Design and Make—and Code?: Technology Education and a Unified Conception of Technology

Jonas Hallström

Book Chapter

Cite this chapter as:


DOI: https://doi.org/10.1163/9789004687912_005

International Technology Education Studies, 1879-8748, No. 20

Copyright: Brill Academic Publishers
https://brill.com/

The self-archived postprint version of this book chapter is available at Linköping University Institutional Repository (DiVA):
https://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-199781
Design and Make – and Code?

Technology Education and a Unified Conception of Technology

Jonas Hallström
Dept. of Behavioural Sciences and Learning, Linköping University, Sweden

Abstract

The aim of this chapter is to discuss how a unified theory of technology could be forged philosophically, and suggest some implications for technology education. A post-phenomenological model of human-technology relations was employed as an analytical tool. It is concluded that both digital and analog technologies could be seen as technical artefacts with a dual nature and technologies of representation. The dual nature of technical artefacts, that is, the functional/intentional and physical dimensions of artefacts and systems, is reflected e.g. in the abstract programming language in conjunction with a specification, which relates to a physical configuration. Representational technologies could include everything from simple control systems to computers to AI systems, and it would be possible to conceive of the concrete and abstract parts of these technologies as different components of their representational capacity; a component could either be seen as representing (concrete) or represented (abstract), but part of the same representational system that makes up the technology. In both these “dual” perspectives on technology, artefacts and systems could be viewed from a common point of view and may consist of both digital, analog, concrete, and abstract components that together make up the technology. One important implication for technology education is that teaching needs to involve both abstract and concrete technological components. When programming, for instance, students need to learn not only about the code or software in itself, but also about what digital technology does in terms of solving real-world problems and achieving technical purposes.

Keywords
Digital Technology – Analog Technology – Programming – Philosophy of Technology – Post-Phenomenology – Technology Education

Introduction

Technology education is a subject with a traditionally strong emphasis on designing, making and using physical artefacts (e.g., de Vries, 2016). This is not surprising given the fact that sloyd, craft and/or vocational subjects – which include the making of products as central components – preceded technology education in many countries over the world (Hallström, Hultén & Lövheim, 2014; Jones, Buntting, & de Vries, 2013). In recent years, however, digital tools, programming, and computational thinking (CT) have also been incorporated on a large scale as curriculum components in technology education in many countries.

When implemented in technology education programming has partly come to be carried out in pure coding tasks (e.g. Bjursten, Nilsson, & Gumaelius, 2022), at the same time as technology education is focused on the designing, making and using of artefacts and systems. This potential mismatch stems from uncertainty about exactly how programming and digital technology should be conceived of in technology education. The aim of this chapter is therefore to discuss how a unified theory of technology could be forged philosophically, and suggest some implications for technology education.

Digital, analog, abstract, and concrete dimensions of technology

When defining technology, it is important to separate between four of its dimensions: digital and analog on the one hand, and abstract and concrete, on the other. Digital technology in everyday usage is a very broad term that encompasses computers and all manner of digital tools, but in essence digital really denotes technology that relies on or processes information in the form of 0s and 1s, such as in binary code. A digital thermometer can thus show temperature as numbers on an LCD screen, in which case both the gauging of the temperature and the display are digital. Analog, on the other hand, means technology that uses or processes information in the form of all values. Examples include a traditional mercury thermometer with a seamless transition between the display of different temperatures due to the expansion or retraction of the mercury, or an oscilloscope.
DESIGN AND MAKE – AND CODE?

showing sinus waves of electronic signals on a tube screen. Concrete technology is usually called artefact but can be all manner of physical devices or physical networks or systems, from coffee cups to windmills to sewer pipes to spaceships. When we talk about computers or information technologies, their concrete manifestations are also referred to as hardware. Abstract technology is essentially information in various forms, both analog and digital, and in the latter case it is also referred to as software. See Figure 1 for an outline of these four dimensions of technology.

**Figure 1. Schematic outline of digital, analog, abstract, and concrete dimensions of technology**

[Diagram showing the outline]

The examples in Figure 1 show that at least in the case of information-processing technologies, be they analog or digital, there is always a concrete and an abstract component. It might be odd to see a steam engine as an information-processing technology, but even in such early controlled systems information had to be fed back for control purposes (Hallström & Kaijser, 2022). The focus of this chapter, however, is mostly on the upper right-hand corner because coding has often been seen as something not really related to technology but rather to mathematics and information processing in that it takes an abstract, or virtual, form (Brey & Hartz Søraker, 2009). Consequently, computing and information technology
have developed their own academic trajectories not always related to technology and engineering (e.g. Denning & Tedre, 2019), but it is also the case that technology education traditionally embraced a view of technology as mainly being physical artefacts, that is, concrete forms of technology.

**Literature review: Technology education and physical artefacts**

Today, therefore, there is a strong focus on designing and making products in technology education, even though there are different emphases in different countries. For example, in the UK there is emphasis on design and manufacturing of products, while in the USA engineering design in the context of STEM (science, technology, engineering and mathematics) education has gained traction (Buckley, Seery, Gumaelius, Canty, Doyle, & Pears, 2021; ITEEA, 2020; Rossouw, Hacker, & de Vries, 2011; Williams, 2018). Unsurprisingly, much earlier research shows that students primarily equate technology with artefacts in general and modern products such as TVs, cell phones and computers in particular (e.g. Jones, Buntting & de Vries, 2013; Solomonidou & Tassios, 2007; Svenningsson, 2020).

The field of philosophy of technology is an important “sister discipline” to technology education because it provides theoretical tools for understanding what technology is, analyzing important concepts, and deciding what technological content is most important to learn (Dakers, Hallström, & de Vries, 2019). Just like technology education, philosophy of technology is a young discipline. Its primary concern is also technological artefacts, which really distinguishes it from the broader field of philosophy in general and philosophy of science in particular. Artefacts are human-made objects and have been thought of as uninteresting and even non-existent in the philosophy of science, because they have been considered as mere applications of science (Dusek, 2006; Meijers, 2009). The philosophy of technology and engineering, on the other hand, concerns itself with philosophical questions related to the human-made world, notably technological artefacts and related knowledge-based practices primarily in engineering but also in other technical professions such as architecture (Meijers, 2009).

One of the most influential philosophers to develop the artefactual perspective is Carl Mitcham (Meijers, 2009). Due to the focus on artefacts in technology education, the philosophical works most cited in a technology education context have related strongly to the artefactual
perspective. Mitcham’s (1994) *Thinking through Technology* has therefore been particularly influential on technology education research, making it a foundation for many current technology education studies (two of several recent examples are Svenningsson, 2020; Xu, Williams, & Gu, 2021). Mitcham (1994) writes:

> Artifacts–material objects such as tools, machines, and consumer products–are what most readily come to mind when the word “technology” is mentioned. [...] Technology as object is the most immediate, not to say the simplest, mode in which technology is found manifest, and it can include all humanly fabricated material artifacts whose function depends on a specific materiality as such (Mitcham, 1994, p. 161).

Because philosophy of technology really started as a field of research after the broader diffusion of computers and information technology in society, and as “tools, machines, and consumer products” are increasingly becoming digital and computerized, our digitalized society has been the object of much philosophical analysis (e.g. Borgmann, 1999; Brey & Hartz Søraker, 2009; Feenberg, 2017). However, abstract conceptions of technology beyond the material artefacts have not yet found their way from philosophy to technology education to any greater extent, for example, in relation to computation and information technology; examples could be coding, information processing, computational thinking, artificial intelligence (AI), and more. One reason for this slow transition may be the preoccupation in technology education with the material aspects of the human-made world, which could make abstract aspects of technology invisible and even contested.

**Post-phenomenological analysis of digital technology**

There has indeed been a preoccupation with the ontology and epistemology of physical artefacts lately even in the philosophy of technology and engineering (see above and e.g. Hahn & Soentgen, 2011; Kroses & Verbeek, 2014). The philosophical concerns of the digital world – which has made attention to abstract aspects of technology particularly pertinent – have instead mostly been addressed in the philosophy of AI, mathematics, or computing, as well as computer ethics (Brey & Hartz Søraker, 2009). On the other hand, a central journal such as *Philosophy & Technology* is so broad in scope that it includes research from all of the
above fields and thus in relation to both concrete and abstract aspects of technology (see, for example, No. 3 2014; cf. Hansson, Belin & Lundgren, 2021).

The problem of coming up with a definition of technology that includes both analog, digital, abstract and concrete dimensions of objects and systems could be framed in post-phenomenological terms. Post-phenomenology deals with the human experience of technology and the different ways that this experience can be embodied, but it also offers ways of categorizing and defining technology through human–technology relations (Rosenberger & Verbeek, 2015). I will here consider what is being mediated in different such relations (Ihde, 2010), as a way of illustrating an integrated view of technology. If we look at post-phenomenology in Ihde’s (1990) version, the relationship between humans and the world is seen as mediated through technology, often conceived of in the following simple model:

\[ I \rightarrow \text{technology} \rightarrow \text{world} \]

Ihde exemplifies technology with a number of concrete instruments such as a telescope, a clock or a pair of eyeglasses, so the conception of technology is essentially physical or concrete, as is the human experience of technology as embodied (Ihde, 1990, p. 25; cf. Ihde, 1993). The corresponding post-phenomenological model for digital technology could be the same if one focuses on digital artefacts such as smartphones as concrete tools:

\[ I \rightarrow \text{digital artefact (smartphone)} \rightarrow \text{world} \]

However, if we look at the abstract code or information as technology the model would have to be revised into something like this, in the context of programming:

\[ I \rightarrow \text{interface} \rightarrow \text{code} \]

Or, when handling information and datasets:

\[ I \rightarrow \text{algorithm} \rightarrow \text{dataset} \] (cf. Wellner & Rothman, 2020)

In both cases the material component of technology is missing, so it is a both abstract and digital form of technology that mediates the relation between I/human and world. Furthermore, the world is also abstract, so it
DESIGN AND MAKE – AND CODE?

is a virtual world. There is therefore also a corresponding problematic regarding what the technology mediates, the real or the virtual. Brey and Hartz Søraker (2009) write:

The software constructs with which computer users interact, such as files, folders, and web pages, exist virtually rather than physically. Although they are realized and sustained by means of physical systems, they do not exist solely or primarily as physical entities. The existence of virtual objects, and correlated virtual spaces, actions and events raises questions regarding their ontological status: what is their mode of existence, and what is their place in philosophical ontology? Let us call nonphysical software-generated objects and spaces with which users interact virtual entities. Virtual entities are represented as part of the user interface of computer programs. They manifest themselves to the user through symbolic or graphical representations, and they interactively respond to actions of the user. Contemporary user interfaces are in most cases graphical, representing virtual objects as ordinary, manipulable physical objects (pp. 1381-1382).

Thus, in many current digital settings and, for example, visual or graphic programming environments, the above post-phenomenological models could be conceived of as:

I \rightarrow \text{virtual object} \rightarrow \text{virtual space}

This is thus a far cry from Ihde’s (1990) post-phenomenological conception of the relationship of humans with – concrete/physical – technology, and may be a reason that there is an unease about digital technology in technology education.

Technologies as technical artefacts

There have been attempts at resolving the duality between the concrete and the abstract, for example, to see software, programs and virtual worlds or objects as de facto technical artefacts. Turner (2014) goes some way towards explaining the relationship between the programming languages and the physical devices by referring to Kroes and Meijers’ (2006) conception of the dual nature of technical artefacts, that is, their physical/structural and functional/intentional dimensions. In Turner’s view, the abstract programming language can stand on its own as having a mathematical semantics, but once we include a specification as the functional dimension that Kroes and Meijers (2006) require for an artefact
to be defined as technical, this changes the program into a technical artefact:

Programming languages have a mathematical semantics and an implementation. It is in the mathematical object that structure is located. And the semantics can stand alone as a mathematical object, i.e., to be investigated as such. But when used as the specification, it is given governance over the implementation. The intentional aspect then comes into play: the agent gives the abstract semantics normative force over the physical implementation: they must be in extensional accord. Abstract structure and physical structure are linked by the intention to take the former as having normative governance over the latter. It is at this point that the piece of abstract mathematics takes on its functional guise. The agent’s intention relates the abstract and the concrete (Turner, 2014, pp. 393-394).

In other words, when viewing the program as a specification (function) of how a (physical) object should behave, we have merged the abstract and the physical components of digital artefacts and systems. A post-phenomenological model of technical artefacts would thus generally look like this:

\[ I \rightarrow \text{technical artefact (function/structure)} \rightarrow \text{world} \]

And, more specifically, based on Turner (2014) a post-phenomenological model of a programming language as a technical artefact could be conceived of as such:

\[ I \rightarrow \text{program specification/physical execution} \rightarrow \text{world (virtual/real)} \]

Many digital instruments such as digital telescopes, computerized tomography scanners and electronic microscopes mediate the world in this way via virtual entities – symbolic representation (Brey & Hartz Søraker, 2009) – and Ihde (2011) therefore calls this translational mediation. The world translates and even transforms through these technologies. There is thus less distinction between the virtual and the real if we acknowledge that the virtual is some form of representation of a real (or imagined) world.

**Technologies as technologies of representation**

There is thus from the perspective of the dual nature of technical artefacts a way of merging the concrete and the abstract into one conception of
technology, which also translates into a reasonable post-phenomenological model. However, I want to suggest a further perspective that might also broaden the conception of technology. In a sense, virtual entities and objects are rather graphical-visual models of real, or imagined, phenomena, that is, types of representation (cf., Hallström & Schönborn, 2023). One way of merging the (conflicting?) perspectives on technology entailed by digitalization – epitomized by the dualisms concrete vs. abstract, and real vs. virtual – is to view digital technologies as *technologies of representation* (Denning & Tedre, 2019; Mindell, 2002). Representational technologies could include everything from simple control systems to computers to AI systems, and it would be possible to conceive of the concrete and abstract parts of these technologies as different components of their representational capacity. That is, a component could either be seen as representing (concrete, real) or represented (abstract, virtual), but part of the same representational system that makes up the technology (cf., Hallström & Kaijser, 2022). Furthermore, it is possible to see representational technologies as semiotic technologies, in which case even the abstract, representing components such as a piece of code are, in the semiotic sense, material. (Latour’s (1999, 2005) actor-network approach is semiotic in the sense that the so-called *actants* could be construed as non-human, concrete or abstract, “actors”.) The post-phenomenological model would thus look like this:

I → technology (representing/represented) → world (real/virtual)

James Watt’s centrifugal flyball governor in his late 18th century modified steam engine was the first simple yet effective modern feedback mechanism to be widely used. Modern, more complex technological systems include a multitude of feedback loops, particularly for negative feedback, to make the systems operate in a stable and robust way, and to assist operators and users. Today such control systems use both analog and digital sensors to communicate information back to a regulator or control unit, which is often a computer program of some sort (Mindell, 2002). Whether we talk about the flyball governor or a digital sensor, they are both part of the representing components that help control the system by providing the (represented) information. In the case of the digital sensor, small currents represent and provide information whereas in the case of the flyball governor a lever represented and transmitted information directly to a steam valve (Glad & Ljung, 2000).
Towards a unified conception of technology: Concluding discussion and implications

Technology could be conceived of both as technical artefacts with a dual nature and technologies of representation. The dual nature of technical artefacts, that is, their functional/intentional and physical dimensions, can be mirrored in the abstract programming language that on its own has a mathematical semantics but once we include a specification/intention the program becomes a de facto technical artefact since it must include a physical configuration to operate. Representational technologies could include everything from simple control systems to computers to AI systems. It would be possible to conceive of the concrete and abstract dimensions of these technologies as different components of their representational capacity; a component could either be seen as representing (concrete, real) or represented (abstract, virtual), but part of the same representational system that makes up the technology. In both these “dual” perspectives on technology, artefacts and systems could be viewed from a common point of view and may consist of abstract and concrete components that together make up the technology, regardless of whether it is analog or digital. See Figure 2 for a visualization of how these two perspectives play out in relation to the digital, analog, abstract, and concrete dimensions of technology.

Figure 2. Schematic outline of digital, analog, abstract, and concrete dimensions of technology, in relation to the dual nature of technical artefacts and technologies of representation
Both these dual conceptions make sense in a technology education context. Denning and Tedre (2019) argue that “internally the computer does not process numbers and symbols. Computer circuits deal only with voltages, currents, switches, and malleable materials. The patterns of zeroes and ones are abstractions invented by the designers to describe what their circuits do” (pp. 55-56). From the computer scientists’ point of view, according to Turner (2014), they build:

programs, data types, type inference systems, etc. [that] seem to have an abstract guise that enables us to reflect and reason about them independently of any physical manifestation. […] One can reason about these in a mathematical way that is independent of any physical representation. Much the same applies to programs, types, compilers, virtual machines, interpreters, etc. All of these notions seem to have an abstract guise that is independent of their physical realization or implementation. On the other hand, these objects must have a physical implementation that enables them to be used as artifacts in the physical world. For instance, a program that has no physical realization is of little use as a practical device for performing humanly intractable computations (pp. 394-395).

In technology education, therefore, it would be limiting for students if the teaching merely focused on the programming in itself, because there needs to be a connection to a physical realization in the form of a design, product
or technical solution. On the other hand, teachers in a study by Vinnervik (2021) deemphasized procedural programming knowledge with the argument that the technology students were not going to become programmers. With regard to the two dual perspectives on technology discussed in this chapter, however, it is clear that technology teachers who deal with the digital world in general and programming in particular should see them as an integrated part of the designed, human-built world that is at the core of technology education. Knowledge about analog and digital technology cannot be separated, but it is, like technology, dual.

Thus, when teaching about, for example, programmed technological solutions (PTS) in a technology classroom, teachers need to include the nature of the abstract code as well as the concrete technical solutions that are to be programmed and controlled. As Cederqvist (2020) writes: “The ability to understand and handle the programming material encompasses both procedural and conceptual knowledge that refers not only to the material itself but also to a general understanding of programming concepts that are necessary for fulfilling the PTS” (p. 24). Such PTS projects should appeal to technology teachers since they could restore some of the design aspects of teaching programming, in the sense that the design encompasses both analog and digital, concrete and abstract dimensions and coding could be conceived of as a craft (Bratteteig, 2010; Lingel & Regan, 2014).

References


Buckley, J., Seery, N., Gumaelius, L., Canty, D., Doyle, A., & Pears, A. (2021). Framing the constructive alignment of design within technology subjects


Notes

1 The analysis on which this chapter is based also provided material for Hallström (2022).