# Social Machines for All

Blue Sky Ideas Track


Petros Papapanagiotou
University of Edinburgh
Edinburgh, United Kingdom
ppapapan@inf.ed.ac.uk

Alan Davoust
University of Edinburgh
Edinburgh, United Kingdom
adavoust@inf.ed.ac.uk

Dave Murray-Rust
University of Edinburgh
Edinburgh, United Kingdom
d.murray-rust@ed.ac.uk

Areti Manataki
University of Edinburgh
Edinburgh, United Kingdom
A.Manataki@ed.ac.uk

Max Van Kleek
University of Oxford
Oxford, United Kingdom
max.van.kleek@cs.ox.ac.uk

Nigel Shadbolt
University of Oxford
Oxford, United Kingdom
nigel.shadbolt@cs.ox.ac.uk

Dave Robertson
University of Edinburgh
Edinburgh, United Kingdom
dr@inf.ed.ac.uk



## ABSTRACT

In today's interconnected world, people interact to a unprecedented degree through the use of digital platforms and services, forming complex 'social machines'. These are now homes to autonomous agents as well as people, providing an open space where human and computational intelligence can mingle—a new frontier for distributed agent systems. However, participants typically have limited autonomy to define and shape the machines they are part of.

In this paper, we envision a future where individuals are able to develop their own Social Machines, enabling them to interact in a trustworthy, decentralized way. To make this possible, development methods and tools must see their barriers-to-entry dramatically lowered. People should be able to specify the agent roles and interaction patterns in an intuitive, visual way, analyse and test their designs and deploy them as easy to use systems.

We argue that this is a challenging but realistic goal, which should be tackled by navigating the trade-off between the accessibility of the design methods –primarily the modelling formalisms– and their expressive power. We support our arguments by drawing ideas from different research areas including electronic institutions, agent-based simulation, process modelling, formal verification, and model-driven engineering.



## KEYWORDS

Social Machines; Design; Analysis; Modelling; Model-Driven Development

**ACM Reference Format:**

Petros Papapanagiotou, Alan Davoust, Dave Murray-Rust, Areti Manataki, Max Van Kleek, Nigel Shadbolt, and Dave Robertson. 2018. Social Machines for All. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 5 pages.

*Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.*



## 1 INTRODUCTION

As an increasing share of human interactions have become mediated by software systems, a new class of socio-technical systems has emerged, sometimes called Social Machines [45, 46] or Human-Machine networks [14]. Examples include Wikipedia, Twitter, dating applications, citizen science platforms, Q&A sites, MOOCs, and many more. Autonomous agents are slowly acquiring meaningful roles in this space – one can think of the proliferation of *bots* on Wikipedia, Twitter, or Facebook, and many platforms are specifically aiming to support agents via rich APIs as part of their design (e.g. the Slack messaging platform or the Github code repository).

In a nutshell, we are witnessing the rise of open multi-agent systems on the Web, including both human and software agents, communicating via Web technologies.

The original conceptualisation of social machines was that machines would do the administration while humans were free to carry out creative goals, with an intention that this should give rise to "new forms of social process"[5, pp. 172–175]. This drive towards new forms of social process can be seen around the edges of platforms: when hashtags on Twitter emerge as coordination artefacts for sharing scientific knowledge (as in the case of #icanhazpdf[1]), or when blog comments become a vehicle for carrying out research level mathematics [10]. These are situations where existing computational infrastructure was re-purposed by a community— in effect creating a new social machine within existing infrastructure.

However, there is also widespread concern about the centralisation of many of these architectures: monolithic platforms provide useful services, but they stifle innovation, and enforce centralised notions of what sociality may or may not be.

In this paper, we outline a research agenda to democratize the development of Social Machines, with the ultimate aim of supporting a wide range of people in creating mixed human-machine communities, where computational agents support human sociality

[1]Academics (and others) use this Twitter hashtag to request scientific articles which are inaccessible to them due to paywalls, and others share the articles by email or by responding with a link to an accessible copy. Similar activities happen on Reddit (`/r/scholar` subreddit) and on Facebook [19]

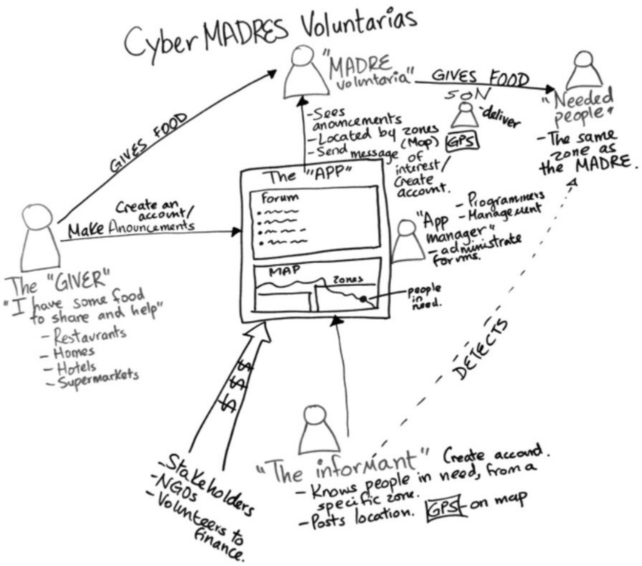


**Figure 1: The *Cybermadres* social machine, sketched by participants of a 2016 Digital Humanities workshop. This represents the activities of a group of volunteers in Mexico (the "Madres"), who collect excess food from restaurants and distribute it to people in need. The diagram shows roles, interactions, implementation hints and social aspects of the system concisely and comprehensibly.**

in an open, extensible manner. We view this initiative as a parallel to the push towards the *re-decentralisation* of the Web, in line with projects such as IPFS [4] and SOLID [11].

Social Machines are a frontier for open agent systems on the web, and as such we want a future where they are available to all.

## Our Vision: Social Machines for All

We have been inspired by working with a range of people over several years to understand how they might go about developing social machines. The *Cybermadres* diagram (Figure 1) is a typical output from a design workshop. This informal sketch of a social machine was the result of an open-ended process, seeded with an initial set of diagrams illustrating interactions in a particular, simplified way, to be extended and subverted as necessary.

Despite its informality, the diagram conveys many relevant aspects of this social machine: we can see actors with different roles, coordination infrastructure, and sketches of interaction between them. Arguably, people familiar with existing social machines can easily interpret the diagram and visualize the finished system.

This demonstrates the ability of laymen to design social machines from a standing start in just a few hours. The next step to produce a working social machine is to create Web-based infrastructure to support it. Enabling the automatic transformation of such designs into working systems would massively enhance the possibilities for bottom-up, democratic creation of social machines. So what are the challenges to make such automatic transformations possible?

In order to do this, we need conceptualisations that are both formal and useful to novice designers, ideally in the form of a visual language that allows non-experts to describe social machines both at a technical and a social level. We then need automation machinery that can transform such designs into usable software, ready to be populated with human and software agents. We also need methods and tools for analysing, debugging and understanding social machines— from "unit tests" to ensure that the interaction patterns make sense to advanced simulation and verification techniques to validate technical and social properties.

To support our contention that this vision is not entirely out of reach, we identify three corresponding main research challenges, discussing the main promising avenues to address them.

## 2 GRAND CHALLENGE: FROM INTUITIVE MODELS TO DEPLOYED SOCIAL MACHINES

The central challenge in democratizing the development of social machines is bridging the gap between intuitive, comprehensible models and fundamentally complex systems. We envision the designers specifying only high-level patterns of interaction (e.g. "make announcement", in the CyberMadres example) or coordination infrastructure (e.g. "a voting system"), and obtaining "out of the box" computational support for those elements, based on a limited set of building blocks.

Parallels exist with Model-Driven Engineering [18], which aims to make models, not code, the primary artefact of the software engineering process. High-level models are transformed into executable code, through the use of patterns, frameworks and templates. While this technique has been applied to multi-agent systems [1, 39], these approaches are intended for high expressiveness and result in extremely complex agent models. Similarly, MDD has been applied to *Virtual Organizations* [2, 3] and the systematic transformation of simple workflow models into executable code (e.g. UML activity diagrams into BPEL[20]).

By viewing hybrid system development as model creation, we can begin to decompose our overarching challenge into different sub-problems, organized around the modelling artefact.

(1) How can a layperson create intuitive models of complex agent systems? This requires simple representations with enough flexibility to incorporate social elements, such as motivations, incentives, influences, yet specific enough to provide a formal enough system that can be analyzed, simulated, and deployed.
(2) How should the agent coordination patterns produced from the model be deployed and monitored on the Web, particularly when both human and software agents are involved?
(3) What tools can be provided to test, debug and predict the success of social machines, creating a design cycle that includes social and technical assurance and feedback?

Our remaining sections unpack these into a series of challenges for different aspects of potential approaches.

## 3 CHALLENGE #1: CREATING INTUITIVE MODELS OF COMPLEX INTERACTIONS

We first consider the following challenge:

> What modelling formalism could enable a non-expert to design a meaningful class of social machines?

MAS researchers are used to formalisms that describe agent interaction in a way that supports execution—for example Electronic Institutions (Islander [15], EIDE [34]), and organizational models based on norms (MOISE [22], OPERA [12]). In particular, recent efforts for hierarchical institution governance [24] attempt to specify the permitted norms of participation in a social machine in a way that fits well with the framework described here. Similarly Process Management researchers work with workflow specification languages (e.g. BPMN [36], BPEL [35]). While we are not arguing against the need for rich and complex languages such as these, for this challenge we are interested in extending existing practice in space of *simple* specifications, with the understanding that this will necessarily constrain the range of systems that can be described. Essentially, most serious development is at the complex end of the spectrum, covering intricate formal systems in unambiguous detail.

In contrast, we hope that formalisms exist where a small number of high-level constructs can be assembled like building blocks to describe complex interaction patterns, such that the proverbial 80% of applications is catered for with 20% of the complexity. Varying specifications of primitives exist: HTTP has a few *verbs* (`GET`, `POST`, `PUT`, `DELETE` etc. [16]) while Twitter is highly constrained, with just 3: `TWEET`, `LIKE`, `RETWEET`. In the agent engineering world, FIPA's communicative acts specification has around 20 *performatives* [17] with complex compositional semantics. At the simple end of the scale, data science became massively parallelised with just two verbs: `MAP` and `REDUCE`.

It is therefore natural to pose the question: can we extrapolate from this observation and determine a set of primitives needed to build social machines? Central to this challenge is finding representational levels that are broadly applicable enough to be re-usable, yet precise enough to support execution. Saying an actor `MANAGES` a collection brings a host of intuitive semantics around create, read, update and delete operations; an `N-WAY` agreement between agents or `RESERVING` physical resources are concisely understandable, and re-usable components of complex systems. The appropriateness of different components may also depend on the particular type of social machine that is being designed. The selected building blocks need to accommodate different categories of social machines, for example based on known taxonomies [45, 46].

These primitives then need to be composed into larger scale machines—in Figure 1, actors need to `ASK_ALL` of the restaurants for excess food and then `RESERVE` it for collection. This requires a formal composition language that systematically links such processes by establishing, for example, information or resource flows and shared data structures.on.

There is an array of theories and tools for supporting such composition. BPEL [35], for example, enables business process composition linking to web services and via executable semantics, but does not have a visual representation and supports centralized orchestration (as opposed to peer-to-peer choreography). Process algebras such as the $\pi$-calculus [29] and session types [23] offer solid theoretical foundations for the systematic analysis of distributed systems, but are far from intuitive for non-expert users. Recent efforts in formally verified process specification and composition [38] have taken steps towards visual process composition with formally verified properties and automatic code generation.

An additional challenge arises in maintaining the social side of these systems: motivations for participation, incentives [43], social norms and expectations [8], knowledge sharing etc.

This vision raises several key questions: first of all, what should a primitive look like and how to develop a usable set of primitives? Secondly, what does it mean to automatically compose these primitives? Finally, how can we fill the gap between a highly abstract primitive, which is by nature underspecified, and a concrete implementation, which needs to be fully functional?

## 4 CHALLENGE #2: CREATING INFRASTRUCTURE FOR PEOPLE AND AGENTS ON THE WEB

Functioning social machines require a combination of infrastructure, interactions that use the infrastructure, and people and agents to carry out those interactions. The process of creating infrastructure is time consuming, and extremely error prone; similarly, community building takes time, and relies heavily on network effects. These requirements are at odds with widening participation, requiring resources not available to the general public.

However, many innovations in Web engineering work by reusing existing infrastructure: "If This Then That"[2] provides simple scripts to work *across* multiple platforms; Zapier[3] allows users to construct workflows based on web applications they already use; and work has been done on integrating electronic institutions with social media platforms [31]. This has the dual benefit of i) abstracting away authentication, security, storage provision etc. and ii) integrating with the practices that users already have.

This stands in contrast to typical development, which requires bespoke infrastructures to be hand-crafted by developers—clearly at the "expressive but difficult" end of the complexity spectrum. At the "accessible but constrained" end of the scale, *Panoptes*[4] has made the deployment of crowdsourcing social machines so easy—in terms of specification, infrastructure and access to a community of volunteers—that several new projects are started each day.

Another observation is that *dedicated* infrastructure is actually not always needed: Social Machines can be built via social conventions, on top of existing communications infrastructure: a good example is the #IcanHazPDF social machine, which has created a community around a particular Twitter hashtag.

These social conventions go hand in hand with having a declarative modelling language, with specifications such as "reach consensus over X", "discuss Y". There are many ways these could be fulfilled that share a set of core executable semantics (i.e. providing a single answer that 'most' actors are comfortable with) while differing implementation details. Several threads of work are relevant here: "Do What I Mean" (DWIM [48]) allows abstract, intuitive, specifications and sensible error behaviours; *convention over configuration* supports extremely minimal configurations by providing sensible, compatible defaults for everything; finally, web service discovery techniques (e.g. [41]) are used to retrieve an appropriate web service implementation given a user-provided specification.

[2] https://ifttt.com/, see also [37]
[3] https://zapier.com
[4] https://panoptes.zooniverse.org/

The challenge then is whether existing platforms can be used as composable components in the design of social machines. A secure multi-agent voting protocol may be appropriate in some places, but sometimes a `VOTE` verb would be better served with a Doodle poll and bindings for computational agents to make use of the results. This requires a combination of both formal systems expertise and a sociological understanding. In particular, what range of human preferences and ad-hoc decisions can be intuitively described by an average user and is manageable automatically in terms of system configuration. We believe such a range would extend much further than a software system or web service.

## 5 CHALLENGE #3: ANALYSING AND DEBUGGING THE SOCIAL COMPUTER

As in traditional software engineering, we expect the design process for social machines to be cyclical: once designs are produced, they need to be evaluated and debugged, then updated accordingly. The designer should be able to analyse a model to validate that it accurately reflects their particular vision. It is also essential towards better understanding the functionality, limitations, and means of improvement of the social machine.

Agent-based simulation techniques [27] adapted to social machines can help test the flow and outcome of particular scenarios (e.g. a defined set of agents and parameters), in the spirit of unit tests. This would also enable explorative what-if scenarios with different parameters to gain insights on quantifiable properties (e.g. costs and delays), information flow, load balancing, etc.

Formal verification techniques, including model checking [9], can also be used to mathematically verify properties across all possible scenarios, ensure the correct system behaviour, eliminate errors, and establish safety. They can also generate counterexamples of unwanted behaviour that breaks expressed properties, such as "the system never reveals information X about an agent", value properties such as "if an agent pays for something then they will receive it or their money back", and safety properties such as "no-one can steal money from another agent".

The trade-off between expressiveness and automation is predominant here too. The logical languages employed by simulation and verification tools are seldom intuitive for the uninitiated, and therefore a more expressive, declarative language would be required. Executable semantics would also be required for such an analysis. These are available for languages such as LCC [42], BPEL (to some extent) and other protocol or workflow specification languages, but not in practical visual languages such as BPMN or flowcharts. Efforts to formalise the semantics of BPMN [13, 50] and develop formal verification tools [47] are clear indicators of the perceived usefulness of such techniques in the community.

Moreover, unlike purely technical systems, social machines include unpredictable human agents, and the overall "behaviour" of the machine depends both on the materiality of the system and on the collective agency of the user population. Testing and analysing this behaviour will require a realistic simulation of choices and social behaviour of human agents and of how the system's regulative infrastructure affects the agents. This knowledge – which currently exists in mostly informal sources [25, 26, 51] – needs to be made available at the point of design.

Another key challenge of social machines is their *adoption* by the community [21, 40]. The social sciences provide a number of models to explain how and why humans engage with technology: these range from highly technical game-theoretic models [52] to empirical models from social psychology [25, 40]. The availability of such techniques requires modelling constructs describing both the technical elements (e.g. protocols) and social behaviour aspects (e.g. economic payoffs), making it a considerable challenge.

Finally, systematic monitoring a deployed social machine is the fundamental drive for refinement, continuous improvement, as well as investigation of the differences between the model and the actual social machine, particularly in the non machine mediated parts of the model. In addition, systems may change through use: new practices emerge which require infrastructure [6], or human behaviour is taken over by computational agents [33]. Related work has shown how social machine observation can be used to better understand social norms and incentives, and refine the interaction models to better support desired practice and optimize efficiency [32].

Such analysis requires automatic recording of event logs, perhaps in the form of provenance graphs [28, 30], and the use of techniques such as process mining [49]. One can then compare expected interaction models with the actual usage of the system (*conformance checking*), identify variances and exceptional behaviour, or even infer new interaction models. Such techniques have successfully been used within multi-agent systems [7, 44]. Making them usable by non-experts through intuitive interfaces and languages would be a major breakthrough towards our vision.

## 6 CONCLUSION

As social machines are rapidly becoming an integral part of today's world and a frontier for the deployment of agent systems, it is paramount that their development becomes accessible to more than just large or well-funded institutions. Non-expert individuals should be able to design, build and analyse social machines that leverage distributed intelligence to benefit different communities and the general public. The challenges associated with this vision are as grand as its potential impact:

*#1.* How can we create intuitive models of complex interactions that balance what people want to express and what can be executed? How can we discover the primitive interactions that can become the building blocks of social machines? How can we incorporate intuitive but abstract, primitive but usable specifications?

*#2.* How can we make social machines easily deployable on the web? How can we leverage existing infrastructure and model-driven development to that end?

*#3.* How can we enable rigorous analysis and debugging that not only reveal system properties from the technical perspective, but also delve into the social aspects of these complex hybrid systems?

We argue that these questions require a unified approach involving research in multiple areas, including model-driven development, process modelling, agent-based simulation, game theoretic analysis, formal verification, software engineering and configuration, and social sciences. This is a unique opportunity to bring these research communities together, drawing strong contributions from them to bring the DIY social machine revolution within reach.

## REFERENCES


[1] Carlos Agüero, Jorge Carrascosa, Miguel Rebollo, and Vicente Julián. 2013. Towards the development of Agent-Based Organizations through MDD. *International Journal on Artificial Intelligence Tools* 22, 02 (2013), 1350002.

[2] Jorge Agüero, Miguel Rebollo, Carlos Carrascosa, and Vicente Julián. 2009. Developing virtual organizations using MDD. In *Proceedings of the workshop on agreement technologies (WAT 2009)*. 130–141.

[3] Jorge Agüero, Miguel Rebollo, Carlos Carrascosa, and Vicente Julián. 2010. MDD for Virtual Organization design. *Trends in Practical Applications of Agents and Multiagent Systems* (2010), 9–17.

[4] Juan Benet. 2014. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561* (2014).

[5] Tim Berners-Lee, Mark Fischetti, and Michael L Foreword By-Dertouzos. 2000. *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor.* HarperInformation.

[6] Axel Bruns and Jean E Burgess. 2011. The use of Twitter hashtags in the formation of ad hoc publics. In *Proceedings of the 6th European Consortium for Political Research (ECPR) General Conference 2011*.

[7] Lawrence Cabac, Nicolas Knaak, Daniel Moldt, and Heiko Rölke. 2006. Analysis of multi-agent interactions with process mining techniques. In *MATES*. Springer, 12–23.

[8] Amit K Chopra and Munindar P Singh. [n. d.]. From social machines to social protocols: Software engineering foundations for sociotechnical systems. In *Proceedings of the 25th International Conference on World Wide Web*. 903–914.

[9] Edmund M. Clarke, Jr., Orna Grumberg, and Doron A. Peled. 1999. *Model Checking*. MIT Press, Cambridge, MA, USA.

[10] Justin Cranshaw and Aniket Kittur. 2011. The polymath project: lessons from a successful online collaboration in mathematics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1865–1874.

[11] Carlos E Cuesta, Miguel A Martínez-Prieto, and Javier D Fernández. 2013. Towards an architecture for managing big semantic data in real-time. In *European Conference on Software Architecture*. Springer, 45–53.

[12] MV Dignum. 2004. *A model for organizational interaction: based on agents, founded in logic*. SIKS.

[13] Remco M. Dijkman, Marlon Dumas, and Chun Ouyang. 2008. Semantics and analysis of business process models in BPMN. *Information and Software Technology* 50, 12 (2008), 1281 – 1294. https://doi.org/10.1016/j.infsof.2008.02.006

[14] Aslak Wegner Eide, J Brian Pickering, Taha Yasseri, George Bravos, Asbjørn Følstad, Vegard Engen, Milena Tsvetkova, Eric T Meyer, Paul Walland, and Marika Lüders. 2016. Human-machine networks: towards a typology and profiling framework. In *International Conference on Human-Computer Interaction*. Springer, 11–22.

[15] Marc Esteva, David De La Cruz, and Carles Sierra. 2002. ISLANDER: an electronic institutions editor. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*. ACM, 1045–1052.

[16] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. 1999. Hypertext Transfer Protocol – HTTP/1.1, RFC2616. (1999).

[17] TCC FIPA. 2008. Fipa communicative act library specification. *Foundation for Intelligent Physical Agents, http://www. fipa. org/specs/fipa00037/SC00037J. html (30.6. 2004)* (2008).

[18] Robert France and Bernhard Rumpe. 2007. Model-driven development of complex software: A research roadmap. In *2007 Future of Software Engineering*. IEEE Computer Society, 37–54.

[19] Carolyn Caffrey Gardner and Gabriel J Gardner. 2017. Fast and furious (at publishers): the motivations behind crowdsourced research sharing. *College & Research Libraries* 78, 2 (2017).

[20] Tracy Gardner. 2003. UML modelling of automated business processes with a mapping to BPEL4WS. *Orientation and Web Services* 30 (2003).

[21] Trisha Greenhalgh, Joseph Wherton, Chrysanthi Papoutsi, Jennifer Lynch, Gemma Hughes, Christine A'Court, Susan Hinder, Nick Fahy, Rob Procter, and Sara Shaw. 2017. Beyond Adoption: A New Framework for Theorizing and Evaluating Nonadoption, Abandonment, and Challenges to the Scale-Up, Spread, and Sustainability of Health and Care Technologies. *Journal of Medical Internet Research* 19, 11 (01 Nov 2017), e367. https://doi.org/10.2196/jmir.8775

[22] Mahdi Hannoun, Olivier Boissier, Jaime S Sichman, and Claudette Sayettat. 2000. MOISE: An organizational model for multi-agent systems. *Advances in Artificial Intelligence* (2000), 156–165.

[23] Kohei Honda. 1993. Types for dyadic interaction. In *CONCUR'93: 4th International Conference on Concurrency Theory Hildesheim, Germany, August 23–26, 1993 Proceedings*, Eike Best (Ed.). Springer, 509–523.

[24] Thomas C King, Tingting Li, Marina De Vos, Virginia Dignum, Catholijn M Jonker, Julian Padget, and M Birna Van Riemsdijk. 2015. A framework for institutions governing institutions. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. 473–481.

[25] Robert E Kraut, Paul Resnick, Sara Kiesler, Moira Burke, Yan Chen, Niki Kittur, Joseph Konstan, Yuqing Ren, and John Riedl. 2012. *Building successful online communities: Evidence-based social design.* MIT Press.

[26] Kimberly Ling, Gerard Beenen, Pamela Ludford, Xiaoqing Wang, Klarissa Chang, Xin Li, Dan Cosley, Dan Frankowski, Loren Terveen, Al Mamunur Rashid, Paul Resnick, and Robert Kraut. 2005. Using Social Psychology to Motivate Contributions to Online Communities. *Journal of Computer-Mediated Communication* 10, 4 (2005), 00–00. https://doi.org/10.1111/j.1083-6101.2005.tb00273.x

[27] C M Macal and M J North. 2010. Tutorial on agent-based modelling and simulation. *Journal of Simulation* 4, 3 (01 Sep 2010), 151–162.

[28] Milan Markovic, Peter Edwards, and David Corsar. 2013. *Utilising Provenance to Enhance Social Computation.* Springer, 440–447.

[29] R. Milner. 1999. *Communicating and mobile systems: the $\pi$-calculus.* Cambridge Univ Presss.

[30] Luc Moreau and Paul Groth. 2013. Provenance: an introduction to prov. *Synthesis Lectures on the Semantic Web: Theory and Technology* 3, 4 (2013), 1–129.

[31] Dave Murray-Rust, Petros Papapanagiotou, and Dave Robertson. 2015. Softening electronic institutions to support natural interaction. *Human Computation* 2, 2 (2015).

[32] Dave Murray-Rust, Ognjen Scekic, Petros Papapanagiotou, Hong linh Truong, David Robertson, and Schahram Dustdar. 2015. A Collaboration Model for Community-Based Software Development with Social Machines. *EAI Endorsed Transactions on Collaborative Computing* 1, 5 (12 2015).

[33] Sabine Niederer and José Van Dijck. 2010. Wisdom of the crowd or technicity of content? Wikipedia as a sociotechnical system. *New Media & Society* 12, 8 (2010), 1368–1387.

[34] Pablo Noriega and Dave de Jonge. 2016. Electronic Institutions: The EI/EIDE Framework. In *Social coordination frameworks for social technical systems*. Springer, 47–76.

[35] OASIS. 2007. Web Services Business Process Execution Language, version 2.0, OASIS Standard. (2007). http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf.

[36] Object Management Group. 2011. Business Process Model and Notation (BPMN), version 2.0. (2011). http://www.omg.org/spec/BPMN/2.0/PDF.

[37] Steven Ovadia. 2014. Automate the Internet With "If This Then That" (IFTTT). *Behavioral & Social Sciences Librarian* 33, 4 (2014), 208–211. https://doi.org/10.1080/01639269.2014.964593 arXiv:http://dx.doi.org/10.1080/01639269.2014.964593

[38] Petros Papapanagiotou and Jacques Fleuriot. 2017. *WorkflowFM: A Logic-based Formal Verification Framework for Process Specification and Composition.* Springer, Cham, 357–370. https://doi.org/10.1007/978-3-319-63046-5_22

[39] Juan Pavón, Jorge Gómez-Sanz, and Rubén Fuentes. 2006. Model Driven Development of Multi-Agent Systems. In *Model Driven Architecture – Foundations and Applications: Second European Conference, ECMDA-FA 2006, Bilbao, Spain, July 10-13, 2006. Proceedings*, Arend Rensink and Jos Warmer (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 284–298. https://doi.org/10.1007/11787044_22

[40] Jos'e Carlos Martins Rodrigues Pinho and Ana Maria Soares. 2011. Examining the technology acceptance model in the adoption of social networks. *Journal of Research in Interactive Marketing* 5, 2/3 (2011), 116–129.

[41] Shuping Ran. 2003. A Model for Web Services Discovery with QoS. *SIGecom Exch.* 4, 1 (March 2003), 1–10. https://doi.org/10.1145/844357.844360

[42] David Robertson. 2005. A Lightweight Coordination Calculus for Agent Systems. In *Proceedings of the Second International Conference on Declarative Agent Languages and Technologies (DALT'04)*. Springer-Verlag, Berlin, Heidelberg, 183–197.

[43] David Robertson and Fausto Giunchiglia. 2013. Programming the social computer. *Phil. Trans. R. Soc. A* 371, 1987 (2013), 20120379.

[44] Anne Rozinat, Stefan Zickler, Manuela Veloso, Wil MP van der Aalst, and Colin McMillen. 2009. Analyzing multi-agent activity logs using process mining techniques. *Distributed autonomous robotic systems* 8 (2009), 251–260.

[45] Nigel R. Shadbolt, Daniel A. Smith, Elena Simperl, Max Van Kleek, Yang Yang, and Wendy Hall. 2013. Towards a Classification Framework for Social Machines. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13 Companion)*. ACM, New York, NY, USA, 905–912.

[46] Paul R. Smart, Elena Simperl, and Nigel Shadbolt. 2014. A taxonomic framework for social machines. In *Social Collective Intelligence*. Springer, 51–85.

[47] Marcin Szpyrka, Grzegorz J. Nalepa, Antoni Ligęza, and Krzysztof Kluza. 2012. *Proposal of Formal Verification of Selected BPMN Models with Alvis Modeling Language.* Springer Berlin Heidelberg, Berlin, Heidelberg, 249–255.

[48] Warren Teitelman. 1966. PILOT: a step toward man-computer symbiosis. (1966).

[49] Wil M. P. van der Aalst. 2011. *Process Mining: Discovery, Conformance and Enhancement of Business Processes* (1st ed.). Springer.

[50] Peter Y. H. Wong and Jeremy Gibbons. 2008. *A Process Semantics for BPMN.* Springer Berlin Heidelberg, Berlin, Heidelberg, 355–374.

[51] Fang Wu, Dennis M Wilkinson, and Bernardo A Huberman. 2009. Feedback loops of attention in peer production. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, Vol. 4. IEEE, 409–415.

[52] Kevin Zhu and John P Weyant. 2003. Strategic decisions of new technology adoption under asymmetric information: a game-theoretic model. *Decision sciences* 34, 4 (2003), 643–675.