The Design of Intelligent Socio-Technical Systems[*]

(Extended Abstract)

Andrew J I Jones[1], Alexander Artikis[2] and Jeremy Pitt[3]

[1]Department of Informatics, King's College London, UK
[2]National Centre for Scientific Research 'Demokritos', Athens, Greece
[3]Department of Electrical & Electronic Engineering, Imperial College London, UK

1 Introduction

………there is no such thing as philosophy-free science; there is only science whose philosophical baggage is taken on board without examination. [5, p. 21]

To a significant extent, research in Computer Science that aims to develop socio-technical systems has to address issues pertaining to the interpretation of social and organizational concepts. The components of socio-technical systems, be they artefacts or humans, carry out their work by interacting with each other against a social, organizational or legal background. The field of Autonomous Agents and Multi-agent Systems has for some time represented an obvious example of this work, but the important part played by social concepts extends into other parts of Computer Science too. Consider – to mention just three further domains – Computer Security, where the notions of *trust*, *reputation* and *role* have figured prominently; E-commerce, where the representation, formation and fulfillment of contracts is fundamental; and E-government, where representing and reasoning about policies and norms are essential.

In Biology and Social Science, in Jurisprudence, and in Analytical Philosophy, among other disciplines, we find examples of conceptual models designed to enhance our understanding of the nature of organized interaction. In writing this paper, our initial question was this: in their construction of so-called *computational* models of social concepts, such as those mentioned in the previous paragraph, have computer scientists been sufficiently informed by *conceptual* models of social phenomena, the construction of which was not motivated by computational considerations, but aimed primarily to reveal, in a systematic fashion, the structure and interconnections of the concepts themselves ? Through its attempt to answer that question, the principal contribution of this paper is a proposed approach to the engineering of socio-technical systems that respects the interdisciplinary nature of the task, in regard to both its theoretical and practical dimensions.

In the full paper, of which this document is an extended abstract, we proceed in Section 2 by giving some examples of work that would justify a negative answer to our initial question, and we explain their shortcomings. Against that background,

[*] AT2012, 15-16 October 2012, Dubrovnik, Croatia. Copyright held by the authors.

Section 3 describes an approach to the engineering of socio-technical systems in which rich, conceptual-analytical models and computational frameworks are combined, providing a basis for *principled operationalisation*, observing that similar methodological concerns have arisen in the field of biologically-inspired computing. We describe the approach in terms of a sequence of steps and, accordingly, in Section 4 we formulate and illustrate adequacy criteria that, ideally, the key steps should satisfy. In the concluding section we suggest, in particular, that if our general methodological proposals were to be adopted, they should have significant consequences for the ways in which researchers are trained, not least in the area of Autonomous Agents and Multi-agent Systems.

2 Motivating Examples

In this section we consider three examples of work on the engineering of socio-technical systems in which social concepts – specifically *trust*, *role* and *normative power* – have figured prominently.

2.1 Normative Power

Any reasonably comprehensive model, formal or informal, of norm-governed multi-agent systems must be able to accommodate norms pertaining to institutionalized normative power, in addition to those that express obligations and permissions. It is a commonplace feature of organizations that particular agents, individually or collectively, are empowered to carry out actions, the consequences of which have a significant bearing on the way the organization is governed or administered. For instance, some public officials/bodies will be empowered to create a state of marriage between two individuals, or to validate wills, or to appoint some other persons to particular roles (including roles that themselves involve the possession of powers), or to create or modify laws and regulations. Powers of this sort are types of rights, or entitlements, that some agents have, and others lack. There is a substantial body of literature, stemming from Hohfeld [8], that focuses on the systematic characterization of types of rights-relations, including in some cases formal analyses of these relations expressed in terms of a small set of basic operators drawn from modal logic [11,12,20,15,9,10].

Oren et al. [17] present a model of what they call ‘normative power’, which they associate with the power to create and/or modify norms. While they refer to the Hohfeldian tradition, they make no use of the analyses offered therein, preferring instead to characterize normative power by means of a first-order logic tuple, the key element of which is called ‘mandators’. “Mandators is a set of predicates identifying agents” [17, p. 817], and “A mandator of the form *professor(x)* means that any agent in the professor role is able to exercise the power”, for instance the power to place a student under an obligation to write a conference paper [17, p. 819]. Note that the interpretation of what it means for an agent to be able to exercise a normative power is not here explicated; rather, it remains implicit in the natural-language reading the authors assign to the ‘mandator’ predicate. This attempt at modelling jumps straight from an *informal* description of the concept of normative power to a first-order logic representation – a transition that is presumably motivated primarily by considerations of computational tractability.

The practice of giving a rather simple, but computationally convenient, representation of complex social concepts is quite widespread in Computer Science – the areas discussed in sections 2.2 and 2.3, to follow, provide further examples of it. But it is a problematic practice because it provides no clear picture of the nature of the simplifications made, and thus also no proper framework for assessing whether a system implemented on the basis of such a computational model behaves in a way that adequately reflects the properties of the social concept itself.

2.2 Role-based Access Control (summary only)

The NIST model for role-based access control (RBAC) [22] formed the basis for the ANSI RBAC standard.

In their introductory section, the authors maintain that “….the basic role concept is simple: establish permissions based on the functional roles in the enterprise, and then appropriately assign users to a role or set of roles” [22, p.47]. But very soon thereafter they allude to a structure that is considerably more complex: “Roles could represent the tasks, responsibilities and qualifications associated with an enterprise”. It is revealing that the latter description of roles is by no means confined to mere permissions, since it appears that some key aspects of the overall NIST model are motivated by the largely unexplicated assumption that agents get assigned to particular roles in virtue of their qualifications, and that – as role-holders – they also acquire obligations associated with the organizational tasks for which they are deemed to be responsible. (Note also the remark: “A role is a job function or job title within the organization with some associated semantics regarding the authority and responsibility conferred on a member of a role” [22, p. 51].)

We note the apparent lack of clarity and uniformity in these informal descriptions of the role concept. Then, with reference to some of the issues that have arisen in the further development of the RBAC approach, we argue that considerable advantage could have been gained had those developments been informed and directed, from the outset, by a comprehensive, precise model of the role concept itself.

2.3 Trust (summary only)

A third source of motivating examples is provided by the literature on the design of socio-technical systems addressing issues of trust in agent interaction. A useful survey of that literature has recently appeared [19], in which the authors present a classification of a range of models in terms of several dimensions.

We discuss aspects of their classification, and note with interest that a principal conclusion they draw very strongly suggests the need for a methodology that brings together *both* conceptual modelling *and* a computational framework informed by it. Just one of the eighteen approaches considered in the survey achieves this synthesis, according to the authors; concerning that one model they say, in their concluding remarks, that it "…..summarizes one of the most prominent future research lines in trust and reputation models: *implementable cognitive models*" [19, Section 5]. In our view, the key point about those cognitive models is that they are *conceptual* models, designed primarily to clarify the *trust* concept itself; non-cognitive analyses of *trust* might also be possible, but the essential methodological requirement emerging from the survey pertains to the need to *integrate* the conceptual and computational aspects.

3 Towards a Method for Designing Intelligent Socio-Technical Systems

This section outlines the structure of an approach to engineering intelligent socio-technical systems in which an abstract analysis of social concepts informs the development of a computational framework, providing a suitable platform for system implementation.

We are here, in part, building on the synthetic method underlying some research in artificial societies and artificial life [25]. The main steps of the synthetic method involve generalizing from some observations of phenomena to produce a theory, on the basis of which an artificial system can be constructed and then used to test predictions deriving from the theory. The outcome of applying the synthetic method is to engineer an artificial system, with the resulting animation, experiments or performance serving to support or refute the theory. Several other attempts to apply ideas from the social sciences to the design of computational systems (see, e.g., [6]) have followed a similar pattern. Furthermore, researchers in biologically-inspired computing, notably those concerned with artificial immune systems [2], have developed a comparable approach.

3.1 Structure of the method (summary only)

The root of the concerns we highlighted in Section 2 may be expressed in the following way: we fully accept that, in the design of socio-technical systems, the need for computational tractability makes it probable that there will have to be some degree of simplification of the principal social concepts involved; but in the interests of good scientific practice – and thus, also, good engineering practice – it is essential to achieve as clear a picture as possible of just what it is that is being simplified. We need first to have a clear characterization of the phenomena, before we set about simplifying them. Any computationally motivated simplifications should be carried out against the background of, and should be properly informed by, precise models of the social concepts themselves. And, crucially, the construction of those conceptual models should not itself be constrained by considerations of computational tractability.

We present our proposals in terms of different steps pertaining to the description and analysis of the members of a set *S* of observed social phenomena, as illustrated in Figure 1. The principal steps are theory construction, formal characterization, and principled operationalisation.

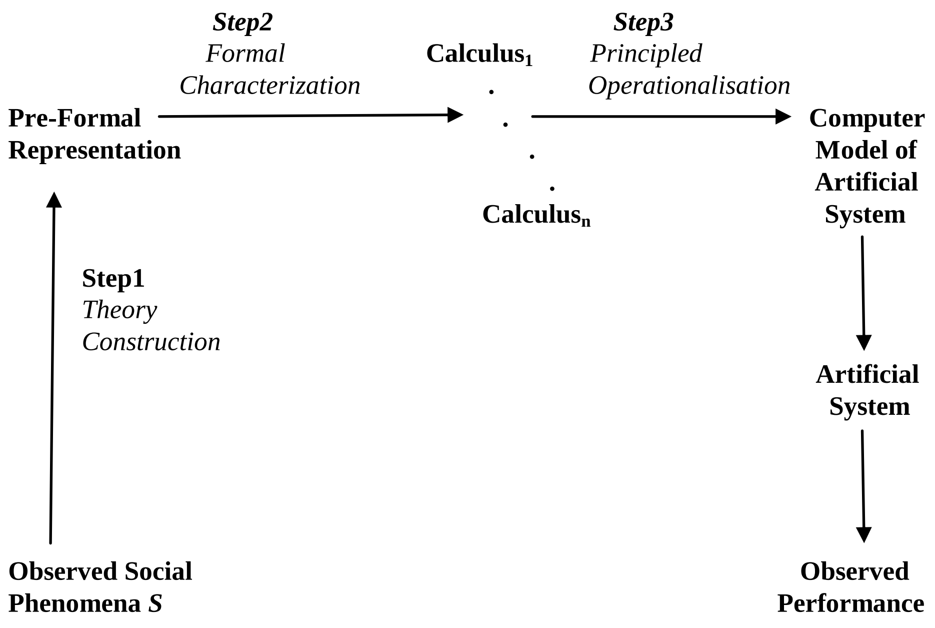

Fig. 1. Simplified Diagram Representing the Proposed Method for Engineering Socio-Technical Systems[1]

Step1 representations of the members of *S* are characterized by the natural-language terms that are used to denote the social phenomena concerned – terms such as *empowerment*, *role*, *trust*, and so on.

The process of *formal characterization* is the process leading from *Step1* representations to *Step2* representations. A *Step2* representation must be expressed in a formal language or 'calculus' of some kind, where by 'calculus' we mean any system of calculation or computation based on the manipulation of symbolic representations.

However, there are *Step2* representations of various sorts, which for our purposes are appropriately divided into two sub-steps, or phases. *Step2-Phase1* representations define a conceptual framework for the phenomena in *S*, in which conceptual analyses are expressed in terms of, for instance, a formal-logical language; the key point about *Step2-Phase1* representations is that they aim to provide an analysis of conceptual structure, identifying the fundamental elements of which complex concepts are composed, and articulating the principles governing their composition and inter-relations.

[1] The diagram is simplified in that it depicts the method as uni-directional. However, there are important aspects of two-way interplay between the key steps. We describe some of these in Section 3.3 of the full paper.

Crucially, *Step2-Phase1* representations are constrained primarily by considerations of expressive capacity, *not* those of computational tractability.

By contrast, it is at the *Step2-Phase2* stage that issues of computational tractability begin to come into play. A *Step2-Phase2* computational framework models the conceptual framework of *Step2-Phase1* in terms of a language, or languages, that are themselves amenable to the development of software implementations; the key points to note about *Step2-Phase2* computational frameworks are that the principles governing their composition are informed and guided by the conceptual characterizations of *Step2-Phase1*, but that they may well involve some degree of simplification, or approximation. Crucially, however, on this approach the designer of a computational framework will have a very clear picture, from *Step2-Phase1*, of the nature of the simplifications or approximations that may have been made.

Step2-Phase1 representations are essentially theory-facing, whereas *Step2-Phase2* representations are essentially implementation-facing. The recommendation to adopt two *Step2* phases is motivated by the need to guard against trying to force subtle societal concepts into the straitjacket of some particular computationally tractable language.

One further observation should be made about the relationship between the two phases of *Step2*. We have emphasized that some of the conceptual detail that is captured in the *Step2-Phase1* model might be omitted from the *Step2-Phase2* computational framework; but we should also point out that there may well be abstractions that can be tolerated at the *Step2-Phase1* level that cannot be ignored in an implementation-facing framework, for example the representation of *time*, and of *the means by which* a particular state of affairs is to be brought about.

Step 3 representations are exemplified not so much by formalisms but by tools that are employed in moving from the computational framework to a *model* of the artificial system, with algorithmic intelligence of the agents embedded in identifiable system processes. This is the transition that we call *principled operationalisation*. Operationalisation may well be selective, vis-à-vis the computational framework; but it is *principled* operationalisation in that it is conducted in the full knowledge of which selections have been made, and why.

3.2 Step2 Exemplified (Summary only)

3.2.1 Step2-Phase1: Formal Characterization using Modal Logic

The focus is on some formal-logical tools, drawn from modal logic, that have been used in the analysis of the group of social concepts that were discussed in Section 2. (This choice is of course not intended to suggest that modal logic is the only tool suited to the formal analysis of social concepts.)

3.2.2 Step2-Phase2: Formal Characterization with Action Languages

Suitable examples come from research in AI on action languages: the Situation Calculus [18,16,21,14]; the Event Calculus ([13]); C+, an action language with transition system semantics [7,1] - Artikis et al. [3] have used this language to develop executable MAS specifications in terms of *institutionalized power*, *permission* and *sanction*. Other relevant references here include [4], [23] and [24]. In general, Sergot's framework was informed by abstract conceptual models of normative systems.

3.3 A Note on the Interplay between Steps (Summary only)

Figure 1 is simplified, and fails to bring out the fact that the design process is frequently two-way, not uni-directional. In the full paper we supply examples to illustrate this point.

4 Adequacy Criteria for Step2 (Summary only)

4.1 Adequacy Criteria for Step2-Phase1

The principal criterion pertains to *expressive capacity*. This sub-section discusses and illustrates various aspects of this requirement.

4.2 Adequacy Criteria for Step2-Phase2

The key criteria discussed and illustrated in this sub-section are: a formal semantics; a declarative semantics; expressive capacity; support for computational tasks; efficient execution.

References

1. Akman, V., Erdogan, S., Lee, J., Lifschitz, V., and Turner, H. (2004), "Representing the Zoo World and the Traffic World in the language of the Causal Calculator," Artificial Intelligence, 153(1-2), 105-140.
2. Andrews, P., Polack, F., Sampson, A., Stepney, S., and Timmis. (2010), "The CoSMoS Process, Version 0.1: A Process for the Modelling and Simulation of Complex Systems," echnical Report YS-2010-453, University of York.
3. Artikis, A., Sergot, M., and Pitt, J. (2007), "Executable Specification of a Formal Argumentation Protocol," Artificial Intelligence, 171(10-15), 776-804.
4. Craven, R., and Sergot, M. (2008), "Agent strands in the action language nC+," Journal of Applied Logic, 6(2), 172-191.
5. Dennett, D. (1995), Darwin's Dangerous Idea: Evolution and the Meanings of Life, London: Penguin Books.
6. Edmonds, B., Gilbert, N., Gustafson, S., Hales, D., and Krasnogor, N. (eds.), Socially Inspired Computing. Proceedings of the Joint Symposium on Socially Inspired Computing, AISB (2005).
7. Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N., and Turner, H. (2004), "Nonmonotonic causal theories," Artificial Intelligence, 153(1-2), 49-104.

8. Hohfeld, W. (1913), "Some Fundamental Legal Conceptions as Applied in Judicial Reasoning," Yale Law Journal, 23(16).
9. Jones, A., and Sergot, M. (1993), "On the Characterization of Law and Computer Systems: The Normative Systems Perspective," in Deontic Logic in Computer Science eds. J.-J. Meyer and R. Wieringa, Chichester: John Wiley and Sons.
10. Jones, A., and Sergot, M. (1996), "A formal characterization of institutionalised power," Journal of the IGPL, 4(3), 429-445.
11. Kanger, S. (1957), New Foundations for Ethical Theory, University of Stockholm, Department of Philosophy. Also in R. Hilpinen (ed.), Deontic Logic: Introductory and Systematic Readings, Dordrecht: Reidel, 1971.
12. Kanger, S., and Kanger, H. (1966), "Rights and Parliamentarism," Theoria 32, 85-115.
13. Kowalski, R., and Sergot, M. (1986), "A Logic-Based Calculus of Events," New Generation Computing, 4(1), 67-96.
14. Levesque, H., Pirri, F., and Reiter, R. (1998), "Foundations for the Situation Calculus," Linköping Electronic Articles in Computer and Information Science, 3.
15. Lindahl, L. (1977), Position and Change: A Study in Law and Logic, Dordrecht: reidel.
16. McCarthy, J. (1963), "A Basis for a Mathematical Theory of Computation," in Computer Programming and Formal Systems, P. Braffort and D. Hirschberg, (eds.), Amsterdam: North-Holland, 33-70.
17. Oren, N., Luck, M., and Miles, S. (2010), "A Model of Normative Power," in Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 815-822.
18. Pinto, J., and Reiter, R. (1993), "Temporal Reasoning in Logic Programming: a case for the Situation Calculus," in Proceedings of Conference on Logic Programming, D. Warren, (ed.), Cambridge, Mass.: MIT Press, 203-221.
19. Pinyol, I., and Sabater-Mir, J. (2011), "Computational trust and reputation models for open multi-agent systems: a review," Artificial Intelligence Review (Springer Online-First).
20. Pörn, I. (1970), The Logic of Power, Oxford: Blackwell.
21. Reiter, R. (1993), "Proving Properties of States in the Situation Calculus," Artificial Intelligence, 64, 337-351.
22. Sandhu, R., Ferraiolo, D., and Kuhn, R. (2000), "The NIST Model for Role-Based Action Control: Toward a Unified Standard," in the 5th ACM Workshop on Role-Based Access Control, RAC '00, 47-63.
23. Sergot, M. (2008), "Action and Agency in Norm-Governed Multi-Agent Systems," in Proceedings of ESAW VIII, A. Artikis, G. O'Hare, K. Stathis and G. Vouros, (eds.), LNAI 4995, Springer, 1-54.
24. Sergot, M., and Craven, R. (2006), "The Deontic Component of Action Language nC+," in Deontic Logic in Computer Science (DEON'06), L. Goble and J.-J. Meyer, (eds.), LNAI 4048, Springer, 222-237.
25. Steels, L., and Brooks, R. (1994), The Artificial Life Route to Artificial Intelligence: Building Situated Embodied Agents, New Haven: Lawrence Erlbaum Ass.