Knowledge of machines: review and forward look

S M Veres

Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering published online 15 July 2011
DOI: 10.1177/0959651811408502

Knowledge of machines: review and forward look

S M Veres
School of Engineering Sciences, University of Southampton, Highfield SO17 1BJ, UK. email: s.m.veres@soton.ac.uk

The manuscript was received on 19 December 2010 and was accepted after revision for publication on 6 April 2011.

DOI: 10.1177/0959651811408502

Abstract: This paper looks at the possibility of knowledge development of machines from an engineer's point of view. First, the increasing need for autonomous operations of vehicles, manufacturing facilities, utility networks, and robots is reviewed. From the point of view of operational design, five desirable ingredients are identified that can facilitate control autonomy. Most of these facilitators can be enabled by using the methodology of natural language programming and writing of documents that machines can read and utilize to improve their feedback control skills, their knowledge of the environment, and also their decision-making skills. A forward look outlines the benefits of 'publishing for machines', to manufacturing, vehicle operations, utility networks, and robots.

Keywords: control engineering, complex systems, autonomous systems, intelligent machines

1 TOWARDS AUTONOMY OF MACHINE OPERATIONS

Modern automated systems utilize finely tuned feedback loops to control the actions of cars, aircraft, ships, manufacturing machines, and also chemical, agricultural, and food production processes. They represent an important functionality that is supervised by humans making command decisions. The professional users of these automated machines have an understanding of the capabilities of these machines. To paraphrase: human supervisors use their higher level of intelligence to make decisions and command these systems.

There is, however, demand for systems that can operate and coordinate numerous feedback loops at various levels. Today, supervisory control (SCADA) systems are used to increase productivity in manufacturing systems and to control utility networks such as power, water/waste, gas, and communications networks [**1**]. There is also demand for systems that need to operate for extended periods of time without the possibility of high-level human supervision. Such systems include survey vehicles and robots used in deep sea applications where the useable bandwidth is limited and communication is often lost. Unmanned aerial vehicles are semi-autonomous in nature and often need the capability of autonomous operation due to loss of communication. Unmanned spacecraft, especially at large distances from the Earth, need an autonomous control capability [**2**–**5**].

Hence, it is necessary to clarify what is meant by an autonomous control capability. The highest level of definition is in fact fairly clear and difficult to argue with [**6**] 'autonomy is the capability of a system to pursue and fulfil its operational objectives without human intervention'.

There can, however, be other reasons for autonomy apart from lack or impossibility of human feedback control for extended periods of time. In the manufacturing industries and utility networks autonomous systems are also highly desirable for two reasons.

1. To increase efficiency so that lower levels of system supervision are needed to guide system operations.
2. To achieve system operational qualities way beyond what could be achieved by human control. Speed of computation and optimization over networks can contribute more financial

value than could any number of human feedback controllers.

There are examples of agent-based control systems in power system optimization that do modelling and perform decision making much faster and more precisely than can humans [**7**]. Agents can also operate over communication networks to optimize traffic speed. In healthcare there is a need for robots that can help patients for extended periods of time thereby reducing nursing requirements. Agents can also provide high-level decision making for low-level feedback control loops [**8**, **9**].

2 ENABLERS OF AUTONOMOUS CAPABILITY

The definition of autonomous capability in the previous section has the advantage of being easily understood by most members of the public. Its disadvantage is, however, that it does not shed much light on the problem of how to achieve autonomous capability. It also does not say what functions are desirable in an autonomous operational capability.

This paper will argue that:

(a) modelling of the changing environment;
(b) learning various skills in feedback interaction with the environment;
(c) symbolic recognition of events and actions to perform logic-based computation;
(d) ability to explain the reasoning behind its actions to humans;
(e) efficient transfer of rules, goals, values, and skills from human users to the autonomous system;

are desirable attributes of autonomously operating technical systems.

2.1 Modelling of the changing environment

The need for this process is not obvious. For instance, an engine control system, ship steering feedback controller, and the carriage position control system in a lift do not require modelling of their environment. This is true until the engine, the ship controller, or the lift become part of an autonomous vehicle, unmanned surface ship, or an intelligent home, respectively. When a feedback controller becomes the 'skill' of a larger intelligent control system, then the logic-based decision-making mechanism of this larger controller needs a model of its environment, especially if the environment can change. A supervisory intelligent controller will decide when to start the engine, when to report lift performance problems to technical staff, or what reference route a ship should track in a harbour, based on a, possibly abstract, model and rules referring to objects in the model.

Not all autonomous systems have the ability to model their environment. For example, no system modeller is used in the swarm robotics and distributed control system approaches. Favourable emerging behaviours need, however, to be proven in all practically important distributed control systems. Modelling is then done *a priori* by the brain of the designer and computers. It is argued in this paper that the highest levels of responsible autonomy, as exemplified by humans, requires some environment-modelling capability at various abstraction/resolution levels for operational safety in time-varying and possibly infrastructure-free environments.

2.2 Learning various skills in feedback interaction with the environment

In essence this is performed to learn appropriate feedback controller parameters for a given environmental situation. It is not obvious why learning the control parameters is needed by the autonomous system supervisor. Currently, many practical applications are pre-tuned by the control system designer as they address a single robust or adaptive control problem. Many autonomous systems need to perform in an environment that may not only change but can also be largely unknown. Thus, pre-tuning and testing of controllers for operation in these complex environments is a daunting task for engineers. These are the reasons why agents need to supervise the learning of feedback controllers. The same agents also need to recognize which controllers to use in a plan to achieve a goal in a given situation [**8**–**12**].

2.3 Symbolic recognition of events and actions to perform logic-based computation

Again it is not obvious whether this is needed. A currently used alternative is to define a hybrid automaton (that can for instance be represented in Stateflow^{TM}) that switches between operational modes in response to environmental changes. The designer formally checks all possible events in the environment and sets up the guard conditions for state changes in the hybrid controller. This requires complete knowledge of what can happen in the environment. This approach is currently used in cars, aircraft, ships, and in manufacturing.

Application of formal verification methods can result in checking of billions of system–environment states [**13**, **14**]. It should not be forgotten that these hybrid automaton-based control systems are still human supervised. When these systems become part of an even more complex autonomous system, the problem arises of maintaining real-time computational efficiency. Naming of key events as they happen, actions, and application of logic, to compute what is allowed to be done and what steps can lead to achieving a goal, is more economic than using finite state machine definitions: simply less coding is needed [**3**, **4**, **15**].

2.4 Ability to explain reasoning behind actions

Currently, control systems rarely 'explain themselves'. One reason for this is that the controllers may not perform their reasoning actions using symbolic events and actions and thus they do not interpret the environment in the same manner as do humans. Another reason is convention: explanations from machines are not expected. There is no general understanding of the need for this process. This paper argues that the introduction of more advanced intelligent autonomous systems is hindered by lack of trust from the users. Trust in a machine by a user can be developed if the machine is not only reliable, but also explains itself when it is not obvious to the human supervisor what is going on.

Self-explanatory systems could, in principle, do a better job of executing some tasks than a human but still may not understand the broader picture of the environment. Human users need to focus their intelligence on understanding these systems rather than expecting them to be perfect. It is time to recognize that even a simple device such as a car's remote controller, becomes far more trustworthy in the eyes of a driver if it explains its reasons as to why it opened or closed a part of the car. It is expected that the driver will learn the habits of the controller rather than the controller attempting to understand the car's owner. This is more important with complex systems such as the operation of a power station. Intelligent agents can supervise feedback controllers and make decisions to stabilize the network [**7**]. Operators must have immediate insight into why a decision was made. Similarly, manufacturing lines and autonomous vehicles need to explain themselves to the user from time to time. Humans can then interact with the autonomous system at a symbolic reasoning level instead of being restricted to simply analysing data [**16**].

2.5 Efficient knowledge transfer to machines

Rules, goals, and values can be communicated to machines in symbolic, natural language formats and in fact the technology for this process is currently available [**16**]. What is also desirable is that physical skills and problem-solving skills are also passed onto the system. In the case of complex systems it is conjectured by the current author that a designer may not have the time or resources to provide the best algorithm, especially in such difficult areas as computer vision and three-dimensional audio perception. An alternative is to release and sell the hardware with a nominal software solution. During the lifetime of the product the producing company's engineers can publish skill documents [**11**], mostly in natural language, which the machine can read. The purchaser or user of the product can also read these updates. This approach develops a shared understanding of operations and also contributes to the development of trust between man and machine.

The last three of these five desirable technical features raises the question of how can they be achieved. Section 5 outlines a new possibility offered by natural language programming (NLP) in sEnglish (stands for 'system English', and pronounced as 's-english') system that enables shared understanding between machines and their users.

3 THE PAST AND CURRENT STATUS

The generic approach to representing human thought is to use conceptual structures [**17**–**19**]. The preface of the 2010 proceedings of a recent Conceptual Structures (CS) Workshop [**20**] provides an insight:

'... CS is to harmonise the uniquely human ways of apprehending the world, with the power of computational information management and reasoning. Thus CS harmonises the creativity of humans with the productivity of computers.'

In NLP each sentence corresponds to a conceptual graph. This is the most important feature that elevates NLP to be more than just a programming language. Through its use of a natural language, NLP structures the procedural flow of programming in terms of conceptual structures of the human mind. The seminal work of Sowa [**17**] stated that the perception processes of a human produce concepts of the physical world and its relationships. The human concepts form hierarchical classes with inheritance of attributes. These are today mainly termed as ontologies [**21**]. A

conceptual structure is essentially a set of concepts or concept samples connected by conceptual relations. There are many natural languages in the world but conceptual structures are language independent: they are formal descriptions of human thought.

Any sentence in a natural language that makes sense to humans can be represented by a conceptual structure that is language independent. NLP associates a sentence with a concept and a code to carry out an action or the code to recognize a physical phenomenon or relationship between world modelling objects.

The most fundamental difference between programming in for instance C, Java, Phyton, or MATLAB on the one hand, and NLP on the other, is that NLP has a top layer of disciplined procedural steps that are associated with conceptual structures and human thought. The lower layer of NLP is ordinary code representing what is often called the 'sub-conscious' part of the human mind. With this classification most procedural codes in artificial intelligence today represent solutions generated in the sub-conscious state of humans. They provide a solution using an algorithm that does not use human concepts to explain how this solution is obtained. This often builds a wall of distrust between the intelligent machine and humans. Confidence in such an approach is only built after extensive successful application of this approach. It often occurs that the solutions offered by an intelligent system are imperfect and users tend to be sensitive to misunderstanding. However, trust is increased when the human mind is aware of how the machine reasons, even if the extent of machine reasoning is limited. This limited level of understanding of the machine can then be taken into account.

Figure 1 summarizes the main relationships from perception and sensing to sharing of meaning in conceptual structures that are expressed in sentences. These sentences are documented in fully sEnglish-based or ordinary publications with executable English sections that can be distributed by traditional or electronic means.

NLP was introduced in [**16**] and its theoretical basis was laid down in [**22**]. The fundamental role of NLP is to explain some of the artificial intelligence procedures of a machine in terms of natural language sentences. In NLP this is not achieved by comments inserted into a computer program but by building the program using a hierarchy of sentences. Some sentences are defined by a sequence of other sentences, while there are basic sentences that are defined purely by traditional computer code. The discipline of writing code in NLP creates formally analysable software that also has the advantage of being easily understandable by colleagues and users.

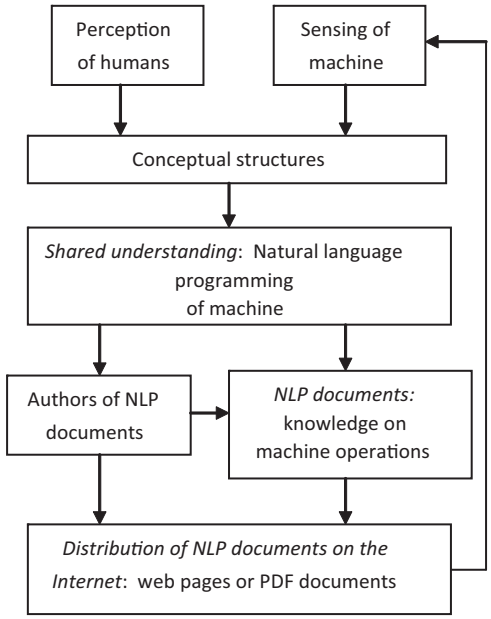

Fig. 1 Block diagram of knowledge generation, sharing between humans and machines and its distribution via publications

4 A FORWARD LOOK

4.1 Benefits of NLP

The use of NLP, and its main version sEnglish, requires a logical build-up of machine operations in terms of sentences. For an existing system the computer code can be reorganized to fit an sEnglish document. For a new project, a careful analysis of the procedures, rules, reasoning, and executable plans of a machine can be performed. These can be written down in terms of sentences that uniquely compile into computer code to control the machine.

4.1.1 Benefits to the manufacturer of the machine

The manufacturer sells a machine with solid operational logic and reliability but with an 'open control system', that can be modified by operators of the machine. This can make the product more attractive and versatile. Change can be done by modifying procedures defined by sentences. The manual of the machine can be changed by the user community to directly improve their behaviour. Operators can write documents that machines can read and utilize

to improve their feedback control skills, their knowledge of the environment, and also their decision-making skills. Using NLP, more complex and efficient decision making by machines can be created than by any currently used finite state machine-based method. NLP programmed agents have the inherent ability to explain their actions to humans.

4.1.2 Benefits to professional authors and programmers

Authors of signal processing and control procedures can use ontology structures generally shared by the professional community as accepted by standard definitions. Computational procedures can be written down by sentences and so can be used as operational logic [**2**, **3**, **22**–**27**]. Complexity for human understanding is always to be kept low by not defining a sentence using excessively long text to define the meaning of sEnglish sentences. The benefit to authors is that their algorithm is written down in a pseudo-code format that machines with an sEnglish interpreter can execute. The more such machines exist the more it becomes worthwhile to write 'executable papers' that both human colleagues and machines can read. Finally, teams of programmers can build complex systems that are formally verifiable. NLP facilitates knowledge sharing in a team of programmers. The transfer of new rules, goals, values, and skills from the human users to the autonomous system becomes possible using publications that can be read by machines.

4.1.3 Benefits to the automation and robotics industry

Market opportunities appear not only in terms of new machines with some shared understanding with users but also an enriched community activity can be cultivated. The hardware becomes a very basic form of a product; its capabilities can develop during their lifetime with initially unforeseeable utilities in later life of the product. This means enhanced value for money for users, due to the free machine knowledge shared, while advanced solutions can be charged for a fee.

5 DESIGNERS AND USERS TO PUBLISH METHODS FOR MACHINES

The five major components of a complex autonomous control system were discussed in section 2. An argument has been developed for symbolic knowledge sharing between the system and its operators.

This paper proposes the creation of product families around the concept of 'continued publication' for them during their industrial lifetime. It is reasonable to call this process 'publication' since the user community of the product can also read these documents in a controlled natural language. The question arises about the conditions required to create this publication system. This is the topic that is addressed in the rest of this paper.

1. The most fundamental condition for 'publication for machines' is compatible hardware for sensing and actuation within a product family. If they have the same hardware (humans all have the 'same hardware') then they have shared symbolic abstractions of sensory events ('feelings') and shared symbolic steps to perform actions ('physical skills'). The machine abstractions can be easily translated by the human mind and interpreted in their correct context.
2. A second condition is that the autonomous system family shares a rich set of basic sensory events and action abstractions from which complex skills can be tailored. This is feasible due to the shared hardware for sensing and actuation.
3. Finally, the third condition is a shared hierarchy of conceptual structures that root their meaning in the machine code for sensing and actuation by the machine. The concepts in this system are to be named using the best matching, concise, and precise descriptions using words of a natural language.

A system that satisfies these conditions also facilitates efficient programming which can reduce development costs via the use of abstract program models [**28**, **29**]. An example of such a publishing system is provided by NLP in sEnglish as documented in a series of publications [**2**–**4**, **22**–**27**]. An alternative solution is to publish journal papers that have executable NLP sections in sEnglish [**30**]. This approach represents a transitional solution towards machine readable publications [**26**, **27**].

6 THE CHALLENGE TO THE CONTROL ENGINEERING COMMUNITY

The NLP concept, sEnglish being the version of NLP for the English language, is not an arbitrary construct. NLP is not a robot programming language, in fact it is not a language at all; it is a method of programming. The choice of whether or not to use it is not a decision between different programming languages.

NLP is a method of building up the functionality of a machine or an autonomous system in terms of human concepts that are used in natural language sentences to express procedures and the logic of the system.

As long as the software enables this, the details of how it is done in the actual programming is irrelevant. The challenge ahead of researchers and design engineers, mainly in industry, and to some degree at universities, is to use such a system to create the benefits of:

(a) efficiency and time savings in system development of autonomous systems;
(b) developing trust in the product by its users;
(c) allowing improvements during the lifetime of a product via its shared understanding with the user.

The available sEnglish system [**24**, **26**, **27**] allows authors to publish self-contained conceptual structures and procedure sentences in a natural language document in English in HTML and LaTex(PDF) formats [**23**, **25**]. Authors can place created documents on the Internet so that they can be read both by the appropriate autonomous systems as well the users of these systems.

This is the birth of publishing for machines. Humans have long used publishing to aid dissemination of information and its comprehension. With the appearance of complex autonomous systems there is a need for shared understanding with these systems. This can be realized via machine understandable publications.

Lack of communication and shared understanding with a machine can make it appear to the user that the machine has a lower capability than it actually possesses. This can limit further development of a product and instead of improving its capabilities through publications so that the machine reaches its optimal intrinsic levels the product can end up being unnecessarily scrapped.

The recording of machine knowledge using sEnglish is 'formal' in that all sentences have a precise meaning in terms of signal processing, control, or logic inference. sEnglish does not have the freedom of pure natural language that heavily depends on the context of its use. Sentence meaning in sEnglish is precisely defined by other sentences. sEnglish is the simplest compromise between what feels like natural language and what is in fact computer programming. Ordinary programming is fully formal but does not carry concept-based meanings at levels of abstraction. sEnglish is a way of expression that has a vocabulary that can be specific for a machine; however, the human mind can easily adapt to it. sEnglish does, however, enable a machine, through suitable agent programming, to make sense of an arbitrary natural language communication for the specific small world with which the machine is designed to be involved.

The challenge is not how to invent such a publishing system but instead how to create products using the NLP paradigm for intelligent systems. This will allow them to use reasoning, explain themselves and thus gain the trust of their users, and read technical documents to allow them to further develop.

The knowledge transferable to machines by sEnglish documents should fundamentally focus on enhancing their sensing and control abilities that can make them safer. Enhanced skills can reduce or nearly eliminate the danger of an incorrect logical abstraction drawn from a situation and hence making the wrong decisions. Alternatives to machines reading technical documents are upgrades by a user, user community, or manufacturer. This paper advocates a system in which the intelligent system discusses its potential upgrades with its user and then performs the upgrade itself thereby saving the users time and effort. For the manufacturer to have the burden of upgrading is to follow the old ways: the modern way is that:

(a) the community of users likes to be involved;
(b) user community resources can be more cost-effective;
(c) the manufacturer can continue to be the main, although not exclusive, provider of knowledge upgrades to their product.

7 CONCLUSIONS

The future requirements for autonomous systems have been outlined. Five facilitators for practical autonomy have been analysed from a control engineering viewpoint. These have implied the crucial nature of developing shared understanding between machines and their users. One quick way to develop shared understanding is offered by NLP; this is not a programming language but rather a methodological approach. It is also not a language of communication; it is a means of building (programming) a shared conceptual base between a machine and its developers that is later available to its users. Various intelligent agent types, that supervise the control of machines, can be made to be able to read technical documents written in sEnglish which is the version of NLP for the English language. Machines can be developed and sold after which they can read technical documents from the Internet to improve their

performance. These documents can be published not only by the original equipment manufacturer but also by their user communities. Without something like NLP, and hence without substantial shared understanding with humans, machine intelligence remains limited when faced with a changing and complex physical world. For high infrastructure environments, where system models are well established and known, the need for machine publications is less relevant: current examples include automated search for flights and ticket booking, route finding on maps, and supervisory control of manufacturing and power grids, etc., but even these systems can benefit from NLP providing friendliness to system operators and automatic upgradability.

Finally, the adoption of the 'publications for machines' approach can bring great practical benefits by making the business of building autonomous systems viable in some critical areas, as they can help develop the trust of customer about what the machines do in certain situations.

FUNDING

This work has been partially supported by EPSRC Grants No. EP/E02677X/1 and No. EP/F037570/1.

REFERENCES

1 **Bailey, D.** and **Wright, E.** *Practical SCADA for industry*, 2003 (Newnes–Elsevier, Amsterdam).
2 **Lincoln, N., Veres, S. M., Dennis, L., Fisher, M.,** and **Lisitsa, A.** Agent reasoning beyond logic – abstractions of a 6DOF spacecraft. In Proceedings of the 16th International Symposium on *Formal methods*, Eindhoven, The Netherlands, 2–6 November 2009.
3 **Dennis, L., Fisher, M., Lisitsa, A., Lincoln, N. K.,** and **Veres, S. M.** Satellite control using rational agent programming. *IEEE Intell. Syst. Mag.*, 2010, **25**(3), 1541–1672.
4 **Veres, S. M., Molnar, L., Lincoln, N. K.,** and **Morice, C.** Autonomous vehicle control systems - a review of decision making. *Proc. IMechE, Part I: J. Systems and Control Engineering*, 2011, **225**(3), 155–195.
5 **Lincoln, N. K.** and **Veres, S. M.** Components of a vision assisted constrained autonomous satellite formation flying control system. *Int. J. Adapt. Control Signal Process.*, 2007, **21**(2–3), 237–264.
6 **Meystel, A. M.** and **Albus, J. S.** Intelligent systems: architecture, design, and control, 2002 (Wiley, New York).
7 **Wiebe, M.** *A guide to utility automation*, 1999 (PennWell, Tulsa, Oklahoma, USA).
8 **Wooldridge, M.** and **Jennings, N. R.** Intelligent agents: theory and practice. *Knowl. Engng Rev.*, 1995, **10**(2), 115–152.
9 **Jennings, N. R.** and **Bussmann, S.** Agent-based control systems. *IEEE Control Syst. Mag.*, 2003, **23**(3) 61–74.
10 **Wooldridge, M.** *Reasoning about rational agents*, 2000 (MIT Press, Cambridge, Massachusetts).
11 **Veres, S. M.** and **Luo, J.** A class of BDI agent architectures for autonomous control. In Proceedings of the 43rd IEEE Conference on *Decision and control*, Bahamas, 14–17 December 2004, vol. 5, pp. 4746–4751. (IEEE Press, Piscataway, New Jersey).
12 **Veres, S. M.** and **Veres, A. G.** Learning and adaptation of skills in autonomous physical agents. The 17th International Federation of Automatic Control World Congress, Seoul, Korea, 2008.
13 **Ezekiel, J.** and **Lomuscio, A.** Combining fault injection and model checking to verify fault tolerance in multi-agent systems. The International Conference on Autonomous Agents and Multi-agent systems, Budapest, Hungary, 23–26 August 2009.
14 **Molnar, L.** and **Veres, S. M.** Verification of AUVs using formal logic. In Proceedings of the *European control conference*, Budapest, Hungary, 23–26 August 2009.
15 **Yaouanc, J.-M., Saux, É.,** and **Claramunt, C.** A semantic and language-based representation of an environmental scene. *GeoInformatica*, 2010, **14**(3), 333–352.
16 **Veres, S. M.** *Natural language programming of agents and robotic devices*, 2008 (SysBrain Ltd, London, UK).
17 **Sowa, J.** *Conceptual structures: information processing in mind and machine*, 1984 (Addison-Wesley).
18 **Peirce, C. S.** On the algebra of logic. *Am. J. Math.*, 1880, **3**, 15–57. (Reading, Massachusetts, USA).
19 **Sowa, J.** Existential graphs MS 514 by Charles Sanders Peirce, available from http://www.jfsowa.com/peirce/ms514.htm (accessed 16 June 2011).
20 **Polovina, S., Andrews, S., Hill, R., Scharfe, H.,** and **Øhrstrøm, P.** (Eds) Preface. In Proceedings of *The first conceptual structures – learning, teaching and assessment workshop, CS-LTA 2010*, Kuching, Sarawak, Malaysia, 26 July 2010.
21 The W3C. Vocabularies – ontologies (OWL web ontology language guide, available fom http://www.w3.org/standards/semanticweb/ontology (accessed 16 June 2011).
22 **Veres, S. M.** Theoretical foundations of natural language programming. In Proceedings of *TAROS 2010: Towards autonomous robotic systems*, Plymouth, UK, 31 August–2 September 2010 (University of Plymouth, UK). Available from: http://www.tech.plym.ac.uk/soc/staff/guidbugm/taros 2010/.
23 **Lincoln, N. K.** and **Veres, S. M.** Sliding mode control for agents and humans. In Proceedings of *TAROS 2008: Towards autonomous robotic systems*, Edinburgh, UK, 1–3 September 2008 (University of Edinburgh, UK).
24 **Veres, S. M., Molnar, L.,** and **Lincoln, N. K.** SDT 1.1 – Sysbrain development tools in sEnglish/Jason (AgentSpeak)/MATLAB, 2010, available from www.cognitive-agent-toolbox.com (accessed 16 June 2011).

25 Veres, S. M. and **Molnar, L.** Documents for intelligent agents in English. In Proceedings of the 2010 IASTED Conference on *AI and applications*, Innsbruck, Austria, 15–17 February 2010 (IASTED, Innsbruck, Austria).

26 Veres, S. M. Authoring tools for sEnglish documents, 2008, available from www.system-english.com (accessed 16 June 2011).

27 Veres, S. M. Reader agent of sEnglish documents, 2008, available from www.system-english.com (accessed 16 June 2011).

28 Ye, Y. Supporting software development as knowledge-intensive and collaborative activity. In Proceedings of the International Workshop on *Interdisciplinary software engineering research*, Shanghai, China, 20–28 May 2006 (ACM, New York, USA).

29 Wang, Y. A hierarchical abstraction model for software engineering. In Proceedings of the Second International workshop on *Role of abstraction in software engineering*, Leipzig, Germany, 10–18 May 2008, pp. 43–48 (ACM, New York, USA).

30 Veres, S. M. and **Adolfson, J. P.** A natural language programming approach to executable papers. *Proc. Comput. Sci.*, 2011, **4** 678–687.