ПОСТРОЕНИЕ МОДЕЛИ ВЗАИМОДЕЙСТВИЯ УЧАСТНИКОВ ПЛАТФОРМЫ МОПС

Лисицин Ян Васильевич, Минева Анастасия Владимировна
(Иркутск, Национальный исследовательский иркутский государственный технический университет)

SIMULATION OF INTERACTION OF PARTICIPANTS IN THE PLATFORM MOPS

Yan Lisitsin, Mineva Anastasia
(Irkutsk, National Research Irkutsk State Technical University)

The article explores the techniques for building simulation systems. Considered popular programs for the simulation. Explains why the selected product AnyLogic. Describes the major problem for modeling the interaction of participants in the platform MoPS. Demonstrate ways to solve these problems

Стремительное развитие компьютерных технологий открывает новые возможности в области построения математических моделей и становится причиной появления нового инструмента - имитационного моделирования. Иммитационное моделирование активно используется в процессе разработки сложных систем. Имитационное моделирование необходимо, когда в системе преобладают случайные параметры, или в системе есть время, причинные связи, нелинейности [1, 11].

При помощи специализированных инструментов моделируется поведение будущей системы. Это дает возможность при проектировании, разработке архитектуры в режиме реального времени проанализировать работу системы, а также изменить ее параметры ―на лету”[2]. Такая наглядность обеспечивает для исследователя работу с моделью на интуитивном уровне посредством восприятия модели как реальной системы.

В данной статье будет рассмотрено применение иммитационного моделирования в процессе разработки платформы МОПС.

Платформа МОПС реализует образовательную сеть, которая объединяет участников через пиринговые технологии. Взаимодействие участников невозможно описать через математический анализ, так как не существует математической модели для описания поведения человека. Реальные эксперименты в учебных группах помогут вычислить только некоторые параметры для будущей системы.

На поведение и состояние МОПС влияют ее участники, а также другие факторы - состояние устройств, стабильность сети. Эти параметры дискретны и случайны. Участники не зависят друг от друга, каждый имеет свой характер поведения и предпочитаемые способы обучения. Также как и используемые устройства: разный расход аккумулятора, ограничения на память. Из этого следует, что каждый узел сети в системе - индивидуален. Поэтому глобальное поведение системы является результатом деятельности пользователей, устройств и средств передачи данных. Данное обстоятельство позволяет применить современные технологии иммитационного моделирования - агентное моделирование.

Агентное моделирование — это представление реального мира в виде отдельных, самостоятельных подсистем. Под подсистемами подразумевают агентов. Агенты обмениваются между собой информацией, которая может влиять на поведение другого агента. Как правило, централизованное управление отсутствует, так как каждый агент функционирует по своим законам[3].

Каждый участник платформы будет агентом. Характеристики поведения, модель устройства, тип связи будут параметрамм агента. Параметры могут задаваться по определенному закону распределения, либо на основе данных от реальных экспериментов в учебных группах.

Были определены следующие задачи моделирования в процессе разработки МОПС:

1. Определение устойчивости платформы в терминах автоматического управления, задание критериев устойчивости, управляющих воздействий.
2. Формирование правил (алгоритмов) для определения возможности пересылки между устройствами с учетом уровеня заряда батареи и объема памяти.
3. Формирование алгоритма выборки адресатов.
4. Формирование алгоритма валидации задач.
5. Определение эффективности учебного процесса в зависимости от формы развертывания платформы.
6. Формирование алгоритма и формулы пересчета рейтинга участника.

Целью моделирования взаимодействия пользователей платформы является проверка устойчивости разрабатываемой платформы, а также определение параметров, которые в дальнейшем будут применяться при конфигурировании платформы.

Для того, чтобы можно было сделать аналитический срез, модель разделена на абстрактные уровни:

Коллективный. Рассматривается каким образом агент ведет себя в окружении. Агент представляет собой участника образовательной сети. Он обладает своим собственным набором характеристик, позволяющих рассматривать агента как отдельную личность. Например, если агент большую часть времени «учиться», то больше генерирует задач. Если агент обладает «общительным» характером, то у него больше друзей.

Транспортный. В данном уровне будет учитываться работа устройств в общей сети: таймауты, которые будут задавать время для повторной рассылки системных сообщений; маршрут через промежуточные узлы и т.д.

Образовательный. Уровень будет отображать процессы создания образовательной единицы, ее валидацию и процесс проверки ответов.

Во время моделирования требуется применение анимации для вывода статистических данных моделирования в виде диаграмм и графиков. Были сформулированы требования к анимации:

- Динамическое отображение агента в зависимости от того, что он в данный момент выполняет.
- Отображение дружественных связей между агентами.
- Отображение гистограммы, которая показывает соотношения количества типов задач, которые в данный момент находятся у агента: количество посланных ответов и т.д.
- Анимация распространения и валидации задачи на примере. То есть показывать, каким образом отсылается задача, как происходит валидация, как выбирается маршрут передачи задачи. Другие задачи должны рассылаться также, только не отображаться. Это необходимо, чтобы отследить поведение и распространение одной задачи.

- Отображение полей общего информативного назначения: сколько задач всего, сколько всего в очереди, сколько не прошло валидацию и т.д.
- Построение графиков активности пользователей, отображающие количество человек, которые в данный момент времени «отдыхают», «читают», «решают».

Рассматривалась возможность создания собственного инструмента для агентного моделирования. Однако, достаточно сложно написать самостоятельно программу, которая реализует сложные формы анимации. За короткий срок существования агентной парадигмы в моделировании было разработано десяток продуктов, которые обладают большим количеством функций и возможностей. Самые известные из них: AgentSheets, Anylogic, Breve, Mason,NetLogo, StarLogo, StarLogo TNG, Swarm,Repast. [4,5].

Исходя из требований, обеспечить анимацию из вышеперечисленных продуктов может AgentSheets, Anylogic, NetLogo, StarLogo (см. Табл. 1)[6].

Таблица 1. Сравнение платформ.

Платформа	**Основное назначение**	**Требуемый язык программирования**	**Требуемая ОС**	**Поддержка пользователя**
AgentSheets	Моделирование обучения.	Visual AgenTalk; может быть экспортирован в Java.	Windows; Mac OS X; запускается на любой JVM.	Руководства пользователя; обучающие видео; FAQ; рекомендуемая литература по программированию и моделированию.
AnyLogic	Системная динамика; дискретно-событийное (процессное) моделирование; агентное моделирование.	Java; UML-RT	Windows Vista, x86-32; Windows XP, x86-32; Mac OS X 10.4.1 или вышел, Universal; SuSE Open Linux 10.2 или выше, x86-32; Ubuntu Linux 7.04 или выше, x86-32.	Демонстрационные материалы; обучение; консультации; база знаний; форум; документация.
StarLogo	Изучение закономерностей и феноменов, в которые вовлечено множество агентов.	StarLogo (an extension of Logo)	Mac OS X v10.2.6 or higher with Java 1.4 installed; Windows; Unix; Linux	Список рассылки; обучающие материалы; FAQ; bug list; документация; developer contacts
NetLogo	Моделирование ситуаций и феноменов, происходящих в природе и обществе(7).	NetLogo	Any Java Virtual Machine, version 1.4.1 or later, is installed. Version 1.5.0_12 or later is preferred.	Документация; FAQ; избранные публикации; обучающие материалы; third party extensions; defect list; списки рассылки

В России популярным инструментом является NetLogo из-за бесплатности и содержательной документации[8]. Однако, данный продукт обладает следующими минусами:

- агенты могут работать только в синхронном режиме;
- используется процедурный язык NetLogo;
- агенты строго привязываются ко времени. Могут выполнять действия только в определенные моменты времени. В промежутки между активностью агенты не способны действовать[9];
- отсутствует русскоязычная поддержка.

Anylogic, в отличие от NetLogo, обладает более широкими функциональными возможностями. Продукт позволяет исследовать динамику и поведение системы на любом

уровне абстракции, также как и в нашем случае, когда необходимо рассматривать поведение участников сети с разных уровней. Популярный объекто-ориентированный подход, реализуемый при построении модели, позволит задать поведение участников, а также добавлять классы для новых объектов взаимодействия участников в платформе. Для наглядного отображения статистики в программе предусмотрены различные элементы: графики, диаграммы, гистограммы[10].

Anylogic - отечественный продукт компании ООО "Экс Джей Текнолоджис". Интерфейс программы реализован на русском языке. Доступно большое количество документации на русском языке и русскоязычная поддержка. Для описания предметной области в Anylogic можно использовать удобный инструмент - активный объект. Активные объекты - это строительные элементы модели, которые могут использоваться для симуляции различных реальных объектов, ресурсов, процессов, людей и контроллеров.

Описание модели. Участник образовательной сети МОПС активное время работает с задачами: создаѐт новые задачи, читает, решает, либо ждет новые. Также проверяет присланные друзьями ответы на задачи. Участник может быть в неактивном состоянии. В реальном мире это может происходить, когда участник не пользуется платформой, например, выключает телефон с установленным приложением. Текущее состояние пользователя (то есть то, что он будет делать в текущий момент времени) определяется случайным образом. В дальнейшем планируется генерировать «характер» для каждого участника сети. «Характер» будет влиять на процессы обучения пользователя.

Связь между пользователями складывается за счет ―дружеских‖ отношений. В начале моделирования при генерации участников случайным образом выбирается, какой конкретно пользователь является другом. Дружеские отношения необходимы для персылки задач.

Когда пользователь создал задачу, случайным образом генерируется список друзей, которые становятся получателями задачи. После передачи, задача складываются в специальный массив входящих задач. Задачи хранятся там, пока пользователь не перейдет в состояние ―Решать задачи‖.

Входящий ящик имеет ограничение на количество задач. Если он переполнится или хозяин ящика находится не в сети, то задача не дойдет до места назначения. Такие задачи хранятся у специального звена в сети - суперпира. Суперпиром может являться как автор задачи, так и участник сети, через которого было передана задача. Суперпир хранит задачи, пока получатель не станет активным или пока в ящике для входящих задач не появится свободное место.

Для того, чтобы за решение задачи пользователь смог получить балл к своему рейтингу, исходная задача должна быть отвалидирована. Валидация различается в зависимости от типа задачи. Задачи бывают с ручной проверкой, с автоматической проверкой и теория. Если пользователь решает задачи с ручной проверкой, то его ответ отправляется автору задачи. Автор ждет, пока на разосланную задачу ответит определенной количество его друзей, и проверяет правильность решения, раставляя баллы для каждого задания. Среднее количество баллов, расчитанное для задачи, сравнивается с параметром ―Прохождение валидации‖. Например, если средний балл задачи больше максимального порога, это значит, что задача оказалась слишком легкой и валидацию не проходит. Если средний балл оказался ниже минимального порога, это значит, что задача оказалась слишком тяжелой и никто еѐ не смог решить.

Стоит отметить, что если задача с ручной проверкой, то все ответы отсылаются автору задачи, а если задачи с автоматической проверкой, то отсылается отправителю, потому что отправитель имеет ответы на задачу с автопроверкой.

Описание диаграммы классов. Для разработки модели были использованы диаграммы классов. На их основе был построен каркас модели, который иллюстрирует взаимодействие участников в сети МОПС(см. Рис. 1).

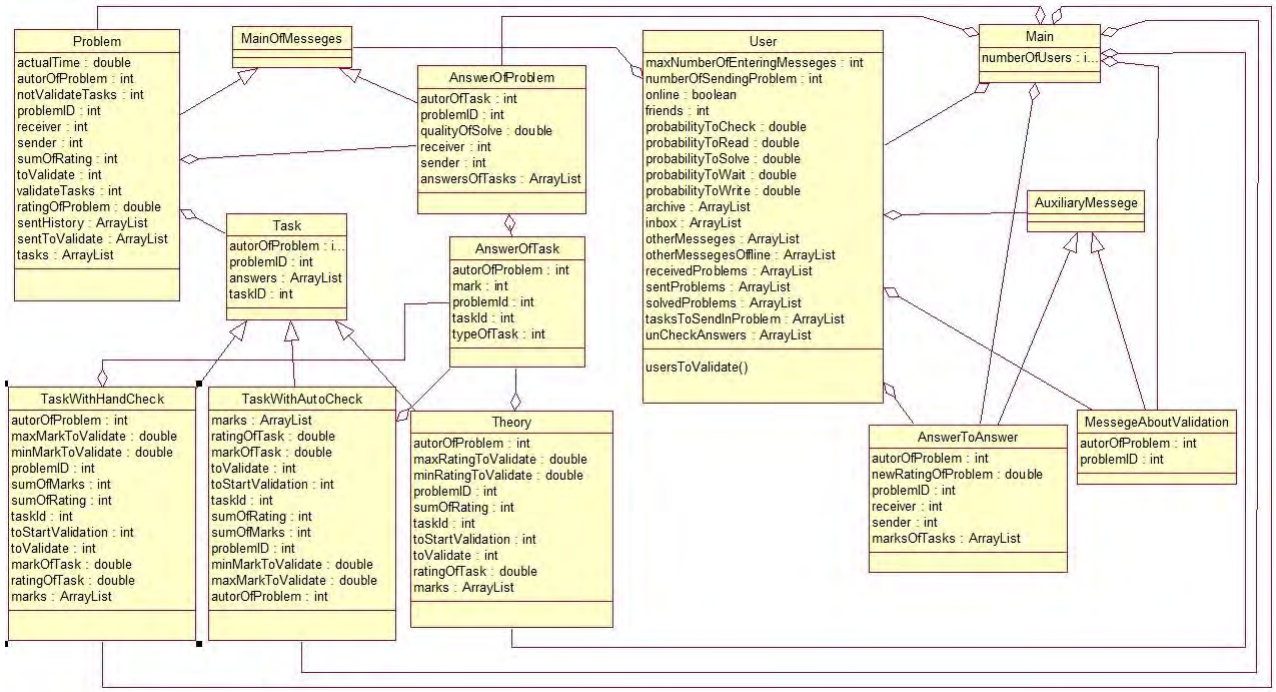

Рисунок 1. Диаграмма классов

Для описания пользователей реализован активный класс User.

На данный момент модель на определенном уровне абстракции позволяет представить процесс взаимодействия пользователей в сети в рамках образовательного процесса. Число пользователей в сети – 10. Пользователи объединены в сеть, связи в которой отражают двусторонние отношения «дружбы». Друзья в сети взаимодействуют посредством передачи сообщений, которые также представлены классами активных объектов:

- Problem – задачи, пересылаемые пользователями;
- Task – задания, входящие в состав задач Problem;
- TaskWithHandCheck – задания с ручной проверкой;
- TaskWithAutoCheck – задания с автоматической проверкой;
- Theory – задание-теория;
- Answer – ответ на задачу класса Problem;
- AnswerToAnswer – сообщение о результатах решения задачи;
- MainOfMessege – родительский класс пересылаемых задач и ответов;
- AuxiliarityMessege – родительский класс служебных сообщений.

Описание класса USER. Пользователь в модели представлен классом активных объектов. Одной из составляющих активного объекта пользователя является statechart, который

предназначен для описания поведения агента в виде возможных состояний и переходов между этими состояниями.

В каждый момент времени пользователь может находиться в состояниях online или offline. Online является сложным состоянием, включающим в себя состояния wait, read, write, solve и check(см. рис. 2).

- Wait – пользователь вышел в сеть, но пока не совершает ни каких действий в рамках образовательного процесса.
- Read – пользователь просматривает нерешенные задачи.
- Write – пользователь создает задачи.
- Solve – пользователь предпринимает попытку решения задачи.
- Check – пользователь проверяет полученные им ответы.

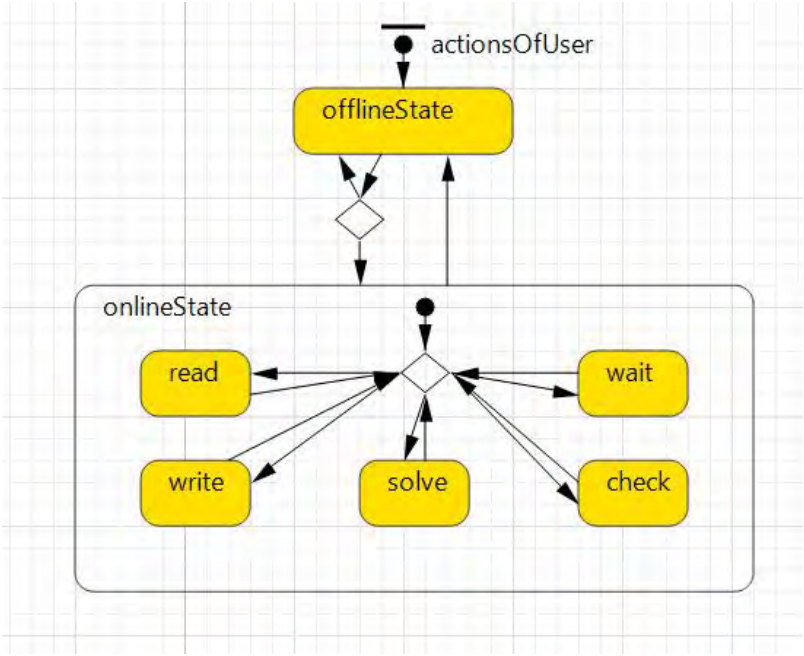

Рисунок 2. Диграмма состояний класса User

Переход из текущего состояния в последующее происходит по таймауту, который является случайной величиной. Для каждого из состояний read, write, solve, wait и check определена вероятность перехода из текущего состояния в последующее (в том числе и для повторения текущего состояния).

У каждого агента имеется несколько ящиков для хранения сообщений, реализованных при помощи инструмента среды AnyLogic – коллекций:

- receivedProblems – принятые пользователем сообщения;
- sentProblems – созданные и отправленные пользователем сообщения;
- archive – архив сообщений (хранит сообщения, у которых вышел срок актуальности);
- inbox – сообщения, адресованные пользователю сети, но еще не полученные им по причине переполнения ящика принятых сообщений или отсутствия подключения к сети;
- solvedTasks – решенные пользователем задачи;

- unCheckAnswers – принятые, но еще не проверенные ответы на задачи с ручной проверкой;
- otherMesseges – коллекция, хранящая служебные сообщения (ответы с автоматической проверкой, сообщения о результатах валидации и т.д.);
- otherMessegesOffline – служебные сообщения, адресованные пользователю сети, но еще не принятые.

При выходе из состояния write агент отсылает сообщения половине своих «друзей» для валидации задачи.

По получении сообщения агент выполняет следующие действия:

1. Если он в этот момент времени находится в состоянии online, то проверяется возможность приема полученного сообщения и его размещения в зависимости от типа в unCheckAnswers или receivedTasks. Если количество уже имеющихся сообщений в этих коллекциях превышает максимальное (задается параметром maxNumberOfEnteringMesseges), то полученное сообщение помещается в коллекцию inbox.
2. Если на момент поступления сообщения агент находится в состоянии offline, то полученное сообщение помещается в коллекцию inbox.

Итогом нахождения агента в состоянии solve является решение задачи: помещение решаемой задачи в коллекцию solvedTasks и отправки ответа автору задачи или пользователю, приславшему эту задачу – в зависимости от типов входящих в задачу заданий.

Полученные ответы Answer проверяются пользователем в состоянии Check: ответ типа Answer помещается в коллекцию ответов в составе соответствующей задачи, отсылается сообщение AnswerToAnswer, содержащее информацию о правильности решения задачи.

Анимация. Окно с анимацией позволяет продемонстрировать работу взаимодействия агентов. Данное окно включает общие масивы переданных данных: количество всех ответов, количество вопросов с ручной проверкой, количество вопросов с автоматической проверкой и количество теории(см. Рис. 3). Линиями обозначены дружественные связи у агентов.

На данном этапе презентация позволяет посмотреть текущие параметры каждого активного класса, будь то агент, передаваемые ответы или задачи. На анимации отображается работа statechart, состояние которых изменяется в зависимости от событий. Текущее состояние подсвечивается красной рамкой, поэтому можно в любой момент времени определить, что делает агент(см. рис. 4).

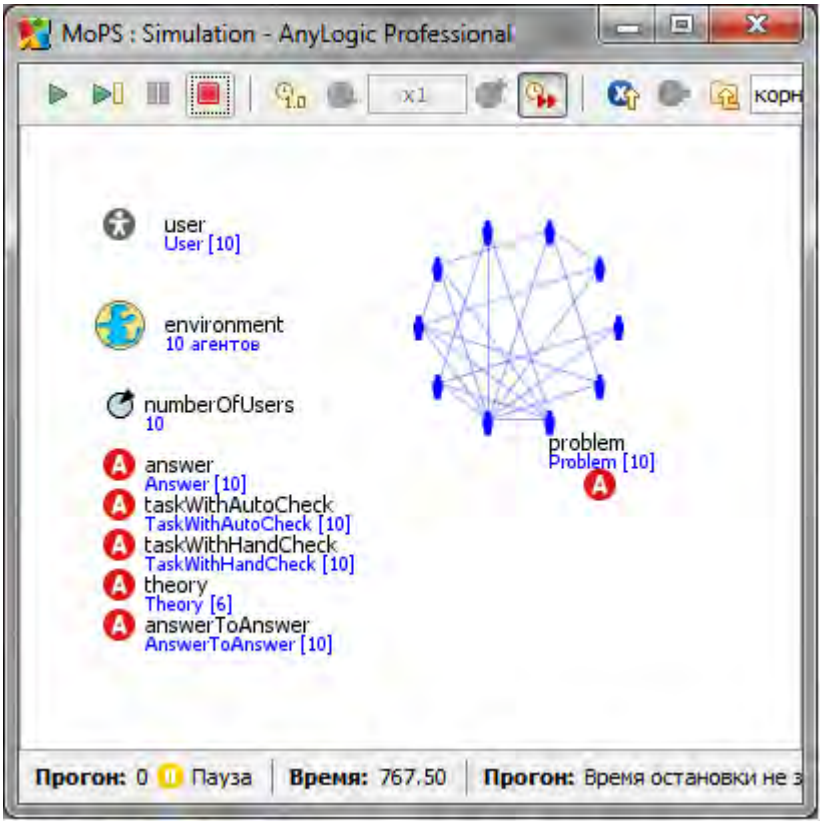

Рисунок 3. Анимация модели

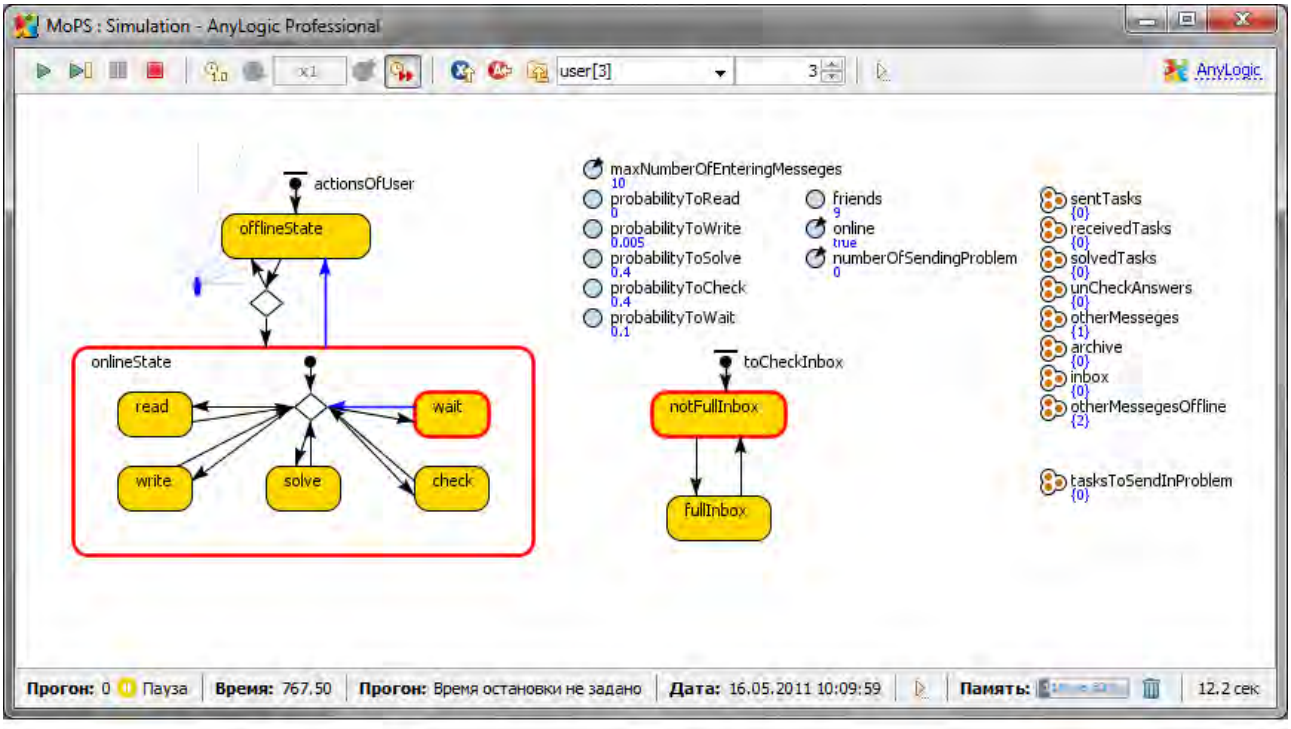

Рисунок 4. Анимация активного класса

Выбор средства моделирования показал, что Anylogic предоставляет все необходимые инструменты для построения модели взаимодействия участников образовательной сети МОПС.

Модель позволяет проследить за процессом создания и отправки задач, а также последующей валидации. В дальнейшем планируется расширять модель посредством реализации в ней алгоритмов взаимодействия агентов с суперпирами и алгоритмов подчета рейтинга пользователя.

Агентное моделирование поволило построить модель процесса взаимодействия участников сети МОПС за короткий промежуток времени. Построенная модель позволила определить параметры платформы, которые влияют на ее устойчивость. Грамотное управление ими препятствует появлению сбоев и отказов в работе платформы. Также, иллюстрация процесса взаимодействия участников платформы с применением анимации наглядно продемонстрировала процессы распространения и обработки задач во времени.

ЛИТЕРАТУРА

1. Кельтон В., Лоу А. Имитационное моделирование. Классика. СПб.: Питер, 2004., 847с
2. Хемди А. Таха Глава 18. Имитационное моделирование // Введение в исследование операций Operations Research: An Introduction. — 7-е изд. — М.: «Вильямс», 2007. — С. 697-737. — ISBN 0-13-032374-8
3. Ю.Г.Карпов, Моделирование агентов - новая парадигма в имитационном моделирвоании, Санкт-Петербургский государственный политехнический университет
4. Строгалев В. П., Толкачева И. О. Имитационное моделирование. — МГТУ им. Баумана, 2008. — С. 697-737. — ISBN 978-5-7038-3021-5
5. D. KORNHAUSER, VISUALIZATION TOOLS FOR AGENT-BASED MODELING IN NETLOGO
6. Сравнение средств разработки для создания мультиагентных систем, http://ru.wikipedia.org/wiki/
7. http://www.letopisi.ru/index.php/NetLogo
8. Е.Д. Патаракин, Б.Б. Ярмахов,Моделирование организационных отношений с использованием "связей" Netlogo
9. AnyLogic vs NetLogo, http://www.xjtek.com/support/kb/?id=19
10. Карпов, Ю. Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. — СПб: БХВ-Петербург, 2006. — 400 с. — ISBN 5-94157-148-8
11. Зинченко О.А, Моделирующие многоагентые системы