Evaluating the Applicability of Multi-Agent Software for Implementing Distributed Industrial Data Management Approaches

Torben Jess[1], Philip Woodall[1], and Duncan McFarlane[1]

[1] University of Cambridge, Department of Engineering, UK

tj282@eng.cam.ac.uk

Abstract. Distributed approaches to industrial control or information management problems are often tackled using Multi-agent methods. Multi-Agent systems – solutions resulting from taking a Multi-agent based approaches - often come with a certain amount of "overhead" such as communication systems, but can provide a helpful tool with the design and implementation. In this paper, a distributed data management problem is addressed with both a bespoke approach developed specifically for this problem and a more general Multi-agent approach. The two approaches are compared using architecture and software metrics. The software metric results show similar results, although overall the bespoke approach was more appropriate for the particular application examined. The architectural analysis indicates that the main reason for this difference is the communication and computation overhead associated with the agent-based system. It was not within the scope of this study to compare the two approaches under multiple application scenarios.

Keywords: Multi-Agent Systems comparison; Multi-Agent Systems for Data Management; Architecture Trade-off Analysis method; Software metrics

1 Introduction

Over the last 10-20 years distributed approaches have been used for various industrial problems, such as holonic manufacturing. It uses separate distributed entities like machines, or products as representations in the algorithm for finding a good production plan. In contrast to centralized approaches, which are trying to optimize the whole production centrally [1]. Distributed approaches have been shown to be beneficial for various industrial cases such as manufacturing [2]. Often distributed approaches are implemented using Multi-Agent systems [1]. However, there have also been various cases where alternative approach such as bespoke object-oriented techniques have been used [3]. Multi-Agent systems have the benefit of having some existing techniques to work with such as negotiation mechanism for example. However, they also create a specific overhead costs in the implementation. With the in-

creasing amount of distributed approaches in various domains the problem of selecting a Multi-Agent system for this is approach is therefore becoming more difficult. The question is if it is worth developing a distributed solution using Multi-Agent systems or should alternative approaches be selected? This paper addresses this question for a data management case. A distributed algorithm has been developed to address the challenge of finding additional relevant data for a supply chain decision problem. We implemented a bespoke solution (specifically designed to implement just this technique using object oriented methods) and a Multi-Agent solution following this distributed algorithm approach. In order to compare both implemented systems, we used architecture evaluation with ATAM and software metrics based on the systems requirements to identify which approach is more suitable. We found that the bespoke solution was more suitable for our scenario due to the high overhead costs and the agent thread management of the Multi-Agent system. Section 2 presents the relevant research background. Section 3 describes the problem, the distributed algorithm and the two-implemented systems. In section 4 the two systems are compared and section 5 presents the conclusion for this paper.

2 Research background

This chapter looks into Multi-Agent systems as a software approach, alternatives for them, and architecture evaluation and software metrics as our comparison approach.

2.1 Multi-Agent software approaches

Multi-Agent systems are used in a various industrial applications [4], such as holonic manufacturing [2], [1] or supply chain management [5]. Different methodologies and techniques for Multi-Agent Systems have been developed and researched. This includes different Multi-Agent architectures (such as BDI for example), different methodologies such as GAIA [6], MaSE or PROMOETHEUS and many more [7], [8] for example. An overview about Multi-Agent systems can be found in Wooldridge [9].

2.2 Alternative software Implementation approaches

Distributed software approaches can be implemented using all kinds of software approaches. In practice and in research projects the main alternatives are standard object oriented techniques. There are various examples of object oriented holonic manufacturing systems for example [3], [10].

2.3 Approaches to Comparing Software Design & Implementation

This review focuses on the techniques applied in this paper, which are software architecture and software metrics. There exist two types of methods for the evaluation of software architecture, software architecture analysis methods such as ATAM or SAM [11] and performance prediction models such as queuing networks or petri nets [12].

This paper uses ATAM as the general accepted industry standard for software architecture analysis methods. ATAM is relying on a set of scenarios (based the system on requirements) defined by users and stakeholders. These scenarios are analyzed using simple prototypes or "back of the envelope" approximation, and combined in a utility tree [11]. ATAM is a qualitative method but showed good results in industrial examples [13], [14]. For software engineering metrics comparison software engineering has intensively studied the field of software system comparison. Various works have addressed different aspects of the evaluation for example bug detection [15]. Various papers provide a good overview about the large number of measures [16],[17]. There have been measures for different aspects of the software system like costs or programming time [18],[19]. This paper is mainly concerned about identifying suitable measures to technical evaluate an implemented software system.

3 An industrial data management problem, solution and implementation approaches

3.1 The industrial data management problem

Currently users in procurement have a lot of effort in finding the relevant data within different databases and also have the risk of potentially missing parts of this information. For example a supply chain users might have to order life vests. He does not know about all the other orders of life vests from his company (previous and current orders) and cannot use this information to get a discount. The goal of this system is to this information gap within a company by giving the user the additional data required. Searching for the data is not always an option since the user is not always aware of its existence.

3.2 A Distributed Solution for the Industrial Data Management Problem

The solution to this problem automatically identifies interesting data for the user. Instead of having a centralized approach going through the data to identify similarities our approach uses a distributed technique in which each data item can find related data items and connect with them. In order to do this each data item (in our case a row) goes through different databases and looks for other data items with similar syntax to itself. This way whenever a specific row is presented to the user the other joined rows can be presented as well.

3.3 Implementation of the Distributed Solution Approach

A bespoke and a Multi-Agent Systems approach were identified as possible architectures for the problem. Both are using the same underlying techniques and therefore also have the same overall results for the user. In the implementation of both systems we used similar coding styles to avoid any potential bias in the evaluation.

Bespoke implementation

The bespoke system has DataWrappers as its main concept. The DataWrapper represents a piece of information and are compared to other DataWrappers. When the user asks for additional information the DataWrapper x of the information the user currently sees (e.g. an order of life vests) compares itself to a list of candidate DataWrappers generated by a DataWrapperManager. These candidates are compared to the presented DataWrapper x representing the life vests. If the comparison value is above a certain threshold then a Join (in this case Join a between two orders of life vests) between the two DataWrappers (x and n in the figure representing two orders of life vests) is created. Once all Joins are generated the best connections are presented to the user (in this case Join a). A diagram of the architecture can be found in figure 1.

465529

Multi-Agent implementation

The Multi-Agent architecture is build around two types of agents GUIAgents and TableAgents. The GUIAgent is used to answer to the requests of the user and forward these requests to the TableAgents (see Figure 1). He interacts with the GUI presented to the user and tries to present additional information to the user. When the user requires additional information the tableAgent (e.g. order of life vests) receives the message from the GUIAgent requesting similar information to be send to the TableAgent. TableAgents represent a database table and its content. He has access to the data from the table. He compares its own information against the information currently presented to the user (using the same comparison criteria that the bespoke system is using) and replies with the information he thinks are relevant for the user.

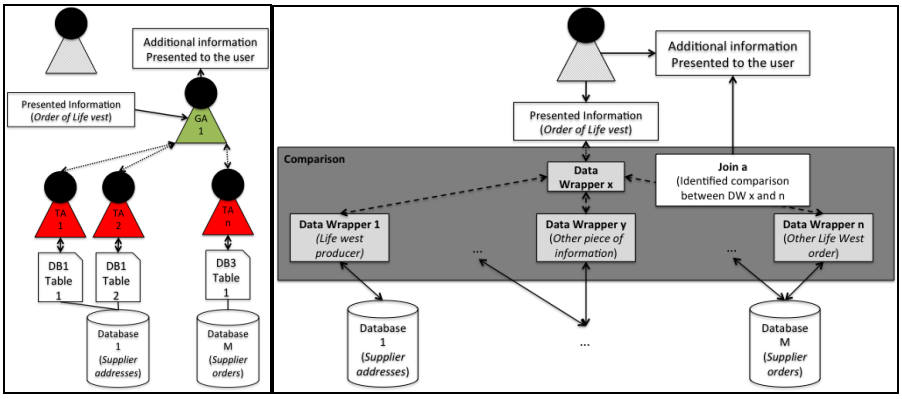

Fig. 1. Description of the Multi-Agent system (left) and the bespoke solution (right)

4 Evaluation of Software design and implementation

This section compares the two systems using ATAM and software metrics.

4.1 ATAM evaluation

The architecture evaluation followed the 9 ATAM [11] steps except for minor changes due to limited stakeholder availability and a small number of developers: In Step 1 and 2 (Present ATAM and Present business driver) the ATAM process and the relevant business drivers were introduced to all developer and stakeholder (see Kazman et al. [11] for details on ATAM and section 3.1 for the business drivers). In step 3 and 4 (Present architecture and Identify architectural approaches) was done using the two architectural approaches (see section 3.3). In step 5 (Generate quality attribute utility tree) a list of scenarios for the evaluation was generated by brainstorming [11]. Three main categories performance, availability, modifiability and implementation were identified. The scenarios were then ranked by importance (see table 1 for details). In step 6 (Analyze architectural approaches) the different scenarios are evaluated using ATAM techniques such as short prototyping or back of the envelope analysis [11]. Both architectures were ranked based on their results in this analysis from 1 (very easy to fulfill scenario requirements with this architecture) to 5 (very difficult to fulfill scenario requirement with this architecture) for each scenario (see results in table 1). Step 7 and 8 (Brainstorm and prioritize scenarios and analyze architectural approaches) are mainly a verification in testing step within ATAM [11]. We used the advantage of having a small group of developers and stakeholders and already incorporated stakeholder feedback in step 5. In step 9 (Present results) the results of the report where gathered and presented.

Table 1. ATAM evaluation (I=Importance [3=high, 2=Medium, 1=Low]), Ag.=Agent and Bes.=bespoke is the rank and the rank times importance (in parentheses) for each architecture

ID	Description	Approach	I	Ag.	Bes.
Performance					
P1	Time for the system to respond to user in demo of approx. 30secs for 2 databases	Sequence breakdown with simple prototypes for time estimate with main computation steps	3	3 (9)	4 (12)
P2	One additional relevant piece of information for every supply chain task found	Depends on syntactic matching algorithm which has potential to deliver additional information based on initial tests	3	2 (6)	2 (6)
P3	Work with 10 databases and in 20secs with same accuracy	Use analysis from P1 and adjust time for larger sizes	1	1 (1)	2 (2)
Modifiability					
M1	Experienced developer can include additional tables within 15min of effort	Analyze architecture to number of places were this change would occur	1	5 (5)	5 (5)
M2	Background programming can easily be implemented	(See M1)	1	5 (5)	3 (3)
M3	Semantic matching can be incorporate by just changes to one method	(See M1)	2	5 (10)	5 (10)

M4	Tables can be accessed dynamically	(See M1)	1	5 (5)	3 (3)
Availability					
A1	Risk of system to fail during a short demonstration is below 1%	Assume failure probabilities for major units and use as basis for whole system failure estimate	3	5 (15)	5 (15)
A2	Risk of database connection failure at the start is below 1%	(See above for failure probability of major units)	3	5 (15)	5 (15)
Implementation					
I1	The system can easily be implemented by a single developer within 2 weeks	Breakdown the different sequences and estimate effort involved in implementing them	3	3 (9)	5 (15)
Total				80	86

4.2 Software metrics comparison

Both systems were evaluated on the measure in table 2; selected based on, first their ability to address the criteria: performance, modifiability, availability and implementation, second their measurability with existing tools, and third based on how established they are within the software engineering/metrics community [16]–[19]. For performance we relied on requirement specific time measures and were looking at two values. The first was performance of existing functionality, looking at the switching between different orders as a measure for existing system time performance. As a second measure we used the time for each system to find additional data. The evaluation was done on a three databases. Database 1 is a parts system containing 2007 part details. Database 2 is an HR system containing 32924 employee entries. Database 3 is a procurement system containing our test cases with 4 orders and 3 items.

Table 2. Software metric comparison results for both architectures (Multi-Agent and bespoke)

	Bespoke	**Multi-Agent**
Performance		
Average time to find additional data and present it in a separate GUI	225,4ms (STDEV: 131.5)	484ms (STDEV: 195.2)
Switching between different cases	30.9ms (STDEV: 12.8)	41.6ms (STDEV: 6.2)
Modifiability and implementation		
Lines of code	1637	1896
Number of methods	161	156
Number of operations	492	639
Decision count	107	130
Number of classes	23	29
McCabe cyclomatic complexity	1.639 avg over all classes	1.808 avg over all classes
Lack of cohesion	0.481	0.378
Coupling factor	0.096838	0.0591133
Halsted program length [20]	5130	5946

Halsted program difficulty [20]	78	81.21
Halsted Time to program [20]	58 hours	72 hours
Halsted delivered Bugs [20]	16.11	19
Availability		
Number of failures during runtime	0	0

4.3 Analyses and Interpretation of Results

We used the criteria: Performance, modifiability, availability, and implementation as a structure for our evaluation. The bespoke system shows better performance having twice the speed of the Multi-Agent system for finding data. ATAM and the high standard deviation in response time indicate the reason is table agents being busy with their different behaviors, so they take more time to answer. In addition the workload is distributed differently among TableManagers. One TableManagerAgent requires more time to answer and check its content then others. However JADE allocates each agent similar time intervals, so some TableAgents block others with their regular behaviors. For modifiability and implementation the Multi-Agent architecture had slightly worse results due to higher measures in complexity and size of the code. However the difference in measures is only small, indicating similar complexity for modifiability and implementations. This is consistent with the ATAM analysis. Both architectures perform well for availability due to the low risks of underlying systems like databases failing. The architecture analysis indicates that a bigger likelihood of underlying system failure would show higher benefits of the agent system.

5 Conclusion

This paper analyzed whether an agent-based approach can help within the implementation of a specific distributed data management algorithm. We implemented two systems: an agent-based and a bespoke solution. They were compared using ATAM architecture evaluation and software metrics. Our results indicate that the bespoke solution is more suitable for the project and its requirements; mainly due to the agent approach being slower and having a higher implementation/maintenance effort. The reasons are the delay based on the thread management for all agents and the problem of misallocation of processing time by distributing time equally to all agents independent of workload within JADE.

References

[1] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Eng. Appl. Artif. Intell.*, vol. 22, no. 7, pp. 979–991, Oct. 2009.

[2] S. Bussmann and D. C. McFarlane, "Rationales for holonic manufacturing control," in *Proc. of Second Int. Workshop on Intelligent Manufacturing Systems*, 1999, pp. 177–184.

[3] P. Valckenaers and H. Van Brussel, “Holonic manufacturing execution systems,” *CIRP Ann. - Manuf. Technol.*, vol. 54, no. 1, pp. 427–432, 2005.
[4] V. Marik and D. McFarlane, “Industrial adoption of agent-based technologies,” *IEEE Intell. Syst.*, vol. 20, no. 1, pp. 27 – 35, Feb. 2005.
[5] M. Metzger and G. Polakow, “A Survey on Applications of Agent Technology in Industrial Process Control,” *IEEE Trans. Ind. Inform.*, vol. 7, no. 4, pp. 570–581, Nov. 2011.
[6] M. Wooldridge, N. R. Jennings, and D. Kinny, “The Gaia methodology for agent-oriented analysis and design,” *Auton. Agents Multi-Agent Syst.*, vol. 3, no. 3, pp. 285–312, Sep. 2000.
[7] C. A. Iglesias, M. Garijo, and J. C. González, “A Survey of Agent-Oriented Methodologies,” in *Intelligent Agents V: Agents Theories, Architectures, and Languages*, vol. 1555, Springer Berlin Heidelberg, 1999, pp. 317–330.
[8] J. Gomez-Sanz and J. Pavon, “Methodologies for Developing Multi-Agent Systems,” *J. Univers. Comput. Sci.*, vol. 10, no. 4, pp. 359–374, Feb. 2004.
[9] M. Wooldridge, *An Introduction to MultiAgent Systems*, 2nd ed. Wiley, 2009.
[10] L. Rannanjärvi and T. Heikkilä, “Software development for holonic manufacturing systems,” *Comput. Ind.*, vol. 37, no. 3, pp. 233–253, 1998.
[11] R. Kazman, M. H. Klein, and P. C. Clements, “ATAM: Method for Architecture Evaluation,” Software Engineering Institute, Carnegie Mellon, Pittsburgh, Pennsylvania, United States, Technical Report CMU/SEI-2000-TR-004, Aug. 2000.
[12] A. Purhonen, “Performance evaluation approaches for software architects,” in *Component-Based Software Development for Embedded Systems*, vol. 3778, Springer, 2005, pp. 275–295.
[13] L. Heikkilä, H. Saarinen, L. Aha, M. Viinikainen, J. Mattila, A. Hahto, M. Siuko, and L. Semeraro, “Analysis of the new architecture proposal for the CMM control system,” *Fusion Eng. Des.*, vol. 86, no. 9–11, pp. 2071–2074, Oct. 2011.
[14] N. Boucké, D. Weyns, K. Schelfthout, and T. Holvoet, “Applying the ATAM to an architecture for decentralized control of a transportation system,” in *Quality of Software Architectures*, vol. 4214, Springer, 2006, pp. 180–198.
[15] A. Bessey, K. Block, B. Chelf, A. Chou, B. Fulton, S. Hallem, C. Henri-Gros, A. Kamsky, S. McPeak, and D. Engler, “A few billion lines of code later: using static analysis to find bugs in the real world,” *Commun. ACM*, vol. 53, no. 2, pp. 66–75, Feb. 2010.
[16] S. D. Conte, H. E. Dunsmore, V. Y. Shen, and W. M. Zage, “A Software Metrics Survey,” Department of Computer Science, Purdue University, West Lafayette, Indiana, Technical Report 87-720, Feb. 1987.
[17] F. Riguzzi, “A survey of software metrics,” Dipartimento di Elettronica, Informatica e Sistemistica, Università degli Studi di Bologna, Bologna, Italy, Technical Report DEIS-LIA-96-010, Jul. 1996.
[18] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, 2 edition. Course Technology, 1998.
[19] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of Software Engineering*, 2 edition. Prentice Hall, 2002.
[20] M. H. Halstead, *Elements of software science*. Elsevier, 1977.