2008

When to use a multi agent system?

Paul Bogg
University of New South Wales, paulb@unsw.edu.au

Ghassan Beydoun
University of Wollongong, beydoun@uow.edu.au

Graham Low
University of New South Wales, g.low@unsw.edu.au

When to use a multi agent system?

Abstract

The decision of which development approach to adopt (e.g. traditional, object oriented, agent oriented) is often taken after identifying the specific features of the problem. If *agent oriented software engineering* (AOSE) is to be taken seriously as a software engineering paradigm, then a clearly identifiable set of criteria of when to apply it, as distinguished from other alternatives such as object-oriented practices, is necessary. The paper is part of an effort to determine when is best to adopt a Multi Agent System approach, identifying a number of critical factors to include in a decision framework.

Disciplines

Physical Sciences and Mathematics

Publication Details

Bogg, P., Beydoun, G. & Low, G. (2008). When to use a multi agent system?. International Conference on Principles of Practice in Multi-Agent Systems (pp. 98-108). Germany: Springer-Verlag.

When to Use a Multi-Agent System?

Paul Bogg[1], Ghassan Beydoun[2], and Graham Low[1]

[1] University of New South Wales
{paulb,g.low}@unsw.edu.au
[2] University of Wollongong
beydoun@uow.edu.au

Abstract. The decision of which development approach to adopt (e.g. traditional, object oriented, agent oriented) is often taken after identifying the specific features of the problem. If *agent oriented software engineering* (AOSE) is to be taken seriously as a software engineering paradigm, then a clearly identifiable set of criteria of when to apply it, as distinguished from other alternatives such as object-oriented practices, is necessary. The paper is part of an effort to determine when is best to adopt a Multi Agent System approach, identifying a number of critical factors to include in a decision framework.

1 Introduction

Agents are highly autonomous, situated and interactive software components. They autonomously sense their environment and respond accordingly. A Multi Agent System (MAS) is a collection of interacting agents which are highly autonomous, situated and interactive software components. They autonomously sense their environment and respond accordingly. Coordination and cooperation between agents that possess diverse knowledge and capabilities facilitate the achievement of global goals that cannot be otherwise achieved by a single agent working in isolation [1]. MASs have been shown to be highly appropriate for the engineering of open, distributed or heterogeneous systems [2-4]. While many AOSE methodologies exist with each focusing on a specific class of problems (e.g. MaSE [5], GAIA [6], PROMETHEUS [7] and TROPOS [8]), a clear and established set of criteria for determining if, and when, an agent approach should be used remains absent. This no doubt has contributed to the delay in the much anticipated adoption of MAS as a preferred development approach in many medium to large-scale projects.

This paper is a part of our ongoing research to remove barriers to the successful adoption of AOSE. We present a preliminary multi-staged approach to identifying a set of criteria to be included in a decision framework that will assist in determining if, and to what degree, a MAS is suitable to solving a particular problem.

2 Related Work

Deciding if a MAS approach should be used to solve a particular problem requires identifying its *suitability* and estimating its *relative cost* (versus alternative approaches)

set of features? To provide answers for these questions we construct a framework that can be applied to a problem instance to assess the suitability of a MAS solution and discuss ongoing work.

3 MAS Features for Determining Suitability

In this section, we lay the foundations to construct a framework to assess whether a MAS is an appropriate solution for a given problem. We first overview properties of agents and MAS. These properties we know that any MAS can be designed to have. After overviewing 25 existing MAS applications from the literature, we establish a comprehensive set of features that affect the decision of whether or not to adopt an agent-oriented approach and we relate these features to the known properties of MAS and agents.

Individual agents show varying levels of autonomy, social ability, reactivity and pro-activity and/or reasoning capabilities. *Autonomy* is the independent control that an agent has over tasks performed as well as its control over the acquisition of inputs and resources required for its tasks. *Reasoning capabilities* is the ability that the agent has to identify reasons for beliefs, conclusions or performing tasks. *Social ability* is the ability that an agent has to interact with other agents or other components of the environment. A MAS solution has *system-level* properties that result from the interactions between the individual agents in the system. *Self-organisation* as a key system-level property of MASs which is a process allowing the system to change its internal organisation to adapt to changes in its goals and the environment without explicit external control [11]. Various characteristics of self-organising systems are (adapted from [11]): *decentralised control, self-maintenance, adaptivity,* and *convergence. Decentralised control* arises in the absence of a central authority when the system operation is governed by the interactions between individual agents. *Self-maintenance* is where a system has the capacity to reproduce or repair itself. *Adaptivity* is the capability of the system to reorganise itself when the environment or goals change in order to continue operating. *Convergence* is the capability of the system to reach a stable state. For example, if the interactions between individual air conditioning unit agents keep the building temperature constant, then we might say the system has decentralised control, is adaptive to changes in the environment, and converges to a stable state.

To establish and validate a comprehensive set of features that affect the decision of whether or not to adopt an agent-oriented approach, we follow an iterative approach based on an initial analysis of specific instances of problems for which MASs have been implemented. These instances are distinct and are identified using a combination of literature and preliminary discussions with experts in MAS development. In all, 25 different MAS solutions from the literature were analysed (Table 1 provides a sampling). A set of features that characterise this spectrum of problems is indetified. In deriving this set, every effort was made to make it domain independent and applicable to new domains not previously considered as candidates for agent-based solutions. Features are identified into two categories: problem-related features and solution-related features (degree of distribution, efficiency, etc).

To elaborate and illustrate the resultant features, we use the following two examples which have been implemented using a MAS solution:

Software solutions often need to *interact* with other software, or components within their environment. Interactions might be as simple as an enquiry for information, or as complex as a negotiation. The properties of these interactions are themselves features of the problem (as adapted from [14]):

- Negotiation – process of coming to agreement on a set of conflicting issues; and
- Deliberation – interactions to establish cooperation on tasks.

For example, software automating an economic business process may require negotiation with external business partners to acquire goods and services.

In Example B, when sensors are destroyed, some *deliberation* between the remaining sensors is required in order to compensate.

3.2 Solution-Related Features

Software solutions may need to perform tasks in or acquire input from geographically separate locations. This distributedness of the solution has the following facets which are also features of the solution:

- Distributed Data – input from the environment is from geographically separate locations;
- Distributed Resources – resources required to perform a task are in geographically separate locations; and
- Distributed Tasks – tasks are performed in geographically separate locations.

In Example A, if the environment is a computer network, then the input is provided by sensors around the network – so there is distributed data. Since the tasks are required to be performed remotely there is a need for *distributed tasks*. (The feature of task distribution might be dependent on the framing of the problem. A similar IDS problem might not require distributed tasks.)

In Example B, *distributed data* is acquired by the battlefield simulation system. While the sensors used to acquire the data might be distributed, a significant proportion of tasks would be performed centrally.

In addition to the above, we identify a set of non-functional requirements that are themselves features of a MAS solution:

- Efficiency – in performing tasks, acquiring inputs, and resource expenditure;
- Reliability – in performing tasks (consistency in doing what it's supposed to do);
- Robustness – in performing tasks (continues to perform in the face of adversity);
- Flexibility – in performing different/new tasks;
- Responsiveness – in performing tasks in response to inputs at runtime;
- Indeterminism at design time – in identifying which tasks to perform; and/or
- Concurrency – in performing tasks (at the same time).

In Example A, the IDS requires that the system be *reliable* in continuing to detect known and new forms of attack. *Robustness* is required in situations where network components fail. *Flexibility* is required in detecting new forms of attack. *Responsiveness* is imperative in order to promptly alert the system administrators of possible intrusions. *Concurrency* is required when many parts of the network are monitored

- Interactions – Negotiation – Agents may be *pro-active* or *reactive* in the process of negotiation in order to achieve their goals. As a *social* entity an agent may negotiate with one or more agents. As an *autonomous* entity, an agent may have control over what information is relevant to the negotiation. Where negotiation is needed to regulate the amount of resources amongst entities, MASs may have *convergent* behaviour that regulates how these resources are distributed. An example of this is in the stabilising of a global market price amongst competing agents in a market place [15].
- Interactions – Deliberation – Agents may be *pro-active* or *reactive* in cooperating with other agents in order to achieve their goals. As a *social* entity, an agent may communicate with other agents. As an *autonomous* entity an agent may have control over if, when and how to cooperate and what information is relevant to the cooperation. When cooperation is necessary to achieve a stable state, MASs may have *convergent* behaviour that regulates the continuity of the state. For example, air conditioning units may be required to cooperate in order to regulate global temperature, and a MAS solution provides the regulation of the global temperature by the local agent interactions.
- Efficiency – As *autonomous* entities, agents may provide efficiency by having control of its input, resources and tasks that is independent from a controller. For example, an autonomous Mars rover reduces the communication control overhead from Earth. Agents may also *reason* about how to perform tasks more efficiently. MASs that have *decentralised control* and *adaptivity* provide efficiency by reducing the overhead that would normally be required of a centralised control to adapt and regulate variables in response to changes in the environment. For example, an IDS is efficient in the sense that new attacks are detected by local agents in order to maintain global security.
- Reliability – As *autonomous, reasoning* entities, agents may provide reliability by reasoning about environment changes that may affect task performance. MASs that have *adaptive* behaviour provide reliability in regulating system-level behaviours when the environment changes.
- Robustness – As *autonomous* entities, agents may provide robustness by having control over self-maintaining tasks in the event of a problem (the Mars rover might be capable of self-healing [9]). MASs that have *decentralised control, adaptivity* and *self-maintenance* provide robustness in task operation by adapting to parts (or agents) of the system that malfunction. For example, a battlefield simulation system with decentralised control is robust in operation because it adapts to the loss of sensors.
- Flexibility – As *autonomous, reasoning* entities, agents may provide flexibility by reasoning about new and different tasks to be performed, and having control over when to perform them. MASs may have *adaptivity* that adjusts the system operation to new problems or tasks. For example, the MAS for an IDS provides flexibility in detecting new forms of attack.
- Responsiveness – An agent is *reactive* to environment changes that affect task performance. As an *autonomous* entity, an agent has independent control over how it reacts to environment changes, which improves its responsiveness compared with a centralised controlled approach. MASs may have *decentralised control* that improves the responsiveness of the system due to less overhead in requiring checks with a centralised control.

Table 1. Features for alternative problem domains

Feature	Intrusion Detection	Battlefield Simulation	Patient Care	Search & Rescue	Document Recommend	MASFIT
Environ – Dynamic	5	5	4	5	3	4
Environ – Uncertain	4	4	2	4	3	3
Environ – Open	4	4	4	2	2	3
Distributed – Data	5	5	5	5	4	4
Distributed – Resource	2	4	3	4	5	4
Distributed – Tasks	4	2	3	5	4	2
Interact – Negotiation	1	1	1	1	3	5
Interact – Deliberation	1	1	4	1	4	3
Efficiency	3	4	5	4	3	3
Reliability	4	4	4	4	3	4
Robustness	4	5	4	4	5	2
Flexibility	2	2	4	2	5	2
Responsiveness	5	5	3	5	3	3
Indeterminism	4	2	1	4	2	5
Concurrency	4	3	5	2	4	4

(Ratings: 5 very high; 4 high; 3 moderate; 2 low; 1 very low)

Problem domains: Intrusion detection [12]; Battlefield information system [13]; Multi-agent patient care [17]; Human-Robot Teaming for Search and Rescue [18]; Document recommendation tools [19]; and Masfit, fish auction MAS [15]

3. Determine the extent that a MAS solution may address the problem and solution-related features identified in step 1.
4. Using the importance of each feature identified in step 1 and the extent that the MAS solution addresses this feature in step 3, determine the appropriateness adopting a MAS approach to solve the problem.

5 Conclusion and Future Work

Our aim was twofold: firstly to abstract key features from a variety of different problem domains. Features were cyclically pruned on the basis that they were significant determinants of whether or not a MAS was suitable. Secondly, to construct a framework which assesses the suitability of a MAS solution, given a set of problem and solution-related features.

Whilst our feature identification may be biased by the chosen examples and the way each problem was originally framed, resultant features actually generalise (and address) all criteria described by [1, 9, 10] with one exception: when extending legacy systems from [1]. We propose extending our set of features to identify any need for wrapper agents in situations such as extending legacy systems and/or interfacing with other non agent-based resources.

10. EUROSCOM: MESSAGE: Methodology for engioneering systems of software agents. Final guidelines for the identification of relevant problem areas where agent technology is appropriate. EUROSCOM Project Report P907 (2001)
11. Serugendo, G.D.M., Gleizes, M.-P., Karageorgos, A.: Self-Organisation in multi-agent systems. The Knowledge Engineering Review 20, 165–189 (2005)
12. Asaka, M., Okazawa, S., Taguki, A., Goto, S.: A Method of Trading Intruders by Use of Mobile Agents. In: 9th Annual Conference of the Internet Society (1999)
13. Maston, E., DeLoach, S.: An Organization-Based Adaptive Information System for Battlefield Situational Analysis. In: Integration of Knowledge Intensive Multi-Agent Systems (2003)
14. Walton, D.: The New Dialectic (1998)
15. Cuní, G., Esteva, M., Garcia, P., Puertas, E., Sierra, C., Solchaga, T.: MASFIT: Multi-Agent System for Fish Trading. In: Proceedings of the 16th European Conference on Artificial Intelligence (2004)
16. Beydoun, G., Gonzalez-Perez, C., Henderson-Sellers, B., Low, G.C.: Developing and Evaluating a Generic Metamodel for MAS Work Products. In: Garcia, A., Choren, R., Lucena, C., Giorgini, P., Holvoet, T., Romanovsky, A. (eds.) SELMAS 2005. LNCS, vol. 3914, pp. 126–142. Springer, Heidelberg (2006)
17. Reed, C., Boswell, B., Neville, R.: Multi-agent Patient Representation in Primary Care. In: Miksch, S., Hunter, J., Keravnou, E.T. (eds.) AIME 2005. LNCS, vol. 3581, pp. 375–384. Springer, Heidelberg (2005)
18. Nourbakhsh, I., Lewis, M., Sycara, K., Koes, M., Yong, M., Burion, S.: Human-Robot Teaming for Search and Rescue. IEEE Pervasive Computing (2005)
19. Pavon, J., Gomez-Sanz, J.: Agent Oriented Software Engineering with INGENIAS. In: Mařík, V., Müller, J.P., Pěchouček, M. (eds.) CEEMAS 2003. LNCS, vol. 2691, pp. 394–403. Springer, Heidelberg (2003)