Intelligent Pedagogical Agents for Teaching OBJECT Oriented Programming Concepts

Felicia GÎZA-BELCIUG

pp. 281-286
Intelligent Pedagogical Agents for Teaching OBJECT Oriented Programming Concepts

Felicia GÎZĂ-BELCIUG

Abstract
The wide spread of mobile devices encouraged the evolution of techniques for distance education. Pedagogical Agents are agents that are designed for pedagogical purposes to support distance learning. Object oriented programming is the most used programming paradigm that is taught in all the universities with computer science profile. Teaching environments based on Multi-Agent architecture support the development of more interactive and adaptable systems. The objective of this paper is to discuss the feasibility of implementing Distributed Intelligent Learning Environments based on the Multi-Agents Architecture approach considering the requirements for an intelligent pedagogical agent.

Keywords
Intelligent pedagogical agent, multi-agent system, teaching object oriented programming.

1. Introduction
Pedagogical agents support human learning and acts like an intermediary in the interaction between the learner and the environment. They usually are simulated human-like interfaces modeling a character enacted by a computer. The intelligent behavior and multitude of functions of pedagogical agents increase the effectiveness of the learning process. They adapt their behavior to the dynamic state of the learning environment, taking advantage of learning opportunities as they arise. They can support collaborative learning as well as individualized learning, because multiple students and agents can interact in a shared environment.

Goyal M. (2011), Soliman M., Guet C. (2010), Bădică et al. (2013) and many other researchers has designed and developed intelligent systems for teaching purposes, first of them considering also the field of object oriented programming as the objective of the learning environment. Soliman M., Guet C. (2010) consider the using of multi-agent system model in 3D Virtual Worlds. They present a conceptual view of the intelligent pedagogical agents.

1 Electrical Engineering and Computer Science Faculty, Stefan cel Mare University, Suceava, Romania, email: felicia@eed.usv.ro.
working in an immersive Virtual Learning Environment to reflect new possible scenarios for learning in the environment and to provide intelligent pedagogical functions in the immersive environment.

Usually, when a student has background experience in structured programming, teaching object oriented concepts is resumed on comparing those two programming paradigms. Learning object oriented programming as a first programming technique may be quite difficult and using pedagogical agents may be more suitable than in previous case. Sarkawi et al. (2008) designed an animated pedagogical agent for teaching object oriented programming in Java and performed a case study about applying this pedagogical agent in the university they work. All considered parameters are above average so they conclude that students were satisfied by the use of agents. Hans van der Meij et al. (2015) designed and tested a motivational animated pedagogical agent (APA) in an inquiry learning environment on kinematics.

2. Agent architecture

Feituri M., Funghi F. (2010) identifies the main roles for pedagogical agent:

**Agent as Expert** - Experts exhibit mastery or extensive knowledge and perform better than the average within a domain.

**Agent as Motivator** - The Motivator suggests his own ideas, verbally encourages and stimulates the learners.

**Agent as Mentor** - An ideal human instructor provides guidance for the learner to bridge the gap between the current and desired skill levels.

Starting from this roles, we designed an architecture of multi-agent system that addresses distance learning in an intelligent way. Specialized agents collaborate to reach the overall goal: student to have desired knowledge in the domain of object oriented programming. JADE multi-agent framework is considered for the implementation as it is the most active in the agent systems community. JADE architecture involves running a middleware for running agents to be executed both on the learning environment central server and on student computer. This middleware consists of multiple containers of agents and they can be of two types:

**Main Container** - is the container that keeps tracking of the other containers and of the all agents in the system. It contains all the agents involved in the teaching process: PedagogicalAgent, KnowledgeAgent, MotivationAgent, AssessmentAgent and also the SupervisorAgent that acts like an supervisor for all the agents in the system.
Remote Container- is the container that is tied to the main container and is running on student's computer.

Figure 1. Multi-agent architecture for Intelligent Learning Environments

SupervisorAgent keeps track of the agent users that are joining to the course, act as a communication agent, dynamically show the list of online students within a course.

The StudentAgent will run in a remote container that is connected to the main container where PedagogicalAgent is running.

The PedagogicalAgent will communicate with KnowledgeAgent and MotivationalAgent to provide an effective teaching. He will have to change his teaching styles to that of a mentor, tutor or facilitator when necessary.

AssessmentAgent will be used for assessing student performance using different assessment strategies.

An important step in this teaching process is getting user profile (like background in programming languages, or any programming knowledge) to configure user agent's first state. This information will be asked by pedagogical agent that will changing the course content to accommodate the student needs and expectations. A list of relevant questions has to be developed to model as close as possible the user programming profile and teaching preferences.

JADE system contains a special PedagogicalAgent responsible for each teaching strategy and for the domain knowledge retrieval. The StudentAgent monitors all actions of the student accessing data. The PedagogicalAgent updates the student history, when required, and sends the data to be updated in the StudentAgent.
3. Overview of the POO course

ACM and IEEE are continuously developing, revising, and refining a set of recommendations, known as Computer Science Body Knowledge - CSBK, to help the academic educators to design and further adapt Computer Science curricula by incorporating the most recent results of Computer Science research and by addressing the new market requirements for Computer Science professionals. The last version of CSBK was issued in December 2013 (Computer Science Curricula 2013).

In developing Object Oriented Programming (OOP) course we took into consideration the definition of course learning objectives and the availability of course prerequisites.

Firstly, the course learning objectives were formulated in accordance with recommendations provided by CSBK on topics that were considered relevant for OOP:

**Objective1:** To introduce the principles and concepts of OOP.

**Objective2:** To provide an opportunity to obtain practical experience in applying these concepts for programming small-scale software applications.

Secondly, the course prerequisites were established taking into account the current structure of the Computer Science curricula. The students must be familiar with structural programming style, theory and practice of Computer Programming.

Thirdly, a course structure was defined including lectures, labs, and project activities.

4. Intelligent Pedagogical agents for teaching OOP

A student that is firstly attending the course will be asked to configure his user profile. It will contain relevant questions about his learning style (especially what is his preferred style for introducing new definition and concepts (graphically, movies, stories)), about his programming background, and also about his capabilities of understanding abstract concepts. This information will be hold by *StudentAgent* and they will be used by *PedagogicalAgent* for assuring following features in the teaching process:

Adaptation: he will adapt the lesson plan accordingly, based on student's learning style and also his understanding level,
Motivation: he will offer encouragement to the students and give them feedback. With the help of MotivationalAgent, he should assist a discouraged student before s/he drops out.

Engagement: based on student's hobbies or areas of interests he will communicate with KnowledgeAgent to get more examples for modeling OOP concepts so its final goal to be accomplished.

Evolvement: based on student's programming background he will provide the new concepts of OOP at the right time. The student should feel at each moment that he is evolving.

KnowledgeAgent acts as a smart search engine, finding specific resources to solve learning needs. Depending on the level of its complexity, it could ”learn” and become more expert and useful as it provides more assistance to the PedagogicalAgent.

PedagogicalAgent monitors the learning process and collaborates with AssessmentAgent for elaborating assessments adapted to the course content that was taught to the student.

Conclusions

In this paper we propose a multi-agent architecture starting from the intelligent pedagogical agent requirements for teaching OOP. Most of the courses for teaching OOP concepts are simple tutorials with fixed examples that has to be watched by students. Using Pedagogical Agents for teaching those concepts is like having your own teacher at home that knows you and explains you all the concepts with examples adapted to your level of experience and interests. The described system meets most of the requirements necessary for an intelligent learning environment: interactivity, adaptable lessons, robustness, direct monitoring of the learning process, empirical evaluation.

References


Volume 296 of the series Advances in Intelligent Systems and Computing pp 267-276