Implementations, interpretative malleability, value-ladenness and the moral significance of agent-based social simulations

Nuno David
Departamento de Ciências e Tecnologias da Informação
Centro de Estudos sobre a Mudança Socioeconómica e o Território (DINÂMIA'CET)
Instituto Universitário de Lisboa (ISCTE-IUL), Portugal
Email: nuno.david@iscte-iul.pt

Abstract

The focus of social simulation on representing the social world calls for an investigation of whether its implementations are inherently value-laden. In this article, I investigate what kind of thing implementation is in social simulation and consider the extent to which it has moral significance. When the purpose of a computational artefact is simulating human institutions, designers with different value judgements may have rational reasons for developing different implementations. I provide three arguments to show that different implementations amount to taking moral stands via the artefact. First, the meaning of a social simulation is not homogeneous among its users, which indicates that simulations have high interpretive malleability. I place malleability as the condition of simulation to be a metaphorical vehicle for representing the social world, allowing for different value judgements about the institutional world that the artefact is expected to simulate. Second, simulating the social world involves distinguishing between malfunction of the artefact and representation gaps, which reflect the role of meaning in simulating the social world and how meaning may or not remain coherent among the models that constitute a single implementation. Third, social simulations are akin to Kroes' (2012) techno-symbolic artefacts, in which the artefact's effectiveness relative to a purpose hinges not only on the functional effectiveness of the artefact's structure, but also on the artefact's meaning. Meaning, not just technical function, makes implementations morally appraisable relative to a purpose. I investigate Schelling's model of ethnic residential segregation as an example, in which different implementations amount to taking different moral stands via the artefact.

Keywords: Social simulation; interpretative malleability; representation gaps; misrepresentations; techno-symbolic artifacts; ethics.

1. Uncovering the nature of implementation in social simulation

The purpose of algorithms is both functional and representational. Some algorithms may be said to have functional purposes, such as deleting an email or ordering a list. The purpose in simulations may be said to be essentially representational – to represent other things in the world, such as a model to investigate the problem of ethnic segregation (see Schelling (1971). Algorithms may also be purposely designed for certain values, according to the requirements of programmers, stakeholders and end-users, such as protecting or undermining privacy. Despite articulating values, algorithms need not be

essentially value-laden, that is, they need not comprise *essential* value judgements. Whether an algorithm comprises an essential value judgement implies knowing its purpose. In general, it can be said that an algorithm comprises an essential value judgement if, for the same purpose and everything else being equal, designers who accept different value judgements would have rational reasons for designing different algorithms (Kraemer *et al*. 2011).

A comprehensive way to approach the purpose of a computational artefact is to take as a starting point that computerization involves an implementation process, of which the algorithm is only one ingredient. The focus of this article is to investigate what implementations are in social simulations and to consider the extent to which they have moral significance on account of essential value judgements. I use the term *social simulations* to refer to computational artefacts whose purpose is to simulate the social world in the field of agent-based computational social science.[1] Constructing a computational artefact implies an intention and an action on the part of a designer to implement it from an idea or specification. Unlike other fields of computerization, the purpose of representing something through social simulation does not lead the designer to the implementation of a systematic representation of the structural characteristics of the thing modelled. What is at stake is not only whether simulations, like other computational artefacts, may comprise value judgements. Modelling the social world involves more than representing facts about the physical world, and includes representing values and institutional facts whose meanings themselves in the social world depend on human agreement. Insofar as we accept that social simulations model social reality, two intrincate questions are raised. At stake is, on the one hand, what implementation is and how it participates in the constitution of a social simulation; and, on the other hand, whether representing the social world by simulation defines its implementations as inherently comprising essential value judgements.

Computers are known to have logical malleability (Moor 1985). In general, software is known to include the biases and particular worldviews of manufacturers, which can affect users in various ways (van den Hoven 2007). Insofar as computers are semantically interpreted, intended to represent other things in the world, they can also be said to have *interpretive malleability*. Their states may be used to stand for anything representable in terms of inputs, outputs and logical operators. If simulations are meant to represent cultural, social and political issues, it is worth addressing the extent to which their designs have moral significance, not just for the sake of epistemic issues of truth with respect to what they model – a subject that is not addressed here – but for the fact that simulations are interpreted by their users.

[1] See e.g. Edmonds and Meyer (2017) and JASSS – The Journal of Artificial Societies and Social Simulation, http://jasss.soc.surrey.ac.uk.

To uncover what implementation is and to assess its moral significance, I will delve into three arguments. First, social simulations have high interpretative malleability. The finding that the meaning of a technology may not be homogeneous among its users is corroborated in different scientific and philosophical domains. Studies in computer ethics and social studies of science and technology have argued that the early stages of design have interpretative flexibility, meaning that design responds to the requirements of different social groups and values, which eventually define the technical functions of the artefact (Pinch and Bijker 1984; Johnson 2006). I will focus on the susceptibility of a simulation to comprise essential value judgements regarding the representation of institutions, as well as the fact that implementations remain interpretatively malleable even after the design stages. Interpretive malleability implies recognizing that social simulations should not be understood as morally neutral, abstract, value-free implementations.

Second, the purpose of representing the social world through a computational artefact implies distinguishing between malfunction and representation gaps. Representation gaps underscore the role of meaning in the purpose of social simulations and whether such meaning is coherently maintained among the conceptual and computational models that make up a single implementation process. This need not imply that the designer holds the wrong conceptions about what the computational artefact is intended to simulate, nor that the implementation does not comply with its functional specification, but that designers with different value judgements may have rational reasons for developing different implementations.

I lay, as a third argument, that implementations of social simulations are akin to Kroes' (2012) techno-symbolic artefacts, constituted by technical function, structure and meaning. Techno-symbolic computational artefacts differ from Turner's (2014, 2018) computational artefacts in that their meaning decisively determines their effectiveness relative to a purpose. Meaning, not just technical function, makes implementations in social simulation susceptible to moral assessment.

The structure of this article is as follows: in section 2, I introduce the purpose of social simulations and the role of internal and external semantics in their development. In sections 3 and 4, I elaborate on the role of simulations in representing the social world and how this defines the kind of thing that implementation is. In sections 5 and 6, I discuss the interpretive malleability of social simulations, using the Schelling model as an example, in which different implementations amount to taking moral stands via the artefact. In section 7, I argue that the moral significance of a social simulation depends on its techno-symbolic ends. In section 9, I elaborate the conclusions.

2. Social simulations, framings and programming languages

To uncover what implementation is and appraise its moral significance, I first lay down two considerations: i) the purpose of social simulations and how designers establish their meaning; ii) and the role of computer programs' internal and external semantics for representing the social world.

2.1 Framings and ontological correspondence

Simulation in computational social science studies social theories and phenomena by specifying and implementing models in computers, viewed as representing aspects of some actual or hypothetical social world. The states of affairs represented from social reality vary and depend on the intended purpose. For the past two decades, the practice has been to computationally represent actors from the social world, in the sense that there can be an *ontological correspondence* between the actors and computational agents. According to Gilbert (2008, p. 14), ontological correspondence is what gives expressive power to social simulation, making it easier to design and interpret its outcomes than would be the case with other kinds of computational social modelling. Organizations, actions, preferences, norms, individual and social values are common ingredients of agent-based artificial societies. Simulation is used to analyse the artificial society, as well as the agents themselves, thereby predicting, explaining or simply describing social systems (see e.g. Edmonds and Meyer 2017).

To build simulations, modellers establish their purpose through a *framing*. This framing gives meaning to the construction of conceptual models, and ultimately to their implementation and interpretation as a computerized model. Take the much-cited culture dissemination model (Axelrod, 1997a, 1997b), whose modelled target is the problem of social influence. The framing elaborates on the theoretical or empirical importance of the subject matter under investigation, establishes the concepts and terms with which the problem will be investigated and sets the research questions. In the cited work by Axelrod, the subject matter is motivated by a variety of topics related to institutional contexts, such as state formation. The term 'culture' is envisioned in a broad sense, with no structural detail or specific meaning, insofar as it is 'the most generic term over which people influence over each other' and thus 'used to indicate the set of individual attributes that are subjected to social influence' (Axelrod 1997a, p. 204). Research questions are set as to how people influence each other and why influence does not lead to homogeneity.

The set of models leading to computerization are conceptual, pre-computerized models. For instance, Axelrod's (pre-computerized) model defines a *territory*, computationally specified with a grid, where each position represents a *culture* of an actor and each culture is specified as a finite word of integers. The simulation as a computerized model is explored by varying parameter values. The analysis investigates properties of aggregates of agents and the conditions upon which they form, such as the concepts of *regions* of cultures that are, post-hoc, formally modelled. Hence, a good deal of the simulation involves new, post-computerized concepts, which had not been considered in pre-

computerized models. Like pre-computerized models, post-computerized models receive their meaning from the target social theory or phenomenon, or the framing thereof.

In sum, it is not only aggregates of agents that emerge from the simulation, interpreted as new concepts, but also a succession of new models. These can be defined in arbitrary ways, and some might not be in a computational form. In any case, pre-computerized models must be implemented as computerized, physical, executable models, by passing through intermediate symbolic models, such as programs written in programming languages.

2.2 External semantics, simulation programs and specifications

The idea of 'emergence' in computer simulation should be placed with reference to a chain of models. The nature of simulation outcomes cannot be analysed without recognizing that the product of a simulation is a sequence of models transformed/embedded into other models (see Fetzer, 1999, in the philosophy of computer science, Sargent, 2005, in simulation and David et al., 2007, in social simulation). This raises intriguing technical and philosophical challenges. Consider, the idea of weak-emergence (Bedau 1997). In an attempt to characterize emergence in systems with high sensitivity to initial conditions, Bedau sets the notion of weak-emergence as follows: a system macrostate is emergent if and only if it is *derivable* from the system microdynamics and external conditions but only by simulation. Microdynamics are the rules governing the evolution of the system microstates; external conditions are the input to the simulation.

Bedau's definition appears indisputable if the notion of 'derivable' is read in some formal computational sense, which alludes to pure syntactic inference processes detached from what the input and the rules – expressed through a computer program – are meant to represent from the external environment. Programmers refer to internal symbols and data structures in programming languages which represent registers and memory locations containing the values to be processed. This need not represent anything from the external environment and may be called internal semantics. Conversely, what a simulation program is intended to represent from the external environment may, for purposes of ontological correspondence, be considered external semantics. Computer programs may be given external semantics to the extent that they are designed as intending to represent other things in the world (Piccinini 2008).[2] Indeed, the idea of social simulation as computational derivation seems insufficient to inquire into how social simulation copes with the semantically rich interpretations required by the modelling of cultural, social and political reality.

[2] These things need not be concrete objects or social structures in the external environment; they may be numbers, abstract structures, imaginary entities.

This difficulty parallels two viewpoints on the role of computing. First, a formal, abstract view that resonates with the idea of simulation as formal inference, in which it is claimed that the theoretical mechanisms acting in the simulation can explain the social world (see, e.g., Epstein 1999; Boero and Squazzoni 2005; Vu *et al*. 2020; and the perspective and objections of David *et al*. 2007; see also Anzola 2021). As we shall see, although this view explicitly acknowledges that the purpose of social simulations is to model the social world, it fails to recognize that implementations are value-ladden, or even takes for granted that implementations are somehow value-free. A second view of computing, initiated in the the field of computer ethics (see Moor, 1985), non-existent or rare in social simulation, recognizes that computing has become a technology constitutive of human life and explicitly acknowledges the value-ladeness of software design.

In any case, computational modelling involves the formulation of *specifications*. Specifications of social simulations need not be different from pre-computerized models. Their distinction comes insofar as the latter emphasize the descriptive character of representing the social world, but when viewed as specifications they carry a normative mood, which is meant to define what the computerized model ought to do. The way specifications are formulated in social simulation range from natural language to formal definitions, such as algorithms or high-level programs. The power of expressiveness varies widely, often inversely to the level of formalization. At one extreme, formal specifications are expressed as functions with some kind of input-output pairs, for which a program is formulated as a solution. At the other extreme, specifications are expressed informally and need not exist as functions; for example, 'Agents interact and exchange cultural characteristics'. They are framed as an invitation to formulate functional solutions by means of algorithms and programs, which eventually must map inputs onto outputs. At any rate, the simulation requires the interpretation of the modelled social theory or phenomenon in terms of high-level programs written in some programming language, and ultimately in terms of low-level programs. So, what is a social simulation program and implementations thereof?

It is relevant to investigate whether the implementation of social simulations has characteristics that are distinct from other fields of computing. Next, I consider the role of representation gaps, interpretive malleability and essential value judgements in the context of the kind of thing implementation is.

3. Semantic gaps

A possible way of viewing implementation is to understand it as semantic interpretation of an abstraction. According to Rapaport (1999), this requires two domains and a relation:

> [A] syntactic domain, the abstraction, characterized by rules of symbol manipulation; a semantic domain, similarly characterised; and a relation of semantic interpretation that maps the former to the latter. Put this way there is not an intrinsic difference between the two domains; what makes one syntactical and the other semantic is the asymmetry of the interpretive mapping. Thus, a given domain can be either syntactic or semantic, depending on the other domain. E.g. a computer process that implements a program plays the role of semantic domain to the program as the syntactic domain. The same program, implementing an algorithm plays the role of semantic domain to the algorithm as the syntactic domain. (Rapaport 1999, p.109).

In this view, implementations are realizations of abstractions in some abstract or physical medium. For instance, one may have an implementation of an abstract data structure – such as a row of student records – through another abstract structure, e.g. a linked list; implement the list in a programming language; implement the program in machine language; and implement the machine language program on a physical computer. Implementing abstractions into semantical domains may come with a semantic gap, which Rapaport attempts to explain through four relations:

A. 'A program P in high-level language is semantically interpreted by real-world objects', for example, a record with data from an actual student.
B. The program P is compiled to an implementation p in machine language. 'The compilation relation is, or includes, the relation between that student's record data structure and a construct of data types in machine language. Both A and B are semantic relations.'
C. The implementation in machine language p 'is in turn semantically interpreted by bits in a computer … All semantic relations are correspondences. The relation between program P and the bits is just another correspondence'.

In Rapaport's view, the semantic gap concerns a fourth relation, between the real-world objects of relation A (the student) and the real-world objects referred to in C (the computer bits), since they *both* would be semantic interpretations of the program P: 'the bits in the computer *simulate* the student. But simulation is, after all, a form of implementation. … The computer bits are a computer implementation of the student' (Rapaport 1999, p. 112, italics added).

Notice the semantic gap considered with reference to something external, relating a program or its implementation to the external environment, such as a real student. It is not blamed with reference to computational representations interpreted with formal semantics, which relate high-level to low-level programs, whose implementation (e.g. through compilation) does not require the assignment of referents from the external environment to the symbols that make up the programs. In this view, any semantic interpretation, including the mapping of an actual student on the machine, would be an implementation. This justifies Rapaport's conclusion that the bits in the computer are a computer implementation of the student, which seems to be either too strong or merely a metaphorical statement. What Rapaport seems not to account for is the role of specifications (Turner 2014). Specifications seem to be what is implemented as programs, not the things in the world they are intended to model. The act of implementing a computational artefact expresses the intention that the

artefact ought to function in a certain way. Rapaport's proposal faces problems in distinguishing between the descriptive and the normative roles of implementations.

4. Normativity, representation gaps and the social world

Consider the relation between programs and something external to the programs. On Rapaport's account, it does not seem questionable to say a program is interpreted semantically by external objects, such as the relation between a data structure and an actual or imaginary student. Although this is consistent with Rapaport's hypothesis that 'the bits are a computer implementation of the student', it does not fit with the thesis that syntactic domains are always abstractions. Still, once a *model of the student* – rather than the student himself – is understood as a specification, it could play the role of abstraction as a syntactic domain to the program as a semantic domain.

We implement on computer specifications, defined by stipulative definitions (Turner 2014; 2018). High-level programs are themselves specifications of low-level programs. How is a definition transformed into a specification? Moreover, how is a definition of a model transformed into a specification-of-a-model? Turner's theory of computational artefacts proposes an explanation to the former question. It is the act of taking a definition to have normative force over the implementation of an artefact that turns it into a specification. The problem remains unresolved with respect to the latter question. The act of specifying does not imply modelling external referents. A program implemented according to an input–output specification, without any external meaning, is still an implementation. Whereas the relation from a specification to a program can be properly characterized as implementation, the relation between a specification and a student – or between a program and a student – is appropriately characterized as modelling, which involves semantic interpretations using the terms of such models.

Yet, it seems unavoidable that the modelling of external referents exerts influence in the artefact's design. The purpose of computerizing models that meet certain specifications is *representation*. Note that I am not advocating that computation is individuated by representation. Piccinini's (2008) distinction between external and internal semantics remains pertinent. But if using computerized models means anything not detached from the world, then the purpose of computing involves representing other things in the world. This need not go as far as Smith's (1995) claim, when he states that there is no computation without representation. The claim that there is no social simulation without representation seems, nevertheless, ineluctable.

In what follows, I use the term representation in the context of external semantics, relating to whatever context of computational objects, interpreted as representing things in the social world. On these grounds, it may be assumed that representations in computers depend on the existence of

interpretations, which require programmers and/or users (Fetzer 1999). Moreover, programs and implementations thereof, like all technical artefacts, are intentionally produced things. Once a model of some social world is viewed as a specification-of-a-model-of-a-social-world, it becomes susceptible to being implemented for the purpose of representation. The model is given a normative mood, towards implementing the representations that it ought to specify. This points to the reason for the semantic gap, which could well be called a *representation* or *interpretation gap*: When representations expressed through an implementation do not live up to the representations as intended by the specification, that counts as a representation gap. If there is any semantic gap between a specification, a program or the computer bits, it is assessed with reference to the same thing modelled that they are all meant to represent.

Hence, if the computer hardware, its resident software and the compiler/interpreter are assumed to work properly, at least two reasons exist for an implementation not doing what it was intended to do. The first is miscomputing (Fresco and Primiero 2013). When computational artefacts malfunction, they are said to miscompute. Miscomputations are objective, non-conformities between an implementation and its functional specification. Another reason concerns *representation gaps* between an implementation and any forerunner specification, in that the implementation may not live up to representing what was intended from the specification. This need not occur due to misconceptions about the modelled target. Several reasons may be advanced: technical reasons such as the use of floating-point variables, which may lack sufficient precision to represent intended aspects of the modelled target, or epistemic reasons, given the low syntactic complexity of formal and programming languages and the resulting difficulties in expressing social and institutional reality.

In characterizing what an ideal social simulation language could be, Edmonds (2003) notes rather pragmatically the lack of expressiveness of formal computational models for representing the social world:

> The characteristics of social phenomena are frequently best approached using semantically rich representations (purpose, emotions, social pressure etc.) and these are difficult to translate into formal models. … The existence of semantic complexity means that modellers have three choices: 1. they can concentrate on those parts of social systems that they think are effectively modellable by syntactic representation; 2. they can adopt a pseudo-semantic approach, where the simulation manipulates tokens which are undefined inside the simulation (where they are computationally manipulated) but which are meaningful to the humans involved; or 3. they can avoid computational simulation altogether. (Edmonds 2003, p. 107)

It might be useful to delve deeper into what Edmonds calls ‘a pseudo-semantic approach’. Viewed as automatic symbol manipulators, computers manipulate marks that are infused with meaning when supplied with interpretations (Fetzer 2001). For reasons often associated with validation or treatability and elegance, simulated social actors are represented parsimoniously. Simple data types are used,

often with primitive variables known as tokens or tags, whose design is timidly constrained by the intended interpretation. Tokens can be ascribed with an arbitrary range of possible meanings. For example, in Axelrod's culture dissemination model an agent is a culture, designed as a word of integers associated with one single rule that loosely constrains its values. The observer has virtually total freedom as to the meanings ascribed to the word and its values – from the colour of a belt to a human intentional value (David et al., 2005). In Schelling's model, agents are devoid of structure, with each agent denoted by a bit taking one of two values – one or zero – expressed in the computerized model as two colours, which may allude to any arbitrary individual distinction, from ethnic characteristics to any kind of economic status.

This does justice to Edmond's sense of 'a pseudo-semantic approach', concerning tokens that are meaningful to the humans involved while undefined inside the simulation. Yet, insofar as the token is meaningful, what is lacking is not semantics but meaningful structure. This is not to say that the simulation is not a useful representation of what it is intended to stand for. For instance, cars in road traffic social simulations, which represent both the human driver and the car, need not have a complex internal structure. Where is the given meaning blamed on? Presumably, on the overall rules of the simulation, which must make sense to the designer and the users according to some purpose.

Nevertheless, simplicity in design is often claimed to have a price. Because most simulations are parsimonious, they have been criticized, with claims that descriptive, richer designs are needed, and that all relevant aspects of the modelled target with respect to a specific research problem should be brought to the design (Edmonds and Moss 2005). Yet, the call for a higher level of detail has not precluded simulations with very simple designs from being regarded as having different application domains, such as the legacy of Schelling's (1971) or Sakoda's (1971) simulation models (see Hegselmann 2017).

The issue may not be simplicity. Relevance of design is not determined by level of detail, but whether the resulting design is susceptible to empirical confirmation for a specific purpose.[3] Nevertheless, insofar as building a simulation requires models to be successively transformed, there could be something illusory, in that it is assumed that all such models have the same expressiveness for representing relevant aspects of the modelled target. Design, of course, is only another word for specifying and implementing, eventually at computational levels, using formal definitions, algorithms or programs themselves as representations of the social world. Once models are expressed at

[3] Arnold (2014), for example, contrasts the empirical usefulness of the Schelling model with the empirical uselessness of Axelrod's reiterated Prisoner's Dilemma simulations of the evolution of cooperation. According to Arnold (2014), while the latter model has 'remained entirely unsuccessful in terms of generating explanations for empirical instances of cooperation' the assumptions on which the former model rests can be tested empirically. As regards the Schelling model, on Arnold's account, whether individuals have a threshold for how many neighbours of a different colour they tolerate, and whether they move to another neighbourhood if this threshold is passed, is an assumption that can be tested empirically with the usual methods of empirical social research.

computational levels, representation gaps among implementations and specifications are likely to occur. This is more likely to be the case when programs are semantically enriched with meanings from the social and institutional world, whose facts they are meant to represent are themselves dependent on human agreement.

In this context, an inescapable question arises. Representation gaps bring about the possibility of misrepresentations. Representations in simulations can become morally problematic. Brey (2014) proposes some reasons for this. If they do not meet standards of accuracy, they thereby misrepresent. Another reason is biased representations; if they do not conform to standards of justice, they unfairly harm certain individuals or groups. People may disagree about the appropriate standards of representation for a specific simulation. I follow Brey in drawing attention to the possibility of misrepresenting and its moral appreciation. However, it must also pointed out that representation gaps between specifications and implementations do not necessarily amount to misrepresenting. Whether it does – and whether it is susceptible to moral evaluation – depends on the framing of the simulation and the values that come with the package. I will give a concrete example using Schelling's simulation model in the following two sections.

5. Interpretative malleability

Schelling's is a spatial proximity simulation model, whose purpose is to explain links between individual preferences and spatial segregation. Tokens move in a checkerboard and have one of two categories, such as whites and blacks, women and men or wealthy and poor. For each token, if the proportion of neighbours of the same category is below a threshold, the token moves to a random adjacent cell.

Many applications have been envisaged, but the interpretation in terms of ethnic segregation in urban environments has made Schelling's model famous. Edmonds puts it as follows (I quote Edmonds because he carefully distinguishes between the purpose of the simulation, the results of the computer model and the intended interpretation):

a) [Purpose]: [T]he highly dynamic nature of many social systems does not allow us even the pretence of ceteris parabis prediction of representative states, but rather we often use [social] simulations to inform our semantic understanding of existing social processes. In the case of the Schelling model, social scientists would not presume that they could actually predict the probability of self-organised segregation based on measurements of the critical parameter in real populations, but rather it informs their perception of the likelihood of the emergence of segregation among relatively tolerant populations, which they will then use as a part of their general evaluation of situations. (Edmonds 2003, p. 108)

Consider two different propositions on the grounds of ontological correspondence, which indicates the emergence of agent clusters with the same colour. Similar conclusions abound in the literature:

b) Results claimed as significant: There is a critical value for parameter C [the maximum proportion of like-coloured agents], such that if it is above this value the grid self-organises into segregated areas of single colour counters. This is lower than 0.5. (Edmonds 2003, p. 123)
c) Intended interpretation: Even a desire for a small proportion of racially similar neighbours might lead to self-organised segregation. (Ibid.)

Sentence b) refers to an empirically tested interpretation, according to the observation of the computerized model. However, the analogy between b) and c) can only be found through semantic enrichment of the former into the latter. It is not difficult to agree that the latter – an interpretation of the former – is not meant to hold any counterfactual evidential meaning in the strict scope of the computerized model. Support for c) can only be found on external evidence.

Strictly speaking, sentence b) may also be said to reflect an interpretation beyond what the behaviour of the computerized model empirically expresses. Unless the term 'segregation' had been defined at the design level and, literally, according to some formal sense of colour-like clustering, it may be said to involve semantic enrichment from the framing, with value judgements concerning the institutional meaning of 'segregation'. These need not be *essential* value judgements in the sense of Kraemer *et al*. (2011). They would be essential if designers who accept different value judgements had rational reasons for designing different implementations, thereby taking a stand on moral values. For example, if agents that strive for a certain proportion of racially similar neighbours were considered 'relatively tolerant'.

If we attempted to remove all traces of external semantics in terms of the social world, an alternative statement to b) could take the following form:

d) There is a critical value for parameter C [the minimum proportion of like-coloured agents], such that if it is above this value the grid results in areas of single colour counters. This is lower than 0.5.

Nevertheless, if no interpretation in terms of the social world is attempted, it amounts to being useless for the research problem. At whatever suggestive level, if it is somehow informative to the research problem, then it amounts to being interpreted. The semantics of a social simulation is pulled as much as it can be stretched. In fact, results and intended interpretation are more often conflated than not, including in reference books in the field. For example:

> The Schelling model assumes that a family will move only if more than one third of its immediate neighbours are of a different type (e.g. race or ethnicity). The result is that very segregated neighbourhoods form even though everyone is initially placed at random, and everyone is *somewhat tolerant*. (Axelrod 1997b, p. 24, italics added)

In social simulation the role of computerized models is to stand as representations of the social world. How is such a role achieved through implementation? A practical answer might be to argue that designers and users strive to implement appropriate representations in order to be interpreted and applied within a given purpose. Whether the designer's intended interpretation is in accord with the users' hinges, on the one hand, on how the designer implemented the computerized model to behave and, on the other, on how both the designer and the users articulate what such behaviour represents in the social world. This requires stipulating definitions in the form of specifications, such as *what is being tolerant*.

I will now introduce the concept of interpretative malleability. A simulation model's specification may be subject to variations and thus to different implementations. For example, Schelling's simulation model proposes specifying both the households' social and physical interactions in a discrete lattice, implemented with different kinds of lattices and neighbourhood forms or sizes, such as the Von Newman or Moore neighbourhoods.[4] Another example is Sakoda's model (1971). Based on Schelling and Sakoda simulation models' intellectual, technical and social history Hegselmann (2017) claims that they are the same, or that the former is an instance of the latter, despite having very different implementations.

Consider a simulation model. I say that a simulation model's implementation has interpretive malleability when different designers, who may or may not accept different value judgements, specify different implementations for the same simulation model. There can be several reasons why a simulation model is expressed in multiple implementation variants, not least, its scientific interest and reputation, but it does not seem possible to have an objective or quantified view of how interpretatively malleable an implementation is.

Notice, however, that the proposed definition does not claim different value judgements for justifying the different specifications, being neutral with respect to the reasons for the different implementations. In effect, a qualitative assessment of malleability can be identified. If the implementation is demonstrated to comprise essential value judgements and the implementation options remain controversial, I say the simulation has *high* interpretative malleability. In other words, given a simulation model's implementation, for a given purpose, when designers who accept different value judgements have rational reasons for specifying different implementations, the implementation has high interpretative malleability. Let me give a concrete example in the next section.

6. The Schelling model: a case of high interpretative malleability

[4] See, for example, Eric W. Weisstein, 'von Neumann Neighborhood', from MathWorld – A Wolfram Web Resource. https://mathworld.wolfram.com/vonNeumannNeighborhood.html.

Schelling's significant result occurs when the simulation reaches states in which tokens appear to be aggregated in colour-like clusters and the proportion of neighbours with the same category is said to be 'high', even if thresholds are said to be 'low'. For instance, if the threshold is 30 per cent, the simulation converges to states in which tokens reach a proportion of colour-like neighbours as 'high' as 70 per cent, and tolerance is said to be high. This is read as a paradoxical relation between tolerance and segregation. The claim is that patterns of ethnic segregation may be produced even in the absence of institutional discrimination factors, such as housing costs or ethnic prejudice. As an invisible hand, a high degree of segregation may be the result of dynamics arising from individual preferences that were not aimed at such levels of segregation.

While this interpretation has mostly been influential in the computational social science literature and beyond, there is not a consensus. For instance, Forsé and Parodi (2010) reject it. They claim that Schelling's interpretation is distorted by the failure to take proper account of the model's institutional and mathematical aspects, and that the former aspects have never been evaluated. According to the authors, the idea that the model produces greater segregation than institutional factors alone would have required a benchmarking of the levels of spatial segregation that result only by chance. They use a level of disorder of the system as an effective segregation measure, where maximum entropy corresponds to the perfect mixture and minimum to maximum segregation. Using this interpretation, the authors claim that the relation between levels of tolerance and resulting segregation is not paradoxical, but linear. If there is near preference of individuals to live in the neighborhoods where they are in the majority, it is not surprising that the system attains a degree of segregation. They further claim that the threshold is presented as a continuous variable, whereas it is in fact discrete, which modifies the interpretation of results. Lattices with small numbers of neighbours create the illusion of high tolerance when effective tolerance is in fact low.[5]

A response in refutation of Forsé and Parodi was made by Alan Kirman (2010), a reputed economist. He disputes the linearity between tolerance and segregation, suggesting a set of design variations in the literature, such as making a continuous approximation of the discrete lattice. On Kirman's account, the interpretation of Schelling prevails:

> [L]et us be absolutely clear about what Schelling claimed. He argued that even though people might have no intention to achieve segregation, their efforts to satisfy some threshold criterion for the number of unlike neighbours they would like to have, might lead to almost total segregation. This, even if the threshold was not very high. He never suggested that tolerance

[5] For instance, if an individual has, at most, three neighbours and a minimum tolerance level of a third of colour-like neighbours, s/he accepts only situations in which two or more of her/his neighbours are of like colour, which corresponds to two-thirds of minimum tolerance level. If, as in Schelling's model, there are at most eight neighbours and a minimum tolerance of one-third, the individual accepts 1 like neighbour out of 1 neighbour in all, 1 like neighbour at least out of 2 (1/2), 2/3, 2/4, 2/5, 3/6, 3/7 and 3/8. After many runs it amounts to approximately one-half of effective tolerance on a weighted average. Under this interpretation, the authors claim the model shows a linear relation between tolerance and segregation levels.

> for racial mixing was the explanation for the existence of ghettos. There are many explanations for their existence and Schelling has always been well aware of this. All that he observed was that despite people being relatively racially tolerant their individual actions could lead to a situation in which individuals of different races were separated. (Kirman 2010, p. 475)

He adds that the possibilities for different designs are endless: 'The possibilities are almost limitless but it is worth concluding by observing that Schelling made a simple observation about the possibility of segregation even though people were not particularly racist.'

Indeed, the profusion of variations with different implementations in the literature reflects the model's significant influence on the mathematics of complex system dynamics as applied to the problem of segregation. Nevertheless, controversy around the different interpretations in terms of the social world suggests high interpretative malleability. In another response to Forsé and Parodi, Rolfe (2010) suggests that the controversy is fundamentally political and not due to representation, but to misinterpretation of popular discourse, which often takes the model as evidence that only individual preferences rather than institutional factors are the main reason for segregation. Rolfe's position encounters difficulties, however, because it does not clarify where preferences come from, or whether preferences themselves reflect underlying institutional factors. It seems to suggest that preferences and institutional factors may be totally disconnected things. On this account, controversies do not stem from implementation, but from interpretation of the mathematical-based dynamics of the model.

In this respect, one aspect which goes beyond the mathematical aspects of the model is lost in the debate. Mathematical structures may be considered neutral in themselves. The question is whether the implementation is neutral. On Schelling's framing account, a token is said to be tolerant if it does not want a majority of colour-like tokens in his/her neighbourhood. Regardless of how the concept of tolerance is mathematically posed, or how the relation between tolerance and segregation is mathematically intepreted, the implementation carries an essential value judgement. Whether the threshold is above or below the majority, there is no objective fact involved in specifying that a certain threshold represents tolerant, somewhat tolerant or intolerant agents. To do so, eventually amounts to take a moral stand via the artefact. Different designers may have different value judgements, possibly conflicting ones, despite these being rationally justified.

Arguably, it may be said that qualifying a threshold as more or less tolerant comprises an essential value judgement, but this is no longer the case if we limit ourselves to saying that a threshold of one-third is *more* tolerant that a threshold of, for example, one-half. Whether or not this judgement is essential in Kraemer's sense, it may be assessed on how controversial the specification of tolerance is. The very idea of specifying tolerance with threshold variables does not come without contention, which is implicit in the debate. For instance, to the extent that tolerance is not framed from the

institutional point of view, with respect to social injustice or discrimination, tolerance and intolerance amount to simple exact synonymous in the simulation model and the terms are used interchangeably in the literature; presumably, because the threshold is a quantitative ratio, viewed as an outcome of some mechanism of social or psychological character that need not be represented. Underlying this is the idea that specification and implementation may be freed from comprising essential value judgements.

This neutrality thesis is difficult to sustain. The implementation of social simulations implies stipulating definitions, through specifications and programming languages. They are meant to represent brute facts that refer to physical reality, as well as to institutional facts in the social world. Representation of institutional meanings explains high interpretative malleability. Institutional facts are dependent on human agreement and require continued acceptance of the status to which a symbolic function is assigned (Searle 1995). The use of a threshold for representing tolerance imposes on the threshold a certain status relative to a particular meaning, that is, a particular understanding of the institutional framework that captures some intended concept of tolerance. Insofar as the institutions are not specified structurally – they are 'undefined' inside the simulation, as Edmonds puts it – the artefact performs its function symbolically. In other words, the function of the artefact is performed through meanings instituted by implementation, in the context of some form of agreement between designers and users, rather than by the computational structure alone.

The same goes for other institutional meanings attributed to tokens in the Schelling model, distinguishable only by its colour attributes, such as 'family residence', 'black neighbourhood', 'white man', 'a woman' etc. Many social simulations are presented as purposely 'abstract', of which Schelling's is a canonical example. Some authors present them also as 'metaphorical'. Indeed, 'abstract', in this sense, seems to convey not value neutrality, but interpretative malleability. Malleability is the condition for the purpose of the simulation as a metaphorical vehicle of representation of the social world, capable of dealing with different value judgements about the institutional world that the computational artefact is specified to represent.

7. Social simulations as computational, techno-symbolic artefacts

I want to wrap up one of the questions posed in this article, about the nature of implementation of social simulations. I do not intend, in the space available, to formulate an exhaustive theory of what implementation is, but to draw attention to the ingredients which give it interpretative malleability and moral significance. Current theories of computational artefacts assume that they are constituted by technical function and structure (Turner 2018). I advocate, however, that to adequately accommodate what an implementation is, additional ingredients need to be considered. If meanings are given as constituent ingredients of social simulations, there are more appropriate conditions to explain both implementation and its moral significance.

Johnson's (2006) perspective on computing posits the focus of moral evaluation as the result of technological intentionality. In this sense, moral significance is not intrinsic to the artefact but results from intentionality. Thus, when humans act with actions, such as implementing a computer program, they are constituted by the intentionality and efficacy of the artefact, insofar as the intentionality of the designer persists in the program via the efficacy of the design. Technological intentionality could suggest that the susceptibility of moral evaluation applies only to artefacts with functional characteristics that interact with and act in the world, such as an ordinary application or a game. This does not deny that the moral evaluation of software is dependent on the contexts of both design and use. But it leaves unexplained whether software is susceptible to moral evaluation whenever its purpose is exclusively representational. It also leaves unclarified whether or how the designer's intentionality persists via the efficacy of the design in interpretively malleable contexts.

The question may be clarified once we consider how the designer assesses whether the simulation fulfils its purpose. Much of the interest in social simulation lies in the surprising character of its results. Suppose that the designer of the Schelling simulation model takes as a surprise that a population of 'somewhat tolerant' agents converges to 'highly' segregated clusters. Does the novel result stem from a consequence of the model? Even if the artefact does not miscompute, and before ascribing the result to the model's outcomes, the designer should consider whether the implementation stands for what it was intended to. In other words, does the claim of 'high' levels of segregation with 'somewhat tolerant' agents stem from the model, from implementation errors, or is it the result of the implementation comprising essential value judgements?

This brings us to Turner's theory of computational artefacts and the problem of malfunction. Turner takes Kroes' (2012) theory of the dual nature of technical artefacts into the computational realm. I think it can go further and bring into the computational context another category of technical artefacts, coined by Kroes' *techno-symbolic artefacts*. This brings into play the role of meaning as a constitutive element of the artefact. Kroes' theory entails that a technical artefact, such as a screwdriver, is co-constituted by technical function, which refers to human intentionality, and physical structure. Functions describe the intentional, normative dimension of the artefact; what the artefact must do in order to be a screwdriver. Structure is the physical dimension. The function of the artefact is performed on the basis of the artefact's physical structure. If the causal, physical properties of the artefact are not in accordance with the function assigned by the designer, then the artefact malfunctions.

Artefacts may also be endowed with meaning (Kroes 2012). For example, a car is not simply a technical artefact in the sense of carrying people. It can represent a certain lifestyle or status, playing a symbolic role with a weaker connection to its physical structure, such as an antique car in a museum. Meanings can be the means through which an artefact performs its function, a symbolic status function

performed on the basis of collective intentionality. Take Searle's symbolic wall, such as a line drawn on the floor, which symbolically performs the wall's function rather than through a physical structure alone.

In Turner's account of computational artefacts there is no role for meaning. The role of technical function is performed by abstract specification and the role of structure by abstract or physical implementation. The function of a program is fixed by a specification and structure is materialized according to the syntax and semantics of some programming language. Meaning is limited to formal semantics: The compositional nature of programming languages provides the structure, describing how each of the functions of the language components is implemented in the abstract program of the physical device. The formal semantics of the programming language articulate the relationship between what a program does and what its specification says it should achieve. Likewise, the high-level program performs the role of specification of the low-level program, which in turn performs the role of artefact, and so on. Hence, structure can be realized abstractly or physically:

> abstract structure and physical structure are linked, not just by being in agreement, but also by the intention to take the former as having normative governance over the latter. On this account, computations are technical artefacts whose function is fixed by an abstract specification. (Turner, 2014)

Viewed in this way, implementation need not carry any moral significance. Turner evaluates malfunction on account of the meaning of the programs, provided by the formal semantics of the programming language. There is no role for representing the external environment.

As I have argued, the purpose of implementing social simulations is both to compute and to stand for something in the social world. What a program is specified to represent hinges on meanings in the social world, with weaker connections to the artefact's structure. Structure alone is not enough to perform the purpose of the simulation, nor to appreciate the occurrence of representation gaps. Purpose must be realized also in virtue of collective agreement regarding institutional meanings that refer to the social world. Kroes' technical artefacts are constituted by technical function, structure and meaning. Likewise, there is no reason to exclude meaning from the computational realm of artefacts. In order to make sense of what implementation is and its moral significance, meanings must be a constitutive ingredient of the artefact. Meaning, not just technical function, makes implementation morally appraisable relative to a purpose.

8. The moral significance of social simulations

Our dependence today on computing technologies stimulates discussion about how these change and impact society. One of the pressing classical questions relates to whether algorithms and computational technologies are value-laden. That information technologies support or hinder some

human values to the detriment of others is a presupposition of several design approaches. In similar but not entirely identical readings, the goal is to design technologies that account for human values (Friedman *et al*. 2008), embody or integrate intended values (Flanagan *et al*. 2008), embed or align with values (Ethically Aligned Design 2019), or are informed by or incorporate values (van den Hoven 2007). These methods assume that the development of computational artefacts is value-laden, implying a proactive approach in evaluating design alternatives to find those that best serve intended purposes. However, while these methods seem to assume that software is susceptible to moral evaluation, current theories of what constitutes an implementation suggest that this applies only to software with functional features, which interact with and act in the world. This leaves unclarified whether software is susceptible to moral significance when its purpose is exclusively representational, such as in the case of social simulation.

The focus of simulation in representing the social world led me to investigate whether implementations are inherently value-laden. While investigating what king of thing implementation is, it turned out that the purpose of representing something does not lead to the implementation of a systematic representation of the structural characteristics of the thing modelled. Insofar as implementation is expressed in several conceptual and computerized models, representation gaps among implementations and forerunner specifications are likely to occur. This is more likely to be the case if implementations are semantically enriched with meanings from the social and institutional world, where the facts they are meant to represent are themselves dependent on human agreement. In this context, conceptual and computerized models comprise value judgements regarding how institutions are interpreted and how they should be represented. Implementations are interpretatively malleable, insofar as different designers specify different implementations for the same simulation model.

The purpose of representing the social world further implies distinguishing between malfunction and representations gaps. Representation gaps reflect the role of meaning in the purpose of social simulations and how meaning may not remain consistent among the models that constitute implementation. This need not entail that the designer holds incorrect conceptions about what the computational artefact is intended to simulate, but that two designers who accept different value judgements may have rational reasons for specifying different implementations, or that two users with different value judgements may have rational reasons for interpreting the implementation differently. If the implementation is demonstrated to comprise essential value judgements and the underlying implementation options remain controversial, the simulation has high interpretative malleability. Schelling's simulation model, in which the design of different implementations amounts to taking a moral stand via the artefact, has high interpretive malleability.

The dependence of social simulations on meaning shows that they are not only computational artefacts with a technical function. Meaning, however, seems not to have a role in current computational accounts of implementation. I take Kroes' idea of techno-symbolic artefacts and consider that social simulations are a kind of techno-symbolic computational artefact, constituted by technical function, structure and meaning. Meaning in terms of the institutional world, not just technical function, makes implementation of social simulations morally appraisable.

The interpretative malleability of social simulations, and, in general, the theory I have here proposed, poses challenges to the empirical and practical field of social simulation, and possibly to other fields of computing whose implementations have representational purposes. It implies recognizing that social simulations are not abstract, morally neutral instruments of study regarding the cultural, social or political reality which they model. In my view, this does not diminish their usefulness for investigating the social world, but it calls for implementation methods which should be able to clarify, for the recipients of the simulation, the values at stake and the design options undertaken. To the extent that social simulation is being increasingly used as a tool in policy-making, the evaluation of results should not be limited to the adequacy of conceptual models, specification models and outcomes. Beneficiaries must be fully informed about the implementation logic and options. A social simulation that provides poor documentation and/or does not provide its source code is not only methodologically inappropriate: It may be deemed morally questionable. Moreover, greater digital literacy and skills will be required among those individuals and groups affected by the resulting policies, so they are able to freely and critically interpret computer simulations of the social world.

Acknowledgments

I thank the anonymous reviewers for their careful reading of the manuscript and their useful comments and suggestions.

This work was partially developed while the author was visiting the Department of Values, Technology and Innovation, Sections Ethics / Philosophy of Technology, at TU Delft, Netherlands. The author was partially supported by the Portuguese

References

Anzola, D. (2021). The theory-practice gap in the evaluation of agent-based social simulations. *Science in Context* (forthcoming).

Arnold, E (2014). What's wrong with social simulations? *The Monist* 97(3):361–379.

Axelrod, R (1997a). The dissemination of culture: A model with local convergence and global polarization. *Journal of Conflict Resolution* 41(2):203–226.

Axelrod, R (1997b). *The complexity of cooperation – Agent-based models of competition and collaboration.* Princeton University Press.

Bedau, MA (1997). Weak emergence. In: James Tomberlin (ed) *Philosophical perspectives: mind, causation, and world*. Blackwell, Oxford, Vol. 11, pp 375–399.

Boero, R, Squazzoni, F (2005). Does empirical embeddedness matter? Methodological issues on agent-based models for analytical social science. *Journal of Artificial Societies and Social Simulation* 8(4):6.

Brey P (2014). Virtual reality and computer simulation. In: RL Sandler (ed) *Ethics and emerging technologies*. Palgrave Macmillan, London.

David, N, Sichman, JS, Coelho, H (2005). The Logic of the Method of Agent-Based Simulation in the Social Sciences: Empirical and Intentional Adequacy of Computer Programs. *Journal of Artificial Societies and Social Simulation* 8(4):2.

David, N, Sichman, JS, Coelho, H (2007). Simulation as formal and generative social science: The very idea. In: Carlos Gershenson, Diederik Aerts and Bruce Edmonds (eds), *Worldviews, science, and us: Philosophy and complexity*. World Scientific Publishing, pp 266–284.

Edmonds, B (2003). Towards an ideal social simulation language. In: Sichman *et al.* (eds) *Multi-agent-based simulation II, LNAI*, V. 2581, Springer, pp 105–124.

Edmonds, B and Meyer, R (2017). *Simulating social complexity – A handbook*. Springer, Berlin and Heidelberg.

Edmonds, B and Moss, S (2005). From KISS to KIDS – an 'anti-simplistic' modelling approach. In: P. Davidsson *et al*. (eds) *Multi-agent-based simulation 2004*, *LNAI, 3415*. Springer, pp 130–144.

Epstein J (1999). Agent-based computational models and generative social science. *Complexity* 4(5):41–59.

Ethically Aligned Design (2019): A vision for prioritizing human wellbeing with artificial intelligence and autonomous systems. The IEEE Global Initiative for Ethical Considerations in Artificial Intelligence and Autonomous Systems, IEEE Standards Association.

Fetzer, J (1999). The role of models in computer science. *The Monist* 82(1):20–36.

Fetzer, J (2001). Thinking and computing: Computers as special kinds of signs. In: M. Bergman and J. Queiroz (eds) *The Commens Encyclopedia: The Digital Encyclopedia of Peirce Studies*.

Flanagan, M, Howe, D, Nissenbaum, H (2008). Embodying values in technology: Theory and practice. In: J. Van den Hoven and J. Weckert (eds) *Information Technology and Moral Philosophy* (Cambridge Studies in Philosophy and Public Policy). Cambridge University Press, Cambridge, pp 322–353.

Forsé, M, Parodi, M (2010). Low levels of ethnic intolerance do not create large ghettos: A discussion about an interpretation of Schelling's model. *L'Année sociologique* 60(2):445–473. https://doi.org/10.3917/anso.102.0445.

Fresco, N, Primiero, G (2013). Miscomputation. *Philosophy and Technology* 26:253–272.

Friedman, B, Kahn, PH, Jr., Borning, A (2008). *Value sensitive design and information systems.* In *The handbook of information and computer ethics*. John Wiley & Sons.

Gilbert, N (2008). *Agent-based models* (Quantitative Applications in the Social Sciences). Sage Publications.

Hegselmann, R (2017). Thomas C. Schelling and James M. Sakoda: The intellectual, technical, and social history of a model. *Journal of Artificial Societies and Social Simulation* 20(3):15.

Johnson, DG (2006). Computer systems: Moral entities but not moral agents. *Ethics and Information Technology* 8:195–204.

Kirman, A (2010). A comment on 'low levels of ethnic intolerance do not create large ghettos' by Michel Forsé and Maxime Parodi. *L'Année sociologique* 60(2):475–480. https://doi.org/10.3917/anso.102.0475.

Kraemer, F, van Overveld, K, Peterson, M (2011). Is there an ethics of algorithms?. *Ethics and Information Technology* 13:251.

Kroes, P (2012). *Technical artefacts: Creations of mind and matter: A philosophy of engineering design*. Springer, Dordrecht.

Moor, J (1985). What is computer ethics? *Metaphilosophy* 16:266–275.

Piccinini, G (2008). Computation without representation. *Philosophical Studies* 137:205.

Pinch, TJ, Bijker, WE (1984). The social construction of facts and artefacts: Or how the sociology of science and the sociology of technology might benefit each other. *Social Studies of Science* 14(3):399–441.

Rapaport, WJ (1999). Implementation is semantic interpretation. *The Monist* 82(1):109–130.

Rolfe, M (2010). A comment on 'low levels of ethnic intolerance do not create large ghettos' by Michel Forsé and Maxime Parodi. *L'Année sociologique* 60(2):481–492.

Sakoda, JM (1971). The checkerboard model of social interaction. *Journal of Mathematical Sociology* 1(1):119–132.

Sargent, R (2005). Proceedings of the 37th Winter Simulation Conference, pp 130–143.

Schelling, TC (1971). Dynamic models of segregation. *Journal of Mathematical Sociology* 1:143–186.

Searle, J (1995). *The construction of social reality*. The Free Press, New York.

Smith BC (1995). Limits of correctness in computers. In: D. Johnson and H. Nissembaum, (eds) *Computers, ethics & social responsibility*. Prentice Hall, pp 456–469.

Turner, R (2014). Programming languages as technical artefacts. *Philosophy and Technology* 27(3):377–397.

Turner, R (2018). *Computational artefacts – Towards a philosophy of computer science.* Springer.

van den Hoven, J (2007). ICT and value sensitive design. In: P Goujon, S Lavelle, P Duquenoy, K Kimppa, V Laurent (eds), *The information society: Innovation, legitimacy, ethics and democracy*, Vol. 233. Springer, Boston, MA.

Vu, TM, Probst, C, Nielsen, A, Bai, HM, Petra S, Buckley, C, Strong, M, Brennan, A, Purshouse, RC (2020). A software architecture for mechanism-based social systems modelling in agent-based simulation models. *Journal of Artificial Societies and Social Simulation* 23(3):1.