

Code as personal data: implications for data protection law and regulation of algorithms

Nadezhda Purtova* and Ronald Leenes**

Key Points

- This article argues that some instances of computer code are personal data under the General Data Protection Regulation (GDPR). We explore what this means for data protection and regulation of automated systems.
- Drawing from information studies, we argue that although computer code often does not contain semantic information and does not mean anything ‘about’ people, it ‘is’ syntactic information and therefore ‘any information’ in the sense of the definition of personal data in the GDPR.
- Code ‘relates to’ its designers when it reveals information about them, such as preferred programming style, but also when it is designed and deployed with the purpose of evaluating, treating in a certain way or influencing the status or behaviour of its users and targets, or has unintended impacts on their rights and interests.
- The designers, users, and targets of code are often identified, eg by their names in their digital dossiers, software use patterns, and software licence numbers, or identifiable, eg when code such as biometric recognition systems is intended to identify people.

- This argument can be used to criticize the scope of the GDPR as too broad. Yet, applying the general principles of data protection and data subject rights to code does not lead to absurd outcomes and adds a layer of legal protection against risks of information society. At the same time, it highlights the need to reconsider if legal protection against digital harms should hinge on the concept of personal data.

Introduction

This article challenges the long-standing distinction between personal data and code. We argue that many instances of computer code can be personal data in the sense of the data protection law and explore the implication of this argument for data protection and regulation of automated systems.

Regulation of algorithms to address the challenges of algorithmic governance and regulation, and automated decision-making has been in the spotlight of the recent scholarly and policy debates.¹ There have been numerous calls for regulatory intervention and regulatory initiatives focusing either on algorithms generally or on their specific applications, eg the opinion of the German Data Ethics Commission on AI² and EU strategy on

* Nadezhda Purtova, Faculty of Law, Economics, and Governance, Utrecht University, Utrecht, The Netherlands. Email: n.n.purtova@uu.nl

** Ronald Leenes, Tilburg Institute for Law, Technology, and Society, Tilburg University, Tilburg, The Netherlands.

1 The reader might benefit from Katzenbach and Ulbricht’s explanation of the concepts of algorithmic governance and algorithmic regulation in Christian Katzenbach and Lena Ulbricht ‘Algorithmic governance’ (2019) 8 Internet Policy Rev1. Automated decision-making refers to the decision-making based solely on automated processing of personal data in the sense of Art 22(1) GDPR. What constitutes automated decision-making in this context is a contested issue in the EU data protection law. Binns and Veale explore what can constitute automated decision under

Art 22(1) GDPR in Reuben Binns and Michael Veale ‘Is that your Final Decision? Multi-stage Profiling, Selective Effects, and Article 22 of the GDPR’ (2021) 11 IDPL 319. According to AG Pikamäe, it is a point of view or opinion of a binding nature, where human participation in the procedure ‘does not affect the causal link between the automated processing and the final decision’ (SCHUFA, C-634/21, [2023] CJEU (ECLI:EU:C:2023:220), Opinion of AG Pikamäe, [37], [44]).

2 Data Ethics Commission of the Federal Government of Germany, ‘Opinion of the Data Ethics Commission’ (Berlin, December 2019) <<https://www.bmi.bund.de/EN/topics/it-internet-policy/data-ethics-commission/data-ethics-commission-node.html>> last accessed 19 April 2023.

Artificial Intelligence,³ and facial recognition,⁴ culminating in the proposed EU AI Act.⁵ At the same time, the General Data Protection Regulation (GDPR)⁶ has been increasingly more often resorted to for legal remedies against harms caused by or in the context of algorithmic systems.⁷ Yet, a dominant view is that the GDPR's potential for regulating automated systems is limited. The GDPR is triggered by the processing of personal data, which arguably does not always happen throughout the entire operating cycle of an automated system.⁸ Also, the effect of the GDPR on computer code that, in addition to data, is another essential component of an algorithmic system, is typically incidental, and a by-product of the requirements for how personal data ought to be processed.⁹ In this article, we offer an alternative view of the role that data protection law can play in tackling risks associated with algorithms. We argue that in many instances computer code can be considered as personal data under the GDPR and explore what impact this has on the level of legal protection against harms associated with the algorithms. This has not yet been done in the literature. Veale, Binns, and Edwards have recently argued that model inversion and membership inference attacks may render Machine Learning (ML) models personal data,¹⁰ and Leiser and Dechesne¹¹ have written a critical reply. Yet, this dispute is confined to the narrow context of ML models, and essentially revolves around two issues, namely (i) whether or not natural persons whose personal data was included in the training set of a model are identifiable and consequently, if a ML model

itself can be considered pseudonymized personal data and (ii) whether such an application of the GDPR is desirable. Veale, Binns, and Edwards argue yes on the first issue and are cautiously optimistic on the second.¹² Leiser and Dechesne argue no on the first issue, citing *inter alia* that any re-identification attacks would have to be illegal, and caution against further expansion of the scope of the GDPR on the second.¹³ In contrast, this article makes a broader argument that computer code generally, as opposed to just one type of it, being a ML model, under some circumstances can be personal data. Unlike the ML model debate highlighted earlier, our argument equally focuses on all the elements of the concept 'personal data' rather than primarily on identification and is broader than a discussion of pseudo- and anonymization. We also do not make any normative claims and do not argue that code 'should' fall within the scope of the GDPR. We merely make an observation that—given how the scope of the GDPR has been interpreted so far—some instances of computer code *are* personal data and hence the GDPR can apply. We present both positive and negative consequences of this state of affairs. Finally, part of the dispute in the existing analyses is whether or not ML models are or contain data. Yet, neither Veale and others nor Leiser and Dechesne devote attention to what data or information—the notion used to define personal data in the GDPR and intimately linked to data—is, thus continuing the pattern of under-theorized use of information concepts in law.¹⁴ In contrast, we heavily draw from

3 European Commission, 'Artificial Intelligence – A European Approach to Excellence and Trust' (White Paper) COM(2020) 65, 19 February 2020.

4 Eg, Consultative Committee of the Convention for the Protection of the Individuals with regard to Automatic Processing of Personal Data, 'Guidelines on Facial Recognition' published 28 January 2021 T-PD(2020)03rev4; European Parliament resolution of 20 January 2021 on artificial intelligence: questions of interpretation and application of international law in so far as the EU is affected in the areas of civil and military uses and of state authority outside the scope of criminal justice.

5 Proposal for a Regulation (EU) 2021/0106 of the European Parliament and of the Council laying down harmonized rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts COM/2021/206 final.

6 Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), OJ 2016 L 119/1.

7 Eg in October 2020 Uber and Ola drivers have invoked the GDPR rights to challenge before the Amsterdam Court the cancellation of their accounts with those platforms, amounting to a *de facto* firing (Anton Ekker, 'Uber Drivers Challenge Dismissal by Algorithm' (26 October 2020) <<https://ekker.legal/2020/10/26/uber-drivers-challenge-dismissal-by-algorithm/>> last accessed 19 April 2023).

8 Wachter and Mittelstadt argue that the data inferred by an algorithm is not personal data and hence does not fall within the GDPR. They call for a creation of a new extra-GDPR right to a reasonable inference. (Sandra Wachter and Brent Mittelstadt, 'A Right to Reasonable Inferences: Re-

Thinking Data Protection Law in the Age of Big Data and AI' (2019) 2 Columbia Bus Law Rev 494). See also Mireille Hildebrandt and Bert-Jaap Koops, 'The Challenges of Ambient Law and Legal Protection in the Profiling Era' (2010) 73 MLR 439. But see Purtova arguing to the contrary [Nadezhda Purtova, 'The Law of Everything. Broad Concept of Personal Data and Future of EU Data Protection Law' (2018) 10 Law Innov Technol 40].

9 Eg the list of safeguards for fair and transparent processing in case of automated decision-making in Recital 71 GDPR among others includes the use of appropriate mathematical or statistical procedures for the profiling, and technical 'measures appropriate to ensure ... that factors which result in inaccuracies in personal data are corrected and the risk of errors is minimized'.

10 Michael Veale, Reuben Binns and Lilian Edwards 'Algorithms that Remember: Model Inversion Attacks and Data Protection Law' (2018) 376 Philos Trans Royal Soc 1. We are aware that outside the debates on data protection law and the legal meaning of personal data models and data are often seen as two distinct things. See eg Sabina Leonelli, 'What Distinguishes Data from Models?' (2019) 9 Eur J Philos Sci 21. See also Mark R Leiser and Francien Dechesne, 'Governing Machine-Learning Models: Challenging the Personal Data Presumption' (2020) 10 IDPL 187.

11 Leiser and Dechesne (n 10).

12 Veale, Binns and Edwards (n 10) 6 ff and 8 ff.

13 Leiser and Dechesne (n 10) 193 ff and 199 ff.

14 See Lee A Bygrave, 'Information Concepts in Law: Generic Dreams and Definitional Daylight' (2015) 35 OJLS 91, 91.

information studies, a body of disciplines studying information and how it relates to reality, to conclude that computer code ‘is’ information in the sense of the definition of personal data used in the GDPR.

The question of whether or not computer code can be personal data and fall within the scope of the GDPR and what consequences ensue is more than simply a piece of legal gymnastics. Our argument can potentially serve three causes. First, it can support the critics of the data protection law who argue that the current broad understanding of the concept of personal data is unreasonable and undesirable since it leads to the problem of overinclusion of the data protection law.¹⁵ Indeed, data protection law was never meant to apply to computer code. Showing that the current broad scope of the GDPR also encompasses software may serve as an argument *ad absurdum*, to argue that the concept of personal data surely should not be interpreted this broadly. Second, our analysis can provide digital rights advocates with ammunition to claim access to and control over the use of algorithms under the GDPR where those rights do not yet exist under alternative legal regimes. This is important given how profoundly computer software, including but not limited to AI, shapes our lives, and considering that the legal protection against possible negative impacts of software use is still being formed. To illustrate, a right to obtain a copy of the code itself under Article 15(3) GDPR could facilitate algorithmic audit and accountability. The principle of purpose limitation applied to code would provide solution to the so-called ‘function creep’ problem where software acquires new unforeseen and undesirable uses, eg smartphone apps originally marketed for tracking children to ensure their safety but repurposed for stalking. The principle of ‘accuracy’ could translate into the requirement that the code is built on a sound scientific method. We specifically consider how applying general data protection principles and some data protection rights to computer code may enhance legal protection.¹⁶ The third and final cause links with the first one but is of a more fundamental nature. The exercise of applying the principles of data protection law to code calls to reassess the regulatory objectives of the EU data protection law and to critically reflect on the chosen mechanisms to achieve those

objectives in light of the regulatory design. We will revisit those causes and what our analysis means for each of them in the conclusion.

The analysis starts with the definition of code, introducing object and source code and compilers. We then briefly deal with the meaning of personal data under the GDPR and apply the definition to computer code. We argue that—while computer code does not always contain meaningful semantic information about people—still, all computer code is information in the sense of the GDPR, since the GDPR also includes the so-called ‘syntactic information’ defined not by its meaning but by its impact on reality. Code may contain information about people, and is increasingly designed and used to assess or otherwise impact our behaviour and interests. Even when the code is not explicitly designed to assess or impact people, it unavoidably does so as an affordance of design. For this reason, code can be said to be information that relates to people in content, purpose of effect. Finally, people to whom code relates—code designers, users, and targets—often are or can be identified. All this leads to the conclusion that some instances of computer code do constitute personal data under the GDPR. Finally, we examine what this conclusion means for data protection and regulation of algorithmic systems generally, among others, in light of the upcoming AI Act.¹⁷

What we mean by code: terminology and basic structures

This section explains what this analysis means by code and introduces other relevant computing terminology. At the risk of oversimplification of realities of modern computing, the considerations below only focus on the basic definitions and structures and go as far as it is relevant for the analysis in the subsequent sections.¹⁸

In the 1970–1980s which were the early years of modern computing, the relevant terminology used to be fairly simple. On the one hand, there was the computer, ie the physical hardware that executed instructions to transform input into output. On the other hand, there were the instructions that told the computer what to do, called computer software, or software for short.¹⁹ Since then the software landscape has become quite complex

15 Eg Gerrit-Jan Zwenne, ‘Diluted Privacy Law’ (Inaugural Lecture, University of Leiden, 12 April 2013) 3 <<https://scholarlypublications.universiteitleiden.nl/access/item%3A2930872/download>> 3ff. arguing that the concept of identification is too broad; Raphaël Gellert, ‘Personal Data’s Ever-Expanding Scope in Smart Environments and Possible Path(s) for Regulating Emerging Digital Technologies’ (2021) 11 IDPL 196; Bert-Jaap Koops, ‘The Trouble with European Data Protection Law’ (2014) 4 IDPL 250, 251.

16 See ‘What this means for data protection and regulation of algorithmic systems’ section.

17 Council of the European Union, Proposal for a Regulation of the European Parliament and of the Council laying down harmonized rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts—General approach, Brussels, 25 November 2022.

18 For a comprehensive review of the nature of code see eg Rob Kitchin and Martin Dodge, *Code/Space: Software and Everyday Life* (MIT Press 2014) 24ff.

19 The term ‘software’ seems to be first introduced in John W Tukey 2.

with many (types) of languages, libraries of reusable functions, services that can be called, etc. Yet, the basic distinction between computer hardware and software remained relevant.²⁰ In this article, we understand ‘code’ as computer software.²¹

Depending on its position in the human-to-machine communication of instructions, computer code generally comes in two types, or layers: source code and object code. There is a third type of code that transforms source code into object code, compilers and interpreters. Source code, otherwise referred to as a program, is a set of instructions written by a human programmer in a high-level human-readable programming language such as Python-, C++, or Javascript. The source code expresses a sequence of instructions envisioned by the programmer (or coder as they are most often called nowadays) to perform the task assigned to the program. There are many kinds (and terms to label those kinds) of programs.

In order for a computer to carry out the instructions written in the source code, they will have to be translated into the object code, ie instructions in a machine language, eg binary code, that the machine can execute. Different computing platforms have different make-up in terms of processor, available and type of memory, input and output, and what have you. While the source code describes what needs to be done by the computer in relatively abstract terms, the object code specifies how this is to be done by the specific hardware configuration, for instance by putting data in the right order for the specific processor (eg, Big Endian or Little Endian). The translation from the source code to the object code is done by a compiler (for a one-time translation) or interpreter (each time the application runs). Thus, the source code is compiled or interpreted into object code. Compilers and interpreters are universal or generic, and they can process any code written in the computer language. Source code is very specific, and describes how to

perform a particular/specific (set of) task(s). In computer–human communication, the source code is a layer of code closest to the human programmer, and any programmer proficient in Python can understand any Python source code.²² Object code is the version of a program that can actually be executed by the Central Processing Unit of the computer. In principle, object code is readable for humans, but it is much more challenging to reconstruct what the code is supposed to do from the object code than from the source code. This is the case because source code generally incorporates meaningful names for variables (eg `TimeOfDay`) and constants (eg `pi = 3.14`) and comments incorporated by the programmer to explain to human readers what the code is supposed to do, whereas object code has the comments stripped and high-level language constructs (loops, conditionals, selectors, etc) and names of variables, etc transformed in simple instructions and registers (eg `mov r1, #(1<<3) str r1, [r0]`). Object code can be transformed into ‘source’ code again by the inverse process of compiling or interpreting called decompiling. Decompiled programs lack (humanly) meaningful names for things such as functions, procedures, variables, and constants as well as the comments that may have been present in the original source code because all of these were lost in the initial transformation from source to object code.

In the Internet era, computers do not only run software that has been compiled or interpreted on the local hardware but instead may use a hybrid mix of software that exists on local hardware and parts that are brought in at runtime, whenever necessary, from remote servers. The local part of the software is relatively stable, and the remote software may be highly dynamic (different in its details today from tomorrow).²³

Another relevant concept to unpack here is ‘algorithm’. A long-standing distinction in computing is between algorithms, data (structures), and programs.²⁴

20 At the same time, we are aware of the phenomenon of hardware as a service and the resulting growing obfuscation between hardware and software. Traditionally, the software is aimed to run on dedicated hardware, eg a mainframe, a PC, a tablet, or smartphone. Nowadays, code can emulate specific hardware on different hardware, thereby further obfuscating the borders between hardware and software. ‘Hardware, once objectivized as a physical computer, is becoming distributed across different data centers and dissolving completely into infrastructures. And software [...] is dissolving in a cascade of services that organize access to data and its processing’ (Irina Kaldrack and Martina Leeker, ‘Introduction’ in Irina Kaldrack and Martina Leeker (eds), *There is no Software, There are Just Services* (Lüneburg, Meson Press 2015)).

21 Following Lessig, both software and hardware constitute code as a regulatory instrument [Lawrence Lessig, *Code. Version 2.0* (New York, Basic Books 2006)]. In this analysis, however, we will abstract away from the hardware and confine the notion of code to the software.

22 In theory at least. As anyone who has ever tried to amend/apply/fix code written by anyone else knows that code usually is not self-explanatory and personal styles and preferences often make it difficult to understand

what the code does or why it does so and not in a different manner. Many performance gains are to be achieved by using expressions closer to object code at the detriment of readability. In C++, for instance if `(hour < 13) {hello="good morning";} else {hello="good afternoon";} and hello = (time < 13) ? "good morning" : "good afternoon" ; produce the same result. Arguably the latter is less readable (for beginners).`

23 An example of such hybrid software is Microsoft Teams. Although it can be run inside a webbrowser, there is also a hybrid version that uses a core that people can download to their PC or Mac, but much of the functionality is provided by online services. The reliance on server components allows for great flexibility for the developers. For a good overview of modern software development and deployment, see Seda Gürses and Joris van Hoboken, ‘Privacy After the Agile Turn’ in Evan Selinger and others (eds), *The Cambridge Handbook of Consumer Privacy* (Cambridge, CUP 2017). For our analysis these dynamics imply that the actual ‘code’ that may be personal data at t1, may cease to be at t2 (or vice versa) because parts of the code may have been altered between t1 and t2.

24 Niklaus Wirth, *Algorithms + Data Structures = Programs* (Englewood Cliffs, New Jersey, Prentice-Hall 1976).

While there is no single commonly used definition of algorithm, the prevalent understanding of an algorithm is as a finite sequence of instructions to transform some input into some output.²⁵ There may be a number of different algorithms to perform a certain task all leading to the same result, but achieving it in different manners. An algorithm for sorting a list of names would for instance be what is known as insertion sort: given an unsorted list, take elements from the list one by one (step 1) and place them in the correct place in a new sorted list (step 2). An algorithm as a set of instructions is encoded into the source code and then compiled into the object code. One could argue that a computer program implements one or more algorithms, where the algorithm represents the main task(s), whereas the rest of the program describes supporting tasks [for instance, how to get input to the sorting algorithm or how to present the results (on screen)].

In the lay usage, notably in legal and ethics debates on the topic of responsible AI and ML, the distinction between an algorithm and a computer program implementing that algorithm has disappeared, and a computer program implementing algorithms is called an algorithm. This article, unless explicitly stated otherwise, will also use ‘code’ or ‘program’ and ‘algorithm’ interchangeably.

Many problems can be tackled by algorithms created by a human programmer. Yet, other problems may be difficult to impossible to address by a human-made algorithm. Differentiating spam from genuine email is an example of a problem that is difficult to grasp by a human-made algorithm because it may be difficult to determine rules that distinguish between them. In those cases, ML on the basis of large sets of examples may produce a tacit way to transform input into output and resolve the problem. For instance, a computer can be taught to detect spam by having humans mark spam in their inbox and then have the algorithm figure out what features may distinguish spam from non-spam.²⁶ In this regard, with the arrival of ML, the role of an algorithm towards data has shifted. While in classical, pre-ML scenario algorithms are executable instructions designed to

perform a specific task in a specific way operating on data, in the ML context the algorithms emerge from data through either explicit feedback by humans labelling output as (non)instances (supervised learning) or by the system discovering commonalities in the data (unsupervised learning).

ML has introduced some new relevant computing terminology. A ML system consists of an application that operates on a general template²⁷ with modifiable parameters, called a ML model.²⁸ A learning algorithm ‘*adjusts the parameters of the template . . . by optimizing a performance criterion defined on the data*’,²⁹ or in other words, ‘trains’ a model on a training dataset. Once an ML system is trained, the model ‘represents’ the knowledge (eg the rules, numbers, data structures), required to perform a particular task, such as recognizing spam emails or faces, the logic connecting input to output. To actually produce functionality, the ML model requires additional software (and hardware) to process input data to output data, similar to traditional algorithms embedded in software. ML is an example of how the boundary between data (structures) and algorithms becomes increasingly less clear-cut. In traditional software, the programmer needed to have a clear idea ‘how’ to solve a particular problem and then formulate the necessary steps in an algorithm that can then be applied to input data. In many ML cases, the programmer may not exactly know how to perform a particular task (recognizing cats) but does know which data are relevant for the task as well as has a clear idea of an evaluation function that allows her (or the system) to determine whether the task is carried out successfully. The ML system will then establish the ‘how’, provided that the tools available are up to the type of task presented.³⁰

Many tasks can be performed by algorithms designed by humans using traditional programming languages and by ML techniques that learn to perform the task from data. Whether one opts for the former or the latter depends on many factors, one being how easy it is to describe the relation between input and output; describing what differentiates a cat from a dog on an image is

25 Eg Algorithm is defined as ‘a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer’ (‘Algorithm’ in *Oxford Dictionary of the English Language*).

26 Ethem Alpaydin, *Machine Learning - the New AI* (Cambridge, MA, MIT Press 2016) 16.

27 In an Artificial Neural Network, such a template would typically consist of a layered network of nodes with ‘weights’ between the nodes in the various layers. An input layer would be where for instance an image array of pixels would be presented to X input nodes (where X could be the total number of pixels in the image). These nodes are connected to a set of (hidden) layers of nodes to ultimately connect to an output layer of nodes. A cat-detecting image processor could at its output have two nodes, one flagging that the image presented is a cat, the other flagging that it is not.

By sequentially presenting the system with images (a typically large number, called the training set) and selecting the appropriate output (cat/non-cat), the system will adapt the weights between the various (many) nodes until it is able to properly classify all images in the training set. If the training set is representative for the task (cat detection), then the system will also be able to correctly classify images not in the training set.

28 Alpaydin (n 26) 24.

29 Ibid 24–25.

30 An image classifier would in principle be able to classify any kind of object (cats, cars, traffic lights, etc). Object classification is a generic task, detecting verbs in a sentence is another, anomaly detection in data traffic flows yet another.

difficult while determining whether the animal on the image is a cat or a dog is a relatively straightforward task for a human. This makes the task more appropriate for ML than for traditional algorithms. For the purposes of this article, to the degree the ML model is a machine-executable body of knowledge, we consider it code.

Personal data under the GDPR and other definitional options

Before proceeding to the main part of the argument, let us revisit the meaning of personal data in data protection law.³¹ Seeing as the GDPR applies ‘to the processing of personal data’ (Article 2(1) GDPR), the concept of personal data serves as a trigger concept to invoke the guarantees of data protection law. The GDPR defines ‘personal data’ as

any information relating to an identified or identifiable natural person (‘data subject’); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person.

The definition thus breaks into three elements of consequence for the argument: (i) information; (ii) relating to (a natural person); (iii) who is identified or identifiable, directly or indirectly. We will provide a brief recap of the meaning of each of these elements according to the case law of the Court of Justice of the European Union (CJEU) and the non-binding but authoritative interpretation by Article 29 Working Party.³²

Information

The lawmaker may have considered the meaning of ‘information’ to be self-evident, and so the GDPR (as well as its predecessor, the 1995 Data Protection Directive)

does not define what information is, or what kinds of information should be included. The former WP29 cites the intent of the legislator ‘to design a broad concept of personal data’³³ and explains that any information can fall under the concept ‘personal data’, regardless of its nature, content, or format.³⁴ Regarding the nature of information, it includes ‘objective’ statements about a person, ‘such as the presence of a certain substance in one’s blood’,³⁵ as well as ‘subjective’ information, like opinions and assessments.³⁶ The information does not have to be true or proven; false or unproven information can be personal data provided other requirements are met.³⁷ Regarding the content, personal data does not have to concern private or family life and could pertain to the life of the individual in his or her professional and other capacities.³⁸ Regarding its format, information in question can be ‘alphabetical, numerical, graphical, photographic or acoustic’, ‘kept on paper [or] stored in a computer memory’ as a binary code,³⁹ structured or unstructured.⁴⁰ The Court of Justice adopts the same broad approach to information in *Nowak*,⁴¹ clarifying that personal data is ‘not restricted to information that is sensitive or private, but potentially encompasses all kinds of information, not only objective but also subjective, in the form of opinions and assessments.’⁴² Whether or not a piece of information is sensitive or seems innocuous or mundane does not make any difference for determining the status of information as personal data.

Relating to

The references to ‘information relating to a natural person’ and ‘identifiable natural person’ invite interpretation as to what constitutes the relevant kind or degree of the link between information and a person, and have been interpreted broadly by both Article 29 Working Party and the Court.

The former WP29 and the Court both share a broad approach to the meaning of ‘relating to’.⁴³ Information

31 This is a very brief account of the meaning of personal data in the GDPR. A much more detailed analysis of the concept of personal data under the GDPR, as interpreted in the opinion of Article 29 Working Party, and in the case law of the CJEU can be found eg in Nadezhda Purtova, ‘The Law of Everything. Broad Concept of Personal Data and Future of European Data Protection Law’ (2018) 10 *Law Innov Technol* 40.

32 Article 29 Working Party, ‘Opinion 4/2007 on the Concept of Personal Data’ (WP 136, 20 June 2007). The current status of the opinion adopted under the 1995 Directive that is not in force by the Working Party that no longer exists is not certain under the GDPR. Yet, the opinion retained its significance under the GDPR, since the concept of personal data has not undergone significant changes. It remains the most comprehensive guide for the controllers on how to apply the concept of personal data, and will be considered as such here.

33 WP 136, 6.

34 Ibid 6ff.

35 Ibid 6.

36 Ibid 6.

37 Ibid 6.

38 Ibid 6–7.

39 Ibid.

40 Ibid 8.

41 *Peter Nowak v Data Protection Commissioner*, Case C-434/16, [2017] CJEU (ECLI:EU:C:2017:994).

42 *Nowak* [34] (‘the expression “any information” [...] reflects the aim of the EU legislature to assign a wide scope to [the concept of personal data], which is not restricted to information that is sensitive or private, but potentially encompasses all kinds of information, not only objective but also subjective, in the form of opinions and assessments’).

43 Note though that the position of the Court regarding the meaning of ‘relating to’ has changed. The *Nowak* judgment where the Court interpreted ‘relating to’ as relating to by reason of content, purpose, or effect reversed a more restrictive interpretation in Joint cases C-141/12 and C-372/12 *YS*

can be considered relating to people by reason of content, purpose, or result, meant as alternative rather than cumulative criteria. This means that information ‘relating to’ a natural person includes but is broader than the information ‘about’ that person.⁴⁴

Information relates to a person in content, ie when it is ‘about’ that person. Information relates to a person in purpose ‘when the data *are used or are likely to be used* ... *with the purpose* to evaluate, treat in a certain way or influence the status or behaviour of an individual’⁴⁵ (emphasis added). Finally, information regardless of its content or any purpose of processing may relate to a person in effect or result when ‘[its] use is *likely to have an impact* on a certain person’s rights and interests’.⁴⁶ The relevant impact does not have to be ‘major’,⁴⁷ ie a minor impact, eg treated differently from other persons, suffices.⁴⁸

This approach is different from eg Australian law which leans towards a narrower understanding of personal data as information ‘about’ a person,⁴⁹ and stands in stark contrast to how personal information, or ‘personally identifiable information’ (PII), is understood in the privacy law of the USA. There the requirement that a piece of information in question should relate to a person has not been articulated as a distinct part of any of the predominant approaches to the definition of personal information and has not been part of the US scholarly debate. The only relevant relationship of information to a person under the US law seems to be of identification or identifiability.⁵⁰ While one can argue that when information is identifiable to a person, a relationship to that person is implied, and the test of ‘relating to’ might be implied in the test of identifiability, it is likely that a relevant relationship in

this case is limited to the relationship by reason of content, ie when information is ‘about’ a person.

Identified or Identifiable [Natural Person]⁵¹

For a piece of information to be personal data, it has to relate to a natural person who is identified or identifiable. While the GDPR itself is silent on the meaning of ‘identified’, Article 29 Working Party explains that a natural person is identified ‘when, within a group of persons, he or she is “distinguished” from all other members of the group’.⁵² Some national courts have interpreted ‘identified’ to mean that a person is ‘individuated’.⁵³ To be ‘distinguished from a group’, an individual does not need to be known by name, passport number, or other type of civil identity. Identification can take many shapes. Five distinct types of identification have been identified so far.⁵⁴ (i) One can be identified by a name, ID number, or another civil identifier stored in a registry and allowing to ‘look up’ an individual in a real-world (*look-up (l-) identification*).⁵⁵ (ii) One can be identified as a known or eligible individual by presenting certain features, tokens, or patterns of behaviour known or recognizable as valid by the recipient (*recognition (r-) identification*),⁵⁶ eg when his or her face matches with an existing facial template, or when a cookie placed on one’s computer earlier identifies a website visitor as a repeated customer. (iii) One can be identified by being described as a member of a particular group or category (*classification (c-) identification*),⁵⁷ eg age, gender, dress style, etc. Classification only constitutes identification in the sense of the GDPR when describing an individual as falling into a certain category is sufficient to distinguish that individual from others in a given context,⁵⁸ eg ‘a man in a black suit standing by that traffic light’ is a group

and *M. and S. v Minister of Immigration, Integration and Asylum* [2016] ECLI:EU:C:2014:2081.

44 WP 136, 11 and Nowak [35].

45 WP 136, 10.

46 WP 136, 11; also Nowak [35].

47 WP 136, 11.

48 Ibid.

49 Joshua Yuvaraj, ‘How About Me? The Scope of Personal Information under the Australian Privacy Act 1988’ (2018) 34 CLSR 47.

50 Two influential pieces of academic commentary on the concept of personally identifiable information discuss only the element of identifiability and do not distinguish ‘relating to’ as a separate threshold that a piece of information has to pass to be considered personal [Paul Ohm, ‘Broken Promises of Privacy: Responding to the Surprising Failure of Anonymisation’ (2009–2010) 57 UCLA Law Rev 1701; Paul M Schwartz and Daniel J Solove, ‘The PII Problem: Privacy and a New Concept of Personally Identifiable Information’ (2011) NYU Law Rev 1814].

51 The meaning of both ‘identified’ and ‘identifiable natural person’ is treated in more detail in Nadezhda Purtova, ‘From Knowing by Name to

Targeting: Meaning of Identification under the GDPR’ (2022) 12 IDPL 163.

52 WP 136, 12. The ‘distinguished from the group’ understanding of identification seems to have been broadly adopted in Europe. The European Agency for Fundamental Rights and the Council of Europe explain that identification ‘requires elements which describe a person in such a way that he or she is distinguishable from all other persons and recognizable as an individual’. European Agency for Fundamental Rights and Council of Europe, *Handbook on European Data Protection Law* (Council of Europe 2018).

53 *R (on the application of Edward Bridges) v The Chief Constable of South Wales Police and Secretary of State for the Home Department* [2019] EWHC 2341 (Admin) 119.

54 Purtova (n 51); but see Marx distinguishing 7 types of identity knowledge (Gary T Marx, ‘What’s in a Name? Some Reflections on the Sociology of Anonymity’ (1999) 15 Inform Soc 100).

55 R Leenes, ‘Do They Know Me? Deconstructing Identifiability’ (2008) 4 Univ Ottawa Law & Technol J 135, 148.

56 Ibid 150.

57 Ibid 151.

58 Purtova (n 51).

reference (there are many men wearing black suits), which is sufficient to distinguish one person from a group of passers-by near that traffic light.⁵⁹ (iv) *Session (s-) identification* is in essence a form of recognition identification which aims to track an individual during a particular interaction, where the lifetime of the identifiers is restricted to the duration of that interaction.⁶⁰ (v) Finally, one can be identified by being selected as a particular individual from a group as an object of attention or treatment in a single moment of time (*targeting (t-) identification*). T-identification can be achieved either by means of a unique identifier that does not need to be persistent or can be based on a rich dataset.⁶¹

The meaning of identifiability is derived from, and hence secondary to the meaning of identification. While identification refers to the process or a fact of being identified, identifiability in the GDPR means a legally relevant chance of identification. The natural person is 'identifiable' when, while the person is not identified yet, it is possible to identify him/her.⁶² Recital 26 GDPR adopts reasonable likelihood of identification as the threshold of what constitutes identifiability:

To ascertain whether a natural person is identifiable, account should be taken of all the *means reasonably likely to be used*, such as singling out, by the controller or by another person to identify the natural person directly or indirectly. To ascertain whether means are reasonably likely to be used, account should be taken of all objective factors, such as the costs of and the amount of time required for identification, taking into consideration the available technology at the time of the processing and technological developments [emphasis added].

Those objective factors also include the intended explicit or implied purpose of processing (when 'the processing ... only makes sense if it allows identification of specific individuals and treatment of them in a certain way',⁶³ the availability of tools of identification should be presumed reasonably likely); the risk of organizational dysfunctions (eg breaches of confidentiality duties) and technical failures.⁶⁴

Identifiability is thus understood as the possibility of singling someone out, ie distinguishing someone from a

group, with the five types of identification providing a baseline for assessing the possibility of identification. The possibility to identify should be assessed not only from the perspective of a given controller but also as an objective possibility, considering the chances of identification by any other person, and taking into account 'all objective factors' of the case. Such objective factors should include cost and the state of art of technology at the time of processing. This objective and broad approach to identifiability was confirmed by the CJEU in *Breyer*.⁶⁵

Personal data as a dynamic and inclusive category

In effect, the definition of 'personal data' in the GDPR provides a 'method' for 'a tailored, context-specific analysis for deciding whether or not personal data is present'.⁶⁶ With regard to the same piece of information, as technology and other relevant circumstances that this method dictates have to be taken into account change, so does the outcome: the same piece of data or information can transition from being clearly non-personal to personal.

Applying this method leads to increase of the range of situations where personal data is processed.⁶⁷ Hence, the definition is considered broad, certainly when compared to the approaches to what 'personally identifiable information' (the US functional equivalent of personal data) means in the privacy law of the USA.⁶⁸

The GDPR definition of personal data is dynamic and broad, to the point that some argue it is overinclusive.⁶⁹ The track record of the CJEU decisions suggests that the Court is unlikely to interpret the definition of personal data narrowly as this would undermine the aim of the data protection law to ensure high level of protection of the fundamental rights and freedoms with respect to the processing of personal data.⁷⁰

In sum, there is no information that by definition cannot be or become 'personal data' in the sense of European data protection law, provided it meets the requirements of the definition. The following sections of

59 WP 136, 13.

60 Leenes (n 55) 152.

61 Purtova (n 51).

62 WP 136, 12.

63 Ibid 16.

64 Ibid 17.

65 *Patrick Breyer v Bundesrepublik Deutschland*, Case C-582/14, [2016] CJEU (ECLI:EU:C:2016:779).

66 Paul M Schwartz and Daniel J Solove, 'Reconciling Personal Information in the United States and European Union' (2014) 102 Calif Law Rev 877.

67 See Purtova (n 8) for analysis demonstrating that even weather measures can be personal data under certain circumstances like a smart city

where data from various sensors including a weather station is used for predictive policing.

68 Bryce Clayton Newell and others, 'Regulating the US Consumer Data Market: Comparing the Material Scope of US Consumer Data Privacy Laws and the GDPR' (forthcoming).

69 Sources cited in n 15.

70 Purtova (n 8) section 4.3, citing among others Bodil Lindqvist, Case C-101/01, [2003] ECR I-12992 (ECLI:EU:C:2003:596), *Google Spain SL, Google Inc v Agencia Española de Protección de Datos and Mario Costeja González*, Case C-131/12, [2014] CJEU (ECLI:EU:C:2014:317); and Nowak.

this article examine how the elements of the concept of ‘personal data’ apply to code and argue that some code under some circumstances constitutes personal data in the sense of the GDPR.

Why code can be personal data

Code and data are rarely seen as one. Indeed, in the eyes of a computer scientist, there is a clear and unquestionable separation between them.⁷¹ Code is code and data are data. Therefore arguing that computer code is or can be personal ‘data’ might be counterintuitive. Yet, in this section, we will do exactly that, because the legal definition of personal data allows for this argument. We argue that all software is ‘information’ and so, in principle, all software may be(come) personal data, provided the remaining two elements of the legal definition are present. We then proceed to build a layered argument that at least ‘some software’ certainly ‘relates to’ people by reason of content and that a growing range of code, especially in the context of algorithmic governance and algorithmic decision-making, relates to people by reason of purpose or result. Finally, we submit that code relating to people relates to people who are increasingly identified or identifiable, and hence is personal data.

All code is ‘Information’

Data protection scholarship, likely due to its historical connection to privacy law and scholarship, has been dominated by the silent assumption that ‘information’ in the definition of personal data is something that has ‘meaning to a human’ and either result in or has the potential, alone or in combination with additional information, to result in some—more or less sensitive—‘knowledge’ about or relevant for that person, eg his/her

medical or credit history, salary and price of one’s house, the content of one’s correspondence revealing personal secrets or telephone communications, or geo-location observed over time, revealing patterns of one’s movements.⁷² A computer program can hardly be said to be that kind of information. Code—on its own⁷³—does not often result in knowledge about or relevant to a person. Yet, the meaning- and knowledge-based understanding of information is just one way to approach it, while there are other ways where knowledge, meaning and, as a matter of fact, even human cognition, play no role, or their substance is highly contested. The legislator’s intent to design a broad concept of personal data⁷⁴ to provide the most complete legal protection⁷⁵ leaves room to argue that all meanings of the concept ‘information’, including those embracing code, should be accounted for in the definition of personal data, and the definition of personal data should not be limited to give preference to one particular understanding of information, eg knowledge based.

We explained earlier that personal data in the sense of the EU data protection law does not refer to any particular kind of information or any particular theoretical understanding of information. Let us briefly—and very roughly—sketch the broad and complex landscape of the various approaches to defining information,⁷⁶ and situate code in that landscape.

Information is a concept which is notoriously nebulous, with different meanings which ‘vary over time, across and within disciplines or depending on one’s philosophical inclination’.⁷⁷ It has Latin and Greek roots. Latin *informatio* and *informo* have been used to denote the act of giving a form to something, including in a biological context (eg that a fetus is being ‘informed’ by head and backbone).⁷⁸ Currently, the word is used to

71 But note a recent development in software engineering where code is increasingly shaped by data and hence, the two in a sense become one [Gürses and Hoboken (n 23)].

72 Other uses of the term ‘information’ and the related terms in the GDPR (eg ‘informed consent’) rely on the same understanding of information as carrying a meaning to a human, eg in the context of the information rights of the data subject [for an in-depth analysis of the meaning of information in the GDPR see Dara Hallinan and Raphaël Gellert, ‘The Concept of “Information”: An Invisible Problem in the GDPR’ (2020) 17 SCRIPT-ed 269].

73 This is as opposed to when used as an instrument for information analysis.

74 See Article 29 Working Party summarizing the relevant considerations during the drafting of the 1995 Data protection directive: ‘It needs to be noted that this definition reflects the intention of the European lawmaker for a wide notion of “personal data”, maintained throughout the legislative process. The Commission’s original proposal explained that “as in Convention 108, a broad definition is adopted in order to cover all information which may be linked to an individual”. The Commission’s modified proposal noted that “the amended proposal meets Parliament’s wish that the definition of “personal data” should be as general as possible, so

as to include all information concerning an identifiable individual”, a wish that also the Council took into account in the common position.’ (WP 136, 4).

75 We refer here to the principle of effective and complete protection of data subjects first articulated by the EU Court of Justice in *Google Spain SL, Google Inc v Agencia Española de Protección de Datos and Mario Costeja González*, Case C-131/12, [2014] CJEU (ECLI:EU:C:2014:317) [34].

76 For a comprehensive overview of the various meanings of the term ‘information’ across disciplines see Rafael Capurro and Birger Hjørland, ‘The Concept of Information’ (2005) 37 *Ann Rev Inform Sci Technol* 343; Mark Burgin, *Theory of Information. Fundamentality, Diversity and Unification* (Hackensack, NJ, World Scientific Publishing 2010).

77 Luciano Floridi, ‘Philosophical Conceptions of Information’ in Giovanni Sommaruga (ed), *Formal Theories of Information: From Shannon to Semantic Information Theory and General Concepts of Information* (Heidelberg, Springer-Verlag 2009) 13–53, 16; Pieter Adriaans, ‘Information’ in *The Stanford Encyclopedia of Philosophy* (2013) <<http://plato.stanford.edu/archives/fall2013/entries/information>> last accessed 8 September 2023.

78 Capurro and Hjørland (n 76) 351.

convey two related phenomena: the act of molding the mind and communicating knowledge.⁷⁹ The arrival of computing has turbocharged efforts to conceptualize information. By now many disciplines as diverse as philosophy, psychology, biology, and cybernetics have adopted their own understanding of information. It is beyond the ambition of this article to provide a comprehensive catalogue of all these conceptualizations and definitional attempts. For the purposes of our argument, it suffices to divide all these approaches into three broad categories.⁸⁰

The first category is semantic. A 'semantic' understanding of information is contingent on the presence of 'meaning'. Information has meaning to someone, and when the meaning is understood and internalized, this results in knowledge. Meaning makes information subjective. Information carries different meanings to different recipients who have different background knowledge and experience.⁸¹ A typical example is a text written in a certain language, say, French. This text will have a meaning to a French speaker but appear nonsensical to the one who does not speak French. Privacy and data protection scholarship seem to have relied on the semantic approach so far. Within the semantic approach, several analyses have adopted a General Definition of Information (GDI) as an (ideal type)⁸² operational standard: '*information is data + meaning*'.⁸³ Data in this definition play a representational role, ie data are 'a description of something that allows it to be recorded, analysed, and reorganized'.⁸⁴

The second category is the syntactic or statistical approaches to information which are not dependent on meaning. Even more so, in these approaches meaning is irrelevant. Consider Shannon's mathematical theory of information, originally called communication theory. It defines information as 'the statistical probability of a sign or signal being selected from a given set of signs'.⁸⁵ Meaning appears to play no part:

[I]nformation must not be confused with meaning. ... [T]wo messages, one of which is heavily loaded with meaning and the other of which is pure nonsense, can be exactly equivalent, from the present viewpoint, as regards information. (Shannon and Weaver, 1972, p. 8)

The trend fitting under the second category is 'naturalization of information' where the existence of information is not only regardless of meaning but also of human cognition and for some—of human presence altogether. For instance, some physicists see 'structural and kinetic information [as] an intrinsic component of the universe [...] independent of whether any form of intelligence can perceive it or not'.⁸⁶ Biological systems and life itself, notably in molecular biology, can be understood as networks of information processes.⁸⁷ According to Stonier, who in addition to being a philosopher and a theorist of information, was also a biologist, information just exists, independent of human cognition.⁸⁸

Under the third category, we put the approaches that embrace the variations in understanding information and see the variations as not mutually exclusive. According to Floridi, the concept 'information' refers to three mutually compatible phenomena: information 'about' reality (semantic information); information 'as' reality (eg 'as patterns of physical signals' such as DNA or fingerprints); and information 'for' reality ('instructions like genetic information, algorithms, orders or recipes').⁸⁹ Capurro and Hjørland argue that the semantic and mathematical approaches to information can be reconciled, since Shannon's engineering perspective on information and meaning- and cognition-based theories share a common feature of selection.⁹⁰

The ultimate expression of the acceptance of conceptual variation is the proposal of Capurro and Hjørland to see the various concepts of information as being in a relationship of Wittgensteinian 'family resemblance',⁹¹ where not all 'family members' look alike and share a

79 Ibid 351.

80 This classification is a broad-brush sketch rather than an accurate portrait. Undoubtedly, scholars studying information primarily will be able to produce alternative, more nuanced classifications.

81 Burgin (n 76) 94.

82 The GDI should serve as an ideal type definition since the boundaries between its elements, while sharp in theory, may be fuzzy in real life. For instance, there is also a widely adopted understanding that data are never meaning-free.

83 Luciano Floridi, 'Is Information Meaningful Data?' (2005) 70 *Philos Phenomenol Res* 351.

84 Viktor Mayer-Schönberger and Kenneth Cukier, *Big Data: A Revolution that will Transform How We Live, Work, and Think* (Boston, MA, Houghton Mifflin Harcourt 2009) ch 5.

85 Fritz Machlup, 'Semantic Quirks in Studies of Information' in Fritz Machlup and Una Mansfield (eds), *The Study of Information: Interdisciplinary Messages* (Wiley 1983) 641–71, 658.

86 Burgin (n 76) 33, and in-text references.

87 Capurro and Hjørland (n 76) 363 and footnotes. For an interesting account of the similarities between computer code and DNA, see: Bert Hubert, 'DNA Seen Through the Eyes of a Coder (or, If You are a Hammer, Everything Looks Like a Nail)' (*Berthub*, 9 January 2021) <<https://berthub.eu/articles/posts/amazing-dna/>> last accessed 30 July 2023.

88 Tom Stonier, *Information and Meaning: An Evolutionary Perspective* (London, Springer 1997) (he also argues that it is necessary to distinguish between information and meaning. Information, eg letters of an alphabet or a strand of DNA, yields a message when processed, and a message yields meaning if processed by a recipient.)

89 Luciano Floridi, *The Philosophy of Information* (Oxford, OUP 2011) 30.

90 'Even in the extreme case in which any interpretation is supposedly excluded [...] we can still recognize a process of selection. [W]e state a resemblance between interpreting meaning and selecting signals. The concept of information makes this resemblance possible' [Capurro and Hjørland (n 76) 368].

91 Capurro and Hjørland (n 76) 368.

common semantic core, but all are held together through a complex network of similarities.

Computer code is information in several senses of the word. It is in any case an instance of syntactic information: a set of symbols jointed in patterns following syntactic rules. In fact, if the code does not comply with the syntax of the language in which it is written it will not compile nor run on a computer. It can also be seen as semantic information which has different meanings to different recipients, eg a software engineer interprets source code written by another software engineer and understands what it is supposed to be doing, hence draws some meaning from it. However, a lay person may see no meaning in it. If we accept that syntax is never free of semantics, computers draw meaning from object code, too, in a sense that code makes machines operate in a certain way. In Floridi's classification, any computer code is a set of commands and hence is information shaping, or 'for' reality. It is 'informing' the computer on what it is expected to do. This understanding of code as information is also compatible with the etymology of the word (recall its Latin and Greek roots *informatio* and *informo* denoting the act of giving a form to something). Hence, code is also information in the sense of the data protection law, since the latter does not specify a particular theoretical perspective on the information it relies on. Recall the WP29 explanation of the definition of personal data that 'any' information can fall under the concept 'personal data' regardless of its nature, content, format, medium, or form, which could be 'alphabetical, numerical, graphical, photographic or acoustic', 'kept on paper [or] stored in a computer memory' as a binary code,⁹² structured or unstructured,⁹³ provided the other criteria of the definition are met.

So, the range of the possible meanings of 'information' in the definition of personal data in the GDPR is broad enough to include computer code. Even more so, information should not be interpreted any narrower, to the effect that it excludes code, since it would be going against the current context of data protection law. As Capurro and Hjørland aptly note with regard to scientific discourse, theoretical concepts are not true or false, better or worse at representing reality; they are constructs that are more or less useful to do a particular job in a certain context.⁹⁴ Similarly, in the data protection law particular notions of information are not correct or

incorrect, but more or less fruitful in the context of data protection. Data protection has long ago embraced some kinds of syntactic information (or, in Floridi's terms, information for reality), eg genetic information, as established categories of personal data. It is debatable whether or not some kinds of personal data, like IP addresses, cookies under some circumstances, or geolocation, do in fact always constitute semantic information in the purist sense, ie where meaning requires human cognition. Therefore, excluding non-semantic meanings of information from the scope of the GDPR will take a large chunk of previously and more conventionally protected data out. Further, as discussed earlier (the presence of) meaning is inherently subjective, ie contingent on a recipient of information. Hence, adopting semantic theories of information as the only standard for the legal definition of information will make the concept 'personal data' and hence the scope of the GDPR highly unstable.

Software that 'Relates to' people

An argument that software is information 'relating to' a natural person and hence, may be personal data, can be constructed in several ways. We envisage at least three and will call them first-type, second-type, and third-type arguments. These are complementary to each other rather than mutually exclusive. While all three argument types have significant implications for data protection law, the arguments of the second and third types are, in our opinion, more profound and pertain to the core of the rationale of data protection, namely, what kind of relationships between information and people data protection law aims to control.

First-type argument: code may relate to (a) natural person(s) in content due to its semantic component

Some code may be information relating to people due to its semantic component. In other words, some software may be personal data because it is or allows inferences of semantic information about a person that is more conventionally seen as personal data. For instance, the analysis of source code, similar to the analysis of a written text, may reveal information about its author, eg his or her programming style,⁹⁵ or source code's authorship.

Some instances of code are susceptible to specific kind of inferences revealing information about people. Veale, Binns, and Edwards have recently argued that

92 WP 136.

93 Ibid 8.

94 Capurro and Hjørland (n 76) 346.

95 Recall the if statement in n 13. Ronald, who writes a fair amount of Arduino code will not use the shorthand version of a conditional because

he always has to look up its syntax, so he simply resorts to the easier to memorize long version. From this, a hand of a particular coder can be recognized.

model inversion and membership inference attacks may render ML models personal data.⁹⁶ If we consider ML models to be code, as we do, this would be an example of code relating to a person in the sense of Article 4 GDPR. Specifically, inversion attacks on a model allow the attacker to recover at least some of the training data which by itself likely was personal data relating to people in content, purpose, or effect.⁹⁷ Membership inference attacks allow to estimate whether or not the data of a particular person was in the training set of a given ML model.⁹⁸ In this way, as Veale, Binns, and Edwards convincingly argue, an ML model is encoded personal data analogous to pseudonymized data.⁹⁹

Finally, another argument can be made, namely that all code reveals information about people. According to Agre, the design of any information system begins with modelling human behaviour, be it individual or organizational, breaking it into smaller tasks ('grammar of behaviour').¹⁰⁰ Hence, code represents a model of human behaviour and therefore is information about people, ie about how people behave. As Gürses and van Hoboken powerfully argue, this characteristic of code is reinforced in the context of the growing deployment of the so-called agile software development methods where software is modelled around behaviour of its user, sometimes in real time. Based on the cloud delivery of software as a service, code is constantly adjusted to how it is used,¹⁰¹ where 'users must be treated as co-developers'.¹⁰²

Granted, the models of behaviour 'built' into the design of code are rarely based on the behaviour of people as individuals, but rather on the behaviour of a group of people, representing an 'average user', ie in abstract. This highlights an issue that was brought forward by the various audiences to whom the argument advanced in this article was presented as it was taking shape. Does

information have to relate to one person in order to be personal data? The EU Court of Justice answers this with a no. In *Nowak*, it ruled that a piece of information (examiner's marks) can still be regarded as personal data even when it relates to a number of people (the examiner and the candidate), as long as those people are identified or identifiable.¹⁰³

Second-type argument: some software is designed and deployed with the purpose to evaluate, treat in a certain way or influence the status or behaviour of an individual, and hence relates to (a) natural person(s) in purpose

As a second-type argument, we submit that software may also be information relating to people regardless of any semantic component, but because some of it is designed and deployed 'explicitly with the purpose' to evaluate, treat in a certain way, or influence the status or behaviour of an individual. Examples of such software are systems used in 'smart' hiring, risk profiling, preventive policing, and others. In fact, this is the entire point of what is now called 'algorithmic decision-making' or 'algorithmic regulation',¹⁰⁴ where algorithms are deployed to make decisions about or affect people in a wide range of contexts for a wide range of purposes. In this section, we will discuss two instances of software designed and deployed with the purpose to evaluate, treat in a certain way, or influence the status or behaviour of an individual: (i) automated decision-making and (ii) digital decision guidance (or nudging). We submit that the software involved in both instances 'relates to' people in purpose in the sense of the GDPR definition of personal data.

'Automated decision-making' is involved in a variety of contexts, from ticket dispensing to consumer credit,

96 Veale, Binns and Edwards (n 10). We are aware that outside debates on data protection law and the legal meaning of personal data models and data are often seen as two distinct things. See eg Sabina Leonelli, 'What Distinguishes Data from Models?' (2019) 9 Eur J Philos Sci 21. See also Leiser and Dechesne (n 10) 187.

97 Congzheng Song, Thomas Ristenpart and Vitaly Shmatikov, 'Machine Learning Models that Remember Too Much' in Bhavani M Thuraisingham, David Evans, Tal Malkin and Dongyan Xu (eds), *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)* (ACM 2017) 587–601, cited in Veale, Binns and Edwards (n 10).

98 Reza Shokri and others, 'Membership Inference Attacks Against Machine Learning Models' (2017 IEEE Symposium on Security and Privacy (SP), San Jose, 2017).

99 Veale, Binns and Edwards (n 10) 6. The notion of pseudonymized, or pseudonymous, data are based on the Art 4(5) GDPR definition of pseudonymization as 'the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an

identified or identifiable natural person'. A ML model is analogous to pseudonymous data but will not always 'be' pseudonymous in the sense of the GDPR. This is because pseudonymization is a process that does not occur by chance or as a by-product, but requires conscious efforts of the controller to achieve data minimization (Art 25 GDPR). For this reason, the status of personal data as pseudonymous grants the controller certain 'privileges', eg plays a role in assessment of compatibility of processing with the original purposes (Arts 6(4)(e) and 89 GDPR), and may invoke exceptions from some data protection obligations under Art 11 GDPR.

100 Philip E Agre, 'Surveillance and Capture: Two Models of Privacy' (1994) 10 Inform Soc 101.

101 Gürses and Hoboken (n 23).

102 Tim O'Reilly, 'What Is Web 2.0' (September 2005) <<https://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html?page=4>> last accessed 1 June 2023, cited in Gürses and van Hoboken (n 100).

103 'The same information may relate to a number of individuals and may constitute for each of them [...] personal data' (*Nowak* [45]).

104 Karen Yeung, 'Algorithmic Regulation: A Critical Interrogation' (2018) 12 Regul & Gov 505.

employing techniques of ranging degrees of complexity and sophistication.¹⁰⁵ Some argue that these decision-making processes ‘automatically issue some kind of “decision” without any need for human intervention beyond user input of relevant data . . . and thus constitute a form of action forcing (or coercive) design’.¹⁰⁶ The algorithms involved in such decision-making are either specifically intended to assess or influence individuals’ behaviour or, even when there is no such explicit purpose, will likely impact individual rights and interests. This is the case regardless of whether or not a human is involved in the decision-making, as the automated decision-making techniques are still meant to have an impact on the outcome of the entire decision-making process even when their outcomes are critically examined by ‘a human in the loop’. For instance, the (banned and now notorious) System Risk Indication (SyRI) is a system that was used by the Dutch government to carry out risk analysis for the purpose of preventing and combating tax and social security fraud. Drawing authority from the Labour and Income Implementation Structure Act, various public bodies, eg government agencies for social services and taxes, government supervisory bodies, police, prosecutor’s office and municipalities, pooled all the personal data they possess ranging from marriage and property registration to the record of fines and health insurance, into the system.¹⁰⁷ SyRI would analyse these data, using risk profiles, and flag people who fit the suspect data patterns. The ‘hits’ then were investigated. Depending on the results of investigations, prosecution for fraud and/or withdrawal of social security benefits could follow. In this instance, we argue that not only the data ‘fed into’ SyRI are information relating to people by

reason of the purpose of processing but also SyRI itself.¹⁰⁸

However, software that explicitly influences behaviour, rights, and interests of an individual is not limited to the software which ‘makes decisions’. Such software is also involved in what Yeung calls the ‘digital decision-guidance processes’¹⁰⁹ where not the machine, but the targeted individual makes the relevant choices, ‘prompted’ or nudged by a choice architecture determined by an algorithm.¹¹⁰ Several scholars from the fields of governance, behavioural economics, and cyber-law have explored the idea of regulation by design or code. While the scholars of governance and cyber law focus on ‘code’ as a mode of regulation of human behaviour,¹¹¹ behavioural economists study behaviour and choices. Among them, Thaler and Sunstein developed their theory of nudge around the idea that choices are not always made rationally but are influenced by numerous visible and invisible elements of the environment. The one controlling the choice environment is a ‘choice architect’, ie controls the choice and the resulting behaviour.¹¹² The often-cited example of choice architecture is a student canteen where healthier food options are placed in front of unhealthy options, thus nudging students towards healthier food choices.

Code can be used to build such a choice architecture. For instance, an option to decline cookies on a website can be made very visible and easy to find and click, or hidden in the depths of the website, with the purpose to dissuade people from uncovering and opting for it.¹¹³ Both are instances of website (and code) design meant to either facilitate or obstruct one’s choices and behaviour. In the algorithmic context, Yeung speaks of ‘Big

105 Karen Yeung, ‘Hypernudge’: Big Data as a Mode of Regulation by Design’ (2017) 20 *Inform Commun & Soc* 118, 121.

106 Ibid.

107 The conditions of deployment of SyRI are established in the implementing provision to the Dutch Labour and Income Implementation Structure Act (‘Besluit van 1 september 2014 tot wijziging van het Besluit SUWI in verband met regels voor fraudeaanpak door gegevensuitwisselingen en het effectief gebruik van binnen de overheid bekend zijnde gegevens met inzet van SyRI’) *Staatsblad van het Koninkrijk der Nederlanden*, 11 September 2014, <<https://zoek.officielebekendmakingen.nl/stb-2014-320.html>> last accessed 28 September 2023. Under Section 3 of the implementing provisions, the following categories of data are used: data on employment, fines and penalties, taxation and property, data from the Chamber of Commerce, living and lodging data, identifying information, including name, address, place of residence, postal address, date of birth, gender and administrative characteristics, information on possible integration and civic integration obligations, information on obedience with the law, education, pension, if a person after a period of illness keeps or has adhered to the re-integration obligations, information on debts, benefits, charges and subsidies, licences and permits, and health insurance.

108 Both the input data and SyRI are personal data when the individuals they relate to are identified or identifiable. For discussion of identification and identifiability see (n 51) and the related text.

109 Yeung (n 104) 121.

110 Ibid 121.

111 Bronwen Morgan and Karen Yeung, *An Introduction to Law and Regulation* (Cambridge, CUP 2007); Lawrence Lessig, *Code and Other Laws of Cyberspace* (New York, Basic Books 1999); Jonathan Zittrain, ‘Tethered Appliances, Software as Service, and Perfect Enforcement’ in Roger Brownsword and Karen Yeung (eds), *Regulating Technologies* (Oxford, Hart Publishing 2008) 125–56; Ronald Leenes, ‘Framing Techno-Regulation: An Exploration of State and Non-State Regulation by Technology’ (2011) 5 *Legisprudence* 143; Bibi van den Berg and Ronald Leenes, ‘Abort, Retry, Fail: Scoping Techno-Regulation and Other Techno-Effects’ in Mireille Hildebrandt and Jaenne Gakeer (eds), *Human Law and Computer Law: Comparative Perspectives* (Berlin/Heidelberg, Springer 2012) 25.

112 Richard L Thaler and Cass R Sunstein, *Nudge* (London, Penguin Books 2008).

113 Eg Ronald Leenes and Eleni Kosta, ‘Taming the Cookie Monster with Dutch Law – A Tale of Regulatory Failure’ (2015) 31 *CSLR* 317, describing various strategies websites employ to obtain consent for placing cookies.

Data-driven' or 'hypernudging', a 'dynamic' form of regulation by design with both the standard of behaviour and its execution 'continuously updated and refined'.¹¹⁴ Hypernudge occurs

within a networked environment that enables real-time data feeds [...] used to personalise algorithmic outputs [...]. Big Data-driven digital-guidance technologies thus operate as self-contained cybernetic systems, with the entire tripartite regulatory cycle continuously implemented via a recursive feedback loop which allows dynamic adjustment of both the standard-setting and behaviour modification phases of the regulatory cycle, enabling an individual's choice architecture to be continuously reconfigured in real time [...].¹¹⁵

Yeung names predictive policing as an example of hypernudge where Big Data analytics is used 'to assist enforcement officials determine their inspection priorities' in terms of high-risk individuals or other policing targets.¹¹⁶ In this case, police officers' behaviour in the form of choices of policing strategies is subject to regulation by code. We submit that not only the input and output data but also the hypernudging software itself that operates the hypernudge regulatory cycle are used with the purpose to assess individuals or are likely to impact their rights and interests and hence are information relating to people in purpose or effect.

For Yeung, hypernudge operates by configuring the user's 'informational choice context',¹¹⁷ eg displaying to a police officer 'information' that would influence his or her decision on policing strategy, or, in the case of Google maps, suggesting to a driver an optimal route from point A to point B. However, behaviour design strategies have also been implemented in the design of computer systems beyond providing information feedback. Think of the cookie opt-out example from earlier in this section. Fogg discusses how physical and complex psychological cues can be designed into a computing product ('persuasive design'), eg to 'lead people to infer, often subconsciously, that the product has emotions, preferences, motivations, and personality', and hence to trigger desirable behaviour.¹¹⁸ The code implementing such persuasive design thus also constitutes information relating to people by reason of purpose.

The scenarios are easily imaginable where automated decision-making and the hypernudge merge. As argued elsewhere,¹¹⁹ the narratives of the 'smart' environment that surreptitiously adjusts to the needs and desires of its users¹²⁰ are steadily on the way out of the realm of science fiction. For example, in a 'smart' city an increasing number of aspects of the environment and people living in it are datified, and the inhabitants are subjected to a certain code-mediated treatment in real time based on the processing of the data, both in decision-making (eg whether or not an automated gate opens to a passer-by or a neighbourhood is policed more intensively based on predictive policing) and constructing choice architecture and nudging (eg in the form of adjustments in the warmth and intensity of street lighting to prevent undesirable behaviour and targeted policing system's recommendations to police officers as to the policing strategies and locations). Some interactions between automated systems and people amount both to the automated decision-making and choice architecture at the same time, eg a hypothetical automated system adjusting the speed at which escalators are running based on a profiling passer's-by in order to promote physical activity. Zuboff described these processes in a broader context of information technology-powered automated systems, information is not simply extracted from people but also is imposed on them in the form of programmed instructions, ie code.¹²¹ We submit that not only the data that feeds that 'smart city' or informs the persuasive design of a computing product but also the software itself that runs its regulatory cycle, is information processed with the purpose to assess individuals or impact their rights and interests and hence are information relating to people in purpose or effect.

Third-type argument: code relates to (a) natural person(s) by reason of effect due to unintended modifications of behaviour

The third-type argument goes to the core of the relationship that exists between human behaviour and code. We submit that, besides particular instances of code deployed or designed with an explicit 'purpose' to assess a person or his behaviour or impact his or her rights and

114 Yeung (n 104) 122.

115 Ibid 122. 'Regulatory cycle' here refers to information gathering and monitoring, setting goals or targets (standard-setting), and 'behaviour modification' [Christopher Hood, Henry Rothstein and Robert Baldwin, *The Government of Risk* (Oxford, OUP 2001), cited in Yeung (n 104)].

116 Ibid 121.

117 Ibid 122 ('Hypernudging relies on highlighting algorithmically determined correlations between data items within data sets that would not otherwise be observable through human cognition alone (or even with standard computing support [...]), thereby conferring "salience" on the highlighted data patterns, operating through the technique of "priming",

dynamically configuring the user's informational choice context in ways intentionally designed to influence her decisions.')

118 BJ Fogg, 'Persuasive Technology: Using Computers to Change What We Think and Do' [2002] *Ubiquity* 89.

119 Purtova (n 8) 55.

120 Mireille Hildebrandt, 'Law as Information in the Era of Data-Driven Agency' (2016) 79 *MLR* 4.

121 Shoshana Zuboff, 'Big Other: Surveillance Capitalism and the Prospects of an Information Civilization' (2015) 30 *J Inform Technol* 30, 76.

interests, many instances of code¹²² determine human behaviour intentionally or unintentionally and hence are information relating to people in terms of effect. This is again inspired by Philip Agre's work on the capture model of privacy and in part, leans on the work of Seda Gürses and Joris van Hoboken where they explore the shift from what they call 'shrink-wrap software' to 'agile' methods of software engineering and its implications for privacy law.¹²³ This argument is related to the second type argument as it too builds on the affordances of technology, including code, to model human behaviour. The difference between the two arguments is in the degree of intentionality of this modelling. In contrast to the second-type argument considering intentional impacts such as automated decision-making and choice architecture, the modelling considered here is not knowingly intentional but comes with the nature of designing code.¹²⁴

In a nutshell, according to Agre, the design of any information system begins with modelling human behaviour, be it individual or organizational, breaking it into smaller tasks (what he calls 'grammar of behaviour') and then assigning these tasks to humans and/or machines. 'Capture' occurs when 'grammar of behaviour' is imposed on people via technology, and they adjust their behaviour according to this model. Their captured activity then produces records which can be used 'to "listen to" the ongoing activities for purposes of error detection, advice-giving, performance measurement, quality control, and so forth'.¹²⁵ Thus, code by virtue of a model of behaviour built into it—even when it is not explicitly intended by its makers—impacts behaviour and under some circumstances is likely to have an impact on people's rights and interests, which amounts to the relationship by reason of effect as the WP29 and CJEU understand it. While it can be argued that any modelling of behaviour imposed on the code's users impacts those users' autonomy, it is not clear from the current state of interpretation of 'relating to by reason of effect' what degree of impact on behaviour would satisfy the standard of impacting rights and interests. Clearly, the impact of a word processing software which does or does not allow, say, to format text in a certain way will be less than some design feature leading to systematic exclusion

from certain benefits or limitation of rights, unintended but baked into the design of the code. Consider a hypothetical example of software originally developed for tracking inmates, where the tolerance for the unplanned movement of the inmates is low, being repurposed for monitoring the whereabouts of children, where the tolerance for the unplanned behaviour is higher. In this example, a piece of software designed for one purpose (with all sorts of assumptions regarding this purpose built in), can be used by a person for a different purpose while being unaware of the original purpose and its underpinning assumptions and hence the grammar of behaviour of one context is transposed onto another.

With the arrival of agile methods of software development that Gürses and Hoboken describe, this dynamics has intensified. When the 'shrink-wrap software' was the norm, the software was relatively static and the 'grammar of behavior' once imposed did not change. The agile turn in software development in the early 2000s meant that pervasive connectivity enabled a shift to cloud-based software as a service which became a dominant way to provide computing functionality.¹²⁶ One of the affordances of the agile turn has been testing-led dynamic software design, where the end-users are being tracked in their use of software to test capture's efficiency. According to Gürses and Hoboken, 'every feature [of design] is a real-time probe into the continuous reorganization of users' activities. End-user activity becomes a key ingredient in managing the system'.¹²⁷ As Tim O'Reilly explained, 'if users don't adopt [features], we take them down. If they like them, we roll them out to the entire site'.¹²⁸ This goes to demonstrate how software, and the modern software design increasingly more so, is based on the 'grammar of behaviour' and intends to capture that 'grammar of behaviour' at the same time. Software influencing human behaviour in the sense of Agre's capture may be regarded as information that impacts people, and hence information relating to people by reason of effect.

'Identified or Identifiable [Living Natural Person]'

A common misconception about what constitutes personal data is that the piece of information in question

122 This is perhaps, with an exception of the famous 'Hello, world!' 'Hello World' is one of the basic computer programs and often the first for a beginning coder to learn. It makes a version of 'Hello, World!' appear on a device's display. It can be created in most programming languages, 'making it some of the most basic syntax involved in the coding process'. It is often used to illustrate how the process of coding works or to ensure that a language or system is operating correctly. 'The History of Hello World' (2015) 17 July 2015 Medium, <<https://medium.com/the-software-guild-blog/the-history-of-hello-world-175440f77776>> last accessed 28 August 2023.

123 Gürses and van Hoboken (n 23).

124 See also van den Berg and Leenes (n 111).

125 Agre (n 100) 110.

126 Gürses and van Hoboken (n 23).

127 Ibid 22 (preprint page numbering).

128 Tim O'Reilly, 'What Is Web 2.0' (September 2005) <<https://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html?page=4>> last accessed 1 June 2023, cited in Gürses and van Hoboken (n 23).

needs to identify a specific person. This is hard to conceive of in the context of code. However, it is not the information that needs to be identifiable to a person, it is a person—to whom the information relates—who needs to be identified or identifiable ‘given all means reasonably likely to be used’ (Recital 26 GDPR), meaning, identified or identifiable not necessarily by the piece of information under consideration. Therefore, depending on the context, code can be information relating to an identified or identifiable person either on its own or if there is sufficient other information that in combination with the code would ‘link’ to that person. Given the broad meaning of identification (the five identification types) and identifiability explored earlier in the article, there can be a number of ways how various types of individuals can be considered identified or identifiable. Below are some illustrative examples.

Creators of code can be identified or identifiable by name, ie in the sense of the look-up identification, by their employers or clients, for instance where there is a service or employment contract between them which includes the coder’s name. In the case of a contractual relationship, both employers and clients likely know a coder by name, or are able to establish the name. Programmers can also be identified or identifiable in the sense of recognition identification by their distinct style of coding, especially in the case of a source code. Recognition by the unique coding style may lead to the look-up identification. In terms of how reasonably likely such identification is, it will depend on the particular circumstances of the case. For instance, identification would be reasonably likely when the police is investigating a cybercrime and tries to establish the identity of the author of a piece of malware. Technological context also makes such attempts of identification more likely. For instance, parts of the code can be compared to the existing repositories, such as on GitHub, which are often tied to individuals. It is also a widespread practice to reuse own code, which increases the likelihood of linking code to its author.

Code users can be identified or identifiable by name (look-up) or other persistent identifiers such as a software licence number or device ID (recognition identification). Currently, the agile software design methods target many code users to model and impose the ‘grammar of behavior’. However, it is also possible to imagine a scenario of highly personalized code adjusting to the behaviour of its one user, eg a ML model operating locally, in which case that user would be identified in the sense of targeting identification.

Finally, the ‘targets’ of code, ie the individuals to whom the code is applied, can be identified or identifiable by name or another civil identity identifier (look-up identification) when the code is used in the context of automated decision-making, eg by public authorities for welfare fraud investigation or risk profiling, or by private bodies in the context of hiring. The targets of code are identified in the sense of targeting identification if the code delivers targeted personalized content like news, recommendations or ads, or other personally targeted treatment, eg a decision on price tailored to a unique individual and his or her surrounding context. Finally, a target of code is considered identifiable if, while he or she is not identified, but the purpose of code is to identify him or her¹²⁹ in any one out of five types of identification, regardless of the actual ability to do so. An example could be a facial recognition system, the purpose of which is to either identify a match to a known facial template, eg for authentication purposes, which would constitute recognition identification, or to establish the civil identity of the targets based on the facial template, which would constitute look-up identification.

To sum up, we have demonstrated that all code is information about people, and sometimes is explicitly meant or unintentionally has a potential to impact people and hence can relate to people. Finally, those people are often identified or identifiable, if not by the code itself, then by a combination of other information. At the time when these circumstances coincide, computer code displays all the characteristics of personal data and should be considered as such.

What this means for data protection and regulation of algorithmic systems

If computer code under some circumstances can be personal data in the sense of the GDPR, what does it mean both for data protection and regulation of algorithmic systems? We will discuss these consequences in terms of positive and negative implications.

The positives: new legal protections

The GDPR

The immediate practical result of applying the GDPR principles and rights to code is that it generates new legal protections for the people impacted by the code. Generally, this is a positive development when many digital harms still do not have a legal remedy other than

129 Recall WP136, 16 where identifiability is determined, a.o. by the intended explicit or implied purpose of processing.

the data protection law. While we do not aim to provide a complete inventory of the effects of the GDPR's provisions when applied to code, reviewing how the general data protection principles under Article 5 GDPR and several specific GDPR provisions could apply to code should illustrate the point. Note that the analysis below applies not to code generally, but only to the code that displays characteristics of personal data.

The principle of 'fair, lawful and transparent data processing' would translate into the principle of fair, lawful and transparent deployment of code as far as it constitutes personal data. This could be interpreted as requiring fairness both in terms of the process (no deception or deployment without the data subject's knowledge¹³⁰) as well as the outcome of the use of computer software and perhaps even software design. The latter could help address design aspects of code that have been considered unfair, such as bias in ML. As Bygrave aptly points out, the principle of fair data processing has a broad scope and an 'open texture' that 'gives EU data protection law a powerfully flexible norm with which to hold automated decisional systems to account'.¹³¹ Transparency would mean transparency towards the data subject both about the fact of use of code in relation to them, as well as the purposes of such use, and other aspects concerning the substance and form of informing the data subjects along the lines of the Article 29 Working Party guidelines on the principle of transparency.¹³² The principle of lawfulness would require a choice of an appropriate legal ground for the design and deployment of code.

The principle of 'purpose limitation' could potentially require that the use of code is limited to the originally intended legitimate purpose stated upfront in a clear, explicit, and specific manner and ban the uses for illegitimate or incompatible purposes. This principle would prohibit covert code deployment, ie for purposes other than declared to the data subject, such as surveillance or remote control of the data subject's device, eg a smartphone camera and microphone. It could also provide solution to the so-called 'function creep' where a technology acquires new unforeseen uses often

perceived as undesirable.¹³³ An example of such function creep in the case of code are smartphone apps originally marketed for the legitimate purpose of tracking the whereabouts of children by their parents or other guardians for their safety but repurposed for spying on or stalking of (former) love partners and anyone else whose phone a perpetrator manages to get hold of and install the app.

The principle of 'data minimization' reinforced by Article 25(2) principle of data protection by default would translate to code minimization, which means limiting both the design and use of code to what is necessary and proportionate for the declared legitimate purposes, forbidding excessive and disproportionate code uses, and functionalities. Code minimization and purpose limitation would also require a more curated approach to the code design where new code cannot be created by mindless copy-pasting of pieces of existing source code from github or similar platforms, which is a widespread practice especially in more 'guerrilla', not professional coding communities.¹³⁴ In many cases, such curated approach to coding would likely improve software quality.

The principle of 'accuracy' could translate *inter alia* into the requirement that the code is built on a sound scientific method. 'Storage limitation'—when applied to code—would mean not using code longer than necessary for a specific and legitimate purpose. It would also reinforce the principle of storage limitation applied to data and require that the use of software enabling identification (such as biometric recognition, detection and personalization, or 'nano-targeting' algorithms¹³⁵) is proportionate to the legitimate purpose.

The principle of 'integrity and confidentiality' would require the protection of code from unauthorized access and tempering, adding another layer of cybersecurity obligations beyond data security. Finally, the principle of 'accountability' would translate into the accountability for the deployment and design of algorithms in the form of an obligation to be able to demonstrate compliance and justify design and use choices.

130 Cécile de Terwangne, 'Article 5 Principles Relating to Processing of Personal Data' in Christopher Kuner and others (eds), *The EU General Data Protection Regulation (GDPR): A Commentary* (New York, 2020; online edn, Oxford Academic), <<https://doi.org/10.1093/oso/9780198826491.003.0034>> last accessed 19 April 2023.

131 Lee A Bygrave, 'Minding the Machine v2.0: The EU General Data Protection Regulation and Automated Decision Making' in Karen Yeung and Martin Lodge (eds), *Algorithmic Regulation* (Oxford, OUP 2019).

132 Article 29 Working Party, 'Guidelines on Transparency under Regulation 2016/679' (WP 260, 29 November 2017).

133 Bert-Jaap Koops, 'The Concept of Function Creep' (2021) 13 *Law Innov Technol* 29, 30.

134 We thank Catalina Goanta for bringing up the point of the 'messy' coding practices in a discussion.

135 Nano-targeting is a practice of unique targeting of a user at the large population scale, eg identification of a unique Facebook user from the population of all Facebook users based on a combination of the user's non-unique interests [see Jose Gonzalez-Cabanas and others, 'Unique on Facebook: Formulation and Evidence of (Nano)targeting Individual Users with non-PII Data' (ACM Internet Measurement Conference, New York 2–4 November 2021)].

Much of these new applications of the data protection principles to code will likely echo and reinforce the protections arguably already created when those principles apply to data because deployment of code often constitutes data processing. Yet, new protections would be created as well that would go further and touch the code itself.¹³⁶ The application of the general data protection principles to code would provide a push for further exploration and articulation of the meaning and the protective potential of those principles both in the legal scholarship and practice, which is a welcome development seeing how the potential of those general principles has been underused so far.¹³⁷

Applying other data protection provisions to code could deliver interesting results.¹³⁸ Article 15(3) GDPR right to obtain a copy of the data, when applied to code, would result in a right to obtain a copy of the code itself which could facilitate algorithmic audit and accountability beyond what is currently possible under the right to obtain meaningful information about the logic involved in automated decision-making under Articles 13–15 GDPR.¹³⁹ Since the code would be most likely considered as data not obtained from the data subject and Article 14(1)(d) GDPR would apply, a data subject would receive a right to be informed both of the fact of deployment of an algorithm in their regard and the category of the algorithm. Article 16 GDPR will translate into the right to obtain from the controller corrections of the code, should it be ‘incomplete’, eg a model trained on incomplete data, or ‘inaccurate’, eg code not complying with scientific standards or a model trained on a faulty dataset. Articles 17 and 18 GDPR would enable a data subject to request erasure or restriction of the use of code, eg when it is no longer necessary for the declared purpose or it has been used unlawfully, etc. Article 25 (data protection by design) would create an explicit obligation to design both technology and organizational processes around software creation and use in a manner that respects the data protection principles and the rights and freedoms of the data subjects. In cases where software would be high risk in the sense of Article 35

GDPR, designers and users of code would be under the obligation to conduct impact assessment which includes assessment of necessity and proportionality of the code’s design and use, as well as the risks to the rights and freedoms of data subjects and how those could be mitigated.

Considering code as personal data under the GDPR brings us to think about the role of the software providers and (the degree of) their responsibility for the consequences of software use. Arguably, this idea has been already expressed in the preamble to the GDPR which states that ‘producers’ of products, services, and applications that involve processing of personal data ‘should be encouraged’ to implement data protection by design and by default (Recital 78).¹⁴⁰ However, the recital has been dormant because the preamble does not have binding effect and the obligations under the GDPR do not apply to technology designers unless they are data controllers. Applying Article 4(7) GDPR to code would suggest that at least some software providers would be (joint) controllers in relation to the use of code. This would strengthen legal protections against digital harms given that what causes them often happens ‘upstream’, ie on the level of design of algorithms. The downside of making software providers controllers is that it would further exacerbate the already complex distribution of responsibilities for compliance with the data protection law where ‘everyone is a controller’,¹⁴¹ and now software providers too.

Of course, the GDPR was not designed to be applied to software, and it is to be expected that at least some of its provisions—when used in relation to code—would not make sense or lead to absurd results. Yet, we struggle to think of many such examples, and those we can think of are not as absurd as one could expect. Invoking data portability rights in relation to code is the only instance of a possibly nonsensical application of the GDPR we could think of. We argued earlier that a piece of computer code developed by a software engineer where that software engineer is identified or identifiable (eg because his or her employer knows or is able to find out who authored the code) could be considered as personal data

136 Eg Veale and Edwards explore how the GDPR provisions other than Art 22 and the right to explanation could be used to address risks of automated decision-making, but they only consider cases where both algorithms and data in the conventional meaning of that word are involved. Lilian Edwards and Michael Veale, ‘Slave to the Algorithm? Why a ‘Right to an Explanation’ Is Probably Not the Remedy You Are Looking For’ (2017) 16 *Duke Law Technol Rev* 18, 44ff.

137 Eg Orla Lynskey, ‘Delivering Data Protection: The Next Chapter’ (2020) 21 *German Law J* 80.

138 See also Veale, Binns and Edwards (n 10), exploring how data protection rights would apply to ML models as personal data.

139 Specifically, Art 13(2)(f), Art 14(2)(g) and Art 15(1)(h) GDPR. On the limited potential of the right to obtain meaningful information about the

logic involved see eg Lilian Edwards and Michael Veale, ‘Slave to the Algorithm? Why a ‘right to an explanation’ is Probably not the Remedy you are Looking for’ (2017) 16 *Duke Law Technol Rev* 18.

140 Bygrave (n 131).

141 Eg Christopher Millard, ‘At this Rate, Everyone will be a [joint] Controller of Personal Data!’ (2019) 9 *IDPL* 217; Lilian Edwards and others, ‘Data Subjects as Data Controllers: a Fashion(able) Concept?’ (2019) *Internet Policy Rev* <<https://policyreview.info/articles/news/data-subjects-data-controllers-fashionable-concept/1400>> last accessed 19 April 2023; Fashion ID, Case C-40/17, [2018] CJEU (ECLI:EU:C:2018:1039), Opinion of AG Bobek; Michele Finck, ‘Cobwebs of Control: the Two Imaginations of the Data Controller in EU Law’ (2021) 11 *IDPL* 333.

in relation to the engineer. In that case, the software engineer would have a right under Article 20(1) GDPR to seek data portability and request a copy of the code in a structured, commonly used and machine-readable format and have the right to transmit the code to another controller, at least when the processing is based on the ground of necessity for the performance of a (labour) contract under Article 6(1)(b) GDPR. While Article 20(4) GDPR stipulates that the right to data portability shall not adversely affect the rights and freedoms of others, such as intellectual property rights that the company employing the software engineer has over that particular piece of code, according to Article 29 Working Party authoritative guidelines, even when conflicting IP rights are considered before answering a portability request, ‘the result of those considerations should not be a refusal to provide all information to the data subject’.¹⁴² A possible criticism of invoking the right to data portability in relation to code is that its protective effects are not clear, except for protection of the economic interests of software engineers and other data subjects, eg against being locked in by an employer or a service provider. Yet, this is not different from when the portability requests are made in relation to the more conventional personal data; if the concern is a potential conflict with IP rights in code, it is addressed by the limitations under Article 20(4); and should the IP-rights holder not be willing to run any risks and not port software, this should serve as an excellent incentive for more anonymization.

Relation to the upcoming EU AI regulation

One could argue that only confusion and no additional benefits will come from applying the GDPR to computer code to increase legal protection, especially because the recent legislative initiatives, specifically the proposed EU AI Act (‘AIA’),¹⁴³ already address many concerns associated with the impact of computer software on people. We disagree. While we do not aim at a comprehensive analysis of the AIA, we will highlight the four most

notable points where applying the GDPR guarantees to code would exceed protections created under the AIA.

First, the AIA only focuses on a subset of computer code, ie the AI systems characterized by a degree of autonomy and a capability to make inferences based on data and using ML and logic- and knowledge-based approaches.¹⁴⁴ In contrast, the GDPR would potentially apply to any type of computer software with intended or unintended impact on identified or identifiable natural persons.

Second, the AIA approaches AI systems as tools and therefore mostly focuses on their design and technical characteristics,¹⁴⁵ while the GDPR applied to computer code offers a systemic approach to addressing code-induced harms throughout the entire lifecycle of code from inception and design to use and destruction via its general principles, rights, and obligations.

Third, the AIA as any product-safety type legislation does not foresee any rights that either natural or legal persons affected by AI systems could invoke against AI providers or users. Importantly, in contrast to the Council’s general approach, the Parliament’s negotiation position introduces some individual rights, such as the right to lodge a complaint with a national supervisory authority (Article 68a), the right to an effective judicial remedy against a national supervisory authority (Article 68b), and the right to obtain an explanation of individual automated decision-making (Article 68c).¹⁴⁶ Yet, this issue is still subject to negotiations, and it remains unclear if and in what shape these rights will be present in the adopted Act.

Of course, the AIA does not emerge in legal vacuum and the AI targets often will be able to invoke many existing rights in other pieces of legislation. In addition to already existing implications of the data protection legislation for AI¹⁴⁷ and as discussed above, the GDPR can be one source of such complementary rights in relation to code rather than conventional data.

Finally, while the AIA draws some ‘red lines’ in the form of prohibition of some AI systems¹⁴⁸ and implements the values of transparency¹⁴⁹ and explainability of AI,¹⁵⁰ the GDPR approach is to a much greater extent

142 Article 29 Working Party, ‘Guidelines on the Right to Data Portability’ (WP 242 rev.01 (April 2017) 12, referring to Recital 63 GDPR concerning right of access which is similar to the right to receive personal data in a structured, commonly used and machine-readable format, the first element of the data portability right.

143 Unless explicitly acknowledged, we are relying on the general approach on the AI Act adopted by the Council, which was the latest document available at the time of writing [Council of the European Union, Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts—General approach, Brussels, 25 November 2022].

144 Art 3(1) AI Act.

145 Title III AI Act.

146 European Parliament, Amendments on the proposal for a regulation of the European Parliament and of the Council on laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts (COM(2021)0206 – C9-0146/2021 – 2021/0106(COD)), 14 June 2023.

147 Eg as discussed in Lilian Edwards and Michael Veale, ‘Slave to the Algorithm? Why a ‘Right to an Explanation’ Is Probably Not the Remedy You Are Looking For’ (2017) 16 Duke Law Technol Rev 18, 44.

148 Title II AI Act.

149 Title IV AI Act.

150 Eg Art 12 AI Act.

normatively underpinned, as manifested in the GDPR general principles of lawfulness, fairness, transparency, and proportionality (as a part of the data minimization and storage limitation).

The negatives: the unbearable burden of the GDPR

While applying the GDPR to computer code is tempting for its many advantages, the negative implications of construing code as personal data are significant and might outweigh the gains. The immediate and core negative implication of viewing code as personal data is that it might be stretching the already broad material scope of the GDPR too far, leading to the increased enforcement and compliance burden which for many will likely be unbearable. Even now, with code not considered within the scope of the GDPR, the Data Protection Authorities (DPAs) cannot cope with the volume of work. They continue to be understaffed¹⁵¹ and underfunded¹⁵² for the enforcement tasks they are formally given while the volume of personal data processing and possible GDPR violations grow exponentially as a result of increasing digitization. On the controllers' side, compliance with data protection law has also been considered labour-intensive, complex, and costly. The GDPR envisages a highly intensive regime of controller's obligations before, during, and after personal data is processed.¹⁵³ The expertise and manpower necessary to comply with the broad scope of the GDPR fully are often available at a premium price. Small- and medium-sized business that cannot afford the services of big law firms and consultancies either choose not to comply at all¹⁵⁴ or opt for compliance shortcuts, which do not do justice to the normative assessment and balancing exercises intended by the various GDPR provisions, eg if the intended processing is fair or really necessary for the purpose, or if a controller's legitimate interest is outweighed by the rights and interests of the data subjects.¹⁵⁵ Finally, even without code included, the concept 'personal data' is already sufficiently complex and

remote from its colloquial use to the effect that controllers often do not fully understand it and therefore may not always apply the GDPR where they should.¹⁵⁶ Should the GDPR obligations be extended to the code producers and users, this will further complicate the already complex division of responsibilities for GDPR compliance. In the circumstances of a multi-layered architecture of the Internet and modern provision of IT services, defining who a controller is and where their respective area of responsibility begins and ends will become even more difficult.¹⁵⁷

Considering computer code as personal data and thus including it within the scope of the GDPR would exacerbate these problems further and contribute to the image of data protection law as a piece of legal fiction, one of 'the most disregarded legislative frameworks' in the EU law,¹⁵⁸ which would collapse under its own weight if implemented properly.

Conclusions and discussion

This article puts forward a provocative argument that, when computer software is used to structure, assess, and nudge our behaviour, many instances of computer code are information relating to an identified or identifiable natural person, ie personal data in the sense of the GDPR. The argument is made in three steps.

While the silent presumption underlying data protection law and scholarship has been that personal data is information that means something about one human being to another (semantic information), there are other approaches to information where meaning to a human is irrelevant. While some code can contain semantic information about people, eg the notes its designer makes, all code constitutes information in the syntactic or functionalist sense as it shapes reality, ie it dictates to a computer what to do. Both syntactic and functionalist approaches to information are embraced by the definition of personal data as 'any information'.

Code is information that relates to people in three ways: by reason of content as in the examples of

151 According to a 2020 KPMG report, Dutch Data Protection Authority (DPA) needs 326 full-time staff to perform its envisaged tasks but has only 173. KPMG, *Onderzoek taken en financiële middelen bij AP* (Ministerie van Justitie en Veiligheid en AP, 2 November 2020) 2. According to the same report, the Dutch DPA is in a better financial position compared to most DPAs in other EU Member States.

152 Dutch DPA has repeatedly asked for additional funding but did not receive it. Autoriteit Persoonsgegevens, 'Miljoenennota: geen verhoging budget AP' (21 September 2021) <<https://autoriteitpersoonsgegevens.nl/nieuws/miljoenennota-geen-verhoging-budget-ap>> last accessed 19 April 2023.

153 Purtova (n 8) 76ff.

154 Christopher Kuner and others, 'The Business of Privacy' (2013) 3 IDPL 65.

155 Discussed in Purtova (n 8) 76ff.

156 To illustrate this point in the context of automated decision-making, FRA recommended providing more guidance 'as regards the meaning of personal data and its use in AI, including in AI training datasets'. EU Fundamental Rights Agency (FRA), 'Getting the Future Right: Artificial Intelligence and Fundamental Rights' (14 December 2020) <<https://fra.europa.eu/en/publication/2020/artificial-intelligence-and-fundamental-rights>> last accessed 19 April 2023.

157 René Mahieu, Joris van Hoboken and Hadi Asghari, 'Responsibility for Data Protection in a Networked World: On the Question of the Controller, 'effective and complete protection' and its Application to Data Access Rights in Europe' (2019) 10 JIPITEC 39.

158 X, Z v Autoriteit Persoonsgegevens, Case C-245/20, [2021] CJEU (ECLI:EU:C:2021:822), Opinion of AG Bobek [65].

semantic information above; but also by reason of purpose when code is intended to assess or influence human behaviour; or by reason of effect since all computer code imposes a ‘grammar of behaviour’ on its users and thus structures human behaviour.

Finally, people to code relates in the sense discussed above are often either identified or identifiable in the various meanings of identification: creators of code can either be identified to their clients or employees or identifiable by their distinct style of coding, etc. In sum, some code is personal data in the sense of the GDPR.

What should we make of this analysis? In the introduction to this article, we have identified three different and partially conflicting causes for which our argument could be useful. All three have merit but do not have our equal sympathies. First, our analysis can be used to further strengthen criticism of the data protection law as too broad and overreaching its limits. It has been argued before that personal data covers much more than what appears at first sight, and such a broad application threatens to make the GDPR a paper tiger which mostly looks threatening but lacks teeth. We share this concern to the point. We have shown how the DPAs and controllers alike already engage in the GDPR compliance theatre, and DPAs are lacking the resources to do their job. Extending the GDPR scope to software will exacerbate this problem even further. Yet, leaving compliance and enforcement aside, even though data protection was never meant to apply to code, our exercise of applying the GDPR general principles and data subject rights to code did not lead to nonsensical results. Here is where the second cause comes in.

Applying the GDPR to computer code can provide digital rights advocates with ammunition to claim rights to access and control the use of algorithms where those rights do not yet exist under alternative legal regimes. For one, while it is a generally accepted point that computer code regulates human behaviour, no rules limit this impact yet, and the proposed AI Act will only be relevant for a narrow subset of computer software. Some of the most interesting effects of the GDPR on code include the principles of purpose limitation, minimization, and information rights. The principle of purpose limitation will require that computer code is developed and used with a specific and legitimate purpose, and that software developed for one context cannot be used in a context incompatible with the original. This will prevent the values, norms, and assumptions characterizing one context and built into software design to ‘bleed’ into another context where the values, norms, and assumptions are different.

The principle of minimization will call for the assessment of the proportionality of the design features and use of software for a given purpose. The GDPR information rights extended to the code will strengthen algorithmic transparency and accountability. At the moment the information data subjects receive under Articles 13 and 14 GDPR is about ‘the existence of automated decision-making, . . . [and] meaningful information about the logic involved’ and its envisaged consequences, and is redacted by the controller. Applying the Article 15 GDPR right to access to code will give data subjects access to the code involved in the decision-making and therefore fuller, ‘raw’ information, which may enable a data subject to do more with it, for instance, subject to necessary expertise, uncover bias that cannot be detected by only looking at limited samples of input and output (the data subject’s own file or those of multiple data subjects joining forces¹⁵⁹). The added legal protection would also go beyond what the proposed AI Act is doing. We welcome such use of our analysis, not the least because the practice of applying the GDPR to code, eg as part of strategic litigation, will tease out the meaning of those principles as well as core concepts of the GDPR.

The exercise of applying the principles of data protection law to code brings us to the third cause, which is rethinking how legal protection against the risks of the digital society is built. We share the concerns of some critics that the scope of the GDPR has extended beyond meaningful compliance and enforcement capacities. Data protection law indeed was never meant to tackle code on its own, and the fact that data protection principles and rights seem useful in the context of code does not necessarily mean that these are the best way to address risks of computing. Therefore, instead of ridiculing data protection law, we prefer our analysis to inspire a fundamental rethinking of the regulatory design of legal protection against information-induced harms: the objectives behind regulation and the mechanisms chosen to achieve them. If the impact of computer software on people is a concern, as evidenced by the fact that it can trigger the application of the GDPR, what exactly is concerning about it? Are concerns technology-specific, ie come as a result of specific challenges that computing brings, or are they reflecting broader concerns in society, such as transparency and justification of decision-making, and fair commercial practices, where code is merely a facilitating tool rather than the core of the problem? In the former case, the regulatory attention should focus on computing, in the

159 As in the German large-scale data donation campaign to shed light on the black box practices of SCHUFA, the largest German credit rating agency, see <<https://openschufa.de/english/>> last accessed 28 September 2023.

latter case some sector-specific and not necessarily technology-focused measures already exist, although others might be necessary. To conclude, we hope that our analysis will do more than simply generate arguments for digital rights advocates in strategic litigation or for the GDPR critics, and will trigger broader thinking about what legal protection against digital harms should look like.

Funding

This contribution reports on the results of the project ‘Understanding information for legal protection of people against information-induced harms’ (INFO-LEG). This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No 716971). The paper reflects only the authors’ views, and the ERC is not responsible for

any use that may be made of the information it contains. The funding source had no involvement in the study design, in the collection, analysis and interpretation of data, in the writing of the report, and in the decision to submit the article for publication.

Acknowledgments

An earlier version of this paper was presented at the REGGOV conference on 5 July 2018, in Lausanne. The authors wish to thank Dr Michael Veale, Prof. Karen Yeung, Prof. Lyria Bennet Moses and other participants of the panel ‘Mechanisms for Securing Algorithmic accountability: Legal and extra-legal Approaches’ for their helpful feedback and suggestions.

*<https://doi.org/10.1093/idpl/ipad019>
Advance Access Publication 12 October 2023*