figure 16), which is an extraordinary number to remember.

<table>
<thead>
<tr>
<th>Event icons (20)</th>
<th>Activity icons (6)</th>
<th>Task icons (8)</th>
<th>Gateway icons (7)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Event icons" /></td>
<td><img src="image" alt="Activity icons" /></td>
<td><img src="image" alt="Task icons" /></td>
<td><img src="image" alt="Gateway icons" /></td>
</tr>
</tbody>
</table>

Figure 16. Icons present discriminability problems due to their size and numerosity
Another problem is that some icons look very similar to one another (Figure 17). Discriminability is far worse in the case of THROWING EVENTS (where icons are shown in black), which often look like indistinguishable “blobs” on diagrams.

Perhaps the most serious problem with BPMN icons is their size. Icons are the smallest graphical elements on BPMN diagrams and therefore the “weakest link” in terms of discriminability: the ability to distinguish between visual elements decreases with their size [33, 48-49]. When diagrams become complex, icons quickly reach “vanishing point” as can be seen in Figure 18 (from the BPMN 2.0 specification [50]). While it is possible to distinguish between the basic shapes, it is difficult if not impossible to identify the embedded icons.

Even when printed at 100% scale as in Figure 19, icons are around 4 mm² for TASKS, EVENTS and GATEWAYS and around 1 mm² for EVENT SUBPROCESSES, which is approaching the limits of human discrimination ability.
4.3 Recommendations for Improving Perceptual Discriminability

**Exclusive Gateways**

The discriminability problem identified between EXCLUSIVE and PARALLEL GATEWAYS (Figure 13) could be resolved by choosing the “no marker” synograph as the standard (and only) symbol for EXCLUSIVE GATEWAYS. As shown in Figure 20, the difference between them is now unmistakeable. This also resolves the problem of symbol redundancy and the partial homograph between EXCLUSIVE GATEWAYS and CANCEL EVENTS (Figure 8).

![Figure 20. Using the “no marker” version of the Exclusive Gateway (left) resolves the discriminability problem with Parallel Gateways and two semiotic clarity issues (symbol redundancy and partial homograph)](image)

**Resolving Symbol Redundancy: Message Flows**

This principle also provides the basis for resolving the problem of symbol redundancy with MESSAGE FLOWS. Including message icons helps to more clearly discriminate MESSAGE FLOWS from other types of flows. While they add slightly to diagrammatic complexity (an additional token for each MESSAGE FLOW), they add little or no cognitive overhead as they are semantically transparent (so require little or no effort to remember). MESSAGE FLOWS are now differentiated from other flows by 4 characteristics: line style (dashed), line beginning (unfilled circle), line end (unfilled triangle) and attached icon (envelope). Three of these are unique among flows, which facilitates perceptual popout.

![Figure 21. Attaching message icons to message flows (left) increases their discriminability from other flows](image)

**Improving Discriminability of Event Types: Redundant Coding**

There are three types of EVENTS in BPMN, depending on where they appear in a PROCESS (START, INTERMEDIATE, END). These are distinguished by different border styles, as shown in Figure 22. These distinctions are rather subtle, especially when the symbols themselves are so small. When diagrams get complex, the distinction between them quickly reaches vanishing point, as can be seen in Figure 19.

![Figure 22. The distinction between different event types is very subtle](image)

A second problem is that exactly the same border styles are used to distinguish between different types of ACTIVITIES: standard ACTIVITIES, TRANSACTIONS and CALL ACTIVITIES (Figure
This means the border styles for EVENT TYPES are not unique, which precludes perceptual popout. The symbols are unique, but represent conjunctions rather than unique values.

<table>
<thead>
<tr>
<th></th>
<th>Solid border</th>
<th>Double border</th>
<th>Bold border</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Events</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intermediate event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Activities</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard activity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transaction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call activity</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 23. The same border styles are used for both events and activities, thus precluding perceptual popout

One way to increase visual distance between symbols is through redundant coding: using additional visual variables to distinguish between them [51]. Redundancy is a useful technique to reduce communication errors and counteract noise [52-53]. For example, colour could be used in addition to border style to differentiate between EVENT TYPES (Figure 24). Colour is one of the most cognitively effective of all visual variables and is the graphic designer’s first choice for distinguishing between categories, especially when speed and accuracy are paramount (e.g. in traffic signs and signals [54]) [55]. Use of colour also supports perceptual popout as each EVENT TYPE now has a unique colour.

Figure 24. Redundant coding: using colour in addition to border style increases discriminability by increasing visual distance and facilitates perceptual popout

Redundant coding is different to using conjunctions to distinguish between symbols: redundant coding uses variables in an additive manner (each variable is sufficient on its own to distinguish between symbols) while conjunctions use visual variables in a multiplicative manner (each variable is necessary but not sufficient).

Perceptual Popout: Improving Discriminability of Activity Types

Alternatively, colour could be used on its own to distinguish between EVENT TYPES as shown in Figure 25. This represents non-redundant coding, which reduces visual distance among event types: they now differ on only a single visual variable (colour). However, it means that the border styles used to distinguish between different types of ACTIVITIES (Figure 23) become unique, thus enabling perceptual popout. This shows how symbol design decisions are interdependent due to the finite size of the graphic design space: reducing the visual distance among EVENT TYPES increases their visual distance from ACTIVITIES.

3 The astute reader will have noticed that colour distinctions will be lost if the diagram printed in black and white: this is discussed in Visual Expressiveness.
5. Semantic Transparency

5.1 Definition of Principle

Semantic transparency involves the use of graphical representations whose appearance suggests their meaning. Literally, it means that the meaning (semantics) of a symbol is clear (transparent) from its appearance alone. While Perceptual Discriminability simply requires that symbols be clearly different from one another, this principle requires that they provide cues to their meaning. This relates to higher level cognitive processing (interpreting meaning of symbols) while Perceptual Discriminability relates to perceptual processing (perceiving differences between symbols). Semantic transparency is a desirable property of symbols but not always achievable (e.g. no-one has yet designed understandable symbols for opening and closing elevator doors [56]).

Semantically transparent symbols reduce cognitive load because they have built-in mnemonics, making it easier for people to learn and remember what they mean [18]. Empirical studies show that semantically transparent representations improve speed and accuracy of understanding by naïve users [12, 40]. Semantic transparency is one of the most powerful tools in the visual notation designer’s bag for improving understanding by novices: more than any other principle, this provides the basis for delivering on BPMN’s promise of being a “business oriented notation” that is “readily understandable by all business users”.

Semantic transparency is not a binary state but a sliding scale (Figure 26).

- At the positive end of the scale, semantic immediacy means that a novice reader would be able to correctly infer the meaning of a symbol from its appearance alone (e.g. a stick figure for a person).
- At the zero point of the scale, semantically opacity means there is a purely arbitrary (conventional) association between a symbol and its meaning (e.g. rectangles for UML classes). Such symbols require conscious effort to remember and must be learnt by rote.
- At the negative end of the scale, semantic perversity means a novice reader would be likely to guess a completely different meaning from the symbol’s appearance. This often occurs when a familiar symbol is used for a different purpose, so its appearance is mis-
leading (false mnemonic). Such symbols require the most effort to learn and remem-
ber, as they require “unlearning” the familiar (natural or intuitive) meaning.

- In between semantic opacity and semantic immediacy there are varying degrees of semantic translucency, where there is some kind of mnemonic association between the symbol and its meaning. A novice may not be able to guess the meaning of such a symbol seeing it for the first time, but once the association is explained, it aids recogni-
tion and recall.

In general, the more semantically transparent a symbol is, the less effort it will require to learn and remember and less likely it will be misinterpreted. The concept of semantic transparency formalises subjective notions like “intuitiveness” or “naturalness” that are often used informally when discussing visual notations\(^4\), as it can be empirically evaluated e.g. by getting novices to guess what symbols mean and measuring the correlation between guesses and cor-
correct answers: 1 indicates semantic immediacy while \(-1\) indicates semantic perversity.

### 5.2 Results of Evaluation

BPMN currently uses a number of semantically transparent symbols, which draw on perceptual resemblance or familiar metaphors: these are summarised in Figure 27. For example, the message icon perceptually resembles an envelope and is commonly used in everyday life to denote messages (e.g. post, email). This is one of the international standard DOT pictograms [57], which are widely used and understood.

![Figure 27. BPMN 2.0 includes a number of semantically transparent symbols](image)

However the vast majority of BPMN symbols are not semantically transparent. For example, all the GATEWAY symbols are highly cryptic (Figure 28). A novice would be unlikely to be able to guess what any of these mean, and if they did, would be likely to guess wrong in at least two cases (as we will see later). Also, there is no design rationale provided for the choice of symbols, which could help in creating a mnemonic association with the underlying construct (e.g. “plus” stands for “parallel”, making use of alliteration); in the absence of this, they must be learnt by rote.

---

\(^4\) The “naturalness” of notations is a contentious issue in IS research, with authors often arguing that particular representations are more “natural” or “intuitive” than others. In most cases, such claims are based on opinion or conjecture. However, at least for visual representations, it is possible to make such claims based on empirical evidence and to empirically validate them.
Semantic Perversity: Exclusive and Parallel gateways

The symbols used to represent EXCLUSIVE (XOR) and PARALLEL (AND) GATEWAYS are semantically perverse, as people familiar with traditional flowcharts or mathematical notation would be likely to transpose their meaning. The PARALLEL GATEWAY marker is similar to that used for XOR junctions in traditional flowcharts and the logical XOR operator in Boolean algebra. Similarly, the EXCLUSIVE GATEWAY marker is the same as that used for AND junctions in traditional flowcharts. No design rationale is provided for why such apparently counterintuitive symbols were chosen.

<table>
<thead>
<tr>
<th></th>
<th>BPMN</th>
<th>Traditional flowcharts</th>
<th>Mathematical notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>XOR</td>
<td>Exclusive (XOR) gateway</td>
<td>OR junction (XOR join)</td>
<td>Logical XOR</td>
</tr>
<tr>
<td>AND</td>
<td>Parallel (AND) gateway</td>
<td>Summing junction (AND join)</td>
<td>Logical AND</td>
</tr>
</tbody>
</table>

Figure 29. BPMN Exclusive and Parallel Gateway markers are semantically perverse

5.3 Recommendations for Improving Semantic Transparency

Semantic Opacity: Event Types

Currently, different border styles are currently used to distinguish between START, INTERMEDIATE and END EVENTS (Figure 30). However there is no natural association between border style and EVENT TYPE. It is a purely conventional relationship (semantically opaque), which requires conscious effort to learn and remember.

Figure 30. Border styles for events are semantically opaque: they do not suggest any particular meaning or sequence

Using colour to differentiate between EVENT TYPES (as proposed in Perceptual Discriminability) is semantically transparent as it draws on the familiar and internationally-understood metaphor of traffic lights. It would be relatively easy for novices to guess (and hence learn and remember) what the symbols mean by association with the “stop-wait-go” sequence of traffic lights.
As an alternative, icons could be used to differentiate between **START**, **INTERMEDIATE** and **END** events, drawing on the metaphor of a CD/DVD player: this represents use of **shape** to encode **EVENT TYPE**. YAWL [58], another workflow modelling technique, uses “play” and “stop” symbols to represent start and end events, which is an excellent example of semantic transparency. This approach could be extended to intermediate events using the “pause” button (Figure 32)\(^5\). It would be relatively easy for novices to learn and remember what these symbols mean as the controls of a CD/DVD player are familiar to most people and are used across geographical boundaries. BPMN already uses the “rewind” symbol to represent COM-\(\text{PENSA}\)TION and the “repeat” symbol to represent a **STANDARD LOOP**, so using these additional icons would provide a consistent **theme** or **metaphor**, which is an effective strategy for designing understandable graphical user interfaces [47, 59].

**Semantic Perversity: Interrupting Events**

Currently, **NON-INTERRUPTING EVENTS** are distinguished from **INTERRUPTING EVENTS** using border style (dashed vs solid). This graphical convention is **semantically perverse** as dashed lines are commonly called “interrupted” lines (e.g. in military terminology [60]), as spaces literally interrupt the line. In addition, this difference can be very difficult to discern

---

\(^5\) The astute reader will have noticed that using icons to distinguish between **EVENT TYPES** is incompatible with the use of icons for **EVENT DEFINITIONS** (e.g. timer, message). However, there is a strong argument that **EVENT DEFINITION** icons should be removed as they add very little information for such a large impost in graphic complexity: see **Graphic Economy** for further discussion of this issue.
with such small elements (e.g. see Figure 18) and precludes perceptual popout as the same border style is used for EVENT SUBPROCESSES (Perceptual Discriminability).

<table>
<thead>
<tr>
<th></th>
<th>Solid border</th>
<th>Dashed border</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupting</td>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
</tr>
<tr>
<td>Non-interrupting</td>
<td><img src="image3" alt="Image" /></td>
<td><img src="image4" alt="Image" /></td>
</tr>
<tr>
<td>Activities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Normal Activity</td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
<tr>
<td>Event Subprocess</td>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
</tr>
</tbody>
</table>

![Figure 33. The same border styles are used with different meanings for events and activities](image9)

A more semantically transparent way to show INTERRUPTING STATUS would be to use a spatial relationship. A NON-INTERRUPTING EVENT could be shown as touching an ACTIVITY, while an INTERRUPTING EVENT could be shown as overlapping it (i.e. literally “interrupting” the boundary). It is more analogical to represent INTERRUPTING STATUS as a spatial relationship between the two symbols than a visual property of an EVENT, as it represents a relationship between an EVENT and an ACTIVITY (i.e. whether the EVENT can interrupt the ACTIVITY) rather than a property of the EVENT itself. This is also more discriminable as the distinction can be seen even when diagrams get complex and supports perceptual popout as this is now the only situation when one symbol overlaps another.

![Figure 34. The existing convention for showing interrupting events (left) is semantically perverse. Using a spatial relationship to show this (right) is more semantically transparent and more discriminable](image10)

Semantic transparency of icons: drawing on international standards

**Icons** are symbols that perceptually resemble the concepts they represent: this reflects a fundamental distinction in semiotics, between symbolic and iconic signs [61]. Iconic representations speed up recognition and recall and improve intelligibility of diagrams to naïve users [12, 40]. They also make diagrams more accessible to novices [18] and support communication across international boundaries [47]. BPMN makes greater use of icons than any previous IS modelling notation (Figure 16), but most are semantically opaque or perverse. To be effective, icons must have clear associations with the concepts they represent [47]. Also, most BPMN icons don’t fit the definition of icons as they don’t resemble any physical object. For example, all of the GATEWAY “icons” shown in Figure 28 are symbols rather than icons in semiotic terms.

A rich source of semantically transparent icons are international standards such as the ISO Standard Public Information Symbols [62], ISO Standard Safety Signs [63], UN Standard Road Signs and Signals [54] and DOT pictograms [57]. These define symbols that are widely...
used and understood across international boundaries, and have been tested for discriminability and understandability (i.e. semantic transparency). The BPMN notation designers seem to have ignored these in designing their icons.

The most numerous of the BPMN icons are the EVENT DEFINITION icons (11 in total). While some are semantically transparent (TIMER, MESSAGE, COMPENSATION, CANCEL), others are opaque or perverse: they have unclear or misleading associations with their referent concepts. In Figure 35, we suggest some more semantically transparent icons, drawing on established international standards:

- **Error Events** are currently represented by a lightning bolt. This is **semantically perverse** as it is the international standard sign for high voltage [63]. A more semantically transparent icon would be the ISO standard warning sign [63]: this is commonly used in computer software to signify errors.
- **Terminate Events** are currently represented by a black circle, which is **semantically opaque**. A possible improvement would be the international sign for dead end, which perceptually resembles a dead end (so is a true icon) [54]. This looks like the letter “T”, which also provides a mnemonic association with Terminate (by alliteration).
- **Escalation Events** are currently represented by something that looks like a Star Trek symbol. This is **semantically opaque**, as it has no obvious association with ESCALATION. A more semantically transparent representation would be the “up escalator” symbol from the DOT pictograms [57]. This is widely understood and recognised, perceptually resembles an escalator, and has a mnemonic (word) association with ESCALATION.
- **Signal Events** are currently represented by a triangle, which is **semantically perverse** as this is the international standard danger warning road sign [54]. A possible improvement would be the international standard sign for radio waves [63].
- **Multiple Events** are currently represented by a pentagon, which looks like the “home” icon in web browsers. A more semantically transparent representation would be multiple EVENT symbols superimposed on one another: literally, “multiple events”.

<table>
<thead>
<tr>
<th>Event</th>
<th>Symbol</th>
<th>Association</th>
<th>Proposed improvement</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td><img src="lightning_bolt.png" alt="Lightning bolt" /></td>
<td>High voltage warning [63]</td>
<td><img src="warning_sign.png" alt="Warning sign" /></td>
<td>ISO general warning sign [63]; also used for computer software errors</td>
</tr>
<tr>
<td>Terminate</td>
<td><img src="circle.png" alt="Circle" /></td>
<td>UN dead end road sign [54]</td>
<td><img src="T_sign.png" alt="T sign" /></td>
<td></td>
</tr>
<tr>
<td>Escalation</td>
<td><img src="star_trek_symbol.png" alt="Star Trek symbol" /></td>
<td>Starfleet symbol</td>
<td><img src="up_escalator.png" alt="Up escalator" /></td>
<td>DOT up escalator symbol [57]</td>
</tr>
<tr>
<td>Signal</td>
<td><img src="triangle.png" alt="Triangle" /></td>
<td>Danger warning sign [54]</td>
<td><img src="radio_waves_sign.png" alt="Radio waves sign" /></td>
<td>ISO radio waves sign [63]</td>
</tr>
<tr>
<td>Multiple</td>
<td><img src="pentagon.png" alt="Pentagon" /></td>
<td>“Home”</td>
<td></td>
<td>BPMN Event symbol</td>
</tr>
</tbody>
</table>

Figure 35. More semantically transparent icons for events
Redesigning these icons in this way would also resolve the discriminability problems among them: these icons are also the ones identified in Perceptual Discriminability as being most similar to one another (see Figure 17).

**Data Objects**

DATA OBJECTS are currently represented as rectangles with folded corners. These could be shown more semantically transparently by including the international symbol for information [62]. In a sign production study, Arning and Ziefle [64] found that this was the most commonly produced symbol for information by naïve participants (the population stereotype [65]), suggesting that this representation would be widely understood.

<table>
<thead>
<tr>
<th>Current Representation</th>
<th>Proposed Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Current Representation" /></td>
<td><img src="image2" alt="Proposed Improvement" /></td>
</tr>
</tbody>
</table>

Figure 36. Data objects could be shown more semantically transparently using the international symbol for information

**Pools and Lanes**

POOLS and LANES draw on the metaphor of swimming pools and lanes. However, currently both are simply shown as rectangles, so don’t take full advantage of the visual metaphor. The association could be strengthened by filling POOLS with “water” (colouring them blue) and representing lanes using dashed lines as they typically are in real life (Error! Reference source not found.). Using dashed lines for LANES is also semantically transparent as it suggests a “permeable” boundary: this can be used to remember that solid lines (SEQUENCE FLOWS) can cross dashed boundaries (LANES) and dashed lines (MESSAGE FLOWS) can cross solid boundaries (POOLS) but not the other way around. This also resolves the discriminability problem between pools and lanes identified in Perceptual Discriminability.

![Figure 37. Left: a real swimming pool with lanes; Right: using border styles to distinguish between POOLS and LANES increases discriminability and semantic transparency. Use of colour strengthens the visual metaphor.](image3)

**Resolving Symbol Redundancy: Orientation of Pools and Lanes**

The BPMN specification allows pools to be oriented horizontally or vertically, and for processes to run in any direction (left to right, top to bottom, right to left or bottom to top):

“The BPMN spec does not ascribe any significance to whether pools and lanes run horizontally or vertically. These are really matters of personal style.”
This is not a matter of personal style but of semantic transparency: direction of flow does not affect the semantics of diagrams but does affect ease of interpretation. Empirical studies show that left to right arrows are naturally interpreted as causation or sequence, while top to bottom arrows are interpreted as taxonomic or compositional relationships [6]. The reason is that the horizontal (x) dimension is naturally interpreted as a surrogate for the time dimension: this accounts for about 75% of all graphics published [66]. For this reason, POOLS and LANES should be oriented horizontally and SEQUENCE FLOWS shown left to right: this also resolves the problem of symbol redundancy with POOLS and LANES (Semiotic Clarity).

Resolving Symbol Overload: Parallel Multi-Instance Marker

DATA OBJECT COLLECTIONS are currently shown as DATA OBJECTS with a PARALLEL MULTI-INSTANCE MARKER inside. A more semantically transparent representation would be to use multiple copies of the DATA OBJECT symbol (Figure 38): literally, a “collection” of DATA OBJECTS.

<table>
<thead>
<tr>
<th>Current representation</th>
<th>Proposed improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Figure 38" /></td>
<td><img src="image2" alt="Figure 38" /></td>
</tr>
</tbody>
</table>

Figure 38. Data object collections could be shown more semantically transparently using multiple data object symbols (right)

MULTIPLE PARTICIPANTS are currently shown as POOLS with a PARALLEL MULTI-INSTANCE MARKER inside. These could be shown more clearly using a more semantically transparent marker (Figure 39).

<table>
<thead>
<tr>
<th>Current representation</th>
<th>Proposed improvements</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Figure 39" /></td>
<td><img src="image4" alt="Figure 39" /></td>
</tr>
</tbody>
</table>

Figure 39. Multiple participants could be shown more using a more semantically transparent marker

These changes resolve the problem of symbol overload with the PARALLEL MULTI-INSTANCE MARKER (Semiotic Clarity).

Resolving Symbol Redundancy: Message Flows

Attaching message icons to MESSAGE FLOWS as proposed in Perceptual Discriminability also improves their semantic transparency: this is yet another case where Semantic Transparency and Perceptual Discriminability work in harmony. The envelope icon is one of the international standard DOT symbols [57] and is widely understood to represent a message of some kind (e.g. letter, email). All the other characteristics of the MESSAGE FLOW symbol (line style,

---

6 There is recent evidence to suggest that this is culture-specific: English speakers tend to talk about time using horizontal spatial metaphors (e.g. “The best is ahead of us” “The worst is behind us”), whereas Mandarin speakers have a vertical metaphor for time (e.g. the previous month is the “up month” and the next month is the “down month”) [67].
line ends) are **semantically opaque**, but the message icon clearly suggests its meaning. It would be relatively easy for a novice to correctly guess the meaning of the representation on the left of Figure 21, whereas they would have little chance of doing so for the one on the right.

<table>
<thead>
<tr>
<th>Preferred synograph</th>
<th>Removed synograph</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Preferred synograph" /></td>
<td><img src="image2.png" alt="Removed synograph" /></td>
</tr>
</tbody>
</table>

**Figure 40. Attaching message icons to message flows** (left) improves their semantic transparency

Message icons are optional for **MESSAGE FLOWS** in BPMN 2.0, presumably because the notation designers saw no difference between including them or not or could not come to any agreement about which was better. This paper has shown that there are clear cognitive advantages for including them in terms of both **Perceptual Discriminability** and **Semantic Transparency**. This shows the power of having explicit principles for choosing between alternative representations rather than relying on intuition, opinion or social consensus. The principles provide a theoretically sound basis for making graphic representation decisions rather than a subjective/arbitrary basis (i.e. majority vote). They can also help reduce indecision (decision paralysis) and time spent debating alternatives, making the design process more efficient.

6. **Complexity Management**

6.1 **Definition of Principle**

**Complexity management** refers to the ability of notations to present large amounts of information (in this case, complex business processes) without overloading the human mind. This relates to **diagrammatic complexity**, which is measured by the number of elements (**symbol instances** or **tokens**) on diagrams (not to be confused with **graphic complexity**), which is measured by the number of **symbol types**: see **Graphic Economy**.

Complexity management is one of the most intractable issues in designing visual notations: a well-known problem with diagrams is that they do not scale well [68]. The number of diagram elements that can be effectively processed by the human mind is limited by working memory capacity (believed to be seven, plus or minus two elements at a time [69]). When this is exceeded, a state of **cognitive overload** ensues and comprehension degrades rapidly.

To avoid cognitive overload, notations need to provide the capability to divide large diagrams into cognitively manageable “chunks”. **Cognitive load theory** shows that reducing the amount of information presented at a time to within the limitations of working memory improves speed and accuracy of understanding and facilitates deep understanding of information content [70-71]. Complexity management is especially important when dealing with novices, who lack effective strategies for dealing with complexity [30]. Excessive complexity is one of the greatest barriers in understanding of IS models by business stakeholders [72-73].

An exemplar in managing diagrammatic complexity in the IS field (and also one of the best examples in any field) are DFDs [74]. These incorporate hierarchical structuring (levelling) and an explicit complexity limit: 7 ± 2 “bubbles” per diagram, consistent with the known limits of working memory. In this respect, they were ahead of their time, and still are: most modern IS modelling notations – including BPMN – could learn from their example.
6.2 Results of Evaluation

Currently, BPMN lacks effective mechanisms for managing diagrammatic complexity. The issue of complexity management is not even mentioned in the BPMN specification, suggesting that it was not considered in the design of the notation. Almost all the examples in the BPMN 2.0 specification [50] and the accompanying “BPMN 2.0 By Example” document [75] exceed cognitively manageable limits, suggesting that the designers are not aware of the need to manage complexity. Figure 41 shows a diagram consisting of 65 elements and 91 flows, which exceeds cognitive limits several times over.

![Figure 41. Collaboration diagram example (from [76]): most of the examples in the BPMN specification exceed cognitive and perceptual limit](image)

In the absence of effective complexity management mechanisms, BPMN models are typically represented in practice as single monolithic diagrams, often resulting in diagrams that business stakeholders find overwhelming (see Figure 42 for a real world example). While such complex diagrams might be acceptable for experts, they are likely to lead to cognitive overload and map shock [77] when presented to non-experts. This may also be a cultural issue: according to Silver [19], business process modellers traditionally produce “flat” models on large sheets of paper or even walls, so effective complexity management may require a fundamental change of mindset.

![Figure 42. In the absence of complexity management mechanisms, BPMN models are typically represented as single monolithic diagrams](image)
**Link Intermediate Events (page connectors)**

BPMN currently provides only one specialised complexity management mechanism: **LINK INTERMEDIATE EVENTS**, which allow diagrams to be linked across multiple pages (Figure 43). Page connectors are the most primitive and least cognitively effective way of dealing with complexity of diagrams as they involve a total loss of context, analogous to a “cut” rather than a “pan” in cinematography [78-79].

![Figure 43. Link intermediate events allow process models to extend across multiple pages](image)

**LINK INTERMEDIATE EVENTS** may also be used as on-page connectors to avoid crossed lines, long sequence flows or backward flows (Figure 44). This is also an ineffective practice, as it undermines one of the primary cognitive advantages of diagrams over text: diagrams support perceptual inferences through direct visual links rather than symbolic label matching [8].

![Figure 44. Link intermediate events may also be used as on-page connectors (from ???)](image)

**Compound Elements: Subprocesses, Subchoreographies and Subconversations**

BPMN has one major advantage when it comes to complexity management: it includes an in-built hierarchical abstraction capability through its use of **compound elements**: constructs that can be decomposed into smaller sub-elements (corresponding to a subsystem in ontological theory [80]). 3 of the 4 BPMN diagram types include such a construct:

- Process Diagrams: **SUBPROCESSES**, which consist of multiple **ACTIVITIES**
- Choreography Diagrams: **SUBCHOREOGRAPHIES**, which consist of multiple **CHOREOGRAPHY ACTIVITIES**
- Conversation Diagrams: **SUBCONVERSATIONS**, which consist of multiple **CONVERSATIONS**

Each compound element can itself contain compound elements, allowing elements to be decomposed to any number of levels (corresponding to a level structure in ontological theory [80]). This means that the BPMN metamodel provides a sound semantic foundation for incorporating complexity management mechanisms. Surprisingly, compound elements are not used for managing complexity in BPMN – for example:

> “Sub-Processe define a contextual scope that can be used for attribute visibility, transactional scope, for the handling of exceptions, of Events, or for compensation.”
In situ decomposition

The ability to use compound elements to modularise diagrams is limited by their visual representation. BPMN currently only supports in-situ decomposition (also called in-line expansion [19] or diagram nesting), in which elements are expanded within their parent diagram, resulting in a “diagrams within diagrams” representation. Compound elements can either be shown collapsed (with their sub-elements hidden) or expanded (with their sub-elements shown). As shown in Figure 45, diagrams may be nested to multiple levels.

![Diagram showing in-situ decomposition](image)

Figure 45. In situ decomposition: SUBPROCESSES may be shown collapsed (top), hiding their internal structure or expanded (bottom) showing their internal structure.

In general, in situ decomposition is not an effective solution to the problem of diagrammatic complexity, as at the lowest level of abstraction the full complexity of the business process is shown on a single page (e.g. Figure 41). This actually increases diagrammatic complexity compared to a “flat” diagram. In situ decomposition really only works in a tool environment, where it is possible to “toggle” between collapsed and expanded views. It does not translate well to practice, where models are typically printed on paper or drawn on whiteboards.

6.3 Recommendations for Improving Complexity Management

Remove Link Intermediate Events

LINK INTERMEDIATE EVENTS should be removed from the notation, as they undermine cognitive effectiveness, when used as on-page or off-page connectors. This also eliminates symbol excess from the notation (Semiotic Clarity).

Provide Hierarchical Decomposition Capability (levelling à la DFDs)

The most cognitively effective way of managing complexity of diagrams is through hierarchical decomposition (also called hierarchical expansion [19] or levelling [74]). This means that compound elements “explode” to separate diagrams at the next level down rather than being expanding within their parent diagram [30]. This results in a set of hierarchically linked diagrams rather than a single diagram with smaller diagrams nested inside. While this increases the number of diagrams, it means that all diagrams are cognitively manageable in size. Empirical studies show that decomposing IS diagrams in this way can improve end user comprehension and verification performance by more than 50%, and is more cognitively effective than either “flat” diagrams or nested diagrams [72].

Figure 46 illustrates the difference between (a) a “flat model”, (b) in situ decomposition and (c) hierarchical decomposition. The collapsed diagram in the in situ decomposition case is manageable in size but the expanded diagram is not, and is actually more complex than the “flat model”. Only in the hierarchical decomposition case are all diagrams cognitively manageable in size. In addition, all diagrams can be shown on standard-sized paper, making them easy to distribute and copy for review.
"Flat" Model (no subprocesses)

In Situ Decomposition (subprocess diagrams nested within parent diagram)

Hierarchical Decomposition (separate diagrams for each subprocess)

Remove in situ decomposition

Rather than allowing notation users the choice of using in situ or hierarchical decomposition, it may be preferable to remove the in situ option. This would resolve the problem of symbol redundancy for subprocesses and subchoreographies (Semiotic Clarity) and remove the burden of choice as to which approach to use. Hierarchical decomposition is more cognitively effective than in situ decomposition [72], so there is a clear, evidence-based reason for prescribing this as the sole approach. Also, having alternative representational approaches undermines communication and standardisation.
7. Cognitive Integration

7.1 Definition of Principle

This principle applies when multiple diagrams are used to represent a problem: this places additional cognitive demands on the reader to integrate information from different diagrams and navigate between them [9]. According to cognitive integration theory [81-82], for multi-diagram representations to be cognitively effective, they must include explicit mechanisms to support (Figure 47):

- **Conceptual integration**: to enable the reader to assemble information from separate diagrams into a coherent mental representation of the problem.
- **Perceptual integration**: perceptual cues to simplify navigation between diagrams.

Cognitive integration applies both to diagrams of the same type (homogeneous integration) and diagrams of different types (heterogeneous integration).

7.2 Results of Evaluation

BPMN 2.0 currently provides little or no support for cognitive integration. BPMN models are presented as diagrams of 4 different types (Process, Collaboration, Choreography, Conversation) with no overall architecture showing how they are linked together (Figure 47). It is unclear from the specification how the diagram types relate to one another or how they should be used in combination.

![Diagram of Cognitive Integration]

**Figure 47.** Currently, BPMN models are presented as a set of diagrams of different types, with no diagram architecture linking them together

**Mixing diagram types**

BPMN allows diagrams of one type to be embedded (nested) inside diagrams of a different type. For example, Process Diagrams can appear inside Collaboration or Conversation Diagrams, while Choreography Diagrams can appear inside Collaboration Diagrams. This seems to defeat the purpose of defining different diagram types, which should be to show complementary views of a problem domain and thereby partition complexity [30]. Combining them together increases both diagrammatic complexity (Complexity Management) and graphic complexity (Graphic Economy).
Lack of Overview (conceptual integration)

One of the most important mechanisms for enabling conceptual integration is a **longshot diagram**, which provides a “big picture” view of the domain being modelled. This acts as an overall cognitive map into which information from individual diagrams can be assembled [81, 83]. Examples of such diagrams are **rich pictures** in the Soft System Methodology [84] and **context diagrams** in DFDs [74]. Currently, BPMN lacks a longshot diagram: all BPMN diagrams are limited in scope to a single end-to-end business process, resulting in a large number of diagrams at the same level of abstraction, and no way of gaining an overview of the domain as a whole. In the absence of this, practitioners often develop informal diagrams to show the broader context: Figure 49 shows a real world example.

### 7.3 Recommendations for Improving Cognitive Integration

**Combine Collaboration and Process Diagrams**

Process Diagrams and Collaboration Diagrams have the same scope (a single business process) and provide different but complementary views of a process: the Process Diagram de-
An Analysis of the BPMN 2.0 Visual Notation

fines the work performed (internal, “glass box” view) and the Collaboration Diagram defines communication between the process and external participants (external, “black box” view) (Figure 50). The reason for separating them is not explained: both are needed to fully understand a business process. Combining them together would reduce the number of diagram types and therefore cognitive integration overheads.

**Figure 50. Collaboration and Process Diagrams are complementary and could be combined**

**Define a Longshot Diagram (conceptual integration)**

BPMN needs to provide a native longshot diagram rather than forcing practitioners to develop their own informal diagrams at higher levels of abstraction, which undermines standardisation and communication. One way to do this is to create a taxonomy or functional decomposition of business processes in the domain (e.g. Figure 49): this could be represented as a hierarchy chart, a landscape diagram [85] or even a mind map [86], as these are all ways of visualising hierarchies. An alternative approach would be to develop a consolidated Collaboration Diagram showing all MESSAGE FLOWS with EXTERNAL PARTICIPANTS, or at least those MESSAGE FLOWS that initiate or terminate a BUSINESS PROCESS. This is logically
equivalent to a Context Diagram for DFDs, which shows all inputs and outputs to the system being modelled.

**Signposting (perceptual integration)**

Diagram labels are required to support perceptual integration, as they provide **signposts** for navigating between diagrams. Each diagram should be labelled by:

- **Diagram type**: this is necessary to ensure correct interpretation and trigger the appropriate **diagram schema**, if one exists
- **Diagram number**: a unique identifier for each diagram that shows where it fits into the overall diagram architecture, unique within the diagram type
- **Diagram name**: this should summarise the content of the diagram and be unique within the diagram type.

**Hierarchical** or **multi-level numbering** (e.g. as used to show structure in documents) could be used to construct unique identifiers for each diagram. This supports **orientation**, as it shows the user where they currently are in the system of diagrams: this is analogous to a “breadcrumb trail” in user interface design [87]. It also supports **vertical navigation** as it is possible to infer all parent and child diagrams from the level number. For example, if a Sub-process Diagram is numbered 2.2.2, the reader can infer that:

- It is a Level 3 diagram (level of abstraction)
- The parent diagram is “2.2”
- The top level (root) Process Diagram is “2.”
- Child diagrams (if any) will be “2.2.2.n” (indicated by collapsed marker)

![Diagram](image)

**Figure 51. Level numbers orient the reader as to where they are in the system of diagrams and support vertical navigation**

**Collapsed marker (perceptual integration)**

The **collapsed marker** in BPMN means that the symbol is a compound element and can be “opened up” to show its contents. This is **semantically transparent**, as the same symbol is used in directory structures and outlining tools for expanding hierarchies and for “zooming” in interactive maps. This is not currently used for perceptual integration as compound elements are expanded on the same diagram (perceptual integration only applies when linking separate diagrams). However if hierarchical decomposition was used instead of in situ decomposition (as suggested in [Complexity Management](#)), this would provide a perceptual in-
An Analysis of the BPMN 2.0 Visual Notation

tegration cue for linking “parent” and “child” diagrams (vertical navigation) across all diagram types.

![Figure 52. The collapsed marker could be used as a vertical navigation cue for all diagram types](image)

8. Visual Expressiveness

8.1 Definition of Principle

The visual expressiveness of a notation is defined by the number of different visual variables used and the range of values (capacity) used of each: this measures utilisation of the graphic design space. While visual distance (Perceptual Discriminability) measures pairwise visual variation between symbols, visual expressiveness measures visual variation across the entire visual vocabulary. Using a range of visual variables results in a perceptually enriched representation that exploits multiple visual communication channels and maximises computational offloading.

8.2 Results of Evaluation

This is one area that BPMN really excels: it uses the graphic design space more extensively than any previous IS modelling notation. The BPMN designers have shown great ingenuity in using visual variables to generate symbols (though at the expense of graphic complexity: see Graphic Economy). In doing so, they have utilised almost all of the visual variables and a wide range of the capacities of each variable. Importantly, they have resisted the temptation to use text to differentiate between symbols, which languages like UML and i* do to their detriment. Figure 53 summarises the results of the visual expressiveness analysis.

<table>
<thead>
<tr>
<th>Visual variable</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape</td>
<td>Uses a wide range of shapes, and not just 2D geometric shapes like most other IS modelling notations but also icons and 3D shapes</td>
</tr>
<tr>
<td>Brightness</td>
<td>Black vs white icons (for throwing vs catching events and tasks)</td>
</tr>
<tr>
<td></td>
<td>Shading: non-initiating messages (message flows, choreography activities)</td>
</tr>
<tr>
<td></td>
<td>Bold border: call activities, end events, call choreographies, call conversations</td>
</tr>
<tr>
<td></td>
<td>Dotted/dashed/alternate dotted-dashed lines: event subprocesses, message flows, non-interrupting events, data associations, associations</td>
</tr>
<tr>
<td>Texture</td>
<td>Double lines: intermediate events, transactions</td>
</tr>
<tr>
<td>Spatial location (x,y)</td>
<td>Spatial enclosure (pools, lanes, subprocesses, subchoreographies, groups)</td>
</tr>
<tr>
<td></td>
<td>Overlap (boundary events)</td>
</tr>
<tr>
<td>Orientation</td>
<td>Multi-instance markers</td>
</tr>
<tr>
<td></td>
<td>Arrows (sequence flows, message flows, associations and data associations)</td>
</tr>
<tr>
<td>Size</td>
<td>Expanded vs collapsed subprocesses and subchoreographies</td>
</tr>
<tr>
<td>Colour</td>
<td>Not used</td>
</tr>
</tbody>
</table>

Figure 53. BPMN usage of visual variables
Mapping between Information Dimensions and Visual Variables

Different information dimensions should be encoded using different visual variables, so they can be separated in the human mind [11, 33]. In some cases, BPMN encodes a single information dimension using multiple visual variables or encodes multiple dimensions using the same visual variable. As an example of this, EVENTS are differentiated from one another by 4 orthogonal properties (corresponding to information dimensions). Figure 54 shows the mapping between information dimensions and the visual variables used to encode them:

- Two different visual variables (brightness and texture) are used to encode the same dimension (EVENT TYPE). While all EVENT TYPES are differentiated by border styles, START and INTERMEDIATE EVENTS differ in texture, START and END EVENTS differ in brightness\(^7\), while INTERMEDIATE and END EVENTS differ in both brightness and texture.
- The same visual variable (brightness) is used to encode three different dimensions: EVENT TYPE, CATCH/THROW STATUS and INTERRUPTING STATUS.

This means there is a many-to-many relationship between visual variables and information dimensions (Figure 54). EVENT DEFINITION is the only dimension that has a clear mapping to a visual variable.

Recoding EVENTS as proposed in Perceptual Discriminability and Semantic Transparency (using colour to encode EVENT TYPE and spatial location to encode INTERRUPTING STATUS) resolves these problems. Now each semantic property is encoded by a different visual variable, resulting in a clear and unambiguous (1:1) mapping between information dimensions and graphic dimensions.

Non-use of colour

Colour is currently the only visual variable not used in BPMN. It is a strange omission as colour is one of the most cognitively effective of all visual variables [34, 41]. Differences in colour are detected three times faster than shape and are more easily remembered [88-89]. However BPMN is not alone in avoiding colour: no other process modelling notation uses

---

\(^7\) Differences in border thickness count as differences in brightness rather than size as they make symbols darker rather than larger.
An Analysis of the BPMN 2.0 Visual Notation

colour and very few (if any) IS modelling notations do: in particular, UML specifically prohibits it [90]. The failure to exploit such a powerful variable highlights IS notation designers’ lack of knowledge about effective graphic representation principles.

Unlike UML, BPMN does not prohibit the use of colour, but takes an “agnostic” position that it can be used at the discretion of individual users or tool vendors. This supports inconsistent and idiosyncratic use of colour, which undermines standardisation and communication. The notation designers even acknowledge this problem:

“The use of alternative colors in BPMN is non normative. The meaning or semantic of colors might vary from tool to tool or, from user to user, potentially leading to misinterpretations.”

Colour is one of the most difficult variables to use effectively: if not used carefully, it can undermine communication (see Figure 55 for a real world example) [33, 66, 91]. For this reason, use of colour should be prescribed at the notation level to promote effective use and standardisation.

8.3 Recommendations for Improving Visual Expressiveness

Use of Colour

Some examples of how colour could be used to improve the cognitive effectiveness of BPMN have been given in previous sections:

- Using different colours to distinguish between EVENT TYPES (Perceptual Discriminability, Semantic Transparency)
- Using colour to more vividly evoke the metaphor of swimming pools and lanes (Semantic Transparency)

Use of colour could be extended to all BPMN symbols. Figure 56 shows a possible colour scheme that could be used: the colours have been chosen to be as discriminable as possible and to maximise legibility of labels [91]; the colours for EVENTS and DATA OBJECTS (white, in association with paper) are also mnemonic (Semantic Transparency). The addition of colour increases discriminability through redundant coding (Perceptual Discriminability).
**Visual saturation** refers to the use of *all* visual variables: the maximum level of visual expressiveness [30]. If colour is introduced, BPMN would reach the point of visual saturation, the first IS modelling notation ever to achieve this.

**Normal vs exception flow**

The distinction between normal and exception processing (“happy” vs “unhappy” paths) is an important concept in BPMN, but currently there is no visual differentiation between them. Colour could be used to do this by showing exception flows in red: this is semantically transparent, as red is naturally associated with errors. This is an example of how visual variables can also be used to *expand* the meaning of diagrams: the distinction between normal and exception processing is not part of BPMN’s formal semantics, but helps clarify their business meaning. This is called *secondary notation* and plays an important role in interpretation of diagrams [18, 92].

**Robust design**

Colour is highly sensitive to differences in visual perception (e.g. colour blindness) and rendering technology (screens, printers, photocopiers). **Robust design** means designing symbols so they are impervious to such differences [33]. To avoid loss of information, colour should never be used as the sole basis for distinguishing between symbols: it should always be used in combination with at least one other visual variable (redundant coding). Figure 58 shows how the difference between START and INTERMEDIATE EVENTS is lost in conversion to black and white (the acid test for robust design). If a lighter green is used for START EVENTS,
the distinction is preserved due to accompanying differences in brightness. The sequence from light to dark is consistent with the progression from start to end.

<table>
<thead>
<tr>
<th></th>
<th>Colour</th>
<th>Greyscale (black &amp; white)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-robust design</td>
<td><img src="image1" alt="Start Event" /> <img src="image2" alt="" /> <img src="image3" alt="End Event" /></td>
<td><img src="image4" alt="Start Event" /> <img src="image5" alt="Intermediate Event" /> <img src="image6" alt="End Event" /></td>
</tr>
<tr>
<td>Robust design</td>
<td><img src="image7" alt="Start Event" /> <img src="image8" alt="Intermediate Event" /> <img src="image9" alt="End Event" /></td>
<td><img src="image10" alt="Start Event" /> <img src="image11" alt="Intermediate Event" /> <img src="image12" alt="End Event" /></td>
</tr>
</tbody>
</table>

Figure 58. Robust design: varying colour in combination with brightness (bottom) ensures differences are preserved in conversion to black and white.

9. Dual Coding

9.1 Definition of Principle

According to dual coding theory [93], using text and graphics together to convey information is more effective than using either on their own. When information is presented both verbally and visually, representations of that information are encoded in separate systems in working memory and referential connections between the two are strengthened. Labels play a critical role in business interpretation of diagrams as they define the real world semantics of diagrams: their correspondence to the real world domain. In particular, novices depend more on labels than graphical elements to make sense of diagrams [9].

9.2 Results of Evaluation

Currently, BPMN provides no support for dual coding. It focuses exclusively on the graphical content of diagrams, while ignoring the (natural language) labels that play such a critical role in their business interpretation. In doing so, it overloads the graphical representation (visual channel), while underutilising textual elements (verbal channel). This also contributes to its graphic complexity (Graphic Economy).

Lack of rules for inclusion of labels

For a language that is so strict in defining rules for use of graphical symbols, BPMN is surprisingly lax on how symbols should be labelled or even whether they are labelled at all. BPMN contains hundreds of grammatical rules for how graphical symbols can be used but lacks any rules for inclusion, construction or placement of labels. Rather puzzlingly, labels are optional for all symbols:

---

8 Greyscale (black and white) images remove colour (hue and saturation) information and only preserve brightness (intensity) information. Varying colour in combination with brightness ensures differences are preserved in this transformation.
“Both BPMNShape and BPMNEdge may have labels placed inside the shape/edge, or above or below the shape/edge, in any direction or location, depending on the preference of the modeler or modeling tool vendor. Labels are optional for BPMNShape and BPMNEdge.”[20]

Labels play a critical role in the interpretation of process models [94] and diagrams generally [74, 95-98]. Without labels, diagrams lack business meaning and cannot be understood and verified by business stakeholders (see Figure 59).

Figure 59. Left: This is a valid BPMN diagram as all labels are optional. Right: Only when labels are included can the diagram be understood and verified by business stakeholders.

Annotations (labels as explanations)

On a positive note, TEXTUAL ANNOTATIONS are an excellent notational feature of BPMN. Textual explanations can improve understanding of diagrams in the same way comments can improve understanding of programs. According to the principle of spatial contiguity [71], including explanations on the diagrams is much more effective than including them in separate documents. Unlike UML, which uses an explicit symbol to represent annotations, BPMN represents them as simple blocks of text, which avoids symbol excess. The only problem is that annotations use the same line style as DATA ASSOCIATIONS (symbol overload): using a dotted rather than a dashed line as shown in Figure 60 would resolve this problem.

Orientation of labels

BPMN allows labels to be oriented in any direction. This ignores scientific evidence that shows that any deviation from left-to-right, horizontal text significantly reduces reading speed and accuracy. When text is rotated away from the horizontal, oculomotor movements become more complicated and require all six eye muscles to be involved [99].

---

9 This might be described as an “unrule”, as rather than saying what must (= grammatical rule) or should (= recommended practice) be done, it really says that anything goes: labels can be placed anywhere, in any direction and don’t even need to be included at all!
9.3 Recommendations for Improving Dual Coding

*Hybrid Coding (labels as cues to meaning of symbols)*

**Hybrid coding** refers to the redundant use of graphics and text to encode the same information (in contrast to redundant coding, where two or more visual variables are used). Mixed text and graphical symbols can be highly effective in reducing ambiguity of graphical symbols [47]. It is not always possible to find semantically transparent symbols, especially for abstract concepts: if the appearance of the symbol does provide a cue to its meaning, verbal cues can be used instead. After Semantic Transparency, this is one of the most effective tools for improving understandability to novices and reducing effects of graphic complexity: with careful choice of keywords, even the most obscure symbols can become self-explanatory.

Hybrid coding could help with some of the more cryptic symbols in BPMN. For example, the DEFAULT FLOW symbol is a SEQUENCE FLOW that is taken only if no other flow is taken out of an activity. This is shown as a SEQUENCE FLOW with a diagonal bar across it, which is semantically opaque. It is also difficult to think of a semantically transparent way to represent this. The meaning of this symbol could be clarified by attaching a **keyword** (e.g. “otherwise”\(^{10}\)).

<table>
<thead>
<tr>
<th>Graphical encoding (current)</th>
<th>Hybrid coding (redundant graphics and text)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

Figure 61. Hybrid Coding: keywords can be used to clarify the meaning of symbols when they are not semantically transparent

Hybrid coding has no effect on **Perceptual Discriminability** as visual distance is not affected by the addition of labels (as text is not a visual variable). However, it aids interpretation by providing cues to the meaning of symbols and improves retention through interlinked visual and verbal encoding in memory. Like Semantic Transparency, it supports cognitive rather than perceptual processing.

**Define rules for inclusion of labels**

Labels are most commonly used on diagrams not as keywords (literals) but as placeholders (variables) to define their business meaning. Rules should be defined for inclusion of labels, as many symbols make little business sense without them (see Figure 62).

<table>
<thead>
<tr>
<th><img src="image" alt="Diagram" /></th>
<th><img src="image" alt="Diagram" /></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

Figure 62. Left: the conditional start event symbol indicates that a process is triggered by some internal (data-based) condition; Right: this can only be verified if a label is included defining the triggering condition

\(^{10}\) “Default” or “else” could be used instead of “otherwise” but these terms are more commonly used in technical contexts.
Some examples of rules for inclusion of labels on BPMN diagrams are:

- **EXCLUSIVE** and **INCLUSIVE GATEWAYS** MUST be labelled with the decision to be made
- **CONDITIONAL FLOWS** (including outgoing flows from **EXCLUSIVE** and **INCLUSIVE GATEWAYS**) MUST be labelled by the condition under which they are taken; this means that any flow that is NOT labelled represents unconditional (mandatory) flow
- **EVENTS** MUST be labelled with a description of the trigger/result (depending on whether they are catching or throwing events), with the exception of **NONE** events
- **ACTIVITIES** MUST be labelled with a description of the work they perform
- **LOOP MARKERS** MUST be labelled with the applicable loop condition

The purpose of such rules is to make sure diagrams stand alone: that they contain all information required to be understood and verified by business stakeholders without reference to off-diagram information (which should act as “appendices”).

**Define guidelines for constructing labels**

Labels play an important part in interpretation of diagrams, but labelling of IS models is typically done in an *ad hoc* manner in practice [94]. To optimise communication, prescriptive guidelines should be defined for constructing labels for each symbol type. Some examples of guidelines are:

- **ACTIVITIES** should be labelled in active verb + object form: in linguistic terms, a **verb phrase** (VP). Naming processes in this way has been empirically shown to improve understanding of process models [94].
- **EVENTS** should be labelled in the form of **noun phrases** (NP) as they represent states. Most texts on BPMN advocate naming **EVENTS** in the same way as **ACTIVITIES** [e.g. 19], which represent actions. More specific guidelines could be defined for each **EVENT DEFINITION** type e.g. **CONDITIONAL EVENTS** should be labelled by the trigger condition, expressed in natural language.

<table>
<thead>
<tr>
<th>Current representation</th>
<th>Proposed improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive customer claim</td>
<td>Check valid customer</td>
</tr>
<tr>
<td>Customer validity check</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 63.** Left: currently there are no rules for formulating labels for BPMN symbols; Right: **EVENTS** should be labelled using noun phrases and **ACTIVITIES** using verb phrases

Labelling guidelines could also simplify interpretation of some of the more arcane symbols in BPMN. For example, standard templates (**boilerplate labels**\(^{11}\)) could be defined for **EVENT-BASED GATEWAYS** to make their semantics clear even if the symbol is not understood. These are notoriously difficult for business stakeholders to understand as they are unique to BPMN: they have no equivalents in previous process modelling languages. As shown in Figure 64, the labels reinforce and expand the meaning of the graphical symbols and reduce likelihood of misinterpretation.

---

\(^{11}\) A label is one that consists of keywords (literals) plus placeholders (variable text). In Figure 64, keywords are shown in upper case and variable text in lower case.
An Analysis of the BPMN 2.0 Visual Notation

Figure 64. Labelling guidelines could clarify the meaning of some of the more arcane symbols in BPMN. The labels both reinforce and expand the meaning of the graphical symbols.

Such guidelines may seem to cross the line between grammar (“correct” use of a notation) and style (“effective” use of notation) [19], but effective labelling has a significant effect on understanding of process models [94]. A consistent approach to formulating labels would increase standardisation, communication and save notation users having to formulate such guidelines themselves.

Define rules for placement of labels

BPMN also takes a laissez-faire approach to the placement of labels: labels can be placed inside or outside, above or below, right or left of, in front or behind, symbols. Placement of labels determines how clearly labels are associated with their referent objects [100], so should be prescribed by the notation.

In most IS diagrams, labels are placed inside symbols. However, in BPMN, labels are mostly placed outside symbols (e.g. EVENTS, GATEWAYS, DATA OBJECTS, CONVERSATIONS, MESSAGES): this is not formally mandated but is the default label position in the Notational Depiction Library. No rationale is provided for this, but the most likely reason is to maximise page real estate: to enable more symbols to be included on the page (Complexity Management). Placing labels inside symbols (using the Gestalt Common Region Principle [101]) creates a stronger association with the referent object than placing them outside (using the Gestalt Proximity Principle [100]): common region overrides proximity [101]. As shown in Figure 65, placing labels inside EXCLUSIVE GATEWAYS (as in traditional flowcharts) creates a stronger association with the symbol, which provides yet another reason for removing the “X” marker.

<table>
<thead>
<tr>
<th>Current representation</th>
<th>Proposed improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Current representation" /></td>
<td><img src="image2" alt="Proposed improvement" /></td>
</tr>
</tbody>
</table>

Figure 65. Left: currently labels are typically shown outside GATEWAY symbols; Right: placing labels inside symbols creates a stronger Gestalt association between symbol and label
10. Graphic Economy

10.1 Definition of Principle

Parsimony is an important goal for any language, visual or otherwise. The Principle of Graphic Economy states that the graphic complexity of a notation should be cognitively manageable. **Graphic complexity** is measured by the number of visually distinct symbols in a notation: the size of its visual vocabulary [15]. The human ability to discriminate between perceptually distinct alternatives (span of absolute judgement) is around 6 categories [69], which defines an effective upper limit for graphic complexity. Novices are much more affected by graphic complexity than experts, as they must consciously maintain meanings of symbols in working memory: the more symbols there are to remember, the greater the cognitive load and therefore likelihood of interpretation errors. Empirical studies show that graphic complexity significantly reduces understanding of IS models by novices [15].

10.2 Results of Evaluation

BPMN is astonishingly complex and looks more like a visual programming language than a business modelling notation. The BPMN Notation Depiction Library lists a total of 177 visually distinct symbols, which exceeds cognitive limits by an order of magnitude (almost 30 times). Such a level of graphic complexity would be a problem for any visual notation, but is particularly so for one designed for communication with novices. In contrast, DFDs consist of just four symbols, which is within cognitive limits, making the notation easy to learn, apply and understand [74].

Figure 66 shows the BPMN 2.0 poster available from the OMG website, which shows all symbols on a single page. What is remarkable about this is that it requires A2 paper (four times standard size paper) to pack them all in. If this was included as a legend on a diagram (which represents best practice in diagramming [96]) there would be no room for the diagram itself!

**Usability effects**

BPMN’s graphic complexity also represents a barrier to learning and use by analysts, as notation users need to learn each symbol, how to use it correctly and how to combine it with other symbols. This may explain why BPMN practitioners use only a tiny fraction of the symbols
available [102]. The number of grammatical rules increases geometrically with the number of symbols, which increases the difficulty of using it correctly. The BPMN specification contains hundreds of sometimes arcane rules for how symbols can be combined together.

As an example, START EVENTS are the first elements on any BPMN Diagram. Figure 67 summarises the rules for using these on PROCESS DIAGRAMS: there are a total of 24 rules, making it hard for notation users to even get off the starting blocks. In contrast, the grammatical rules for constructing DFDs can be summarised in a single sentence: “A Data Flow may be used to connect a Process to one of the following: (a) another Process, (b) a Data Store (c) an External Entity”. The number and complexity of grammatical rules may explain why BPMN diagrams in practice contain so many errors [19, 21].

<table>
<thead>
<tr>
<th>Rules for use of Start Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. A Start Event is OPTIONAL: a Process level—a top-level Process, a Sub-Process (embedded), or a Global Process (called Process)—is NOT REQUIRED to have a Start Event.</td>
</tr>
<tr>
<td>2. If a Process is complex and/or the starting conditions are not obvious, then it is RECOMMENDED that a Start Event be used.</td>
</tr>
<tr>
<td>3. An Event Sub-Process MUST have a single Start Event.</td>
</tr>
<tr>
<td>4. If a Start Event is not used, then the implicit Start Event for the Process SHALL NOT have a trigger.</td>
</tr>
<tr>
<td>5. An exception to 4 is if the first Activity in the Process is an instantiating Receive Task.</td>
</tr>
<tr>
<td>6. If there is an End Event, then there MUST be at least one Start Event. The use of Start and End Events is independent for each level of the Diagram.</td>
</tr>
<tr>
<td>7. When a Start Event is not used, then all Flow Objects that do not have an incoming Sequence Flow shall be the start of a separate parallel path. Each path will have a separate unique token that will traverse the Sequence Flow.</td>
</tr>
<tr>
<td>8. An exception to 7 is a Compensation Activity. Compensation Activities are not considered a part of the normal flow and MUST NOT be instantiated when the Process is instantiated.</td>
</tr>
<tr>
<td>9. An exception to 7 is a catching Link Intermediate Event, which is not allowed to have incoming Sequence Flows.</td>
</tr>
<tr>
<td>10. An exception to 7 is an Event Sub-Process, which is not allowed to have incoming Sequence Flows and will only be instantiated when its Start Event is triggered.</td>
</tr>
<tr>
<td>11. There MAY be multiple Start Events for a given Process level. Each Start Event is an independent Event. That is, a new Process instance SHALL be generated when each Start Event is triggered.</td>
</tr>
<tr>
<td>12. If the Process is used as a global Process (a callable Process that can be invoke from Call Activities of other Processes) and there are multiple None Start Events, then when flow is transferred from the parent Process to the global Process, only one of the global Process’s Start Events will be triggered. The targetRef attribute of a Sequence Flow incoming to the Call Activity object can be extended to identify the appropriate Start Event.</td>
</tr>
<tr>
<td>13. A top-level Process that has at least one None Start Event MAY be called by a Call Activity in another Process. The None Start Event is used for invoking the Process from the Call Activity. All other types of Start Events are only applicable when the Process is used as a top-level Process.</td>
</tr>
<tr>
<td>14. Start Events can be used for three types of Processes: Top-level Processes, Sub-Processes (embedded), Global Process (called), Event Sub-Processes.</td>
</tr>
<tr>
<td>15. There are seven types of Start Events for top-level Processes in BPMN: None, Message, Timer, Conditional, Signal, Multiple, and Parallel.</td>
</tr>
<tr>
<td>16. There is only one type of Start Event for Sub-Processes in BPMN: None.</td>
</tr>
<tr>
<td>17. There are nine types of Start Events for inline Event Sub-Processes: Message, Timer, Escalation, Error, Compensation, Conditional, Signal, Multiple, and Parallel.</td>
</tr>
<tr>
<td>18. A Start Event MUST NOT have incoming Sequence Flows.</td>
</tr>
<tr>
<td>19. An exception to 18 is when a Start Event is used in an Expanded Sub-Process and is attached to the boundary of that Sub-Process. In this case, a Sequence Flow from the higher-level Process MAY connect to that Start Event in lieu of connecting to the actual boundary of the Sub-Process.</td>
</tr>
<tr>
<td>20. A Start Event MUST be a source for a Sequence Flow.</td>
</tr>
<tr>
<td>21. Multiple Sequence Flows MAY originate from a Start Event. For each Sequence Flow that has the Start Event as a source, a new parallel path SHALL be generated.</td>
</tr>
<tr>
<td>22. The conditionExpression attribute for all outgoing Sequence Flows from a Start Event MUST be set to None.</td>
</tr>
<tr>
<td>23. A Start Event MAY have zero or more incoming Message Flows. Each Message Flow targeting a Start Event represents an instantiation mechanism (a trigger) for the Process. Only one of the triggers is required to start a new Process.</td>
</tr>
</tbody>
</table>

Figure 67. Grammatical rules for use of Start Events on Process Diagrams.
Causes of graphic complexity in BPMN

Why is BPMN so complex compared to other process modelling languages, especially when its raison d’être was understandability to business stakeholders? One possible explanation is the explicit design strategy of showing as much information as possible in graphical form: “modelling conventions should be employed to make the semantics unambiguous from the diagram alone” [19]. BPMN tries to get the diagram to do too much work, even work it is not well suited for, which has catastrophic effects on graphic complexity. In this paper, we question this design approach and argue that this was a tactical error on the part of the notation designers: trying to encode everything in graphical form overloads the diagram and in doing so, the cognitive circuitry of the notation user.

Information density

Not only is the number of symbols a problem, but also the complexity of individual symbols (information density). As discussed earlier, BPMN is unusual among IS modelling notations in that it builds composite symbols from smaller elements (graphemes). It starts with a basic set of shapes (e.g. ACTIVITY = rountangle) and changes their visual characteristics (e.g. border style) and/or adds icons (e.g. LOOP MARKERS). This results in graphically complex symbols that convey multiple messages. The information density of a symbol is measured by the number of individual facts it conveys. In DFDs, each symbol conveys a single fact (e.g. circle = process) while in BPMN, each symbol is like a mini-diagram: in Figure 68 the symbol on the right has an information density of 7.

This means that a much more complex process is required to interpret the meaning of BPMN symbols, in which symbols must be parsed into their constituent graphemes: in most diagramming notations, simple shape recognition is enough. For example, the symbol in Figure 69 requires a 5 level parse tree (the number of levels corresponding to its information density), while only a single level (the first: shape) is required for any symbol on a DFD.

10.3 Recommendations for Improving Graphic Economy

Reducing graphic complexity should be a major priority for BPMN as this represents the single greatest barrier to its usability and effectiveness in practice. Its graphic complexity is likely to deter all but technical experts, which undermines its primary goal of providing a common language for communicating between business and technical experts. However, reduction in graphic complexity does not happen naturally and requires conscious effort: history shows that visual notations tend to increase inexorably in complexity over time [30].
BPMN is no exception to this: BPMN 2.0 extends BPMN 1.2 by adding 2 new diagram types and over 50 new symbols.

Figure 69. Interpreting the meaning of BPMN symbols involves a complex parsing process. Interpreting this symbol requires 5 levels of decision making.

Reduce semantic complexity

Semantic complexity (the number of semantic constructs in a notation) is a major determinant of graphic complexity, as different constructs are usually represented by different symbols (following the Principle of Semiotic Clarity). Thus, an obvious way to reduce graphic complexity is to reduce the number of semantic constructs. However, this is beyond the scope of this paper, which focuses only on syntactic issues. Also, the semantics of BPMN is relatively fixed because of the requirement to “canonically support the semantics of BPEL” [103].

Introduce symbol deficit

Graphic complexity can also be reduced directly (without affecting semantics) by introducing symbol deficit (Semiotic Clarity). This means choosing not to show some constructs in graphical form. The interpretation of any diagram depends on a division of labour between graphics and text [92, 104]. Symbol deficit reduces graphic complexity by offloading some information from the diagram to supporting definitions, so alters this division (the graphics-text boundary). The goal of visual notation design should not be to show everything in graphical form (the approach taken in BPMN) but to find the right balance between graphical and non-graphical encoding that maximises computational offloading while keeping
graphic complexity manageable [18, 68]. Currently, the BPMN graphical representation is overloaded, and some information needs to be offloaded onto textual specifications.

**Rationalise Number of Loop Markers**

A simple example of how to use symbol deficit to reduce graphic complexity would be to replace the three existing LOOP MARKERS by the STANDARD LOOP MARKER, the most semantically transparent one (it is a true icon as it perceptually resembles a loop) (Figure 70). The type of loop could then be defined in supporting definitions (i.e. non-graphically). Business stakeholders would be unlikely to be interested in such technical detail, so little is lost by removing this information from the diagram. This provides a useful basis for making decisions about where to draw the graphics-text boundary: business relevant details can be shown on the diagram and technical details hidden in supporting definitions, which act as “appendices” to diagrams.

![Figure 70. Graphic complexity of BPMN could be reduced by rationalising the number of loop markers](image)

**Remove Event Definition Icons**

A more radical proposal for introducing symbol deficit would be to remove the EVENT DEFINITION icons: this would reduce graphic complexity by around 25%. These are one of the most distinctive features of BPMN and make BPMN diagrams almost immediately recognisable. Clearly, a lot of effort has gone in developing them. However this effort has been largely counterproductive:

- They are difficult to discriminate due to their size and numerosity (Perceptual Discriminability)
- They are mostly semantically opaque or perverse so require conscious effort to remember (Semantic Transparency)
- They are one of the major sources of graphic complexity
- Many are difficult to draw by hand (Cognitive Fit)

In addition, they add very little information for such a massive impost in graphic complexity: effectively they classify events into 12 categories, which corresponds to only about 3 bits of information. In other words, they are not cognitively cost-effective. They are also likely to be a continuing source of graphic complexity in the future: they define a taxonomy of EVENT TRIGGERS and RESULTS, which is likely to grow over time as people think of new trigger and/or result types.
Finally, when EVENTS are properly labelled (which they always should be: see Dual Coding), EVENT DEFINITION icons are redundant and provide only a subset of the information the label provides. For example, in Figure 71, the MESSAGE EVENT icon says that the process is triggered by a message of some kind but doesn’t say what message or who it is from, whereas the label defines all of these things\textsuperscript{12}. Removing the icon would therefore lose no information.

<table>
<thead>
<tr>
<th>Message Start Event</th>
<th>Labelled Message Start Event</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Message Icon" /></td>
<td><img src="image2.png" alt="Message Icon with Label" /></td>
</tr>
</tbody>
</table>

Customer order received

Figure 71. Left: The message icon indicates the process is triggered by a message of some kind. Right: the event label indicates what message triggers the process and who it is received from: the icon could therefore be removed without losing information.

11. Cognitive Fit

11.1 Definition of Principle

According to cognitive fit theory\textsuperscript{[105-107]}, problem solving performance is determined by a three-way fit between the problem representation, task characteristics and problem solver skills (Figure 72). This means there is no one “best” representation in all situations: different representations may be suited for different tasks and/or audiences. Most IS modelling notations define a single visual notation for all purposes. However cognitive fit theory suggests that this “one size fits all” assumption may not be appropriate and that different representations (visual dialects\textsuperscript{13}) may be required for different purposes.

![Cognitive Fit Diagram](image3.png)

Figure 72. Cognitive fit is the result of a three-way interaction between the representation, task and problem solver\textsuperscript{[106]}

\textsuperscript{12} Naming standards can be defined to ensure this information is provided (see Dual Coding).

\textsuperscript{13} A visual dialect is a distinct visual representation of the same underlying notation semantics. Notations that differ in semantics are called semantic dialects.
11.2 Results of Evaluation

Currently, BPMN consists of a single notation used for all purposes, so does not support cognitive fit. It is also poorly adapted for some purposes, suggesting that the needs of all tasks and audiences have not been considered in its design.

Problem solver skills: language differences

On the positive side, BPMN relies exclusively on graphics to differentiate between symbols, so is impervious to language differences. This means that it can be used to communicate across international boundaries.

Problem solver skills: expert-novice differences

There are well-known differences in the way experts and novices process diagrams [41, 108-112], which need to be taken into account when designing notations for communication with novices. The most important differences are:

- Novices have more difficulty discriminating between symbols [40, 111]
- Novices have to consciously remember what symbols mean [41]
- Novices are more affected by diagrammatic complexity [77]

For effective communication with novices, visual notations should have easily discriminable symbols (Perceptual Discriminability), mnemonic conventions (Semantic Transparency), effective mechanisms for dealing with complexity (Complexity Management), verbal cues and labels (Dual Coding) and a simplified visual vocabulary (Graphic Economy) [30]. BPMN fares poorly on all of these principles, which suggests that expert-novice differences were not taken into account in its design. This is particularly puzzling given that one of BPMN’s major claimed advantages is its business-friendliness [19, 50].

Task characteristics: hand drawing

Process models are typically developed in an interactive manner by sketching on whiteboards or paper. It is therefore important that diagrams can be drawn quickly and easily so as not to impede the flow of ideas. Some of the important notational requirements for hand drawing are:

- Perceptual Discriminability: discriminability requirements are much higher for hand drawing because of variations in how symbols are drawn by different people (e.g. rectangles and rountangles are likely to be indistinguishable when drawn by hand).
- Semantic Transparency: icons are more difficult to draw than simple shapes.
- Complexity management: in situ decomposition is difficult to apply as expanding compound elements requires redrawing the diagram.
- Visual Expressiveness: some visual techniques (e.g. shape fills and border styles) are difficult to draw by hand.
- Graphic Economy: graphic complexity and symbol complexity (e.g. use of composite symbols) increase drawing difficulty.

The BPMN 2.0 symbol set presents considerable challenges for hand drawing, as it includes a wide range of icons, composite symbols, shape fills and border styles. For example, the symbols shown in Figure 73 would be difficult for most practitioners to draw by hand, especially in a way they could be reliably interpreted; drawing such complex symbols would also slow down the modelling process. This suggests that requirements for hand drawing were not taken into account in its design, even though this was explicitly mentioned in the BPMN charter [103]:

“If small BPMN processes cannot be easily jotted down by a business analyst on a blank piece of paper then BPMN will not have been successful”
Task characteristics: design stages

In all design disciplines, designs evolve through multiple stages. For example, the Zachman framework for IS architecture identifies 5 stages through which designs progress, from initial sketching of requirements through to a working system [113]. At each stage, additional details are introduced. Different design stages correspond to different tasks (and audiences), which cognitive fit theory says may require different representations. BPMN 2.0 is unusual in that it uses a single notation for all design stages, from analysis of an existing process through to an executable process. Effectively, it coalesces all the rows in the Zachman framework into one. In doing so, it overloads representations in earlier stages with unnecessary (implementation-oriented) details.

11.3 Recommendations for Improving Cognitive Fit

Trying to design a single notation that is suitable for all purposes inevitably involves compromise, and generally results in a notation that is suboptimal for any particular purpose. Defining multiple, complementary visual dialects enables the best of all worlds: notations that are optimised for particular tasks and/or audiences. This does not require changing the BPMN semantics, as visual dialects can be defined as profiles using the Meta Object Facility (MOF), the OMG standard metamodel architecture which is used to define the BPMN metamodel.

Expert-novice differences

One approach to dealing with expert-novice differences is to design the notation so that it is understandable by novices: the “lowest common denominator” approach. However, optimising representations for novices can reduce their effectiveness for experts, due to the expertise reversal effect [114]. Alternatively, different visual dialects can be defined for each audience: a fully functional (“Pro”) dialect for BPM experts and a simplified (“Lite”) dialect that is optimised for communicating with business stakeholders. For example, because novices have a lower tolerance for graphic complexity, EVENT DEFINITION icons could be removed from the “Lite” version and only shown in the ”Pro” version.

Support for hand drawing

Many of the existing BPMN symbols are difficult to draw, which suggests that the visual vocabulary should be simplified to support hand drawing. Alternatively, a simplified visual dialect could be defined for sketching (which may be a modified subset of the full symbol set), making sure that symbols are similar enough to those in the full symbol set not to cause confusion.

Separation of concerns

BPMN currently contains the superset of all symbols required across all design stages. This overloads the representation for upstream participants, particularly for business stake-
holders. As a result, BPMN models contain many technical details that are irrelevant or incomprehensible to a business audience. **Separation of concerns** is one of the fundamental principles of software engineering: to clearly separate between aspects relevant in each design stage to partition complexity and avoid constraining the solution [115]. To support this, BPMN could be partitioned into different dialects for different design phases: for example, “analysis”, “design” and “implementation” BPMN. “Analysis” BPMN would only include information relevant to business stakeholders (which could correspond to the “Lite” version above), “design” BPMN would include additional details required by application developers (which could correspond to the “Pro” version) and “implementation” BPMN would include all information required for execution by process engines (which would correspond to the full BPMN 2.0 notation).

Partitioning the language in this way would make it much easier to learn and use, as few practitioners are currently using BPMN to specify executable processes and most are using it purely for analysis purposes [21]. This means they could get by with learning a relatively small subset of the language (the “analysis” dialect). Prescriptively defining subsets of the notation based on the needs of different tasks and audiences may be preferable to identifying these based on observations of practice [102], which may reflect practitioners’ limited knowledge of the constructs available.

![Diagram](image)

**Figure 74. Different visual dialects of BPMN could be defined for different tasks and audiences, which may be subsets of the existing notation and of each other**

### 12. Conclusion

This paper has conducted a systematic, symbol-by-symbol analysis of the BPMN 2.0 visual notation, based on a set of theoretically and empirically grounded principles for visual notation design). The analysis has revealed some serious flaws in the notation, which represent potential barriers to its usability and effectiveness in practice. Given the widespread, almost universal, adoption of BPMN, the effects of these problems are likely to be magnified many times over in practice. The conclusion from the analysis is that radical surgery is required to make the BPMN 2.0 visual notation cognitively effective. The paper also makes some practical recommendations for improving the cognitive effectiveness of the notation.
12.1 Summary of Findings

The findings for each principle are summarised in Table 3: red = serious problem; amber = moderate problem and green = minor problem.

<table>
<thead>
<tr>
<th>Principle</th>
<th>Description</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Semiotic Clarity</td>
<td>1:1 correspondence between constructs and symbols</td>
<td>Moderate: some violations in all categories but mostly easily fixed.</td>
</tr>
<tr>
<td>2. Perceptual Discriminability</td>
<td>Ease of differentiating between symbols</td>
<td>Moderate: basic symbol categories mostly discriminable but too many subtle variations within each category. Problems mainly caused by numerosity of symbols (graphic complexity) and overuse of icons.</td>
</tr>
<tr>
<td>3. Semantic Transparency</td>
<td>Ability to infer meaning of symbols from their appearance</td>
<td>Moderate: some good points but does not exploit this enough for a notation designed for communication with novices. Represents a major opportunity for improving understandability to business stakeholders.</td>
</tr>
<tr>
<td>4. Complexity Management</td>
<td>Ability to represent complex models without overloading the human mind</td>
<td>Moderate: existing complexity management mechanisms ineffective but compound elements provide a sound semantic basis for incorporating this.</td>
</tr>
<tr>
<td>5. Cognitive Integration</td>
<td>Support for understanding of multiple diagrams</td>
<td>Poor: Lacks overall diagram architecture, overview (longshot) diagram and support for navigation between diagrams (signposting).</td>
</tr>
<tr>
<td>6. Visual Expressiveness</td>
<td>Utilisation of graphic design space</td>
<td>Excellent: outperforms all previous IS modelling notations in exploring the graphic design space. Still room for improvement as it omits one of the most effective visual variables of all (colour).</td>
</tr>
<tr>
<td>7. Graphic Economy</td>
<td>Manageable sized visual vocabulary</td>
<td>Catastrophic: far too many symbols, which reduces usability and understandability; symbols themselves are also graphically complex</td>
</tr>
<tr>
<td>8. Dual Coding</td>
<td>Use of text to reinforce and clarify diagrams</td>
<td>Poor: lacks rules for inclusion, construction and placement of labels</td>
</tr>
<tr>
<td>9. Cognitive Fit</td>
<td>Support for all tasks and audiences</td>
<td>Poor: lack of support for expert-novice difference, hand drawing and different design stages</td>
</tr>
</tbody>
</table>

In many ways, BPMN represents both the best and worst of visual notation design practice: it scores better than any existing IS modelling notation on Visual Expressiveness, but worse than any on Graphic Economy. It represents a giant leap forward from early process modelling notations in terms of its semantics: it has an explicit metamodel defining its constructs and grammatical rules as well as formal execution semantics. However it represents a giant leap backwards in terms of visual representation. Compared to DFDs, one of the earliest process modelling notations, it fares worse on almost all principles. While BPMN has closed
the gap with process execution languages (implementation distance), it has widened the gap with human information processing (cognitive distance).

So where did it go so wrong for BPMN? Why does it perform so poorly given the resources and expertise that went into its development and that its primary goal was “to provide a notation that is readily understandable by all business users” (which represents an informal statement of cognitive effectiveness)? One possible explanation is the lack of involvement by business stakeholders. The major contributors to BPMN so far have been tool vendors [20-21, 103], which explains its technical rather than business focus: it seems better suited for execution by process engines than processing by the human mind. If business stakeholders had been involved in the design process or if usability testing had been conducted using them, it may have revealed some of the problems identified in this paper. Usability testing is routinely conducted for user interfaces and public information symbols [116] but rarely for visual notations, even international standards like BPMN and UML.

Another possible explanation is the terms of the BPMN charter, which asked for a notation that “is acceptable and usable by the business community” and “canonically supports the BPML semantics” (the target implementation language) [103]. These are largely incompatible requirements: it is impossible to optimise a notation for processing by process engines and humans. In trying to achieve the second objective (executability), the notation designers may have sacrificed or downgraded the first objective (business understandability): while executability can be objectively verified, business understandability can only be evaluated subjectively (in the absence of usability testing) so is easier to explain away or ignore. The solution to this conflict lies in Cognitive Fit: defining different notations for different purposes.

12.2 Practical Contributions

The practical contribution of this research is to identify potential problems with the BPMN 2.0 visual notation and suggest possible ways of improving its cognitive effectiveness. The results of the analysis provide a sound scientific basis for improving the notation in future releases.

A more general practical contribution is to highlight the need for IS notation designers (e.g. groups like OMG) to design visual notations in a scientific manner rather than relying on intuition and social consensus. The broad objective of this research is to transform IS visual notation design from:

- An unselfconscious process based on instinct, imitation and tradition (tacit knowledge [117]) to a selfconscious process based on explicit design principles
- A naïve process based on common sense to an informed process based on theory and empirical evidence
- A democratic process based on weight of opinions to a scientific (evidence based) process based on weight of evidence.

12.3 Theoretical Contributions

The theoretical contribution of this research has been to demonstrate a systematic approach to evaluating and improving IS visual notations, which could be applied to any visual notation. The Physics of Notations performs a complementary role to ontological analysis [80] (used for analysing notation semantics) in analysing notation syntax.

The paper also shows how a design theory can be used to improve IS practice. The Physics of Notations synthesises relevant theory and empirical evidence to help notation designers make informed visual representation decisions. Because of the infinite possibilities in the graphic design space, it would be inefficient to resolve every decision by experiment.
Design theories and empirical testing are complementary. Design theories provide a prescriptive basis for making design decisions, which result in outcomes that can be tested against the predictions of the theory. In this paper, changes are proposed to the BPMN visual notation that are hypothesised to improve its cognitive effectiveness based on the predictions of the Physics of Notations. These hypotheses that can be empirically tested. Decisions can be made and justified using a design theory but can only be confirmed by empirical testing.

### 12.4 Limitations of the Research

One obvious limitation of this research is that it focuses only on syntactic issues. Addressing some of the problems in BPMN 2.0 may require re-examining its semantics, which is beyond the scope of this paper.

A second limitation is that it falls short of defining a new, more effective visual notation for BPMN. This was never our intention and is beyond the scope of this research. Our goal was not to redesign the BPMN visual notation but to evaluate existing visual representation choices and open people’s eyes to the alternatives available in the graphic design space.

A third limitation is that some of the recommendations are inconsistent with one another (e.g. new EVENT DEFINITION icons in Semantic Transparency and removal of EVENT DEFINITION icons in Graphic Economy), as the analysis for each principle was conducted relatively independently of one another. There are tradeoffs between objectives in any design task, and these would need to be resolved to produce an improved visual notation.

A final limitation is that the suggestions for improving the visual notation have not been empirically tested. However the Physics of Notations provides a sound theoretical basis for predicting that they will improve cognitive effectiveness. It would be relatively easy to conduct experiments to confirm the predictions of the theory.

### References


56. Donohue, B., *Hold that elevator! The intuitiveness of symbols*, in iQblog. 2009 iQ Content.


