5

Computer Science, Interaction and the World

Carsten Schulte, Sue Sentance and Erik Barendsen

Chapter outline

Chapter synopsis

This chapter addresses the implications of digital technology for everyday life and society for teaching computer science at school. The reader will get an overview on how to address these issues as an integral part of teaching computing. Using the opportunity to address the interaction of computing and the world when discussing computing is preferred, instead of addressing it only in separate teaching units which are then presented as a separated perspective on computing.

5.1 Introduction

Interaction might be a strange term to read in a book about computer science education at school. Given the history of the subject at school in most countries you may think that what we mean by interaction is how to use a computer. That is indeed one aspect of interaction, but here we intend to focus on the implications for the interaction between computing and the world.

While it may seem that the idea of interaction being a key aspect of the teaching of computer science in school is new – particularly as there have been questions recently about future jobs (Frey and Osborne, 2013), the future of education and required skills (Sefton-Green, Nixon and Erstad, 2009) and even the future of humanity (Harari, 2017) – the integration of these aspects of computing education date back to the first approaches to computing education in schools in the 1960s, when computers and robots were first used in industry – although at that time there was only sparse computing power.

Teachers (of computer science) generally want to have an understanding of the broad nature of the discipline (see Chapter 2) and their own attitudes towards it (see Chapter 4). This shapes our whole experience of teaching and our perspectives on effective learning. As well as facilitating a secure understanding of the many different aspects of computer science, we also need to prepare students for an ever-changing world where computers affect the way they and others communicate live and work. We have a responsibility to ensure that students are aware of the ethical, moral and societal implications of advances in computer science as part of their core education: this is why these topics appear on school curricula.

In previous approaches to computing education at school, we have seen a gap between core computing topics and what is seen as the 'softer' side of computer science: the general role of computing in individual lives and in society as a whole. This can make the subject rather distorted, if it is chunked in such a way that its application and impact are regarded as being either somewhat separate or not as an essential part of the discipline. In this chapter, we warn against that view, by considering the importance of a broad view of computer science and its implications for our lives, focusing on how this is a key aspect of the learning of computer science in school. We hope we can help teachers to put the pieces together, which in turn will help learners to understand the relationship between computing ideas, technology, individual use and the general effects on society.

Interaction is not a single event, but a sequence of events, including actions and reactions. Consider a user inputting data into a digital artefact, for example a computer: the input is processed by the digital artefact; this produces output, which in turn leads to the next input. This leads us to ask who is the driver in this interaction and who is merely reacting to the input from the other side? Will technology be shaped so that humans are the drivers or will they be programmed and thus told what to do? By simply using the individual pattern or type of interaction as an example of 'how it works in general' we can very quickly debate the future of society as it is changed by ubiquitous computing. Following on from this, we immediately ask the question 'Is society really being shaped by technology or is it the other way around?' This brings us to interaction on a large scale: the interaction between technology and society.

In the age of ubiquitous computing (Weiser and Brown, 1996) almost all areas and aspects of life are penetrated by computing – even taking a shower triggers some computing systems, measuring and controlling data (e.g. for the landlord, the water supply company or municipal bureaucracy). As Kitchin and Doge put it: 'Turning the tap therefore indirectly but ineluctably engages with software, though the infrastructure appears dumb to the consumer who simply sees flowing water.

In other cases, the elevator arrives, the car drives, the mail is delivered, the plane lands, the supermarket shelves are replenished, and so on' (Kitchin and Dodge, 2014). In the past, it was easy to distinguish between computing systems and the world and capture the interplay between computing and society by the automation of processes in socio-technical informatics systems, in which technological systems had a clear border and interacted by clear interfaces with some surrounding social systems. Nowadays, even mundane activities are permeated by numerous technological systems and aspects; it makes more sense to conceptualize the role of technology with the notion of hybrid systems. Understanding health care, production, distribution, household activities and so on, where computational systems are being used, cannot rely on automation alone, but must take into account interaction processes and the ability to perceive the overall situation as a hybrid system. A hybrid system consists of human and digital actors interacting. From an engineering perspective, the goal is to design such systems so that the potential (skills, abilities) of human and digital actors are combined for the best outcome. For example, humans are often better in pattern recognition, whereas algorithms outperform humans in accuracy and speed of computation. From a general or societal perspective, the goals are more unclear, but questions concerning the future development for individuals and humankind are evolving.

In the next sections, this argument will be deepened and discussed within the following framework of interaction levels:

- **Human and computer interaction** refers to forms of interaction within a 'simple' configuration of a person and a computational artefact.
- **Hybrid network interaction** is a term used to capture various forms of interaction in a networked setting consisting of humans and computational artefacts.
- **Computing and society interaction** widens the perspective to the interdependence of technology and society as a whole.

5.2 Level 1: Human and computer interaction

Who is in control?

In this section, we look at the ways in which we interact with computers. We are used to a traditional Input-Process-Output (IPO) model as the way in which we use/interact with computational artefacts, but this is changing. The original IPO model implicitly assumed the core activity of a computer process was to process data: it stems from batch processing and client-server structures of the 1970s, where terminal input was transferred to the server, processed and awaited some checking of the output by the user. In this model, it is important to think clearly about the structure of the input and how data is processed.

However, this has changed profoundly. Based on much shorter response times and advances in user interfaces, the so-called 'What You See Is What You Get' (WYSIWYG) paradigm allows immediate feedback on user input, so it seems as if a user can directly change an item presented on the output. This concept underlies direct manipulation interfaces. Interaction with a computer thus changes from a batch-processing orientated style to a more interactive style with many short input-processing-output cycles which are perceived as direct manipulation (e.g. of text to be written on a screen).

It has been claimed that in future there will only be two types of interaction: program, or be programmed (Rushkoff, 2010). In *Human and computer interaction,* this is conceived as a technological challenge: the problem is to design useful interactions where the individual is 'in control'. In other words, interactions where the human is the programmer, not being programmed.

These changes can be seen not only as a change in usability or comfort level in using a computer, but also a change in computing itself. From the perspective of theoretical computer science Peter Wegner (1997) argues that this new paradigm allows computing beyond the quality level of Turing machines, which describe the quality or power of computing in terms of the IPO model.

The two perspectives on human control (to be programmed or to program) can be illustrated as follows. Let us consider a flight booking system.

Human as user view

This implies that in an everyday end-user scenario the user is guided or prompted by a system which tells her what to do. A customer can search for a flight and then choose and book a specific flight; however, she can do so only in a very restricted way. The system ensures that no missing input is allowed, checks the input thoroughly and only then confirms the booking. The user can choose only between pre-given choices (e.g. choose from a list of flights to book). The system guides this booking process in a predefined order: search, select, add additional input about the customer, and finalize the booking. If, for example, one airline is not in the system, the user cannot book it. Thus, the computer defines what the user can do and the order in which it is done.

Human as designer view

The opposing view would be that the optimization of the system contains some hard problems, including the optimal distribution of customers to flights, additional cargo to be transported if there is space available or a system that allows airlines to charge the highest price possible without losing customers. There may be no optimal solution. However, where designers exist who have experience and intuition they will be able to choose optimal parameters so the system produces the best results. Thus, the human is still in charge of the process.

In summary, one view conceives the user as passive object – the other as a designer who can influence and manipulate the system: system control vs user control.

We can see that the role of end-users is shifting: this role is less and less restricted to a human-as-user perspective. For example, the new intelligent systems that are based on self-learning algorithms usually start outperforming humans in such tasks.

We now have machines that are more powerful than Turing machines because these systems are ultimately based on huge amounts of input data from external sources. Do these systems only get as good as they are because they are trained on human experts? The additional computing power is not only based in the idea of the training or learning algorithm but also in the quality of the

4.4 Exploring our own Mindset and practices as teachers

In our experience, it seems easy to hear about the Mindset work but not necessarily fully absorb it. The teachers we refer to above are in this category. Dweck herself, on the basis of numerous studies (e.g. Sun, 2015), identifies the notion of a *false growth* Mindset, where teachers state they have a growth Mindset but aren't acting as though they do (Dweck, 2015). Their teaching practices, considered as a whole, do not embody a deep understanding of the Mindset principle. In Dweck's words (2015), 'the path to a growth mindset is a journey, not a proclamation'.

Having read this far, where do you think you stand? What is your own view of the potential for any learner to develop their computer science ability? This is a fundamentally important question as computer science is globally transformed from its erstwhile position as optional subject taken only by those who chose it in upper secondary or tertiary education into a mandatory part of all pupils' education starting in early primary. Even as an optional subject, when one might sensibly conjecture that those taking it would have some expectation of success, computer science has had a notoriously high failure rate (Watson and Li, 2014). If one holds a view of computer science ability as innate (fixed Mindset), how can one then subject young people everywhere to a computer science education programme, knowing that large numbers must fail?

Key concept

Question: Is our stated expectation of what learners can achieve (i.e. our Mindset or belief about what can be learned and by whom) at odds with our own teaching practices or those we have experienced?

Here is a question to facilitate an exploration of our position. *Is our stated expectation of what learners can achieve* (i.e. our Mindset) *at odds with our own teaching practices, or those we have experienced?* A teacher who believes that computer science ability is innate (fixed Mindset) most likely has adopted, consciously or unconsciously, a set of teaching practices that mirror his or her belief. For example, the programme may move too fast, provide little feedback, offer no opportunity for catching up and so on.

More importantly, however, even if a teacher says that he or she has a growth Mindset attitude towards the learning of computer science, we conjecture that very often the learning and teaching approaches used are still not sufficiently supportive – they are still likely to foster a fixed Mindset attitude in learners.

Do you disagree with our conjecture? Are you outraged that we might suggest this disconnection between growth Mindset belief and fixed Mindset practices? If so, then consider the following exercise, which we have used with large numbers of teachers to shed light on this disconnect.

training data, and little tweaks and tricks during the training phase. In the human designer view, in the end this quality stems from human cognition, from human input into the system.

Design and meta-design

From a human and computer perspective, Fischer and Giaccardi argue beyond the dichotomist view of design vs use for a spectrum of design activities for users – and programmers as meta-designers, allowing users to re-design the software at use time (Fischer and Giaccardi, 2004) which no doubt would mean that end-user programming and end-user development is a more relevant skill. Another argument for a range of design possibilities beyond use vs design comes from research in artificial intelligence: traditional views on problem solving and computational thinking in computer science conceptualize the problem-solving process as analysis and understanding the problem, followed by design as an analysis and understanding of the algorithmic solution – no problem could be solved without designing an algorithmic solution. With techniques like deep learning, this changes: the problem needs to be understood, and data or examples of problem solutions collected. The machine can then learn from these examples (learn a model of the problem solution) generalize on its own and use this generalization to solve new instances of the problem (see Figure 5.1).

Figure 5.1 can be described as follows. In the top row is shown the classic view on problem solving as programming and finding an algorithmic solution. This requires (a) an understanding of the problem space, and (b) an understanding of the generalized algorithmic solution. The bottom row demonstrates problem solving via model learning. It requires an understanding of the problem

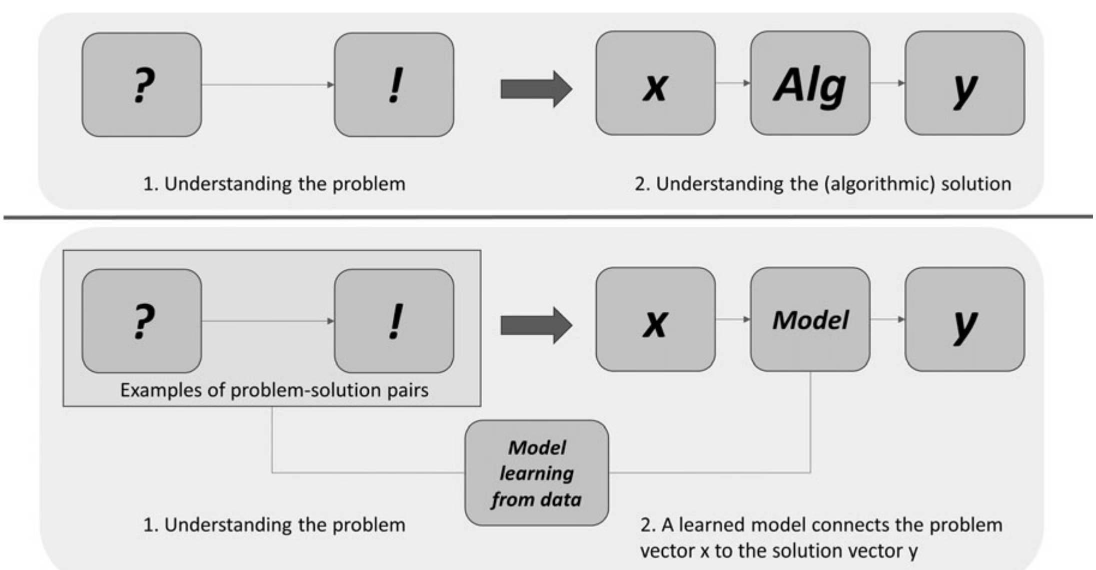

Figure 5.1 Views of problem solving

and getting data through many examples of concrete problem instances and their solutions. By using that data the machine learns the underlying general solution and can solve future problem instances based on the learned model.

These different views (human as designer vs human as user) seem to imply a normative difference (e.g. booking flights is probably optimized so that the airline makes the most profit, not so that the customer gets the cheapest flights). The algorithm uses data from previous users of similar systems, but the humans may not have agreed to this. However, the human as user is not always a negative thing: some people happily use smartphone apps that track their daily routines – the benefit to the person is that the system can present alerts and notifications that get better and better at predicting what they want or need in a certain situation (e.g. they may be prompted to take an earlier train home to ensure they make an appointment).

In summary, the discussed examples should show that human computer interaction cannot solely be evaluated on the basis of the technical features of the system alone. In some situations, it might be very useful if the digital artefact tells the human what to do, in other situations it might be best if the human can adapt or change the system so that it behaves differently in the future.

Key concept: Interaction

Interaction with a computer usually relies on immediate feedback. In Graphical User Interfaces (GUIs) this is also known as WYSIWYG paradigm: What You See Is What You Get. While interaction is based on the IPO-model of Input–Processing–Output, current systems employ quick and short IPO-cycles so that interaction is better defined as *cyclic process of action and reaction between two agents* (e.g. a human and a computer). Interaction is a relationship between structure and function of digital artefacts and the design and use contexts in which the content occurs.

Interaction can be seen on different levels: Human and computer, hybrid network, computing and society.

5.3 Level 2: Hybrid network interaction

Complex configurations of humans and computers

In the beginning, digital devices were perceived as replacement for human computers.[1] From the 1930s on, humans used machines to compute; these machines were called computers. Interaction was carried out by primitive interfaces, which we could call programming. After designing the input, probably consisting of data and an algorithmic description of how to process that data, it was processed within the machine; afterwards – after some time – the output was done. The idea of IPO and the term 'computer' stems from those days. Nowadays, computers are a part of sociotechnical systems.

[1] See: www.nasa.gov/feature/jpl/when-computers-were-human

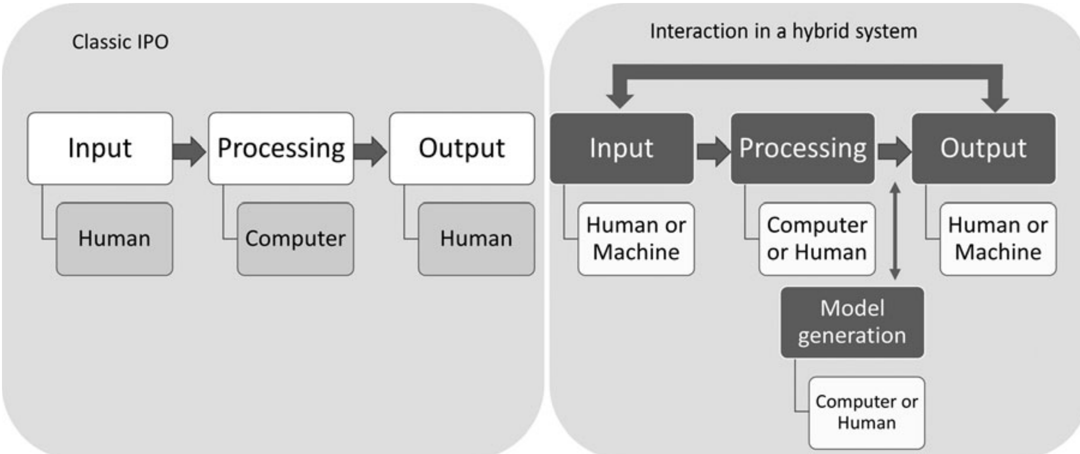

Figure 5.2 Adaptations of the Input-Process-Output model

A hybrid network consists of both human and digital agents participating in data transformation processes, where each partner takes turns and can change its role. Both are important to consider in the different steps of (a) data generation, (b) data transportation and transformation and (c) data consumption.

In such hybrid systems various forms of interactions can be observed (see Figure 5.2). In this figure we see that the old linear model of Input-Processing-Output is transformed by direct manipulation interfaces into cyclic interactions which make the internal computation invisible and make it hard to distinguish who starts the interaction and who is just responding.

The hybrid nature of networks provides possibilities, but also raises the need to protect against unintended interactions. Many webpages containing forms for human use are protected against (presumably malicious) computer-computer interactions by so-called *Captchas* with the purpose of detecting whether or not the interaction partner is human.

Human vs computer traits in interaction

The computer science researcher Keil is interested in designing computational artefacts so that humans and machines in interaction processes can contribute their particular strengths (e.g. Hampel & Keil-Slawik, 2001; Keil-Slawik, 1992) as introduction to his work).

Keil argues that describing technological and human actors as (in principle) the same or as equal partners in interaction unavoidably leads to misunderstandings due to unjustified transfer features of technology to humans, or vice versa. For example:

- A human forgets details; in a machine data can only be deleted.
- A machine can be instructed to exclude some aspects; a human cannot be instructed to (e.g. 'not think about the next break!').
- A human can learn from mistakes; a machine (at least not in the foreseeable future) cannot learn from mistakes, or only in a frame foreseen by the developers.

- How machines process data is based on precisely defined steps (algorithms), human cognition is not yet fully understood, but relies in contrast to machines on the body and physical experiences (Barsalou, 2008).

Keil therefore suggests that we should clearly distinguish between technical *structures* and technical *effects* and social/cognitive *structures* and social/cognitive *effects*. Key to this understanding is the idea that social/cognitive processes to some part entail or are based on technical processes.

Implications for data processing

For example, this idea is visible in the way data is processed in a hybrid network: such data will need to be digitized. Indeed, human perception and memory rely on external stimuli; humans can use media as external memory. By representing data in some media we don't need to memorize all the details (e.g. written language is storing what was oral data beforehand in a new medium). By using external storage mechanisms to store the process of a calculus (e.g. summing up the value of several items), a human can then reperceive and thus trace the calculation process to check for errors. In analogous media, this storing of data is done by inscription. The visual inscription and the visual layout of the data *is* the data. Thus, computers as digital media have brought a fundamental change in that for the first time the data itself and its visual layout are separated. A computer can produce different visual layouts without destroying the original data! Perceiving different representations of the same data can help humans to understand them better. Keil labels this process 'the experience of difference' – the potential for producing all kinds of different representations makes the algorithmic processing of data such a powerful technology in terms of 'mind tools' or thinking tools, which Keil labels as digital media.

Key concept: Hybrid network interaction

A hybrid network consists of human and digital agents participating in data transformation processes, where each partner takes turns and can change its role. Both human and digital agents are important to consider in the different steps of (a) data generation, (b) data transportation and transformation, and (c) data consumption. In such hybrid systems, various forms of interaction can be observed. The hybrid nature of networks provides possibilities, but also raises the need to protect against unintended interactions.

5.4 Level 3: Computing and society interaction

Objectivation

One approach introduced in the German literature on the philosophy of technology and computing education (Frank and Meyer, 1972) proposed a basic or philosophical account of digital technologies, to account for the anticipated tremendous growth and development of that technology. The core idea is that the advent of technology in our lives brings with it the final evolutionary step of the development of humankind. Frank and Meyer suggest three evolutionary phases, each associated with substantial changes in the self-view of people and the overall abilities of humankind:

1 Phase: The so-called objectivation of organs and limbs (e.g. teeth, hands by tools like stones or hand axes).
2 Phase: Objectivation of physical work by machines.
3 Phase: Objectivation of intellectual work by computing devices.

To make sense of this view, we would interpret *objectivation* in Phase 1 as automation. In Phase 2 we can see that formerly essential human skills are becoming obsolete due to new technological developments: machines now do the work of humans. Phase 3 adds even more; it is on a larger scale, and concerns the very nature of what it means to be a human. What is it that distinguishes us from animals? It is not the physical strength nor the ability to socialize, but our intellect: the ability to think, and deductive and logical reasoning. We could regard this as a development that shatters the last resort of humans' uniqueness: our intellectual capabilities. Humans become ever-more marginalized in this view. We should, however, embrace this development, because humans are weak, laden with emotions, prejudices and with a tendency to rush decisions. Humans have individualistic and subjective approaches to any task. Technology, however, can now lead humankind to an era where all decisions can be objectivated (hence the term) and thus become fair, just, devoid from subjective emotions, prejudices and values. For this to happen, the task at hand just needs to be transformed into an algorithmic problem and the input variables transformed into digital data.

The goals of education in the context of moving from the subjective to the objective era are pretty straightforward: instead of objectivation the terms now in use are problem-solving skills, algorithmic thinking or computational thinking (Tedre & Denning, 2016) – but all of them still aim at marginalizing subjective humanity by processes of automation. This is the reason the topic under discussion is *interaction* and not *automation* – hybrid and socio-technical systems have interaction as the dominant-use behaviour instead of conceptualizing automation as Input-Processing-Output where the problem solving is entirely done by and delegated to the machine.

Socio-technical perspective

Another way of viewing change in technology and its impact is through the lens of socio-technical approaches (Magenheim and Schulte, 2006). The aim is to uncover the interplay between technology and society (and the individual actors or agents). In these approaches, the interaction between human and technology is a central notion for understanding how technology is constructed and shaped, in its turn shaping society as well as the human being. In traditional approaches, agency is defined as the purposeful acting (e.g. planning, foresight and making decisions) and is a term used solely for human actors. With the increased responsiveness and autonomy of digital interactive technologies we can give them the label 'secondary agency' (Mackenzie, 2006). This can be seen as a substitute or imitation of human agency. The dichotomy of product and process mirrors the fact that during design time, a solution is built with a world in mind that ceases to exist when the product is in use – thereby changing the situation so that most likely another or different ideas and solutions emerge. This leads to the next generation of products, thus forming a development path.

Structure and function

Kroes and Meijers (2006) argue that artefacts (things made by humans) cannot be fully understood by what we could call the science perspective ('how does it work?'). Physics, for example, could deliver a physical description of a hammer such as weight, material, etc. However, this would not explain why humans call it a hammer, and perceive it as a tool for hammering. Therefore, two perspectives are needed to fully grasp a technological or digital artefact: *structure* and *function*. That gives us another reason why computer science *cannot* exclude the function perspective to teach both concepts relating to computer science and also the interaction between computing and society, and the individual experiences. The concept of duality reconstruction uses this as a basis for an approach that supports developing teaching units for computer science education that relate to interaction (Schulte, 2008).

Key concept: Computing and society interaction

The societal, as well as the ethical and value-laden, implications of the use of computer science can be captured using varying models, giving both sides different roles. Here we suggest capturing the process as interaction between two sides – human and digital – or technological actors glued together by interaction processes. Both can essentially play the same roles – especially influencing the other side – but do so based on different perspectives and skill sets.

5.5 Implications for pedagogy

Interaction is a relationship between structure and function of digital artefacts and the design and use contexts in which the content occurs. What does this imply for teaching computer science?

Teaching interaction is not a new idea. However, we suggest perceiving interaction not as additional content to be taught and also not as a recipe for teaching, but a perspective for thinking and designing teaching. It requires a mindset useful for thinking and reflecting that shapes how content, teaching methods and goals are chosen and integrated. This approach is not claiming to change content, but to think about a wider perspective and the underlying educational goal when teaching computer science content (see chapter 10.5 on page 140 for an example).

In this way, we might avoid the pitfall of teaching computer science by teaching children absolute facts about technology (i.e. by teaching them 'how it really works' before – or even without – focusing on interaction aspects ('what it is good for'). In light of the discussion above, we hope it is clear that by such separation of concerns, the role of digital technology and of the underlying ideas and concepts from computer science for everyday life, together with their impact on the changes and development of contemporary societies in all their aspects, cannot be captured. In order to achieve this when teaching computing, letting students discover the dual nature of digital artefacts is crucial (Schulte, 2008). The same holds for the intertwined relationship between product and process (Magenheim and Schulte, 2006).

When planning teaching, one can analyse and reflect on possibilities for the different levels and the different viewpoints within each level. In Figure 5.3 we can see that interaction within a nd between the three levels can be analysed by looking at structure, function and contexts. Using this scheme it is possible to reflect on societal issues based on knowledge and experiences gained from levels 1 and 2, so that the societal issues are closer linked to concrete examples and experiences. Teaching level 3 in isolation could result in teaching sociology, but by grounding it in level 1 examples, the abstract implications and reflections can be integrated. It therefore seems useful to switch between levels in a teaching unit, to demonstrate the interaction between levels.

This can also be described as a pars-pro-toto principle; students should be enabled to make inferences from a small example to the whole – to transfer and generalize. For planning teaching this means to ask which example can be chosen that allows such generalization.

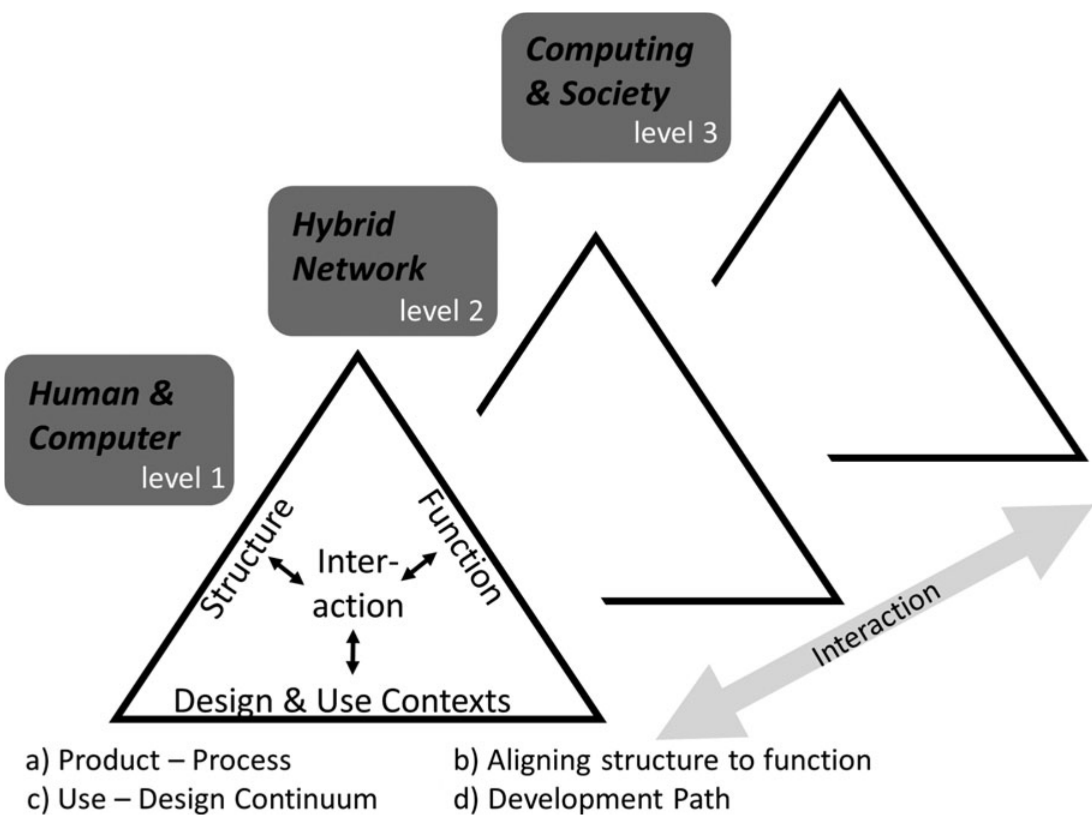

Figure 5.3 Interaction within and between the three levels

Example activity

Interaction can be explored based on analysing a digital artefact.

Your task is to pick a digital artefact of your choice and to describe it with regard to the three levels.

Take a word processor for instance – presumably you are familiar with using one, so the function aspect is easy to describe – but what about the underlying structure? For example, how is digital text represented and saved on a hard drive? How is it rendered – what algorithm is used for text alignment? The interaction between algorithmic layout procedures and user functions can be explored. Such aspects demonstrate interaction level 1. Can you find similar examples at level 2? (E.g. if you are text processing in the cloud, is there any algorithmically generated text? What hybrid interactions (human–computer, computer–computer, human–human) can you recognize?) Finally, at level 3, what are more general implications of the digitalization of text processing (e.g. change in job description, change in text-based media, . . .)?

Extension activity: Can you see interactions between the levels?

With respect to level 1, students should become aware of the different views of the human-as-user vs the human-as-designer; that there is a continuum of interaction pattern between the two extremes (Fischer and Giaccardi, 2004).

As to level 2, it is important to discuss how the hybrid nature of human-computer networks affects the interfacing and the way data is processed, as well as the nature of problem solving (see Figure 5.3).

Regarding level 3, the role of objectivation in society in light of the dichotomy of product and process and how this triggers the intertwined development path of society and technology should become clear.

We illustrate our approach with an example in the area of robotics.

Example: Robotics

Robotics might be a common topic to discuss with students. The starting point for thinking about how to introduce this topic might be to analyse robots on level 1 and the IPO-model. A robot can be seen as a hybrid system where sensors provide input values (measuring distance, colour of underground, etc) and process this to adapt the driving behaviour (e.g. follow a line or avoid bumping into obstacles). This can be observed in small educational robots, but also in everyday households using robot vacuum cleaners and most likely in the future in robot cars – which provides a use context, where we could analyse the role of the human in the interaction. Who is in control when driving the robot car? After specifying the destination, who is in control of choosing a route? Will the human be perceived as a passenger (*Human-as-user view*) or can they still be the driver (*Human-as-designer view*) (i.e. configure and adapt how the robot car operates)?

With regard to the idea of design and meta-design, questions can be asked such as how such cars can be built to allow for design at use time; and where are technical, legal and ethical issues to prevent the user from re-designing the robot car system. This can be discussed largely on the basis of the classical IPO model by experimenting or so-called 'Exploring' (Schulte et al., 2017) how sensor data are processed and control actuators.

This then quickly widens to the perspective of hybrid networked systems, because many features of robot cars will depend on them communicating both with each other and the cloud. In this network view, again the question arises of how best to acknowledge human vs computer traits in this interaction and the implications for data processing (e.g. changing the traffic environment to be more machine readable).

This quite naturally engages a debate on the level of computing and society. Most likely, the behaviour of humans with regard to driving cars will change. Such experiences affect the difference between product and process (Magenheim & Schulte, 2006) and will most likely lead to changes in product as well as in the environment (development path (Schulte, 2008)). One can debate about how robot cars will change transport systems, jobs, ownership, legislation and transform urban space and traffic patterns.

5.6 Summary

It sometimes seems there is a dichotomy between ideas and technology. With interaction as a central notion we aim to lay the foundations for a relational understanding. This provides the depth which is sometimes lacking in the teaching of computer science and prevents us from teaching digital artefacts from a use – or indeed from a structural – perspective only.

Teaching in an integrative way would mean that for each computer science topic, an activity or reflection is added to draw out the implications of this topic for human–computer interaction, human–human interaction and human–computer–society interaction. By also addressing the interaction *between* the different levels it is possible to address societal issues of computing without either overwhelming student on the one hand, and also without detaching the discussion of societal issues from the domain of computing. This approach might be very useful in providing a context for computer science theory and some motivation for studying it.

Key points

- While it sometimes makes sense to teach topics from the field of computing and the world in isolation, we aim for an integrated approach, where the interaction between technology and society becomes more visible.
- Interaction can be analysed on three levels: human and computer, hybrid networks and computing and society. A conceptual analysis gives rise to key properties within each of these levels.

- These key points can be addressed in computing education by analysing interaction in concrete systems. It is crucial to connect these, so examples should allow for an analysis on all three levels.
- The interplay between *structure* and *function* of digital artefacts implies that interaction should not be taught as a separate notion, but rather in conjunction with the question how digital artefacts work.

Further reflection

- Ask students to consider systems that they have used for their own entertainment (e.g. to play, download or store music). Let them reflect on the balance between the system affecting control over the user's use of the system ('listen to this') and the benefits of using the system.
- Ask the students to discuss a system they know and outline some feature or problem of the system that plays a role on each of the levels of interaction.
- Review the local formal curriculum. Look for explicit and implicit occurrences of interaction aspects. Classify them in terms of the interaction levels. Suggest how to integrate the aspects you found with the more 'functional topics' in the curriculum.

References

Barsalou, LW (2008). 'Grounded Cognition' 59 *Annu Rev Psychol* 617–645.

Fischer, G and E Giaccardi (2004). 'Meta-design: A Framework for the Future of End-User Development' (Online-Version) in H Lieberman, F Paternò and V Wulf (eds), *End User Development* (Vol 9) (Dordrecht, The Netherlands, Kluwer Academic Publishers,) 1–26.

Frey, CB and MA Osborne (2013). 'The Future of Employment: How Susceptible are Jobs to Computerisation?' 17 September 2013. Available at: http://archedepotoire/autoblogs/wwwinternetactunet_8a3fe3331e0ad7327e18d9fe6ec3f0ad04dcea58/media/3722fa7dThe_Future_of_Employmentpdf

Hampel, T and R Keil-Slawik (2001). 'sTeam: Structuring Information in Team-distributed Knowledge Management in Cooperative Learning Environments' 1(2es) *J Educ Resour Comput* 3.

Harari, YN (2017). *Homo Deus: A Brief History of Tomorrow* (London, Vintage).

Keil-Slawik, R (1992). 'Artifacts in Software Design' in C Floyd, H Züllighoven, R Budde and R Keil-Slawik (eds), *Software Development and Reality Construction* (Berlin Heidelberg, Springer) 168–188.

Kitchin, R and M Dodge (2014). *Code/Space* (Cambridge, MA, MIT Press).

Kroes, P and A Meijers (2006). 'The Dual Nature of Technical Artefacts' 37(1) *Studies in History and Philosophy of Science Part A*, 1–4.

Magenheim, J and C Schulte (2006). 'Social, Ethical and Technical Issues in Informatics—An Integrated Approach' 11 (3–4) *Education and Information Technologies* 319–339. Available at: wwwspringerlinkcom/content/658435n54601048x/

Rushkoff, D (2010). *Program or Be Programmed: Ten Commands for a Digital Age* (New York, NY, Or Books).
Schulte, C (2008). 'Duality Reconstruction – Teaching Digital Artifacts from a Socio-technical Perspective' in RT Mittermeir and MM Syslo (eds), *Informatics Education – Supporting Computational Thinking: Third International Conference on Informatics in Secondary Schools – Evolution and Perspectives, ISSEP 2008, Torun Poland, 1–4 July 2008 Proceedings* (Berlin, Heidelberg, Springer-Verlag) 110–121.
Schulte, C, J Magenheim, K Müller and L Budde (2017). 'The Design and Exploration Cycle as Research and Development Framework in Computing Education' in *Global Engineering Education Conference (EDUCON), 2017 IEEE* 867–876. Available at: http://ieeexploreieeeorg/abstract/document/7942950/
Sefton-Green, J, H Nixon and O Erstad (2009). 'Reviewing Approaches and Perspectives on "Digital Literacy"' 4(2) *Pedagogies: An International Journal* 107–125.
Tedre, M, and PJ Denning (2016). 'The Long Quest for Computational Thinking' in *Proceedings of the 16th Koli Calling International Conference on Computing Education Research* New York, 120–129. ACM.
Wegner, P (1997). 'Why Interaction is more Powerful than Algorithms' 40(5) *Communications of the ACM* 80–91.
Weiser, M and JS Brown (1996). 'The Coming Age of Calm Technology' 8 *Xerox Parc* 17. Available at: wwwjohnseelybrowncom/calmtechpdf