International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 1, pp: 1204-1208 www.ijaem.net ISSN: 2395-5252

Analysis of Software Metrics in Multi Agent Based Software Development

P.Felcy Judith
T .John College Bangalore

Date of Submission: 01-07-2020 Date of Acceptance: 16-07-2020

ABSTRACT
The objective of the software system is the ability to quickly solve complex requirements and to have a more flexible architecture to apply changes faster. In general the software requirements grow more complex from time to time. Also the software should be highly configurable in no time or very less time the product should hit the software market before the competitor product reaches the market. This paper analyses the metrics of the software developed using automated multi-agent based framework prototyping a process in comparison to other engineering industry.

Keywords - Agent; Multi-Agents; Modeling Agents; Agent Oriented Software Engineering;

I. INTRODUCTION

Recent statistical study says "17 percent of large IT projects go so badly that they can threaten the very existence of the company, on average, large IT projects run 45 percent over budget and 7 percent over time, while delivering 56 percent less value than predicted [11]". These numbers suggests there are still some fundamental issues with respect to software development. After comparing with other engineering disciplines, Jack Greenfield and Keith Short [12]prescribed One-off development or development in isolation, Monolithic system - increasing system complexity, working at low levels of abstraction, process immaturity and rapidly growing demand for software systems as the primary reasons for lack of predictability in software development.

Current software languages and tools work at low levels of abstraction to provide greater flexibility. Even though the third generation languages do justice by increasing the level of abstraction without compromising flexibility, still the level of abstraction is not higher enough to produce reusable code across domain and platforms. Software development processes are not matured enough in comparison with other engineering disciplines to produce flexible and predictable software. Rapidly growing demand of software increases along with the size and the complexity of the software. This brings in the necessity of standardization of software systems like other engineering divisions where two entirely different products could exchange their parts.

AGENTS BASED SOFTWARE DESIGN

Design of multi-agent based software in order to achieve

a. Reusability of behaviors by the way of agent design
b. Reactive, proactive, autonomous and decision making skills
c. Dynamically configure agent behaviors, actuators, sensors and construct agents using configuration
d. Create more modular mechanism
e. Achieve secure communication
f. Achieve greater separation of concern

VALIDATING AGENT BASED SOFTWARE SYSTEM

In the process of quantitative analysis, four open source software is chosen and various parameters from the code metrics were analyzed. They were named as software "A", "B", "C" and "D". Software "D" is agent based software has closer practices followed in this research. Even though software "D" did not use models and code generation but chooses technology of agent framework. In the perspective of validation the model development and code generation is outside the scope of quantitative analysis.

Using software metrics tools the data is collected for all the four software chosen to the method and class level. The abstract data looks like given in figure 1. The summary values for the software "A", "B", "C" and "D" is given in figure 2, 3, 4 and 5 respectively.

	B	C	D	E	F	G	H	I	J	K
1	Cyclomatic Complexity (CC)	IL Cyclomatic Complexity (ILCC)	NbTypesUsingMe	NbTypesUsed	Instability	Depth of inheritance	# Children	Association Between Types (ABT)	Lack of Cohesion Of Methods (LCOM)	Rank
285	N/A	28	1	16	0.941176471	1	0	9	0	0.2614
286	N/A	26	1	16	0.941176471	1	0	8	0	0.2614
287	N/A	26	1	16	0.941176471	1	0	8	0	0.2614
288	N/A	28	1	19	0.95	1	0	9	0	0.2667
289	N/A	2	1	10	0.909090909	1	0	12	0	0.1862
290	N/A	2	1	7	0.875	1	0	2	0	0.1973
291	N/A	1	2	5	0.714285714	1	0	1	0	0.3123
292	N/A	3	1	13	0.928571429	1	0	7	0	0.206
293	4 428	6 179	1 248	4 292	232.5589176	498	128	4 755	96.664	169.85
294	15.702	23.494	4.2887	14.749	0.799171538	1.8652	0.50593	16.34	0.36754	0.58368
295	0	0	0	2	0.043478261	1	0	0	0	0.15
296	203	293	66	95	1	7	18	151	1	7.3356
297	23.873	35.435	7.9374	12.575	0.224458988	1.5005	2.179	23.326	0.37079	0.93329
298	569.93	1 255	63.003	158.12	0.050381838	2.2515	4.748	544.11	0.13748	0.87102

Figure 1: Class level quantitative data sample

fileController	Cyclomatic	IL Cyclomatic Complexity (ILCC)	NbTypesUsingMe(Ca	NbTypesUsed(Ce)	Instability	Depth of inheritance
Sum:	2 577	3 274	648	3 235	163.20701	387
Average:	13.215	18.291	3.3231	16.59	0.83695903	2.25
Minimum:	0	0	0	2	0.03703704	1
Maximum:	188	211	52	72	1	6
Standard deviation:	24.827	30.882	7.0451	13.839	0.1860138	1.4392
Variance:	616.36	953.71	49.634	191.51	0.03460114	2.0712

fileController	# Children	Association Between Types (ABT)	Lack of Cohesion Of Methods	Rank	Level	# lines of code
Sum:	114	3 484	64.042	83.061	129	5 955
Average:	0.72611	17.867	0.35777	0.426	1.5176	33.268
Minimum:	0	0	0	0.15	1	0
Maximum:	51	151	0.96417	5.1719	4	423
Standard deviation:	4.3532	27.755	0.35896	0.6101	0.74537	57.215
Variance:	18.95	770.33	0.12885	0.3722	0.55557	3 273

Figure 2: Summary metrics values for software "A"

types	Cyclomatic Complexity (CC)	IL Cyclomatic Complexity (ILCC)	NbTypesUsingMe	NbTypesUsed	Instability	Depth of inheritance
Sum:	16 314	24 610	7 632	27 935	1280.31822	4 774
Average:	13.35	17.031	5.0745	18.574	0.851275412	3.2279
Minimum:	0	0	0	1	0.015503876	1
Maximum:	535	647	368	147	1	6
Standard deviation:	31.334	37.592	20.429	16.064	0.201096612	1.8643
Variance:	981.83	1 413	417.33	258.06	0.040439848	3.4755

types	# Children	Association Between Types (ABT)	Lack of Cohesion Of Methods (LCOM)	Rank	Level	# lines of code (LOC)
Sum:	1 033	28 869	635.4	592.44	575	44 378
Average:	0.71836	19.195	0.43972	0.39391	1.3039	30.711
Minimum:	0	0	0	0.15	1	0
Maximum:	309	685	1	19.777	8	1 760
Standard deviation:	11.16	38.725	0.43978	1.0866	0.80992	94.8
Variance:	124.55	1 499	0.19341	1.1807	0.65597	8 987

Figure3: Summary metrics values for software "B"

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 1, pp: 1204-1208 www.ijaem.net ISSN: 2395-5252

types	Cyclomatic Complexity (CC)	IL Cyclomatic Complexity (ILCC)	NbTypesUsingMe	NbTypesUsed	Instability	Depth of inheritance
Sum:	4 428	6 179	1 248	4 292	232.5589176	498
Average:	15.702	23.494	4.2887	14.749	0.799171538	1.8652
Minimum:	0	0	0	2	0.043478261	1
Maximum:	203	293	66	95	1	7
Standard deviation:	23.873	35.435	7.9374	12.575	0.224458988	1.5005
Variance:	569.93	1 255	63.003	158.12	0.050381838	2.2515

types	# Children	Association Between Types (ABT)	Lack of Cohesion Of Methods (LCOM)	Rank	Level	# lines of code (LOC)
Sum:	128	4 755	96.664	169.85	228	13 310
Average:	0.50593	16.34	0.36754	0.58368	1.916	50.608
Minimum:	0	0	0	0.15	1	0
Maximum:	18	151	1	7.3356	6	668
Standard deviation:	2.179	23.326	0.37079	0.93329	1.1492	78.56
Variance:	4.748	544.11	0.13748	0.87102	1.3207	6 171

Figure 4: Summary metrics values for software "C"

types	Cyclomatic Complexity (CC)	IL Cyclomatic Complexity (ILCC)	NbTypesUsingMe	NbTypesUsed	Instablity	Depth of inheritance
Sum:	822	1 411	561	1 832	103.8419215	199
Average:	9.1333	11.857	4.218	13.774	0.780766327	1.6179
Minimum:	0	0	0	2	0.064516129	1
Maximum:	54	88	39	74	1	7
Standard deviation:	11.497	16.241	7.1455	12.757	0.20852428	0.87903
Variance:	132.18	263.77	51.058	162.75	0.043482375	0.77269

types	# Children	Association Between Types (ABT)	Lack of Cohesion Of Methods (LCOM)	Rank	Level	# lines of code (LOC)
Sum:	63	1 698	25.009	91.048	96	2 124
Average:	0.6	12.767	0.21016	0.68457	1.6	17.849
Minimum:	0	0	0	0.15	1	0
Maximum:	29	117	1	7.9377	5	116
Standard deviation:	3.1308	18.565	0.31478	1.1432	1.052	26.152
Variance:	9.8019	344.66	0.099089	1.3069	1.1067	683.93

Figure 5: Summary metrics values for software "D"

The metrics and their analysis is given below in Figure 6

i) Lines of Code (Loc) – Number of lines of code available with a specific class

ii) Lines of IL instructions (ILoc) – Number of lines of code available in the intermediate language in Common Language Runtime (CLR).

iii) Afferent coupling (Ca) – Number of types outside this package that depend on types within this package.

iv) Efferent coupling (Ce) – Number of types within this package that depend on the types outside this package.

v) Instability (I) – ration of efferent coupling to total coupling

i.e. I = Ce /(Ce + Ca)

Instability = 0 indicates completely stable package, painful to modify.

Instability = 1 indicates completely instable package.

vi) Cyclomatic Complexity (CC) – The number of decisions that can be taken in a procedure, this depends on number of branching, looping statements available in the code of a particular method.

For a method CC > 15 then it would be hard to understand CC > 30 the method is extremely complex.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 1, pp: 1204-1208 www.ijaem.net ISSN: 2395-5252

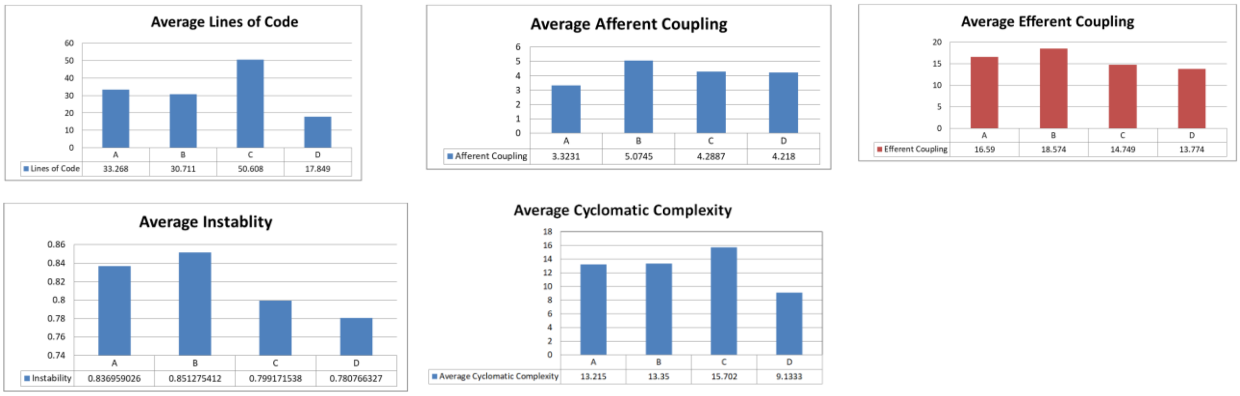

Figure 6: Software Metrics comparison of agent based software with other software's

II. CONCLUSION

The analysis proves the agent based software development addresses the key issues of software development by raising the level of abstraction, providing necessary modularity and providing better results for the range of software metrics. Additionally the design carried out for simulation indirectly satisfies the challenges of solving complex requirements, reducing time to market, faster implementation of change in requirement, lowering cost of production and maintenance. Overall the agent based software developments provides a step towards resolving challenges in the software industry in providing practices equivalent to other engineering disciplines.

REFERENCES

[1]. J. Greenfield and K. Short, Software Factories: Assembling Applications with Patterns, Models, rameworks, and Tools, John Wiley and Sons, 2004.

[2]. P. C. Smolik, Mambo Metamodeling Environment, A dissertation submitted in partial fulfillment of the requirements for the degree of doctor of philosophy, Brno University of Technology, Czech Republic, 2006.

[3]. OMG Trademarks, http://www.omg.org/legal/tm_list.htm, January 2007.

[4]. S. Cook, Domain Specific Modeling and Model Driven Architecture, MDA Journal, January 2004,

[5]. H. Jonkers, M. Stroucken, and R. Vdovjak, Bootstrapping Domain-Specific Model-Driven SoftwareDevelopment within Philips, In Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling (DSM'06), pages 204-213, 2006, http://www.dsmforum.org/events/DSM06/Papers/21-Vdovjak.pdf, January 2007.

[6]. B. Selic, Model-Driven Development: Its Essence and Opportunities, In Proceedings of the Ninth IEEEInternational Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'06),pages 313-319, IEEE Computer Society, Washington, DC, 2006.

[7]. OMG Model Driven Architecture (MDA) Main Page, http://www.omg.org/mda/, January 2007.

[8]. Microsoft Software Factories Main Page, http://www.softwarefactories.com/, January 2007.

[9]. D.C. Schmidt, Model-Driven Engineering , Computer, 39(2): 25--31, 2006.

[10]. K. Balasubramanian, A. Gokhale, G. Karsai, J. Sztipanovits, and S. Neema, Developing Applications Using Model-Driven design Environments, *Computer*, 39(2): 33--40, 2006.

[11]. Study on large scale IT Projects, McKinsey & Company in conjunction with the University of Oxford, 2012

[12]. Jack Greenfield and Keith Short, Software Factories: Assembling Applications with Patterns, Models,Frameworks, and Tools (Indianapolis, IN: Wiley, 2004)

[13]. The Case for Software Factories, Jack Greenfield, Microsoft Corporation, July 2004

[14]. Ferber J. 'Multi-agent systems: An introduction to distributed artificial intelligence', 1999 Published by: Addison-Wesley, ISBN: 0201360489

[15]. Trichakis, Pavlos(2009) Multi Agent Systems for the Active Management of Electrical Distribution Networks, Durham theses, Durham University.

[16]. Teamwork in Multi-Agent Systems: A Formal Approach Barbara Dunin-K ¸eplicz and RinekeVerbrugge, 2010 John Wiley & Sons, Ltd
[17]. Multiagent Systems, A Modern Approach to Distributed Modern Approach to Artificial Intelligence, 1999 Massachusetts Institute of Technology
[18]. The Art of Agent-Oriented Modeling, Leon Sterling and KuldarTaveter, 2009 Massachusetts Institute of Technology
[19]. Wooldridge, M., Jennings, N. R. and Kinny, D. (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3, (3), 285-312.
[20]. Wagner, G. (2003a). The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behavior. Information Systems28:5 (2003), 475-504. Available at http://aor.rezearch.info/

International Journal of Advances in Engineering and Management
ISSN: 2395-5252

IJAEM

Volume: 02 Issue: 01 DOI: 10.35629/5252

www.ijaem.net
Email id: ijaem.paper@gmail.com