# Lightweight Coordination Calculus for Agent Systems: Retrospective and Prospective

David Robertson

Informatics, University of Edinburgh, UK

**Abstract.** The Lightweight Coordination Calculus was presented in a paper to DALT 2004 as a method for specifying a class of social norms for multi-agent systems. This was intended for use in the engineering of a range of applications but at the time the original paper was written this was an aspiration and we had little experience of actual use of the method. In this paper I summarise how experience with this approach has developed in the seven years from 2004 to date.

## 1 Introduction: Original Aims of the Lightweight Coordination Calculus

The Lightweight Coordination Calculus (LCC) was first presented at DALT [1] and at ICLP [2] in 2004. The aim of these papers, and of the DALT paper particularly, was to provide a means by which declarative programming might apply directly to the problem of coordinating agents that had not previously worked together. This sort of problem had been tackled previously but the primary means of attack had been either to standardise agent ontologies or standardise on performatives for agent illocutions. The former is difficult to scale to large and open systems because of problems in making sure that agents actually use language in comparable ways (and being able to check that they have). The latter is difficult to scale to complex social interactions in which a standard set of performatives leaves too much to the interpretation locally of agents. It appears to be very difficult in practice to build agents independently but with enough innate commonality to reliably perform complex social interactions. Work on institutions in the multi-agent community was, in 2004, already providing a partial answer to this problem by providing systems for specifying the desired interactions, separately from the agents involved. The practical aim of LCC was to turn this into a programming problem by viewing interactions between agents as executable specifications that could be communicated between agents that wished to coordinate. The theoretical aim of LCC was to form a bridge to multi-agent institutions from more abstract work on languages for communicating processes, then use this to bring techniques from formal reasoning into electronic institutions. A more detailed overview of the broader aims of LCC appears in [3].

## 2 Relating LCC to Other Languages: Translators and Meta Interpreters

One of the most frequent questions asked about LCC was how it related to other languages. This question was hard to answer definitively because a wide variety of

C. Sakama et al. (Eds.): DALT 2011, LNAI 7169, pp. 84–89, 2012.

languages are in use for coordination between systems, inside and outside of the multi-agent community. Several translators were written from such languages to LCC. Li built translators to LCC from service orchestration languages such as the Business Process Execution Language for Web Services BPEL4WS [4] (an XML-based language for describing workflow amongst Web services). Sierra's group at IIIA, Barcelona, added a translation facility to the ISLANDER electronic institution specification system [5]. A translator is the most obvious bridge between languages but languages like these are quite complex so the translators themselves are non-trivial to build. An alternative solution is to use LCC to specify an interpreter for another language, similarly to the way in which one traditionally uses a declarative language to define interpreters for other languages. Although it is unusual to think of a protocol language, like LCC, as an interpreter this works well in practice because the structure of LCC is suited to the task, as demonstrated by Li [6,7]. By bringing LCC into contact with other systems it became possible to explore more extensively how it could apply more broadly, particularly for Web service choreography [8].

## 3 Protocol Brittleness: Ontologies, Constraints and Adaptation

A strength of LCC is its ability to be used as, effectively, a programming language that coordinates agents. This is also a weakness, however, because the autonomy expected of agents often demands flexibility in interaction. If LCC protocols are too brittle then the interaction simply fails. One way to tackle this is, of course, to write LCC specifications that are more sophisticated but that creates work for LCC ”programmers” so various routes for adding various forms of more generic flexibility have been explored. A principal cause of brittleness is ontology mismatch - agents that could cooperate but fail because each describes its world in different ways. Mechanisms were invented for assisting in mapping local agent ontologies to terms in LCC specifications [9] and for using statistical information on the correlations between LCC terms to infer ontology mappings [10,11]. A second cause of brittleness in LCC was the inability to commit precisely to a constraint without committing to specific values for variables - this gave brittleness to interactions that required progressive refinement of the constraint space. Mechanisms were invented to add finite domain constraints to LCC-based systems, thus allowing one form of constraint representation and providing for constraint relaxation [12]. A third cause of brittleness was that agents originally had no control over the structure of LCC specifications - they could only choose whether or not to participate in a particular LCC-supported interaction, with no option to adapt the rules of interaction as they participated. This is a particular issue in argumentation systems, where the course of future interaction may be influenced strongly by the structure of arguments used previously within the same interaction. Mechanisms based on protocol synthesis were invented to produce these forms of interaction in a class of argumentation systems [13]. Later, LCC was used as a prototypical low level language for the protocol level of the Argument Interchange Format [14] which is a general purpose framework for describing argumentation systems.

## 4 Community Formation: Discovery, Group Formation and Trust

LCC is, deliberately, neutral to the manner in which it is used to coordinate agents. Nevertheless, in practice agent coordination that is sustainable over time has to occur in environments that support the interactions between agents and this requires mechanisms for helping agents to discover agents that are likely to be compatible; to form appropriate groups to achieve tasks and to establish trust. A range of methods have been developed to address aspects of this problem. At one end of the range, there are statistical methods for recommending compatible groups of agents based on previous successful/unsuccessful interactions [15]. These sorts of methods give crude measures of compatibility but have the advantage of requiring only simple statistical data on the history of interactions and no adaptation of the agents themselves. At the other end of the range are methods that check deontic specifications of agents (their permissions, obligations, etc) in real time against the LCC interaction specifications in which they are involved [16]. These allow more subtle control at the interface between agents and their interactions but at the cost of additional representation of deontic specifications for agents and of the inference machinery needed to perform the checking.

## 5 Application Areas

Although originally developed with multi-agent systems in mind, LCC has been used in a wide variety of contexts. In proteomics it has been used to share data on protein structure between protein data bases [17] and, in subsequent research, between research labs in Spain's ProteoRed network. In astrophysics, LCC has been used as a high level, executable specification language for data intensive experiments [18]. In crisis management, LCC has been used in simulation experiments comparing methods of centralised and peer to peer response to emergency flooding situations in the Trentino region of Italy [19]. In healthcare, LCC provided the basis for peer to peer sharing of healthcare workflows based on the ProForma system of medical protocol specification - this formed the basis for the Safe and Sound initiative (`www.clinicalfutures.org`) [20,21]. In computer games, LCC has been used as a language for specifying coordination between game agents in Unreal Tournament [22]. In service environments, LCC has been used in the development of market systems for confederations of services [23]. The common theme across all of these applications is the need for a compact language for specifying desired interactions plus a relatively straightforward way to make these easy to share and connect to local systems (whether these are autonomous agents or more traditional services).

## 6 Work in Progress

In the seven years since the DALT 2004 paper the world has changed considerably. Personal devices have become more sophisticated, more capable of data intensive processing and are much more ubiquitous. Social use of computation is also much more extensive and intimate than ever before. This has created many more potential applications for agents and in particular the coordination of agents. Potentially, these could operate across very large sectors of the population to harness individual sensing and problem solving for problems that hitherto resisted attack. We have already seen examples of this in the numerous social computing and crowdsourcing applications familiar to many. We are also experiencing the social effects of commercial interest in this area (through Facebook and other major companies) and the resulting conflicts over anonymity, privacy and ownership of information. Most of these issues are at least one step removed from declarative agent languages but they do increase the need for such languages and the need for scale and (perhaps) openness of operation places additional demands on specifications for agent interaction. Given this, future developments of LCC focus on community formation (driven from interaction data); security (in the context of electronic institution sharing in open systems); and the ability to synthesise/adapt specifications locally without breaking the coherence of interactions. These issues are not new but we lack methods that apply at current global scale. Further discussion of these issues will appear in [24].

**Acknowledgements.** This work was initially supported under the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (`www.aktors.org`), which is sponsored by the UK Engineering and Physical Sciences Research Council (EPSRC). Development of the OpenKnowledge system (`www.openk.org`) was supported by the European Union Framework 7 programme. Continuing work is supported by an EPSRC e-Science Platform grant. I am grateful for the discussions and inspiration provided by many other researchers on the AKT, OpenKnowledge and e-Science initiatives, as well as those on the other related efforts described in this paper.

## References

1. Robertson, D.: A Lightweight Coordination Calculus for Agent Systems. In: Leite, J., Omicini, A., Torroni, P., Yolum, p. (eds.) DALT 2004. LNCS (LNAI), vol. 3476, pp. 183–197. Springer, Heidelberg (2005)
2. Robertson, D.: Multi-agent Coordination as Distributed Logic Programming. In: Demoen, B., Lifschitz, V. (eds.) ICLP 2004. LNCS, vol. 3132, pp. 416–430. Springer, Heidelberg (2004)
3. Robertson, D., Walton, C., Barker, A., Besana, P., Chen-Burger, Y., Hassan, F., Lambert, D., Li, G., McGinnis, J., Osman, N., Bundy, A., McNeill, F., van Harmelen, F., Sierra, C., Giunchiglia, F.: Models of interaction as a grounding for peer to peer knowledge sharing. In: Chang, E., Dillon, T., Meersman, R., Sycara, K. (eds.) Advances in Web Semantics I. LNCS, vol. 4891. Springer, Heidelberg (2008)
4. Andrews, A., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S.: Business process execution language for web services, version 1.1 (2003), `http://www-106.ibm.com/developerworks/webservices/library/ws-bpel`

5. Esteva, M., de la Cruz, D., Sierra, C.: Islander: an electronic institutions editor. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 1045–1052 (2002)
6. Li, G., Robertson, D., Chen-Burger, J.: Using a multi-agent platform for pure decentralised business workflows. Journal of Web Intelligence and Agent Systems 6 (2008)
7. Li, G., Robertson, D., Chen-Burger, J.: A novel approach for enacting distributed business workflow on a peer-to-peer platform. In: IEEE Conference on e-Business Engineering, Beijing, China (2005)
8. Barker, A., Walton, C., Robertson, D.: Choreographing web services. IEEE Transactions on Services Computing 2 (2009)
9. Giunchiglia, F., McNeill, F., Yatskevich, M., Pane, J., Besana, P., Shvaiko, P.: Approximate, structure-preserving semantic matching. In: 7th International Conference on Ontologies, Databases and Applications of Semantics (2008)
10. Besana, P., Robertson, D.: How Service Choreography Statistics Reduce the Ontology Mapping Problem. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 44–57. Springer, Heidelberg (2007)
11. Schorlemmer, M., Kalfoglou, Y., Atencia, M.: A formal foundation for ontology-alignment interaction models. International Journal on Semantic Web and Information Systems 3 (2007)
12. Hassan, F., Robertson, D.: A constraint relaxation approach for over-constrained agent interaction. In: 11th Pacific Rim International Workshop on Multi-Agent Systems, Kuala Lumpur, Malasia (2008)
13. McGinnis, J., Robertson, D.: Realizing Agent Dialogues with Distributed Protocols. In: van Eijk, R.M., Huget, M.-P., Dignum, F.P.M. (eds.) AC 2004. LNCS (LNAI), vol. 3396, pp. 106–119. Springer, Heidelberg (2005)
14. Chesnevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S.: Towards an argument interchange format. The Knowledge Engineering Review 21 (2006)
15. Lambert, D., Robertson, D.: Matchmaking Multi-Party Interactions Using Historical Performance Data. In: 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, The Netherlands (2005)
16. Osman, N., Robertson, D.: Dynamic Verification of Trust in Distributed Open Systems. In: 20th International Joint Conference on Artificial Intelligence, Hyderabad, India (2007)
17. Quan, X., Walton, C., Gerloff, D., Sharman, J., Robertson, D.: Peer-to-peer experimentation in protein structure prediction: an architecture, experiment and initial results. In: International Workshop on Distributed, High-Performance and Grid Computing in Computational Biology, Eilat, Israel (2007)
18. Barker, A., Mann, R.: Integration of multiagent systems to AstroGrid. In: Proceedings of Astronomical Data Analysis Software and Systems XV. European Space Astronomy Centre, Spain (2005)
19. Marchese, M., Vaccari, L., Trecarichi, G., Osman, N., McNeill, F.: Interaction models to support peer coordination in crisis management. In: 5th International Conference on Information Systems for Crisis Response and Management, Washington, DC (2008)
20. Besana, P., Patkar, D., Barker, A., Robertson, D., Glasspool, D.: Sharing choreographies in openknowledge: A novel approach to interoperability. Journal of Software 4, 833–842 (2009)
21. Fox, J., Glasspool, D., Patkar, V., Austin, M., Black, E., South, M., Robertson, D., Vincent, C.: Delivering clinical decision support services: There is nothing as practical as a good theory. Journal of Biomedical Informatics 43, 831–843 (2010)

22. Graham, P.: Multi-agent coordination in complex virtual environments. PhD thesis, Informatics, University of Edinburgh (2011)
23. Guo, L., Darlington, J., Fuchs, B.: Towards an Open, Self-Adaptive and P2P Based e-Market Infrastructure. In: Proceedings of the IEEE International Conference on e-Business Engineering, Macao, China (2009)
24. Robertson, D., Giunchiglia, F.: Programming the social computer. Royal Society Philosophical Transactions A (special issue: Web Science: A New Frontier) (in press)