Mugla Journal of Science and Technology

ARCHITECTURAL DESIGN OF MULTI-AGENT MASSIVE OPEN ONLINE COURSES (MOOCs) WITH GAIA METHODOLOGY

Bekir AFŞAR[1], Rıdvan ATA[2]*

[1]Department of Computer Engineering, Faculty of Engineering, Muğla Sıtkı Koçman University, 48000, Muğla, Turkey
bekirafsar@mu.edu.tr
[2]Department of Computer Education and Instructional Technology, Faculty of Education, Muğla Sıtkı Koçman University, 48000, Muğla, Turkey
ridvanata@mu.edu.tr

Received: 08.09.2015, Accepted: 28.08.2015
*Corresponding author

Abstract

This study aims to illustrate an architecture of Massive Open Online Courses (MOOCs) structure from a perspective of Gaia methodology. MOOCs are current popular online teaching and learning trend in the fields of open and distance education. Software agents that come together to create the structure is called multi agent systems (MAS). The methodology of Gaia is based on the metaphor of organization in developing multi-agent systems. Limited researches have been conducted for the use of agents in the management or delivery of MOOCs that could provide improved design for MOOCs. This article presents an analysis of the architectural structure of MOOCs with multi-agent systems and sets goals for further research. The study proposes a design that includes the use of multi-agent based software systems with the aim to improve design, delivery, assessment and personalized management of MOOCs.
Keywords: Massive open online courses, Multi-agent systems, Gaia methodology

KİTLESEL AÇIK ÇEVRİMİÇİ KURS (KAÇK)'LARIN GAIA YÖNTEMİ İLE MİMARİ TASARIMI

Özet

Bu çalışma Gaia yöntemi perspektifinden Kitlesel Açık Çevrimiçi Kurs (KAÇK) yapılarının mimarisini göstermeyi hedeflemektedir. KAÇK'lar açık ve uzaktan eğitim alanlarında yaygın popüler çevrimiçi öğretme ve öğrenme eğilimleridir. Yazılım etmenlerinin bir araya gelerek oluşturdukları yapıya çok etmenli sistemler denilmektedir. Gaia yöntemi çok etmenli sistemlerin (ÇES) geliştirilmesinde organizasyon metaforunu temel almaktadır. KAÇK'ların yönetim ve kitlelere iletilmesinde KAÇK'lar için daha geliştirilmiş tasarım sağlayabilecek etmenlerin kullanımı konusunda sınırlı sayıda çalışma yürütülmüştür. Bu makale KAÇK'ların mimari yapılarını çok etmenli sistemler ile analizini sunmakta ve gelecek çalışmalar için hedefler belirlemektedir. Çalışma KAÇK'ların tasarımını, ulaştırılmasını, değerlendirilmesini ve kişisel yönetimini geliştirme amacı ile çok etmenli tabanlı yazılım sistemlerin kullanımını içeren bir tasarım önermektedir.
Anahtar Kelimeler: Kitlesel açık çevrimiçi kurs, Çok etmenli sistemler, Gaia metodolojisi

1 Introduction

The advancement and use of e-learning systems substantially increase because of innovations brought by current technology. E-learning system is an important area of research to increase the quality of education, to promote information sharing and to enhance dialogues between tutors and learners. For this purpose, Massive Open Online Courses (MOOCs) are developed that a large number of participants can benefit as online or distance via the internet [1]. Hundreds of educators and thousands of learners can involve in MOOCs. This massive structure of MOOCs cause some pedagogical and technical challenges. For instance, giving feedback to thousands of learners and assessing their works appear very challenging in terms of educators. In this case, MOOCs need to be transformed into a more intelligent and autonomous structures.

Various improvements in MOOCs have been seen through the use of software agents in the autonomous, intelligent, reactive and proactive structure [2]. Software agents that come together to create the structure is called multi agent systems (MAS). Various MAS architectures and software development methodologies are put forward to the design of MAS and their implementations. As MOOCs are intended to involve large numbers of learners, it is considered that they are naturally appropriate for a multi-agent structure.

In the literature, there are various MAS methodologies such as Gaia [3], Tropos [4], Adelfe [5], MaSE [6], Prometheus [7] and INGENIAS [8] that offer a variety of approaches and define development activities for the process of development of a MAS software, interactions between activities and how to performing these activities in order to the fulfillment of the requirements of the agent-based software systems. While some of these methodologies such as MAS-CommonKADS [9] is based on artificial intelligence directly, most of them (e.g. PASSI [10] and Prometheus [7]) either in the extension of direct object-based methodologies or in the structure of including both artificial intelligence and object-based approaches (e.g. Gaia [3], Adelfe [5], MaSE [6] and INGENIAS [8]).

Gaia that is a MAS development methodology [11] has used in the design and analysis of MOOC MAS system with the widespread acceptance in agent-based software development due to offering a set of system development steps to define agent organizational models and determine internal relations of system components with the widespread acceptance in

agent-based software development. Gaia is a MAS methodology that use the society organization metaphor and aim to produce interaction and coordination models to define social relations, dependencies between agents and agent roles due to organizational nature of MAS. As an example of implementation of Gaia for a conference management system was achieved by other studies [12]. This study attempts to illustrate an architecture for MOOCs from a perspective of Gaia. Limited researches have been conducted for the use of agents in the management or delivery of MOOCs that could provide improved design, delivery and assessment of MOOCs.

Remained sections of this paper is structured as follows. The Gaia methodology is introduced in Section 2. MOOCs are defined in Section 3. Section 4 explains how the analysis and design of a MAS MOOC with Gaia is performed. In Section 5, discussion and future work is presented.

2 Gaia Methodology

The methodology [11] of Gaia is based on the metaphor of organization in developing multi-agent systems. A software system in Gaia is considered as the whole of organizations including a number of agents that play one or more roles and interact with each other in order to achieve its various purposes. In addition agents of the system appear in an environment where they interact to achieve their objectives. Apart from roles, interactions and the environment, Gaia takes into account two other abstractions that are organizational rules and organizational structures [13]. There are 3 main phases of the Gaia methodology that includes modelling of these abstract structures: Analysis Phase, Architectural Design Phase and Detailed Design Phase. Detailed information is given about these phases and models that Gaia includes in the following subsections.

2.1 Analysis Phase

The aim of this phase is to reveal the premise structure of the organization in terms of roles of the organization, the relationships of these roles with each other and interactions between these roles. In this phase, a set of organizational rules are prepared with “environmental model”, “preliminary role model” and “preliminary interaction model”.

An environment that an agent settled in an environmental model is considered as the whole of abstract resources where agents can detect and then act. These resources may be information the agents hold. A simple graphical illustration is recommended to illustrate environment resources in Gaia.

Roles of an agent is defined with 4 features in Gaia. These are; responsibilities, permissions, protocols and activities. Responsibilities that define the functionality of a role are represented by “liveness” and “safety” features. “Liveness” refer to benefit of the agent assumes the role and its liveness on this basis. “Safety” refer to avoid damages from the system and always being in a state of acceptable. On the other hand, permissions refer to rights of roles by identifying information resource of roles that can be used or not. Activities of a role are atomic tasks of agents carried out without interacting with other agents. Lastly, protocols are defined as being special patterns that belong to interactions of a role with other roles [14].

Liveness features are identified with liveness expressions in Gaia. Operators that can be used in these expressions are given in Table 1 the general structure of liveness expression is as follows.

ROLE_NAME: expression

ROLE_NAME is the name of a role where liveness features are defined. Expression defines liveness features of this ROLE_NAME. Liveness expressions are given in the form of activities and protocols. Activities in the liveness expressions are shown as underlined. Safety features are specified with a predicate list.

Table 1. Operators for Gaia liveness expressions.

Operator	Interpretation
$x.y$	x followed by y
$x \mid y$	x or y occurs
x^*	x occurs 0 or more times
$x+$	x occurs 1 or more times
x^w	x occurs infinitely often
$[x]$	x is optional
$x \mid\mid y$	x and y interleaved

A role model consists of a set of role schema. In each role schema, characteristics that belong to a role mentioned above are illustrated.

Interaction protocols are defined in terms of characteristics and dynamics of the roles that interact in the premise interaction model which is a second output of the analysis phase. Defined protocol characteristics are the definition of protocol name, protocol initiator, protocol partner, inputs, outputs and textual protocol [11].

On the other hand, rules to be followed in the organization need to be determined in the analysis phase for system organization to be defined properly in the architecture phase. Organizational rules can be instantaneous or temporal by taking into account a moment in time or the time slot[b]. First-order temporal logic expressions are proposed to be used in Gaia for formal representation of organizational rules. Temporal connectives that can be used in representation of organizational rules are listed in Table 2. Temporal models use ○, ◊, □, *U*, *W*, *B* and symbols to indicate these time status respectively: next, in the end, always, until it is, without it is, previous.

Table 2. Temporal connectives.

Operator	Interpretation
$\circ\varphi$	φ is true next
$\Diamond\varphi$	φ is eventually true
$\Box\varphi$	φ is always true
$\varphi\ U\ \psi$	φ is true until ψ is true
$\varphi\ W\ \psi$	φ is true unless ψ is true
$\varphi\ B\ \psi$	φ is true before ψ is true

2.2 Architectural Design Phase

The primary aim of this phase is the determination of the most appropriate organization structure for functional characteristics of multi-agent system worked on and organization characteristics identified in the analysis phase of multi-agent system environment.

The organizational structure of the system is introduced in line with a topology and control regime. The most appropriate organizational structure of the agent for multi-agent system is aimed to be chosen by defining the organizational structure according to a topology. Control regime determines control mechanism and how the interactions between organization members are going to be. Distribution of work needs to be reasonable and at the same time system liveness and safety conditions need to be ensured due to limited information members in an organization can handle per unit of time. The organization structure is determined in accordance with above information. As the organization of multi-agent systems is important for the real world, this arrangement in Gaia is based on the Organization Theory introduced by Simon [15].

Once multi-agent system organizational structure is determined in this phase, preliminary role and interaction models are detailed and completed in the analysis phase. The

system designer determines which roles would interact with which roles (topology) and which protocols would be operated during these interactions (control regime).

2.3 Detailed Design Phase

After general architecture of the system with all roles and interactions is determined, agent and service models are prepared for the implementation of multi-agent system in this phase. In agent model agent categories, roles and instance quantities specified and assigned. As identical matching between agent categories and roles would be, more than one role can assigned to an agent category in terms of the efficiency of the system. At the same time, all services associated with agent roles are defined in the Service Model. Services can be considered as functional methods of agent categories in terms of the object based perspective. However, there are some differences from the conventional object oriented approach due to the autonomous structure and self-organization of agent systems. For each service, inputs, outputs, working preconditions and termination conditions of the service in Gaia are specified. These features are generated from the information involved in models, e.g. protocols, rules, etc. that are outputs of previous phases of the Gaia development.

3 Massive Open Online Courses (MOOCs)

The Massive Open Online Course (MOOC) is a current popular online teaching and learning trend based on the learner-cantered approach in the field of open and distance education. Key elements of a MOOC are being massive, open and online with more interactive ways [16]. A typical MOOC provides opportunities to deliver a higher education program as online for large numbers of learners. It can be said that the power of the MOOC is based on engaging participants actively and enabling to interact with each other. In this context, Coursera, Khan Academy, Udacity, EdX and FutureLearn are well-known pioneer platforms of MOOC.

The principles of MOOC base three distinctive characteristics which are Design, Delivery and Assessment. According to Siemens [17]; MOOCs are currently classified as xMOOCs, cMOOCs and quasi-MOOCs depending on their different pedagogical approaches (Design). Learning Management Systems (LMSs) store the data of thousands of participants from different origins (Delivery). Automated computer tools are facilitated to grade and assess participants' practices (Assessment).

It is this paper's intention that with the inclusion of Gaia, MOOCs can experience significant improvements in content quality, course delivery and assessments.

4 Analysis and Design of a Multi-Agent MOOC System with Gaia

In this study, the structures of MOOCs are investigated and delivering a course process is proposed. Steps of submitting a course proposal, evaluating the course proposals and adding to the system, delivering the content of the course added to the system to learners and assessing learners' process take place in the proposed system. Tutors send prepared course proposals to the course manager. The course manager direct course proposals to the reviewer assigner to be evaluated by the appropriate referee. The reviewer assigner send course proposals to be evaluated by appropriate referee according their speciality. The course reviewer evaluates course proposals and send the review to the decision maker. According to the review, the decision maker decides to include the course in the system. The course content prepared by the tutor is transferred to the course designer after adding the course. Pop-up quizzes are asked about the course content in the identified times and whether learners follow the course is traced. In addition, feedbacks are sent to learners about insufficient subject by analyzing answers given to the quizzes. In specified period of the course, learners are subjected to tests and their grades are determined according to these test results.

Software agents that will take place in MOOC systems are expected to display behaviors of people take part in such a system in the real world. It is almost impossible for a tutor to assess thousands of learners or personalization of the course content according to learners' educational background in the real world. For instance, a course designer identified above carried out as a software agent will be able to customize the course content based on students' educational background and/or according to their physical disability situations (This is beyond the scope of this study and not performed in the design). Similarly although asking pop-up questions through the course process and giving feedback to thousands of learners appear to be very challenging, a software agent can give feedback and asses each learner thanks to an answer key given.

Analysis and design of MOOCs architecture, which is introduced in Section 3, is fulfilled with the stages Gaia suggests for developing multi-agent software. Analysis, architecture design and detailed design is explained respectively in the following subsections.

4.1 Analysis

Analysis models and organizational rules that are introduced in subsection 2.1 are prepared for MOOCs based on guidance for a system analysis in Gaia methodology.

4.1.1 Environmental Model

In the environmental model, abstract information take place that agents would have and use. Resources considered to be used by agents included in MOOCs are course proposal, course content, test, quiz, feedback, assessment, answer, answer key, speciality, and review. Figure 1 illustrate the environmental model includes these resources. Tutors prepare course proposals and course contents for the courses that they want to register to the system. Course managers refer these proposals to review assigners. Appropriate reviewers that have already registered their specialities to the system are assigned and the assessment is carried out. Reviewed courses are added to the system. Course designer transfers the content of the courses that added to the system to the learner. Tutor assesses learners with tests and quizzes during the class process. The system gives feedback by evaluating answer of quizzes and tests received from learners and grade them with assessment at the end of the course.

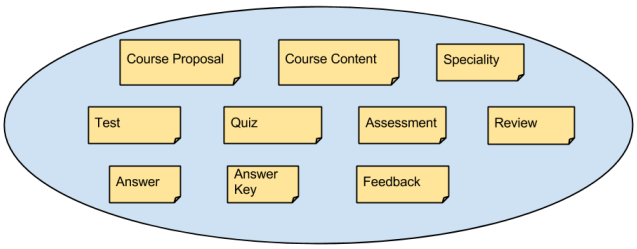
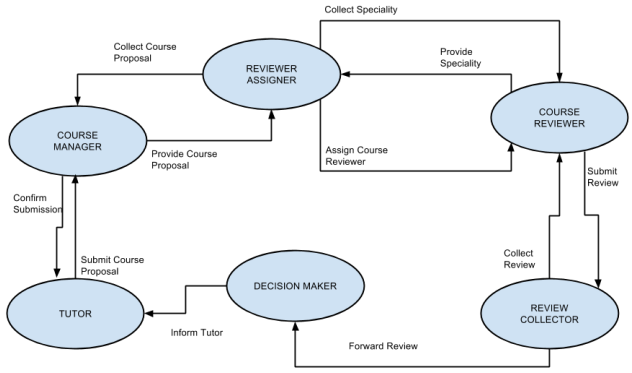


Figure 1. Environmental model of MOOC System.

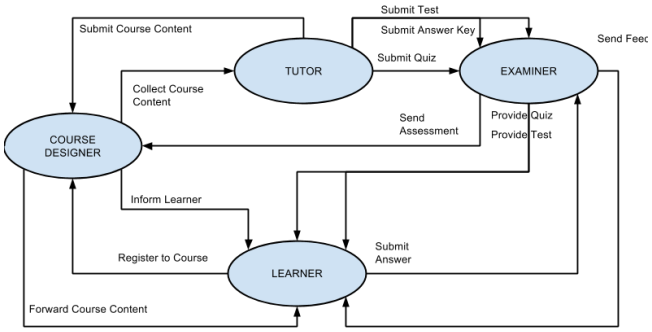
4.1.2 Preliminary Role and Interaction Model

The preliminary role that involve necessary roles and interactions to carry out the scenario explained in the environmental model and the interaction model is illustrated in Figure 2. This model provides a basis for the desired role model and interaction model in the stage of architectural design in

Gaia. When organizational roles and the structure is determined, role definitions and interactions turn out to be clearer. The preliminary role and the interaction models are framed in two parts in order to be clarified. In Figure 2-a, roles and interactions that are necessary for the role of tutor to register a course to the system are illustrated. In Figure 2-b, providing the course content to the role of learner from the role of tutor as well as roles and interactions required to evaluate the course are given.



a- Preliminary roles and interactions for course registration.



b- Preliminary roles and interactions for course registration.

Figure 2. Preliminary role and interaction model of MOOC System.

4.1.3 Organizational Rules

Zambonelli et al. [13] suggest three terms and a function to express organizational rules with primary temporal logic:

1. *plays* (*i*, *r*): agent *i* plays role *r*.
2. *initiate* (*r*, *p*): role *r* initiates *p* protocol.
3. *participate* (*r*, *p*): *r* role is involved to operate *p* protocol.
4. *card* (*r*) : it informs how many agents undertake *r* role.

In this study, necessary rules are defined for Tutors to suggest a course proposal, for reviews to evaluate and for learners to access the content after the course was added and assessment steps that are considered for MOOC. Parametric expression are used to determine the constraints identified below as required in the implementation phase. For instance, at least three referees can be asked for the review to approve a course or this number may also vary. Another system administrator may ask for 5 referees. Therefore this value is expressed parametrically.

Some of the organizational rules that are prepared to use in designing the system are:

1. Each course proposal (c) is evaluated by at least n referee:
   $\forall c.card\ (REVIEWER(c)) \geq n$
2. A Tutor (t) cannot evaluate his/her own course proposal:
   $\forall t.\forall c.plays\ (t, TUTOR(c)) \Rightarrow \Box\neg plays\ (t, REVIEWER(c))$
3. A Tutor cannot enroll his/her own course as learner:
   $\forall t.\forall c.plays\ (t, TUTOR(c)) \Rightarrow \Box\neg plays\ (t, LEARNER(c))$
4. A Learner cannot administrate the course enrolled as Tutor:
   $\forall l.\forall c.plays\ (l, LEARNER(c)) \Rightarrow \Box\neg plays\ (l, TUTOR(c))$
5. The same course proposal should not be sent more than one time to a referee to evaluate (safety property):
   $\forall r.\forall c.plays\ (r, REVIEWER(c)) \Rightarrow \Box\neg plays\ (r, REVIEWER(c))$

Some of the organizational rules prepared to operate protocols are given below:

1. If a course proposal is taken, the proposal must be evaluated (liveness property):
   $\forall r.\forall c.participate\ (r, AssignReviewer(c)) \Rightarrow \Diamond\ initiate\ (r, ProvideReview(c))$
2. Examiner cannot send an assessment the students who have not enrolled the related courses (safety property):
   $\forall e.\forall c.participate\ (e, RegisterCourse(c))\ B\ initiate\ (t, ProvideAssessment(c))$
3. The number of assessments sent for a course cannot be larger than the number of students enrolled in the course (safety property):
   $\forall c.SubmittedAssessments(c) \leq card\ (LEARNER(c))$

4.2 Architectural Design

In architectural design, the appropriate organizational structure for MOOCs is determined and the preliminary and interaction model which were introduced in the analysis phase are detailed and completed.

4.2.1 Organizational Structure

The multi-level hierarchical organization structure is considered as appropriate for a MOOC where thousands of learners can register and hundreds of tutors can propose a course that were evaluated by referees and involved in the system. MOOC Manager is considered as the administrator. Reviewer assigners who direct the course proposals to referees, review collectors who collect the evaluations from referees, decision makers who decide whether a course is given according to evaluations from referees, course managers who deal with course process management are considered as MOOC members. MOOC members operate necessary procedure by interacting reviewers and tutors. This hierarchical structure can be seen in Figure 3.

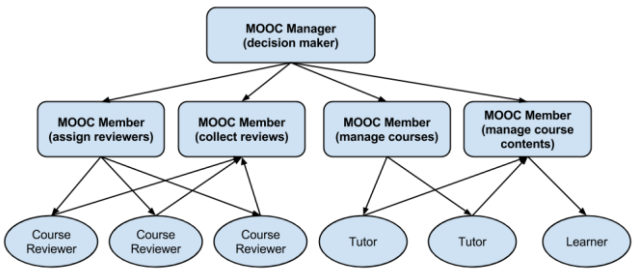


Figure 3. Organizational Structure of MOOC System.

4.2.2 Role Model

A multi-level hierarchical multi-agent organizational structure considered for MOOCs is compatible with the preliminary role and interaction model carried out in the analysis phase. Some

roles that were not involved in the organizational structure are going to be performed by autonomous software agents. In this context, roles initially considered were finalized. These roles are; Course Manager, Reviewer Assigner, Course Reviewer, Review Collector, Decision Maker, Tutor, Course Designer, Examiner and Learner. Appropriate role schemas are prepared for each role in the architecture design. Properties such as liveness, safety etc. belong to roles are prepared in these schemas according to notation explained in Section 2.1.In role schemas there are role name, its description, protocols and activities, permissions and responsibilities. Actions that are embodied by the role are expressed in activities and written underlined. Interactions in other roles are displayed with protocols. Authorities of roles on resources in the environmental model at permission part are seen. In the responsibilities part, liveness and safety features of that role take place. The role schemas for the Tutor and Examiner roles are seen in Figure 4.

Role Schema: TUTOR

Description:
This role is responsible for generating course proposals, course contents, tests, answer keys and quizzes.

Protocols and Activities:
GenerateCProposal, AwaitConfirmation, GenerateCContent, GenerateTest, GenerateAnswerKey, GenerateQuiz, SubmitCProposal, SubmitCContent, SubmitTest, SubmitAnswerKey, SubmitQuiz

Permissions:

changes	*course proposal*	*// Only his/her own course proposal*
generates	*course content*	*// Only his/her own course content.*
generates	*test*	*// Only his/her own course tests.*
generates	answer key	*// Only his/her own tests answer keys.*
generates	*quiz*	*// Only his/her own course quizzes.*

Responsibilities:
Liveness:

TUTOR = (HandleCContent || HandleTest || HandleAnswerKey || HandleQuiz || GenerateCProposal.SubmitCProposal.AwaitConfirmation) ^{w}

HANDLECCONTENT = (GenerateCContent || SubmitCContent)
HANDLETEST = (GenerateTest || SubmitTest)
HANDLE_ANSWERKEY = (GenerateAnswerKey || SubmitAnswerKey)
HANDLEQUIZ = (GenerateQuiz || SubmitQuiz)

Safety:
- *course proposal* is successfully submitted.
- *course content* is successfully generated.
- *test* is successfully generated.
- *answer key* is successfully generated.
- *quiz* is successfully generated.

Role Schema: EXAMINER

Description:
This role is responsible for providing tests, quizzes and sending feedbacks and assessments.

Protocols and Activities:
GenerateFeedBack, GenerateAssessment, ProvideTest, ProvideQuiz, SendFeedBack, SubmitAssessment

Permissions:

generates	*feedback*	
generates	*assessment*	
reads	*answer*	*// All answers*
reads	answer key	*// All answer keys.*

Responsibilities:
Liveness:

EXAMINER = (ProvideTest || ProvideQuiz || GenerateFeedBack.SendFeedBack || GenerateAssessment.SubmitAssessment) ^{w}

Safety:
- *feedback* is successfully sent.
- *assessment* is successfully submitted.
- number of registered learners = number of *assessments*

Figure 4. Role Schemas of Tutor and Examiner.

4.2.3 Interaction Model

Clarified protocols in the role model, that is, details of interactions between roles take place in the interaction model. A protocol started by which role, its partner role, inputs, outputs and description are illustrated in the interaction model. Interactions between roles took place in MOOCs design have been developed in the context of this study. In Figure 5, interaction model of TUTOR and EXAMINER roles that hold an important place in the design were illustrated.

The role of TUTOR is to interact with the roles of COUCE MANAGER, DECISION MAKER, COURSE DESIGNER and EXAMINER. The responsibilities of this role are creating course proposal, course content, tests, quizzes and applying for a course proposal. TUTOR send the course proposal to COURSE MANAGER via SubmitCProposal protocol. TUTOR is informed about confirmation of submitting the course proposal with the ConfirmSubmission protocol. The decision after revision of the course proposal is sent to the TUTOR role by the role of DECISION MAKER with InformTutor protocol. If the course has been added at the end of course review process, COURSE DESIGNER role asks for the course content from the role of TUTOR with CollectCContent protocol. The role of TUTOR presents the course content to the role of COURSE DESIGNER via ProvideCContent protocol.Then the role of TUTOR delivers tests, quizzes and answer keys to be used assessing learners to the EXAMINER role with SubmitTest, SubmitQuiz and SubmitAnswerKey protocols. The role of EXAMINER presents tests and quizzes received to the role of LEARNER via ProvideTest and ProvideQuiz protocols. The role of LEARNER transmits answers of tests and quizzes to the EXAMINER role via SubmitAnswer protocol. According to answers of quizzes the role of EXAMINER gives feedbacks to the role of LEARNER via SendFeedBack protocol. According to answers of testes assessment is fulfilled and this evaluation is delivered to the role of COURSE DESIGNER with SubmitAssessment protocol.

Protocol Name	Initiator	Partner	Inputs	Outputs	Description
SubmitCProposal	Tutor	CourseManager	Course proposal	Confirmation details	Signals the submission of a course
ConfirmSubmission	CourseManager	Tutor	Tutor details	-	Confirms the receipt of a course submission
InformTutor	DecisionMaker	Tutor	End of decision period	-	Informs the acceptance/rejection of a course
CollectCContent	CourseDesigner	Tutor	End of course review period	Course Contents	Requests the course content from the Tutor
SubmitCContent	Tutor	CourseDesigner	CourseDesigner details	-	Indicates contents of the courses
SubmitTest	Tutor	Examiner	Examiner details	-	Indicates tests prepared by the tutor
SubmitQuiz	Tutor	Examiner	Examiner details	-	Indicates quizzes prepared by the tutor
SubmitAnswerKey	Tutor	Examiner	Examiner details	-	Indicates answer keys prepared by the tutor
ProvideTest	Examiner	Learner	Learner details	Test details	Shows the test questions to the Learner
ProvideQuiz	Examiner	Learner	Learner details	Quiz details	Shows the quiz questions to the Learner
SubmitAnswer	Learner	Examiner	Answers	-	Learner answers the test and quiz questions
SendFeedBack	Examiner	Learner	Learner details	Feedback	Feedbacks for the answers
SubmitAssessment	Examiner	CourseDesigner	End of the assessment period	Assessment information	Indicates the assessments completed by examiners

Figure 5. Interaction Model of Tutor and Examiner Roles.

4.3 Detailed Design

In the detailed design phase agent model and service models are formed for multi-agent MOOCs of which analysis and architectural design has been completed in the light of information described in Section 2.3.

4.3.1 Agent Model

9 different roles are determined as the result of analysis and architecture design of MOOCs following Gaia methodology. 6 software agent that play 9 different roles determined in the Agent model (Figure 6) have been identified.

The role of DECISION MAKER is the role of deciding if the course has been added according to reviews of referees after peer review process and this role will be played by MOOC Manager Software agent. Roles of COURSE MANAGER, REVIEW COLLECTOR, REVIEWER ASSIGNER, COURSE DESIGNER, TUTOR and COURSE REVIEWER will be played by MOOC Member software agent in line of the organizational rules. In the considered MOOC system, roles of TUTOR and COURSE REVIEWER will be played by Tutor and Course Review software agents respectively as non-members referees and tutors of MOOC to take part in the system.

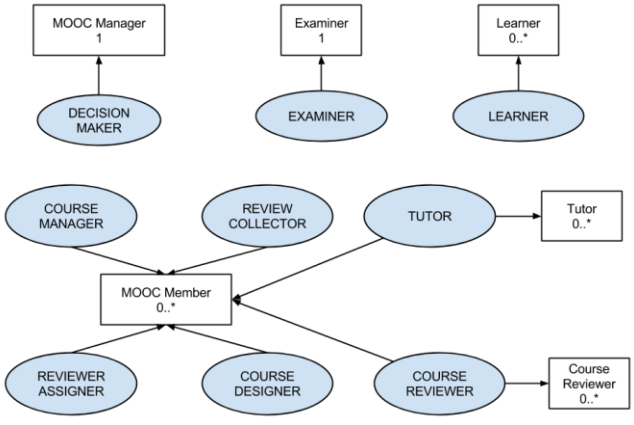


Figure 6. Agent model of MOOC system.

4.3.2 Service Model

A service model has been prepared for each role in the MOOC system. Inputs and outputs of activities determined for roles in the role model, precondition and termination conditions are located in the service model. Due to its importance in the MOOC system, service models of the role of EXAMINER are given in Figure 7.

Important services in the role of EXAMINER are services of Generate FeedBack and Generate Assessment. In the Generate FeedBack, feedbacks are formed according to answers of

quizzes incoming from the role of LEARNER. To the start of this service, answers of quizzes should be given by the role of LEARNER. The termination condition of this service is that feedback has formed successfully. In the service of Generate Assessment, an evaluation is formed according to test results. Precondition of this service is that the role of LEARNER answers tests. This service will be terminated only if the assessment has formed successfully. Services of Provide Test and Provide Quiz is used to present quizzes and tests received from the TUTOR role to the role of LEARNER. Tests and quizzes information from the TUTOR are necessary in order to start the these services and successful arrival of texts and quizzes by the role of LEARNER is necessary for termination of these services.When the service of Generate FeedBack terminates, received feedback is sent to the role of LEARNER with Send FeedBack. When the service of Generate Assessment terminates, received assessment is sent to the role of COURSE DESIGNER with Submit Assessment service. The last service of the role of EXAMINER is used to ensure that assessments are formed for all students enrolled.

Service	Input	Output	Pre-condition	Post-condition
Generate FeedBack	*answer* details	*feedback*	*answer* for quizzes is submitted by Learner	*feedback* is successfully generated.
Generate Assessment	*answer* details	*assessment*	*answer* for tests is submitted by Learner	*assessment* is successfully generated
Provide Test	-	*test*	A *test* is provided by Tutor	Learner is taken the *test*
Provide Quiz	-	*quiz*	A *quiz* is provided by Tutor	Learner is taken the *quiz*
Send FeedBack	-	*feedback*	*answer* for quizzes from the Learner are analyzed.	Learner is given the *feedback*
Submit Assessment	-	*assessment*	*answer* for tests from the Learner are analyzed.	Course Designer is given the *assessment*
Ensure all registered learners are assessed	number of registered learners, number of *assessments*	-	True	number of registered learners = number of *assessments*

Figure 7. Service Model for EXAMINER role.

5 Conclusion and Future Work

It is considered that MOOCs are going to be used widely in the future where thousands of learners and tutors across the world come together for education. We believe that multi-agent systems could be solution for MOOCs that embody various challenges besides a lot of advantages brought.

In this study, how a MAS methodology that uses the organization metaphor can be used in analysis and design of MOOC organization is illustrated. Analysis and design steps proposed by Gaia methodology for MOOCS have been implemented and product obtained were shared. We believe that qualified course contents, course process and effective assessment can be delivered with the proposed MOOC MAS system.

In the future work, performing the designed developed in this study is aimed to put into life. Investigation of studies for MAS designed by Gaia in the literature and proving MOOC MAS with proposed process is aimed. In addition, personalization of the course content for each learner who may have different level of educational background, design details of delivering the course content based on learners physical disabilities, if any, and attaining a more intelligent structure of assessment process are considered to be added.

6 References

[1] Kop, R., Fournier, H., and Mak, J. S. F., "A pedagogy of abundance or a pedagogy to support human beings? Participant support on massive open online courses.", *The International Review Of Research In Open And Distributed Learning*, Vol. 12, no. 7, pp. 74-93, 2011.

[2] Daradoumis, T., Bassi, R., Xhafa, F., and Caballé, S., "A review on massive e-learning (MOOC) design, delivery and assessment.", *In P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on,* 2013, pp. 208-213.

[3] Zambonelli, F., Jennings, N.R. and Wooldridge, M., "Developing multiagent systems: The Gaia methodology", *ACM Transactions on Software Engineering and Methodologies*, Vol. 12, no. 3, pp. 317-370, 2003.

[4] Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. and Mylopoulos, J., "Tropos: An agent-oriented software development methodology", *Autonomous Agents and Multi-Agent Systems*, Vol. 8 no. 3, pp. 203-236, 2004.

[5] Bernon, C., Gleizes, M-P., Peyruqueou, S. and Picard , G.,"ADELFE: A methodology for adaptive multi-agent systems engineering", *Lecture Notes in Artificial Intelligence*, Vol. 2577, pp. 70-81, 2003.

[6] DeLoach, S.A., Wood, M.F. and Sparkman, C.H., "Multiagent systems engineering", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 11, no. 3, pp. 231-258, 2001.

[7] Padgham, L. and Winikoff, M., "Prometheus: A methodology for developing intelligent agents", *Lecture Notes in Computer Science*, Vol. 2585, pp. 174-185, 2002.

[8] Pavon, J., Gomez, J. ve Fuentes, R., "The INGENIAS Methodology and Tools", *Agent-Oriented Methodologies*, Idea Group Publishing, pp. 236-276, 2005.

[9] Iglesias, C.A., Garijo, M., Gonzalez, J.C., and Velasco, J.R., "Analysis and design of multi-agent systems using MAS-CommonKADS", *Lecture Notes in Artificial Intelligence*, Vol. 1365, pp. 313-326, 1998.

[10] Cossentino M., "From Requirements to Code with the PASSI Methodology", *Agent-Oriented Methodologies*, Idea Group Publishing, pp. 79-106, 2005.

[11] Zambonelli, F., Jennings, N.R. and Wooldridge, M., "Multi-Agent Systems as Computational Organizations: The Gaia Methodology", *Agent-Oriented Methodologies*, Idea Group Publishing, pp. 136-172, 2005.

[12] Tamersoy, M., Afsar, B., Erata, F., ve Kardas, G., "Gaia ile Çok-Etmenli Konferans Yönetim Sistemi Analiz ve Tasarımı" *4. Ulusal Yazılım Mühendisliği Sempozyumu (UYMS 2009)*, 2009, ss. 83-90.

[13] Zambonelli, F., Jennings, N.R. and Wooldridge, M., "Organisational rules as an abstraction for the analysis and design of multi-agent systems", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 11, no. 3, pp. 303-328, 2001.

[14] Wooldridge, M., Jennings, N. R. ve Kinny, D. "The Gaia methodology for agent-oriented analysis and design", *Journal of Autonomous Agents and Multi-Agent Systems*, Vol. 3, no. 3, pp. 285-312, 2000.

[15] Simon, H. A., "Models of man: social and rational", *Wiley Publishing*, New York, 287 p., 1957.

[16] Haggard, S., Brown, S., Mills, R., Tait, A., Warburton, S., Lawton, W., and Angulo, T., "The Maturing of the MOOC: literature review of massive open online courses and other forms of online distance learning.", Department for Business, Innovation and Skills, UK Government, 2013.

[17] Siemens, G., "Massive open online courses: Innovation in education.", *Open educational resources: Innovation, research and practice*, 5, 2013.