Conceptual Integrity without Concepts

Giorgio Grasso
Department of Cognitive Science
University of Messina
gmgrasso@unime.it

Alice Plebe
Department of Mathematics and Computer Science
University of Catania
alice.plebe@gmail.com

Abstract

It is commonly asserted that one of the most crucial factors in software design is the adoption of consistent and appropriate concepts. This widespread assumption, underlying several researches in software engineering, tacitly postulates the very existence of concepts. What, then, if concepts do not exist? This is the question asked in this paper, where the articulation of possible answers is attempted. As weird as it might sound, doubts on the existence of concepts have been recently cast by few distinguished philosophers of mind, with compelling arguments. One such argument is the heterogeneity of cognitive assets that can be referred to a single concept, and we show that it might be the case for design concepts in the example of Blender, a 3D computer graphics software.

1. Introduction

There is a widely perceived view that there exists something that we call "concept". Concepts are allegedly used to categorize the world, to store knowledge, for reasoning, and for drawing analogies. This is certainly the view held by Frederick Brooks when he famously stated that the success of software system design rely heavily on the precise identification and preservation of "concepts" [4]. It is reasonable to assume that the integrity of design concepts throughout software development was based on the implicit postulate that concepts are objective and stable entities. We will show that in fact this is not always a necessary requirement, but it is certailny usuallly assumed.

This tacit confidence on the ontology of concepts has surrounded all the research domain in conceptual design for software engineering that followed Brooks' ideas [9, 2, 8]. Moreover, even the precise nature of concepts, with very few exception [16], has never been inquired.

Yet, the ontological status of concepts is highly controversial, and, recently, compelling arguments have been proposed to challenge the very existence of something we call "concepts" [20, 21, 25]. The difficulties in finding a suitable ontological status for concepts, and the reasons for eventually denying any such status, will be briefly reviewed in §2.

While not entering into this debate as such, this paper questions how the issue of conceptual integrity in software design will fare, if it turns out that concepts in fact do not exist. One of the main troubles affecting concepts, as traditionally conceived, is their heterogeneity, that will be discussed in §2.1. If a linguistic label used to refer to a specific design concept actually corresponds to a plurality of heterogeneous cognitive assets, then, arguably, a serious threat undermines the whole idea of conceptual integrity. Scholars arguing for the elimination of concepts have also proposed new theoretical entities, more congruent with the current knowledge of human cognition. One, here discussed in §2.2, is that of "unicept", a sort of concept unique for a human agent. The trouble for conceptual integrity can be now traced to the simple fact that, unlike concepts, unicepts are not objective entities, thus cannot be generalized across individuals.

In §3 we will show that it may be the case for Blender, a popular open-source software for 3D computer graphics. This software is developed under a wide collaborative basis, and it is built on to a set of firm design concepts concerning its user interface. Yet, Blender is often not regarded as easy to use, even by experienced graphics developers familiar with other software applications. We argue this is a case of heterogeneous cognitive assets of, apparently, same concepts, nothing exceptional if concepts do not exist.

However, we argue that the lack of a well-founded ontology for concepts does not undermine completely the need and practice of conceptual integrity, as will be demonstrated in the case of the software Blender. Rather, it requires awareness that design concepts are not necessarily carved on stone tablets just because they are expressed in words, therefore the ideas of software architects may not coincide with those expected by implementers or users.

DOI reference number: 10.18293/SEKE2017-203

2. Do concepts exist?

Doubts about the existence of concepts were raised by the extreme difficulties found in every attempts to define exactly what concepts are, and how they work.

In modern philosophy extensive work on the nature and origin of concepts was carried by the English empiricists, especially John Locke [19] and David Hume [15]. In their idiom concepts were called "ideas", similarly to Frederick Brooks who, in his books, uses "design ideas" as a perfect equivalent of "design concepts". Ideas, for Locke and Hume, are perceptually based, and reason gets its contents from the senses. The empiricist position has been often refuted in the history of philosophy, but recently proposed in a modernized account supported by cognitive and neurocognitive evidences [29].

A large body of work on concepts was carried out by psychologists in the second half of last century, but marginally concerned with a precise characterizations of what concepts are. Concepts are often conflated with categories [23], knowledge [1], and representations [33].

In most contemporary philosophy of mind and philosophy of language "concepts" are used as a non technical equivalent for "propositional attitudes" [27, 10, 22, 29], i.e. propositions that, like beliefs, can be true or false, or, like desires, can be satisfied or unsatisfied. Most of the times philosophers content themselves with this account of concept, and their effort is in explaining how concepts are individuated. For instance, they are supposed to explain what distinguishes our capacity to have propositional attitudes about keyboards as such from our capacity to have propositional attitudes about hard-disks as such. Even if the focus of philosophy and psychology on concepts is different, in the last decades the discussions become entrenched, with philosophers evaluating the virtues of psychological theories of concepts with respect to their own criteria [10, 24, 29].

2.1. (Too) Many concepts of "concept"

Details of the countless theories of concepts in psychology and philosophy are out of the scope of this work, but the diversity of existing theories is at the very core of concerns related to the existence of concepts. For sure, both philosophers and psychologists have always acknowledged the wild differences between concepts and between kinds of concepts, but the received view is that, over and beyond the differences between concepts and between kinds of concepts, there is a set of common relevant properties [26, 13]. Few have proposed that the apparent divergence between concepts is due to a pluralism in scope, for example few properties could be shared by cats and computer keyboards because biological kinds and artifacts have little in common [18]. Others have argued for a pluralism of competences, different kinds of concept are involved in different cognitive competences [28], for example a concept of keyboard used when we categorize visually an object as keyboard could have little in common with a concept of keyboard when reasoning about developing the software driver module for a keyboard.

A different line is held by Machery with his "heterogeneity hypothesis" [21], the proposal that most categories of physical objects, most types of event, and most substances are represented by several concepts that belong to kinds that have little in common. There are at least four different theoretical entities that have been conceived as theories of concepts, and they have little in common.

The classical theory states that a concept is the set of properties which are separately necessary and jointly sufficient for an object to belong to the class of that concept [32], this theory has been largely endorsed before the end the 1960s [17]. Notably, the classical theory had profound influence on computer science, it has been assumed as the foundation for the domain of formal concept analysis. One of the most favored mathematization of concepts in computer science is by sets of objects and attributes, that define, respectively, the extension and the intention of a concept, and the interdependence of concepts is formalized by mathematical lattice structures [12]. This formalization of concepts has proven fruitful in several applications, including text retrieval and mining [7], support in software engineering for identification of object-oriented structures or refactoring activities [34]. While formal concept analysis is still today an active domain of research, his underlying classical theory of concepts has been mostly abandoned in philosophy for decades. It was replaced in the 1970s by the *prototype* paradigm, assuming that we have in mind some properties that are believed to be typically possessed by the members of a class [31, 14]. Few years later a different paradigm of concepts was proposed, that of *exemplar* [5], the idea is that concepts are sets of exemplars, which are bodies of properties believed to be possessed by few particular members of a class. The fourth theory is the *theory* paradigm [6, 30], proposing that concepts are some kind of folk theories, knowledge that can explain the properties of category members. Machery compellingly illustrated how all those theoretical paradigms of concept have little in common, from both the point of view of the types of knowledge they encode, and the kinds of cognitive process they involve.

2.2. Unicepts

The elimination of concepts for Machery is a consequence of the perplexing heterogeneity of theoretical construes, reviewed above, which in fact denote heterogeneous

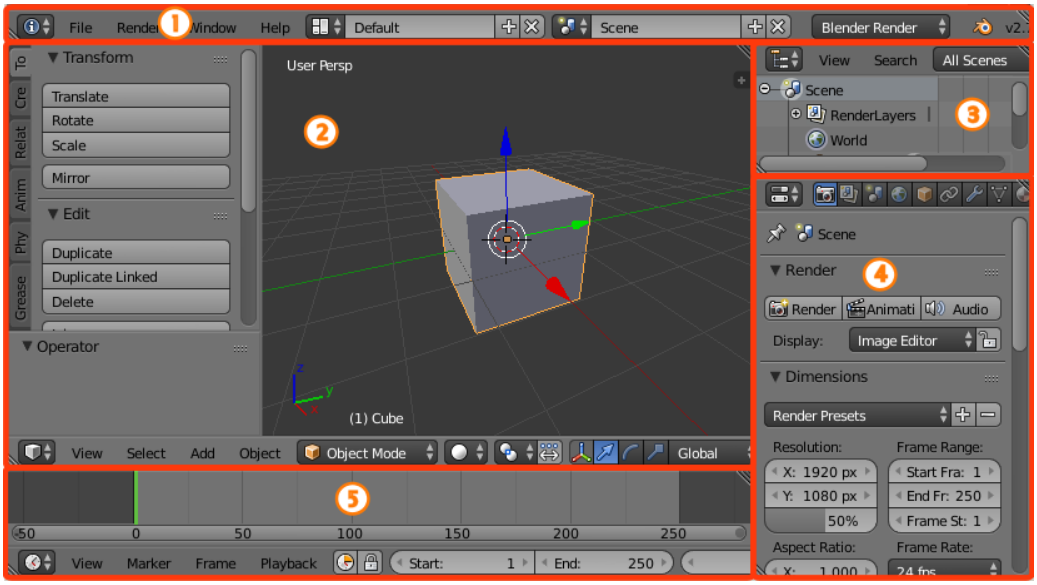

Figure 1. The Blender interface, with highlighted in red five "Editors": Info (1), 3D View (2), Outliner (3), Properties (4) and Timeline (5).

ways in which the mental/neural vehicles hold information about things and properties of things. It is this heterogeneity that led Millikan to introduce a new notion different from "concept", as the fundamental units of cognition: the "unicept". In her words:

> "Uni" is for one, of course, and "cept" is from Latin *capera*, to take or to hold. One's unicept for an object, or property, or kind, or relation etc., takes in many proximal stimulations and holds them as one distal entity. A developed unicept reaches through a radical diversity of sensory impressions to find the same distal thing again. [...] A unicept is a specific individual *faculty* developed for a very specific purpose, the purpose of collecting and integrating information about some particular thing. [25, p.16]

This new theoretical entity differs from "concept" in several ways, which are not essential in this paper. What count here is that, unlike the common view on concepts, unicepts are not entities that people share. Every individual has his/her own private stock of unicepts. In most cases unicepts of different people succeed in gathering information about exactly the same things in the world, but they do this in different ways, sometimes using overlapping cognitive strategies, but also many that are distinct. Moreover, having a unicept involves having a certain kind of ability or capacity to deal, successfully, with an aspect of the world, and there is no genuine unicept unless what it pulls information about is indeed a precise entity in the world. Otherwise, in the words of Millikan, "if it pulls together information about many things, using this as though it were about one thing, then [...] it is an *empty* unicept or at best an *equivocal* unicept.

3. Unicepts in Blender

In order to illustrate cases where design concepts are much better defined in terms of Millikan's unicepts, rather than traditional, universal, concepts, we analyize the user interface of Blender (`www.blender.org`), a widely used open-source computer graphics software [3]. Originally developed in 1995 by Ton Roosendaal as an in-house tool, in 2002 turned into an open source project, that since 2007 has been coordinated by the Blender Institute in Amsterdam, under Roosendaal direction.

Elements in the interface of Blender are organized in strict hierarchical manner, based on the following "concepts":

Window ⇒
Screen ⇒
Areas ⇒
Editors ⇒
Regions ⇒
Tabs ⇒
Panels ⇒
Controls

"Screens" are window layouts, Blender offers a set of pre-defined possible layouts, but it is possible to customize

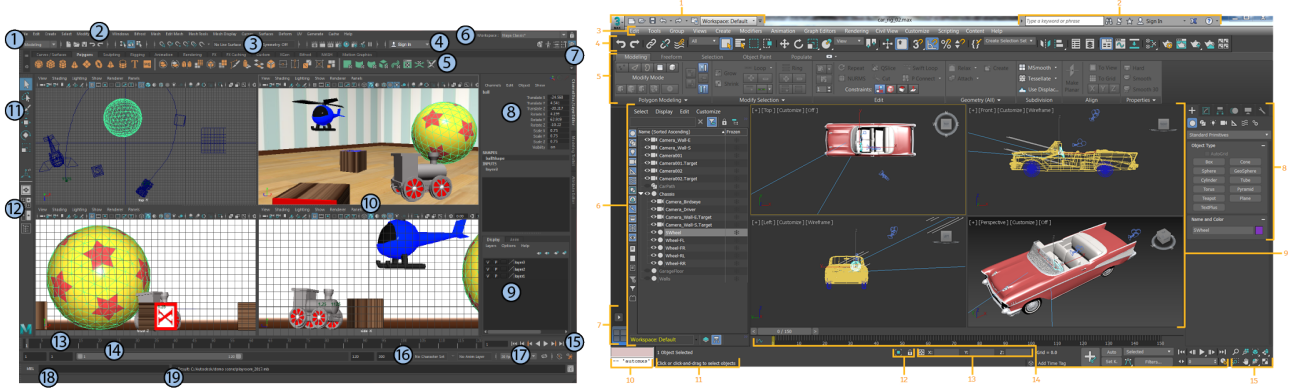

Figure 2. Comparison with interfaces of other 3D computer graphics software: Maya on the left, 3Ds Max on the right.

any of them. Screens are divided up into a number of re-sizable "Areas", each of which containing a particular type of "Editor", like a 3D View Editor (the main interaction with the 3D scene), an Outliner (the overview of scene graphs hierarchy), a Graph Editor (for keyframe interpolation), a Node Editor (node-based shading and compositing tools), and so on. In Fig. 1 a screenshot of the "Window" is given, with a layout of 5 "Areas" hosting different "Editors". "Editors" in turn are split into "Regions", portions of the space inside the "Area" containing smaller structuring elements like tabs and panels with buttons, controls and widgets placed within them. Blender uses a peculiar screen-splitting approach to arrange areas: the "Window" can be divided into an arbitrary number of smaller adjacent "Areas", and each "Area" is always fully visible, and it is straightforward to work in one area and hop over to work in another.

3.1. What "Editor" means

Most of the words here quoted to address the main "concepts" of Blender's interface are of common usage among professional sofware, and not only in the computer graphics domain. The point is that several of this words, even if including properties shared by other software, are characterized by aspects of meaning, and usage, which are distinctive in Blender. For sexample, in the two main alternative to Blender, Maya and 3Ds Max (see Fig. 2) there is a sharp distinction between "editors" and other tools like toolbox shelves (Maya) or ribbons (3Ds Max). In Blender, every instance of an "Area" contains an "Editor", words that, arguably, refers to an unicept, kind of concept held by accustomed users of Blender, and by its developers, but diverges considerably from the set of properties and features other users include in their unicept of "editor".

It is noteworthy that despite this deviation of several concepts from their account in other software, Blender achieved in its user interface a remarkable integrity. There is a high consistency of interaction across all "Editors", with several non-modal tools, that can be accessed efficiently without taking time to select them according to the type of editing. Inside every "Editor" there is a consistent and predictable usage of mouse and keyboard actions for interaction.

For example, most single-key commands have their reverse action with the same key pressed together with the `Alt` modifier. The key `A` means the selection of "All", where this quantifier can refer to a variety of entities depending on the "Editor", like 3D meshes in the case of 3D View, keyframes in the case of Graph Editor, nodes in the Node Editor, and so on. These features are among the most appreciated in the community of Blender users, who find the working efficiency offered by its interface unsurpassed.

Therfore, it is possible to state that Blender, in its user interface, is a good example of, say, *Uniceptual Integrity*. This integrity, which is certainly an important reason of its success, comes at a price. Unicepts are not universally shared, and users that formed their concepts of editor and the likes via mediate inference and practical learning over time on different graphic software of their acquaintance, will find themself discomforted and confused working with Blender. Arguably, the amount of overlap of unicepts between people is a matter of degrees, in the case of software we may have groups of users where the amount of sharing is high, and other gropus where is it minimal. We believe that the case of Blender is rather unexceptional, that in most complex software there can be examples of unicepts, that may or may not be succesfully preserved in their integrity, but can be alien to unicepts named with the same words in other software.

4. Conclusions

This article takes on the issue of how conceptual integrity in software design can be pursued, under the assumption that there is no such thing as "concepts". We addressed this issue not as a logic exercise of counterfactual reasoning, but because recently a few distinguished philosophers have argued against the very existence of concepts. In trying to articulate the consequences for conceptual integrity, we adopted one of the best alternatives, in our view, proposed so far: Millikan's "unicept". It is a mental entity that is built by collecting and integrating information about some thing of the world, in a large variety of ways, and differs substantially from concepts in that unicepts are not shared as such by people. We analyzed the case of Blender software, and the items ordinary called "concepts" that characterize its user interface, arguing that they are better construed as "unicepts", because, despite the same verbal labels, differ intrinsically from those used in other graphics software. However, the lack of generality does not hamper the software design to pursue a high degree of integrity, that now we can call *uniceptual integrity*. It is probably an integrity less ambitious of that fostered by Frederick Brooks, that works well for group of users that share, at least, an important part of the unicepts founding the design, and not for others. It is our opinion that the reasoning carried out in this paper has a relevance for application in software engineering, especially when large teams of developer are involved and the conceptual integrity issues have a significant impact.

References

[1] L. W. Barsalou. Perceptual symbol systems. *Behavioral and Brain Science*, 22:577–660, 1999.

[2] G. A. Blaauw and F. P. Brooks. *Computer Architecture: Concepts and Evolution*. Addison Wesley, Reading (MA), 1997.

[3] A. Brito. *Blender 3D: Architecture, Buildings, and Scenery: Create photorealistic 3D architectural visualizations of buildings, interiors, and environmental scenery*. Packt Publishing Ltd, Birmingham (UK), 2008.

[4] F. Brooks. *The Mythical Man-Month: Essays on Software Engineering*. Addison Wesley, Reading (MA), 1975.

[5] L. Brooks. Nonanalytic concept formation and memory for instances. In E. Rosch and B. B. Lloyd, editors, *Cognition and concepts*, pages 169–211. Lawrence Erlbaum Associates, Mahwah (NJ), 1978.

[6] S. Carey. *Conceptual change in childhood*. MIT Press, Cambridge (MA), 1985.

[7] C. Carpineto and G. Romano. Using concept lattices for text retrieval and mining. In *Formal Concept Analysis – Foundations and Applications* [11], pages 161–179.

[8] F. Détienne. *Software design: cognitive aspects*. Springer-Verlag, Berlin, 2002.

[9] J. Dvorak. Conceptual entropy and its effect on class hierarchies. *IEEE Computer*, 27:59–63, 1994.

[10] J. Fodor. Concepts: A potboiler. *Cognition*, 50:95–113, 1994.

[11] B. Ganter, G. Stumme, and R. Wille. *Formal Concept Analysis – Foundations and Applications*. Springer-Verlag, Berlin, 2005.

[12] B. Ganter and R. Wille. *Formal concept analysis: mathematical foundations*. Springer-Verlag, Berlin, 1999.

[13] R. L. Goldstone and A. Kersten. Concepts and categorization. In I. B. Weiner, editor, *Handbook of psychology*. John Wiley, New York, 2003.

[14] J. Hampton. Prototype models of concept representation. In I. Van Mechelen, J. Hampton, R. S. Michalski, and P. Theuns, editors, *Categories and concepts: Theoretical views and inductive data analysis*, pages 67–95. Academic Press, New York, 1993.

[15] D. Hume. *An Enquiry Concerning Human Understanding*. A. Millar, London, 1748.

[16] D. Jackson. Towards a theory of conceptual design for software. In *ACM International Symposium on New Ideas, New Paradigms and reflections on programming and software – Onward!*, pages 282–296. ACM, 2015.

[17] J. Katz and J. Fodor. The structure of a semantic theory. *Language*, 39:170–210, 1963.

[18] L. K.Komatsu. Recent views of conceptual structure. *Psychological Bulletin*, 112:500–526, 1992.

[19] J. Locke. *An essay concerning human understanding*. Meridian Books, Cleveland, 1690.

[20] E. Machery. Concepts are not a natural kind. *Philosophy of Science*, 72:444–467, 2005.

[21] E. Machery. *Doing without concepts*. Oxford University Press, Oxford (UK), 2009.

[22] E. Margolis and S. Laurence, editors. *Concepts: Core readings*. MIT Press, Cambridge (MA), 1999.

[23] E. M. Markman. *Knowledge representation*. Lawrence Erlbaum Associates, Mahwah (NJ), 1999.

[24] R. G. Millikan. *On Clar and Confused Ideas: An Essay About Substance Concepts*. Cambridge University Press, Cambridge (UK), 2000.

[25] R. G. Millikan. An epistemology for phenomenology? In R. Brown, editor, *Consciousness Inside and Out: Phenomenology, Neuroscience, and the Nature of Experience*, pages 13–26. Springer-Verlag, Berlin, 2014.

[26] G. L. Murphy. *The big book of concepts*. Cambridge University Press, Cambridge (UK), 2002.

[27] C. Peacocke. *A study of concepts*. MIT Press, Cambridge (MA), 1992.

[28] G. Piccinini and S. Scott. Splitting concepts. *Philosophy of Science*, 75:390–409, 2006.

[29] J. Prinz. *Furnishing the Mind – Concepts and Their Perceptual Basis*. MIT Press, Cambridge (MA), 2002.

[30] L. J. Rips. The current status of research on concept combination. *Minds and Language*, 10:72–104, 1995.

[31] E. Rosch. Principles of categorization. In E. Rosch and B. Lloyd, editors, *Cognition and Categorization*. Lawrence Erlbaum Associates, Mahwah (NJ), 1978.

[32] K. L. Smoke. An objective study of concept formation. *Psychological Monographs*, 42:1–46, 1932.

[33] K. Solomon, D. L. Medin, and E. Lynch. Concepts do more than categorize. *Trends in Cognitive Sciences*, 3:99–105, 1999.

[34] T. Tilley, R. Cole, P. Becker, and P. Eklund. A survey of formal concept analysis support for software engineering activities. In *Formal Concept Analysis – Foundations and Applications* [11], pages 250–271.