Role Based Multi-Agent System for E-Learning (MASeL)

Mustafa Hameed, Nadeem Akhtar, Malik Saad Missen
Department of Computer Science & IT
The Islamia University of Bahawalpur
Baghdad-ul-Jadeed Campus, Bahawalpur, PAKISTAN

***Abstract*—Software agents are autonomous entities that can interact intelligently with other agents as well as their environment in order to carry out a specific task. We have proposed a role-based multi-agent system for e-learning. This multi-agent system is based on Agent-Group-Role (AGR) method. As a multi-agent system is distributed, ensuring correctness is an important issue. We have formally modeled our role-based multi-agent system. The correctness properties of liveness and safety are specified as well as verified. Timed-automata based model checker UPPAAL is used for the specification as well as verification of the e-learning system. This results in a formally specified and verified model of the role-based multi-agent system.**

Keywords—Information Management System (IMS); Multi-Agent System (MAS); Role Based Multi-Agent Systems; Agent-Group-Role (AGR); Agent-based Virtual Classroom (AVC); Intelligent Virtual Classroom (IVC); E-Learning; Information and Communication Technologies (ICTs); Formal verification; Model Checking

I. Introduction

Use of information and communication technologies plays a vital role in a virtual classroom. E-learning systems are widely being deployed by many universities. Several new techniques like blended learning, peer-to-peer learning, and collaborative learning are being introduced [21]. Collaborative learning is an approach which builds up confidence in students and helps students to be independent in learning as well as an efficient team member to build up a knowledge base [17]. It also helps to learn from the experiences of others. This is practically possible in real classrooms but intelligent systems can also provide this as a feature in virtual classroom systems for e-learning.

Agent-based computing system developed for e-learning has concurrent processes and is distributed in nature. Along with these characteristics this type of system demands intelligence and social learning abilities. A multi-agent system is equipped with these features; and provides many fringe benefits like *autonomy*, *social ability*, *reactivity* and *pro-activeness* [3]. A role-based approach for multi-agent systems based on AALAADIN meta model [18] called AGR (Agent / Group / Role) have potential to prototype components of an organization (i.e. persons, environment) in the form of intelligent agents [1].

At the same time, the software systems are error prone [4]. And if these systems are distributed and concurrent, errors are enormous due to complexity. There are many techniques for error detection, error removal, and error reduction in software systems like software testing, inspection, or simulation [5]. Model checking is a popular technique for formal verification which formally verifies system properties to prove the correctness of the system. This enables an Intelligent Virtual Classroom with correct intended behavior.

In our paper, Section 2 elaborates state-of-the-art, agents, multi-agent systems, e-learning, correctness, formal verification, model checking, and timed-automata. In Section 3, we propose a role-based multi-agent system for an Intelligent Virtual Classroom (IVC) to be used in e-learning environment.

Our IVC is the core of *MASeL* (Multi-Agent System for E-Learning) featuring collaborative learning in an intelligent and flexible manner with Agent-Group-Role (AGR) architecture. We have elaborated on how a new skill is coined by the teacher; how students share knowledge; how students consider each other as peers; and how group leaders are selected using role-based intelligent agents. Furthermore, in Section 4, the states of the systems are defined in timed-automata and correctness of the system is formally verified with the help of model checking. Concluding notes about our proposed approach and suggestions for future work are presented in Section 5.

II. State of the Art

A. Agents and Multi-Agent Systems

An agent "is an encapsulated computer system that is situated in some environment" [39]. An agent-based system has autonomy, social ability, reactivity, and proactive-ness properties [39][27]. Multi-agent systems may have heterogeneous or homogeneous agents. Multiple interacting agents in some environment for a common goal are called multi-agent systems, which are true implementation of agents. Agents in multi-agent systems can either have their own personal goals or an overall system goal. According to [20] multi-agent systems evolved from distributed artificial intelligence. Multi-agent systems are attractive for designing and implementing open and distributed systems because of their modularization and abstraction capabilities. Odell has noted several applications of multi-agent system [35]. ZEUS, JADE [6], agenTool, RETSINA, JATLite, FIPA-OS, JAFMAS, Agent Building Shell, OAA, Cougaar, AgentSpace, Cybele and MADkit [24] [25] are tools for multi-agent system development [4]. Multi-agent development kit was developed at LIRMM (France). It is based on a model AALAADIN. This model shapes agents as agent, group, and role in an organization. MADKIT is used for many applications like agent-based social networks or agent-based robots [24] [25].

There are various approaches for multi-agent systems based on roles of agents [10]. Almost each approach faces the common problems of heterogeneity of languages, multiple operations, and languages and security. [18] proposes a model AALAADIN which address all these issues. AALAADIN, ROPE (Role-Oriented Programming), TRUCE (Tasks and Roles in Unified Coordination Environment), Yu and Schmid's proposal, Kendall's proposal, RoleEP (Role-based Evolutionary Programming), BRAIN (Behavioral Role for Agent INteraction), Fasli's proposal, Gaia [40], TRANS (Tractable Role-based Agent prototype for concurrent Navigation Systems), RICA (RICA-J) Role/Interaction/Communicative Action, Zhu and Zhou's proposal [10], Role based BDI framework [31] are different approaches for role-based multi-agent systems development. AALAADIN meta-model, as its advocates [18], [24], [19] claim that it addresses three main problems (heterogeneity of languages, multiple applications and architecture, security) of multi-agent systems for their design and implementation. Agent, group role is the notation used for creation of organization of multi-agent systems in AALAADIN. Community/Group/Role (CGR) as a variation of AGR is also used in MadKit [24]. MadKit is a toolkit based on java for development of organization-centered multi-agent systems based on AGR (Agent/Group/Role) architecture [24]. AGRE (Agent-Group-Role-Environment), OCMAS (Organization Centered Multi-Agent Systems), MASQ (Multi-Agent Systems based on Quadrants) are extension of AGR [19].

B. e-Learning

The most comprehensive definition of e-learning is "the use of any kind of internet or communication service or electronic device that supports learning process" [17]. One good thing for e-learning is that now it is being widely accepted by teachers and students regardless of any particular discipline of education [29]. Modern e-learning systems use artificial intelligence to make it more efficient and productive [21]. [33] proposed a collaborative learning method based on multi-agent system. [34] studied and proposed an architecture of an intelligent tutoring system to support distance learning. Tutoring systems for distance education are not new to research and academic community. There are many early examples for these types of systems like [8] and [9]. Virtual classrooms are the core of any tutoring systems. Agent technology is being widely used to model students and teachers and handle their interaction, address their dynamic and run-time needs by their novel characteristics like intelligence and autonomy. An agent-based Virtual Classroom (AVC) represents virtual professor agent, virtual student agent, and the interfaces between them. A content agent is also present in AVC to provide relevant content and handle the changes in content [37].

C. Correctness: Safety and liveness properties

Correctness verifies that the software behavior (i.e. functionality) is exactly according to its requirement specifications. It is a mathematical property that is absolute. Thus a program is functionally correct if it behaves according to its stated functional requirements [22]. Correctness can be accessed systematically and precisely by rigorously specifying the functional requirements [41].

The fundamental correctness properties are safety and liveness. Safety property is an invariant which asserts that something bad never happens, that an acceptable state of affairs is maintained. [32] has defined safety property as a deterministic process. ERROR conditions are like exceptions which state what is not required, as in a complex system we specify safety property by directly stating what is required. The liveness property asserts "something good happens" which describes the states of a system that an agent must bring about given certain conditions [41]. Both safety and liveness complement each other and play an important role in system verification. Progress [23] is also a type of liveness property. Reachability of state in the system under study means that, a particular state is reachable. Deadlock freeness means absence of deadlocks. This is achieved by proving safety and liveness properties. It assures that system will not stop working until in a decided terminal state. Timed automata [28] is an approach for model checking in which systems states are explicitly defined and correctness of the systems is assured by these formally verifying the safety, liveness, and reachability of these states of the system.

D. Formal verification

Formal methods are techniques based on mathematics to design and develop software systems. Formal methods include techniques like formal specification, formal verification and automatic theorem proving [7]. Formal verification is the mathematical demonstration of the correctness of a system. Formal verification, on its mathematical foundations examines the system in accordance with the given formal specification of that system [38]. If an error is in early phases of development, it can cause big losses at the later phases. Formal verification provides ways for early detection of errors.

In [41], we have proposed a mathematical model based on UPPAAL for the design and formal verification of a multi-agent based transport system. Formal verification proves or disproves the correctness properties of the system.

E. Model Checking

Model Checking [12][13][14][15][16][28][36] is a method for automatic and algorithmic verification of finite state concurrent systems [41]. OBDD (Ordered Binary Decision Diagrams), SAT-based translation [2], Fix-point characterization of CTL [5] and timed automata are techniques for model checking.

Model checking is for finite systems but can be scaled up for complex systems (i.e. multi-agent systems) having infinite number of states. It involves the formal verification of system properties to prove the correctness of the multi-agent system.

A complex system has a large number of independent interacting components, with non-linear aggregate activity, with concurrency between components and constant evolution [41].

Model checking due to its dynamic and automatic capabilities is more suitable for multi-agent systems; it is executed as an in-depth state space exploration that is guaranteed to terminate since the given model is finite; its algorithms are used to improve the system design; it can handle a large number of states and can be used for evolving systems [22]. There are many tools for model checking, for example, MCMAS (Model Checking Multi-Agent Systems), SMV (Symbolic Model Verifier) [11], SPIN (Simple PROmela Interpreter) [26], VDM (Vienna Development Model), LTSA (Labeled Transition System Analyzer) [32], Petri-nets and UPPAAL [30].

F. Timed Automata

Timed automaton models finite state real-time systems. A timed automaton is a finite-state automaton equipped with a finite set of real-value clock variables called clocks, which are used to measure the elapse of time. In [41] we have detailed timed-automata.

Timed automata are described with two elements: (1) Automata and (2) the passage of time. Control states for the system and their transition are defined in automata along with the instances of the states. Time constraints are imposed on transitions of the state with the help of clocks [5].

UPPAAL [30] is a formal tool for symbolic model checking of real-time systems developed at the University of Uppsala (Sweden) and Aalborg (Denmark). In [41], we have discussed UPPAAL as well as formally designed and verified a multi-agent based transport system.

III. Role-Based Multi-Agent System for E-Learning (MASeL)

A multi-agent system based on roles for e-learning named MASeL is proposed. The core of the system is Intelligent Virtual Classroom (IVC). Features proposed in MASeL are:

1) *Employ collaborative learning*
2) *Enable real-time group discussion*
3) *Dynamic synchronization of learning process*
4) *Real-time question and answer sessions between students and teachers*
5) *Dynamic delivery of lessons*
6) *Adaptive assessment*
7) *Study habits of the people involved in learning process*
8) *Create a sense of community by virtual teachers and students*
9) *Multi-platform access*
10) *Using project based learning methods*

G. Layers of MASeL

Depending upon the functionalities and roles of individual agents, the agents can form group to form a community of agents. Groups of these intelligent agents are placed in layered architecture as shown in Figure 1.

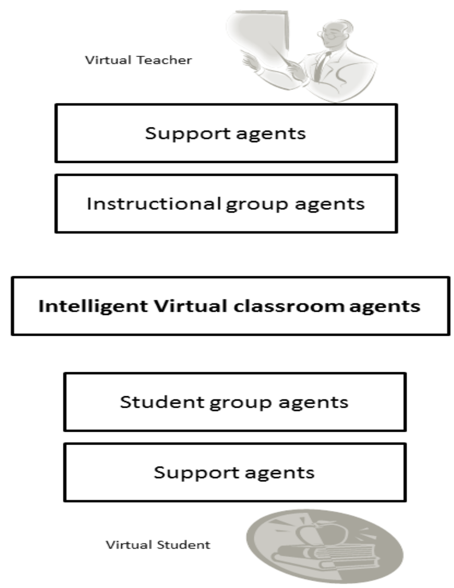

Fig. 1. Layers of MASeL

IVC agents are a group of agents which constitute the core of MASeL. Virtual students play an active or passive peer role with the real student. Virtual teachers have the ability to demonstrate a content-centered as well as student-centered role as demanded by the environment.

H. Roles of Agents in MASeL

Each agent in MASeL has a specific role. Based on the role of agents, they are classified in groups.

I. Intelligent Virtual Classroom (IVC)

The IVC consists of instructional group, student group, and support group agents. These three group of agents interact in IVC and this intelligent interaction results into a learning environment.

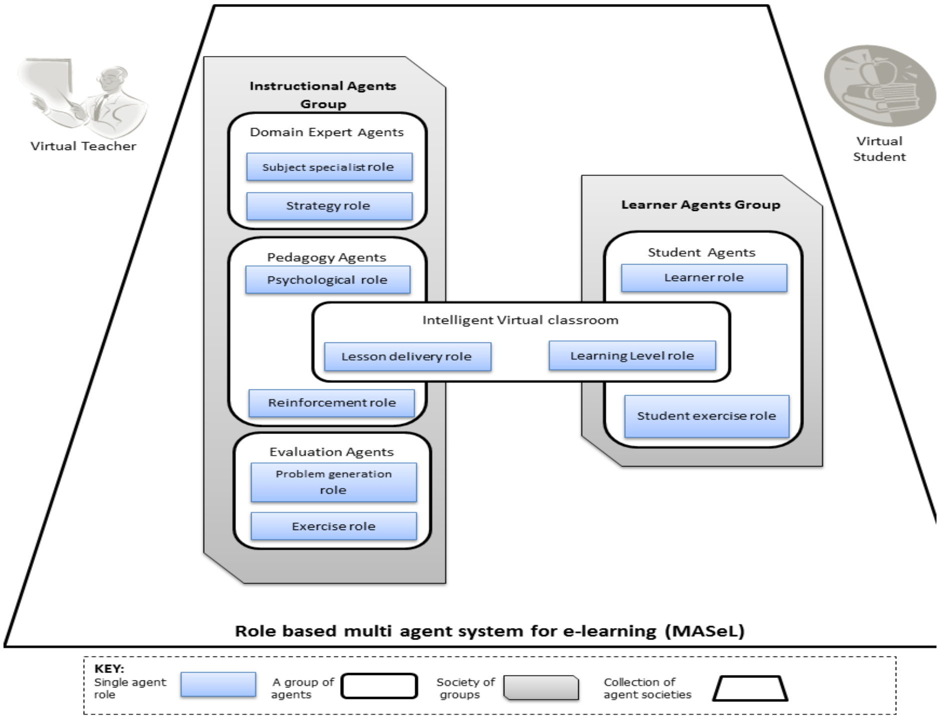

Fig. 2. MASeL: Agent roles

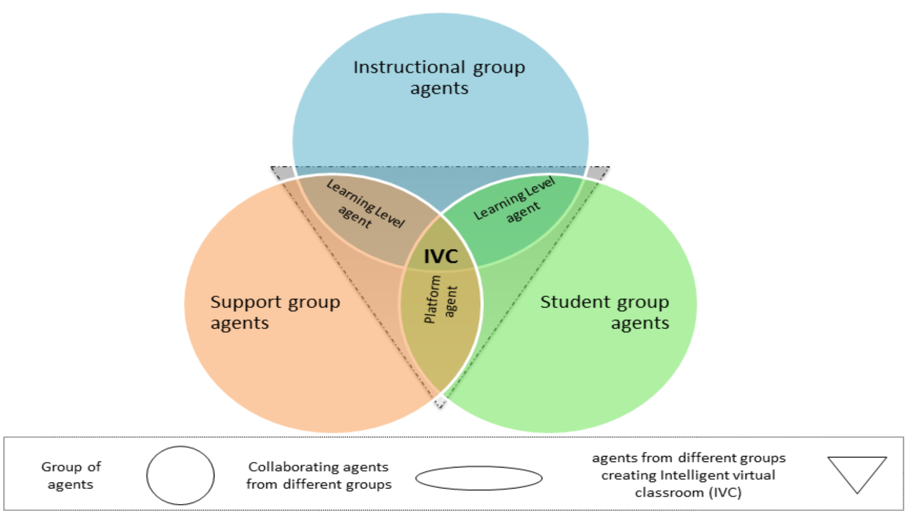

Fig. 3. MASeL: Intelligent Virtual Classroom

IV. TIMED-BASED MODEL CHECKING OF MASEL

J. Scenario-1: Active discussion session between students and teacher

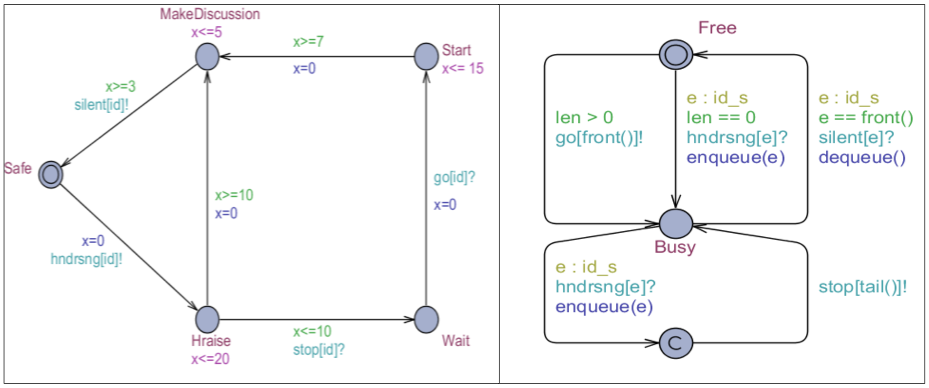

Fig. 4. UPPAAL: Timed-based automata model of active discussion session

K. Scenario-2: Student asking question in IVC

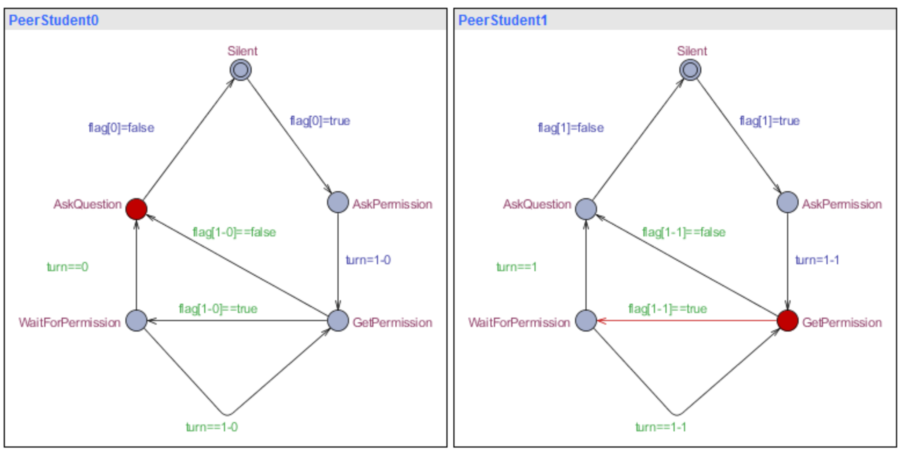

Fig. 5. UPPAAL: Timed-based automata model of student asking question in IVC

L. Scenario-3: Use of White-board by participants of IVC

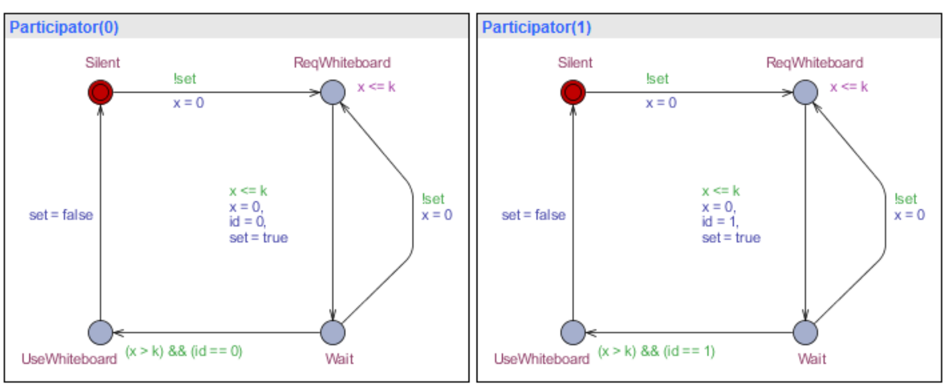

Fig. 6. UPPAAL: Timed-automata model of the use of white-board by IVC participants

M. MASEL Safety properties specified and verified in UPPAAL

Scenario	Properties satisfied by model checking	Description
White board	A[] forall (i:presenterid_t) forall (j:presenterid_t) Participator(i).UseWhiteboard && Participator(j).UseWhiteboard imply i==j	At any time instance there is only one individual using the White board (Mutual Exclusion requirement)
	A[] not deadlock	The system is deadlock-free
Two students	A[](not (PeerStudent0.AskQuestion and PeerStudent1.AskQuestion))	Mutex property
	A[] not deadlock	The system is deadlock-free
Active discussion session	E<> Teacher.Busy	Teacher can receive (and store in queue) messages from hand raising students
	E<> Student(0).Make_Discussion	Student(0) can make discussion with teacher
	E<> Student(0).Make_Discussion and Student(1).Wait	When student(0) makes discussion with teacher. Student(1) waits till student(0) ends discussion
	E<> Student(0).Make_Discussion and forall (i : id_s) i != 0 imply Student(i).Wait)	Student(0) can make discussion while the other students are waiting to start their own discussion
	A[] forall (i : id_s) forall (j : id_s) Student(i).Make_Discussion && Student(j).Make_Discussion imply i == j	At any given time instance there is never more than one student making discussion with the teacher
	A[] Teacher.list[N] == 0	There can never be N elements in the queue (thus the array will not overflow)
	Student(0).Hand_Raise --> Student(0).Make_Discussion	Whenever a Student(0) raises hand for a query, it will eventually entertained by teacher agent
	A[] not deadlock	The system is deadlock-free

V. Conclusion and Future Work

Education is of fundamental importance in a developing country like Pakistan. Due to the lack of funds for educational infrastructure, less-expensive alternatives should be proposed to promote education. E-learning is one of the cheapest solutions for spreading education. And a correct e-learning system is fundamental. A model of multi-agent e-learning system that is formally specified and its correctness properties are verified is prorposed in this paper. An interactive correct working model of the system is proposed. This model can be translated into a full working implementation of the system.

References

[1] G., Abrami, O., Barreteau, F., Cernesson. *"An Agent-Group-Role based modelling framework for participative water management support"*. In: *Proc. of the iEMSs 3.* s.l.:s.n., pp. 497-502, 2002.

[2] N. Amla, X. Du, A. Kuehlmann, R. Kurshan, and K. McMillan. "An analysis of SAT-based model checking techniques in an industrial environment". In *CHARME'05 Proceedings of the 13 IFIP WG 10.5 international conference on Correct Hardware Design and Verification Methods.* Springer-Verlag Berlin, 2005

[3] C., Badica, Z., Budimac, H. D. Burkhard, M. Ivanović. "*Software agents: languages, tools, platforms". Computer Science and Information Systems/ComSIS,* 8(2), pp. 255-298, 2011.

[4] F. M. Batista, M. G. B. Marietto, G. C. O. Barbosa, R. S. Franca, E. A. Noronha. "Multi-Agent Systems in a Computational Environment of Education: A Chatterbot Case Study". *International Journal for Infonomics (IJI) (Online),* 3(3), pp. 285-295, issn: 17424712, 2010.

[5] Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, Ph. Schnoebelen. *Systems and Software Verification: Model-Checking Techniques and Tools*. Springer-Verlag, 2001.

[6] F. Bellifemine, A. Poggi, G. Rimassa, "Jade. A FIPA-compliant agent framework", *In fourth international conference on Practical Applications of intelligent Agents and Multi-agent technology (PAAM'99)* (pp. 97-108), 1999.

[7] R. H. Bordini, M. Fisher, W. Visser, M. Wooldridge, "Verifying multi-agent programs by model checking". *Autonomous agents and multi-agent systems*, 12(2), pp. 239-256, 2006.

[8] P. Brusilovsky, S. Elmar, and W. Gerhard, "ELM-ART: An intelligent tutoring system on World Wide Web". In: *Intelligent tutoring systems.* Berlin Heidelberg: Springer, pp. 261-269, 1996.

[9] P. Brusilovsky, S. Ritter, E. Schwarz, "Distributed intelligent tutoring on the Web". Kobe, Japan, *8th World Conference of the AIED Society*, pp. 482-489, 1997.

[10] Cabri, G., Leonardi, L., Ferrari, L. and Zambonelli, F., "Role-based software agent interaction models: a survey". *Knowledge Engineering Review*, 25(4), pp. 397—419, 2010.

[11] Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. "NuSMV version 2: An opensource tool for symbolic model checking". *In Proc. International Conference on Computer-Aided Verification (CAV 2002)*, Volume 2404 of Lecture Notes in Computer Science, pp. 359-364, Copenhagen, Denmark, 2002. Springer Berlin Heidelberg. http://dx.doi.org/10.1007/3-540-45657-0_29

[12] E. M. Clarke, E. A. Emerson, "Design and synthesis of synchronization skeletons for branching time temporal logic". *In Logics of Programs* (pp. 52-71). Yorktown Heights, NewYork: LNCS volume-131, May 1981.

[13] E. Clarke, O. Grumberg, D. Peled, D., *Model Checking*. MIT press, 1999.

[14] E. M. Clarke, E. A. Emerson, A. P. Sistla, "Automatic verification of finite state concurrent systems using temporal logic specifications". *ACM transactions Prog. Lang. Syst.*, 8(2): 244-263, 1986.

[15] E. M. Clarke, O. Grumberg, D. E. Long, "Model checking and abstraction". *ACM Transactions Prog. Lang. Syst.*, 16(5):1512-1542, 1994.

[16] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, H. Veith, "Counter example-guided abstraction refinement for symbolic Model Checking". *Journal ACM*, 50(5): 752-794, 2003.

[17] G. Conole, M. d. Laat, T. Dillon, J. Darby, "Disruptive technologies, pedagogical innovation: What's new? Findings from an in-depth study of students' use and perception of technology". *Computers & Education* , 50(2), p. 511–524, 2008.

[18] J. Ferber, O. Gutknecht, "Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems", *ICMAS 98 (International Conference on Multi-Agent Systems)*, Paris, Y. Demazeau (ed), IEEE Press, pp. 128-135, 1998.

[19] J. Ferber, T. Stratulat, J. Tranier, "Towards an integral approach of organizations in multi-agent systems: the MASQ approach". *Multi-agent Systems: Semantics and Dynamics of Organizational Models in Virginia Dignum (Ed), IGI*, 2009.

[20] C. E. Georgakarakou, A. A. Economides, "Software Agent Technology: an Overview Application to Virtual Enterprises". *In: N. Protogeros, ed. Agent and Web Service Technologies in Virtual Enterprises.* New York: IGI-Global, pp. 1-24, 2008.

[21] D. Gheorghiu, L. Stefan, A. Rusu, "E-Learning and the Process of Studying in Virtual Contexts". *In: M. I. a. L. C. Jain, ed. E-Learning Paradigms and Applications - Agent-based Approach (Studies in Computational Intelligence 528).* Berlin Heidelberg: Springer-Verlag, pp. 65-95, 2014.

[22] C. Ghezzi, M. Jazayeri, D. Mandrioli, *Fundamentals of Software Engineering*. 2nd edition, Prentice-Hall, 2003

[23] D. Giannakopoulou, J. Magee, J. Kramer, "Fairness and priority in progress property analysis". *Technical report*, Department of Computing, Imperial College of Science, Technology and Medicine, 1999.

[24] O. Gutknecht, J. Ferber, "MadKit: A generic multi-agent platform". In: *Proceedings of the fourth international conference on Autonomous agents.* s.l.:ACM, pp. 78-79, 2000.

[25] O. Gutknecht, J. Ferber, "The madkit agent platform architecture. In: *Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent.* s.l.:Springer, pp. 48—55, 2001.

[26] G. J. Holzmann, "The model checker SPIN". *IEEE Transactions on Software Engineering*, 23(5), pp. 279-295, 1997.

[27] N. R. Jennings, M. Wooldridge, "Agent-oriented software engineering". In: *Multi-Agent System Engineering.* s.l.:Springer, pp. 1-7, 1999.

[28] J. P. Katoen, "Concepts, Algorithms, and Tools for Model Checking". *Lecture Notes of the course Mechanized Validation of Parallel Systems*, 1999.

[29] P. Lam, C. McNaught, J. Lee, M. Chan, "Disciplinary difference in students' use of technology, experience in using eLearning strategies and perceptions towards eLearning". Computers & Education, Volume 73, pp. 111-120, 2014.

[30] K. G. Larsen, P. Pettersson, W. Yi, "UPPAAL in a nutshell". *Software Tools for Technology Transfer*, vol-1: pp.134-153, 1997.

[31] J. Leng, J. Li, L. C. Jain, "A role-based framework for multi-agent teaming". In: *Knowledge-Based Intelligent Information and Engineering Systems.* Springer: s.n., pp. 642-649, 2008.

[32] J. Magee, J. Kramer, *Concurrency: State Models & Java Programs.* New York: John Wiley and Sons, 2006.

[33] H. Mazyad, I. Tnazefti-Kerkeni, H. Basson, "A Multi-Agent System to Implement a Collaborative Learning Method". In: *ICIW 2013 : The Eighth International Conference on Internet and Web Applications and Services.* s.l.:s.n., pp. 266-271, 2013.

[34] M. T. Mitchell, "An architecture of an intelligent tutoring system to support distance learning". *Computing and Informatics*, 26(6), pp. 565-576, 2012.

[35] J. Odell, "*Agent Technology: An Overview"*. [Online] Available at: http://www.jamesodell.com/Agent_Technology-An_Overview.pdf [Accessed 2 February 2014], 2010.

[36] J. P. Quielle, J. Sifakis, "Specification and verification of concurrent systems in CESAR". *Proceedings of the 5th International Symposium on Programming*, (pp. 337-350), 1982.

[37] V. Trajkovic, D. Danco, K. Goran, Z. Petanceska, "Web-based virtual classroom". In: *Technology of Object-Oriented Languages and Systems (TOOLS).* s.l.:IEEE, pp. 137-146, 2000.

[38] M. Winikoff. "*Assurance of Agent Systems: What Role should Formal Verification play?"* Chapter 12 (pages 353-383) in *Specification and Verification of Multi-agent Systems*. Edited by Mehdi Dastani, Koen V. Hindriks, and John-Jules Meyer. ISBN:978-1-4419-6983-5, 2010.

[39] M. Wooldridge, "Agent-based software engineering". *IEE Proceedings-software*, 144(1), pp. 26-37, 1997.

[40] F. Zambonelli, N. R. Jennings, M. Wooldridge, "Developing Multiagent Systems: The Gaia Methodology". *ACM Transactions on Software Engineering and Methodology*, 12(3):317-370, 2003.

[41] N. Akhtar, M. Nauman, "Timed-Automata Based Model-Checking of a Multi-Agent System: A Case Study", *Journal of Software Engineering and Applications*, Vol. 8, pp. 43-50, 2015, DOI: 10.4236/jsea.2015.82006

Author Profile

Dr. Nadeem Akhtar is Assistant Professor at the Department of Computer Science & IT, The Islamia University of Bahawalpur (IUB), PAKISTAN. He has a PhD from the Laboratory VALORIA of Computer Science, University of South Brittany (UBS), FRANCE with the highest honour "*Tres Honorable*". He has a Masters with specialization in Information System Architecture from Institut Universitaire Professionnalisé (IUP), University of South Brittany, Vannes, FRANCE. He is the recipient of Research award in year 2014 from The Directorate of Research and Development, The Islamia University of Bahawalpur. His research areas are formal methods, formal verification, multi-agent systems, system-of-systems, bigdata and self-adaptive systems.