5th International Conference on Emerging Trends in Engineering, Technology, Science and Management
Institution of Electronics and Telecommunication Engineers, Ganganagar, Bengaluru, Karnataka, India (ICETETSM-17)
20th August 2017, www.conferenceworld.in ISBN: 978-93-86171-60-3

DEVELOPING DYNAMIC CAPABLITY FOR MULTI-AGENTS

P. Felcy Judith

Associate Prof, T. John College, Bangalore, Karnataka, (India)

ABSTRACT

In comparison to the other engineering discipline software industry needs lot of maturity in order to provide technologies for sustainable development. The software industry need to focus on improving the level of abstraction, separation of concern or modularity, improvement in software life cycle process, software measurements and other software concerns in order to achieve such technologies. The other engineering discipline works with tangible material and it is confined to the natural laws and boundaries. These boundaries are not modifiable and design needs to be concrete with the understanding of the boundaries and limits. Whereas the software goals and boundaries are more abstract in nature and the limitations are more flexible in contrast to the other industries. Since it is more abstract human knowledge plays a crucial part in software design. The introduction of multi-agent software development methodologies into main line software development could revolutionize the way software is made. Agents are the best replacement of human and could be a tool to capture human knowledge. The design of agents could provide greater modularity in organizing the agents and achieving the results of collective interaction of agents. In object oriented development the objects represent the real world entity, but in addition to that the agents also capture the way human interact with the entity.

Keywords: Agent Development, Multi-Agents, Agent based Sustainable Development.

I.INTRODUCTION

The following are some of the needs for achieving sustainable development

1. The need for software change is greater in communication technologies like telephones, Mobile technologies, satellite television and other communication technologies
2. The need for software innovation in house hold application product development, security devices and in consumer electronics domain is large.
3. The reduction in software complexity is going to reduce the cost of devices consuming the software technologies. This would highly revolutionize the dependent technologies.
4. Under development and developed countries needs governance and growth in other areas including industrial growth. The changes in the software development and the reduction in the cost of software could revolutionize the growth of those countries.
5. The need for software technologies in the agricultural development in order to redefine the way agricultural practices happening around the world. The low cost affordable technologies could revolutionize agro industry and which would make a greater impact in reducing poverty and hunger across the world.
6. The need for revolution in terms of automation is high and comparing with the peer electronic devices technology. There are several layers of automation in place with the electronics chip design. The complexity is highly managed and this helped reduction in the production cost of electronic devices.

ISBN: 978-93-86171-60-3

7. In comparison with the automotive industry the automation is mainly done for the areas where the key need of human arises. The technology is developed keeping man machine interaction in mind.
8. Technology could no longer be only available for very few people in the world. The developing and under developed countries needs more technology in order to solve their problems.
9. The mobile technology now revolutionized the way communication could make a difference in the under developed and developing countries. Similar technology is need in the software world. The simpler easy to use and quickly configurable software should hit the market and software development should be possible for people without higher knowledge.
10. The software and technology is needed for satellite and other technologies but more than that it should be available for the common man. Software revolution up to now already provided such a revolution. But now the need is to go still beyond the boundaries of social and economic level.

II. MULTI AGENT SYSTEM

Multi Agent Systems (MAS) is based on distributed artificial intelligence computing, where the aim is to split a complex problem into several subtasks and distribute the management of these tasks to individual software entities [15]. This allows system intelligence to be distributed across the system components rather than being concentrated on a single point. "An agent is a computer system, situated in some environment that is capable of flexible autonomous action in order to meet its design objectives [14]" this is the commonly accepted definition for agents. An agent controls the environment through sensors and actuators and possess well defined boundary. Agents are flexible by exhibiting the following characteristics [17].

Reactive – Agents were able to perceive their environment and respond to the changes occur in their environment in order to satisfy their design behavior

Pro-Active – Agents exhibit goal-oriented behavior by taking initiative in order to satisfy their design behavior

Being Social – Agents communicate with other agents in order to satisfy their design behavior

"Multi-agent systems (MAS) are computational systems in which a collection of loosely coupled autonomous agents interact in order to solve a given problem. [16]" Agents communicate, cooperate, coordinate and negotiate for

a) Achieving a common goal
b) Monitor the progress of the team effort
c) Help another in need
d) Co-ordinate individual action so that they do not interfere with one another
e) Communicating success and failures
f) To establish zero competition among team members

III. DESIGNING AGENTS RULES FRAMEWORK

Agents are dynamic in nature. Agent rules needs to be dynamic in nature. Business rules for general software are static and coded in business layer and could not be changed without altering code and making a new deployment. This works well when there is fixed business rules and no major changes to the rule in frequent happens. But considering the agent framework where agent works as a team to produce results. Also agents

ISBN: 978-93-86171-60-3

generally adopt itself to the environment and change in the environment. So the agent working should not be decided by predictable business rules. Every individual agent posses a goal to achieve and in collaboration they achieve common goals. This agent prototype design looks for pluggable rules to individual agents. Also the agents can able to receive their rules on the runtime so that change in rules could be incorporated in runtime without changing any code. Pluggable rules could be achieved by creating DSL specifically textual DSL, where the business users could configure rules using DSL interface and could plug-in the rules to the agents. Here in this prototype we look for windows workflow based rules instead of a textual DSL. The visual studio provides graphical environment for orchestrating the rules which could be used by the business users to make rules.

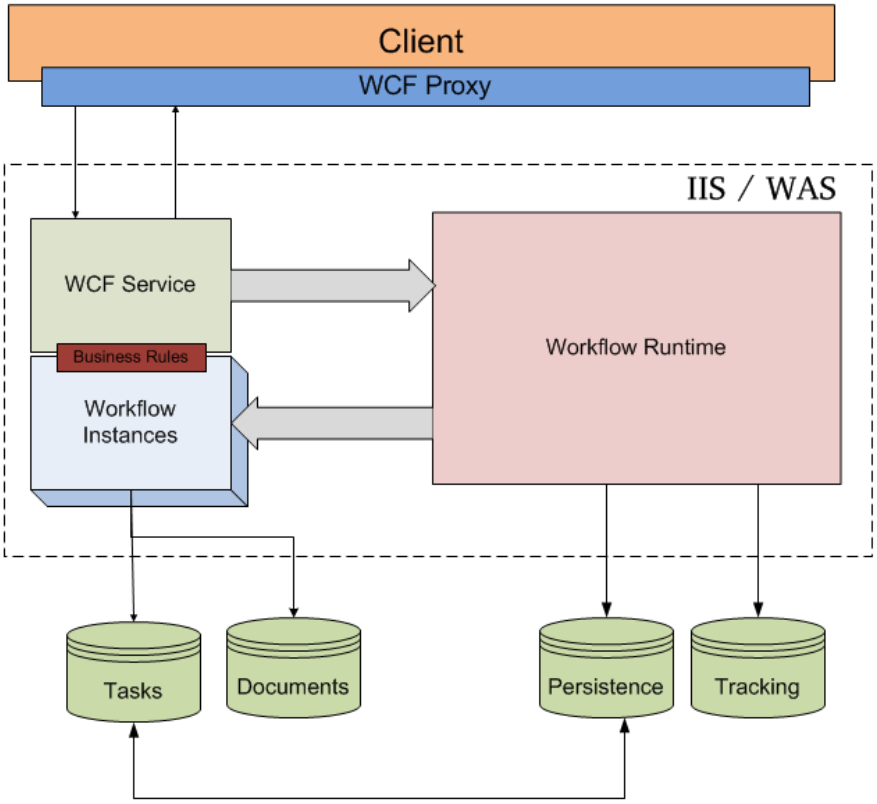

Figure 1: Architecture Business Rules Workflow Process

Windows Workflow Foundation (WF) is a Microsoft Technology platform for building workflow enabled applications. The rules engine in our case is implemented with the workflow. Workflow is specifically useful for the coordinating business process including human workflows where people drive the processes. This workflow carefully integrates the system tasks with the people and automates the process. The primary goal of the human workflow is to automate business processes to reduce the time spent waiting between human activities and to increase productivity. This also provides metrics to measure time spent on independent activities, identify the most time consuming resources which will help remove bottlenecks. Figure 1 shows the architecture of workflow based business rules implementation. The business rules are configured thru xml and the rules could be change in the configuration XML so that the rules will be always latest. These rules could be made persist in the database and could be exposed using Windows Communication Foundation (WCF) web services.

ISBN: 978-93-86171-60-3

Agents developed with .NET framework 4.0 or higher will be able to communicate with the workflow services hosted in IIS shown in Figure 2. Agents can communicate with HTTP or TCP with these services.

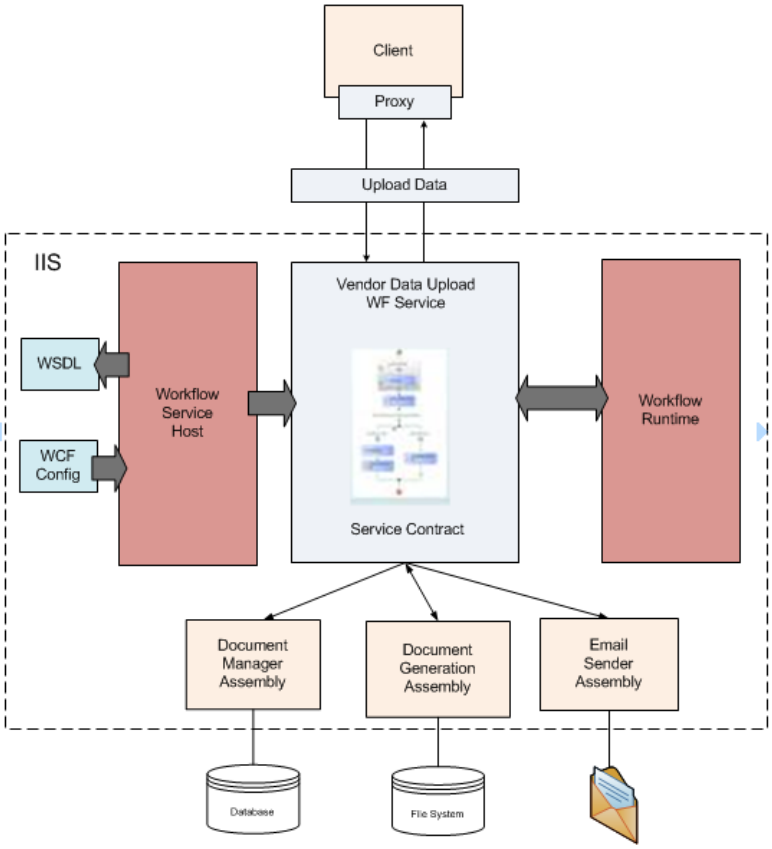

Figure 2: Logical Architecture of Agent Service

IV. CONCLUSION

Multi agent system is one of the solutions for sustainable technologies. This paper discusses one of the dynamic aspects of agents. There are several other aspects of agents will be dealt in future. The primary goal of this paper is to create sustainable solutions for affordable usage for all sorts of people.

REFERENCES

[1] J. Greenfield and K. Short, Software Factories: Assembling Applications with Patterns, Models, Frameworks, 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/

[21]Sterling, L.; Taveter, K.; The Daedalus, Team (2006). Building Agent-Based Appliances with Complementary Methodologies. In: Proceedings of the Joint Conference on Knowledge-Based Software Engineering 2006 (JCKBSE'06): August 28-31, 2006, Tallinn, Estonia. (Eds.)Tyugu, E.; Yamaguchi, T. . IOS Press, 2006, 223 - 232.