Пловдивски университет „Паисий Хилендарски“

Факултет по математика и информатика

Катедра „Компютърни системи“

РАЗРАБОТВАНЕ НА БАЗОВА АРХИТЕКТУРА И ПРОТОТИП НА ПЕРСОНАЛЕН АСИСТЕНТ НА ПРЕПОДАВАТЕЛ

АВТОРЕФЕРАТ

на дисертационен труд за присъждане на образователна и научна степен „доктор“ в област

4. Природни науки, математика и информатика, професионално направление:

4.6. Информатика и компютърни науки, докторска програма: Информатика

Докторант: Пенчо Йорданов Малинов

Научен ръководител: проф. д-р Станимир Стоянов

Пловдив

2023

Дисертационният труд е обсъден и насочен за защита пред научно жури на заседание на катедра „Компютърни системи“ при Факултета по математика и информатика на ПУ „Паисий Хилендарски“ на 09.12.2022 г.

Дисертационният труд съдържа **138 страници**. Библиографията включва **116 източника**. Броят на авторските публикации е 4.

Защитата на дисертационния труд ще се състои на …….. в Заседателната зала на Нова сграда на ПУ „Паисий Хилендарски“, гр. Пловдив.

Материалите по защитата са на разположение на интересуващите се в секретариата на ФМИ – каб. 330 в Нова сграда на ПУ „Паисий Хилендарски“, всеки работен ден от 8:30 до 17:00 часа.

Автор: Пенчо Йорданов Малинов

Заглавие: „Разработване на базова архитектура и прототип на персонален асистент на преподавател“

Пловдив, 2023 г.

СЪДЪРЖАНИЕ

ОБЩА ХАРАКТЕРИСТИКА НА ДИСЕРТАЦИОННИЯ ТРУД

Съществуват различни похвати за оценка на получените знания в процеса на обучение. Все по-съществена роля заема електронното образование и респективно дистанционното обучение. Изграждат се различни системи, предоставящи електронни образователни услуги, които съдържат в себе си, както множество учебни материали, така и тестове за оценка и самооценка.

В същото време се утвърдиха различни стандарти, включващи различни аспекти на образователния процес. Някои от тях са Sharable Content Object Reference Model 2004 (SCORM 2004, създаден от ADL [1]) и Question and Test Interoperability Standard (QTI 2.1, създаден от IMS Global Learning Consortium [2]).

Отчитайки това развитие, Центърът за Електронно Обучение – Distributed e-Learning Center (DeLC) [3,4,5,6] се променя концептуално и от динамична разпределена мрежова структура, състояща се от възли и връзки между тях, преминава във Виртуално образователно пространство(ВОП).

За да се повиши ефективността от процеса на обучение във ВОП е необходимо да се създаде интелигентен модул за анализ на резултати от електронни тестове (генетичен преподавателски помощник), като резултатите трябва да се предоставят, както на преподавателя, така и на обучаемите, за да бъдат отчетени пропуските в усвояването на учебния материал и да бъде дадена възможност на преподавателя да анализира и открие причините за тях.

ЦЕЛ И ЗАДАЧИ НА ДИСЕРТАЦИОННИЯ ТРУД

Целта на дисертационния труд е да се създаде интелигентен модул за анализ на резултати от електронни тестове (генетичен преподавателски помощник), който да предоставя резултатите, както на преподавателя, така и на обучаемите. Базовата версия на преподавателския помощник да се разшири с възможности за отчитане на състоянието на заобикалящия физически свят.

Актуализиран с новите възможности преподавателски помощник ще може да оперира като IoT персонален асистент.

Основни задачи

- Да се направи анализ на текущото състояние- среди за доставка на тестово съдържание;
- Да се разработи базова версия на преподавателски помощник.
- Да се създаде тестова система на базата на IMS QTI стандарт
- Да се разработи допълнителен модул, възприемащ и анализиращ състоянието на физическия свят.

СТРУКТУРА НА ДИСЕРТАЦИОННИЯ ТРУД

Дисертационният труд се състои от увод, пет глави и заключителна част.

Уводът обосновава актуалността на проблема, дефинира целите и задачите, които авторът си поставя за изпълнение на софтуерната разработка.

В **първа глава** са разгледани основните понятия, свързани с темата на дисертационния труд. Разгледани са различни стандарти за електронно обучение, среди за доставка на тестово съдържание и видове персонални асистенти.

Във **втора глава** е направен анализ на процеса на мигриране на DeLC във VES и общата архитектура на ViPS

В **трета глава** авторът представя софтуерната архитектура на генетичен преподавателски помощник. Представени са софтуерните модули, тяхната взаимовръзка и използваните алгоритми, които са неизменна част от софтуерната реализация на генетичен преподавателски помощник.

В **четвърта глава** е представена и обяснена софтуерна реализация на тестова система, имплементираща стандарта QTI. Обърнато е внимание на функционалността на всеки един модул и е обяснена неговата реализация.

В **пета глава** е разгледана структурата и реализацията на модул, възприемащ и анализиращ състоянието на физическия свят.

Дисертационният труд завършва със **заключение**, бъдещи цели, списък с участието в проекти на автора, списък с публикации и използваната литература.

КРАТКО СЪДЪРЖАНИЕ НА ДИСЕРТАЦИОННИЯ ТРУД

ГЛАВА 1. СЪСТОЯНИЕ НА ИЗСЛЕДВАНИЯ ПРОБЛЕМ

Основни понятия

Терминът е-обучение или е-учене (e-learning) е обобщаващо понятие за обучение с компютър, обикновено свързан в мрежа. Може да се провежда където и да е и по всяко време. Отдавна е прието, че е поне толкова богато и ценно, колкото традиционното обучение. Със своите уникални характеристики е-обучението е процес, който води до овладяване на нови умения и знания с множество предимства. Обединява различни дейности - логистика (административно управление), създаване, разпространение и използване на обучаващ софтуер (courseware). Най-често срещаните (непълни) синоними на е-обучението са: онлайн обучение (online learning); уеб-базирано обучение (web-based training, WBT); сравнително по-старите термини - компютърно-базирано обучение (computer-based training, CBT) и обучение с помощта на компютър (computer-assisted learning, computer-aided learning, CAL, computer-assisted instruction, CAI), които не означават непременно, че компютърът е свързан в мрежа и др. [7].

Стандарти за електронно обучение и среди за доставка на тестово съдържание

Основната цел на стандартите за електронно обучение е да бъде предоставена стандартизирана структура на данните и протоколи за комуникация на обектите за електронно обучение, както и междусистемни работни процеси. Използването на тези стандарти дава възможност на потребителите на електронно обучение да използват съдържание и системни компоненти от множество доставчици, въз основа на тяхното качество и уместност, с увереност, че те ще работят заедно ефективно.

Съдържанието е в основата на електронното обучение. Учебното съдържание и каталожните предложения трябва да бъдат етикетирани по последователен начин, за да поддържат индексиране, съхранение, откриване (търсене) и извличане на учебни обекти, чрез множество инструменти в множество хранилища. Използваните данни за тази цел се нарича метаданни на учебния обект.

Спецификациите и стандартите за пакетиране на съдържанието позволяват транспортирането на курсове (учебни дисциплини) от една учебна система към друга. Пакетите със съдържание включват както учебни обекти, така и информация за това как те трябва да бъдат обединени, за да образуват по-големи учебни единици. Те могат също да посочат правила за

предоставяне на съдържание на обучаемия. Въпроси, тестове и групи от тестове могат да бъдат създадени в една среда и използвани в друга. В случая най-подходяща за пакетиране на съдържанието, е спецификацията за оперативна съвместимост на въпросите и тестовете на IMS Question and Test Interoperability (QTI).

Стандартите, които се разработват в тази област, позволяват на компонентите да споделят резултати на ниско ниво, каквито са въпросите за индивидуална оценка или пък на по-високо ниво, като оценка на курс или състояние на завършеност. Това се постига, чрез създаване на протоколи и модели на данни за стандартизирана комуникация, които позволяват на учебното съдържание да комуникира със системата, която го е доставила.

Персонални асистенти.

Персоналните асистенти е концепция, въведена през 90-те години на 20-тия век. Целта е подпомагане на хората в ежедневните им дела – били те служебни или лични дела. Различни модели, подходи и технологии от областта на изкуствения интелект представят достатъчни възможности за изграждане на интелигентни машини, които самостоятелно изпълняват задачи от името на потребителя [8]. Агентно - ориентираните системи могат да бъдат класифицирани в два големи класа [9]:

- Разпределени автономни, реактивни и социални системи.
- Лични софтуерни

Персоналните асистенти могат да помогнат както за ежедневното, така и за дългосрочното управление, изпълнението и контрола на различни видове задачи. Днес тези асистенти обикновено работят на мобилни устройства и могат да използват ресурсите на социалните мрежи и да се самоубочават посредством комуникацията с потребителя.

Агентите въобще и в частност персоналните асистенти оперират в някаква околна среда.

За целите на дисертацията интересни среди са такива, които интегрират виртуални и физически светове. Теоретична основа на такива среди са концепциите за Cyber-Physical Systems (CPS), Cyber-Physical-Social Systems (CPSS) и IoT.

ГЛАВА 2. ВИРТУАЛНО-ФИЗИЧЕСКО ПРОСТРАНСТВО

Първи идеи за разработване на пространство, интегриращо виртуалния и физическите светове са дадени в [10]. Общо описание на една стабилизирана версия на ViPS може да бъде намерено в [11]. В последните години са провеждани различни експерименти за адаптиране на ViPS в три приложни области – интелигентно земеделие, дигитализация на културно-историческото наследство на България и електронно обучение [12]. Virtual Physical Space (ViPS) не възниква изведнъж – то има своя предистория или по-точно два предшественика, наречени Distributed eLearning Center (DeLC) и Virtual Education Space (VES) съответно.

Обща архитектура на ViPS.

Virtual Physical Space (ViPS) се изгражда като референтна архитектура, интегрираща виртуалния и физическия светове [13], която може да бъде адаптирана за различни CPSS-like applications

Достъпът до ViPS е контролиран и персонализиран. Персоналните асистенти, работещи от името на потребителите и запознати с техните нужди, ще подготвят сценарии за изпълнение на заявките на потребителите и ще управляват и контролират тяхното изпълнение чрез взаимодействие с мидълуера на ViPS. Освен това, персоналните асистенти трябва да могат да работят също така и в режим на „изпреварващо действие“ (превенция), включително да поддържат система за „ранно и навременно предупреждение“.

Събитиен модел

Теоретичната основа на ViPS е събитийният модел [14, 15]. Той е основният механизъм, с който се синхронизира работата на компонентите на пространството. Събития могат да се случват в произволно място и по всяко време в пространството. Събитията се разглеждат като пасивни елементи на пространството.

За реализиране на модела се предвижда имплементиране на специализиран интерпретатор, наречен Event Engine – това е една от задачите на настоящата дисертация. Event Engine трябва да интерпретира и управлява възникналите събития в реално време, получавайки данни от физическия свят, напр., от сензорни мрежи. Първи експерименти за реализиране на събитийната машина са представени в [16]. При случване на едно събитие Event Engine (имплементация на събитийния модел) генерира съответен събитиен обект, който може да бъде сериализиран и препратен

към асистентите [17]. Те от своя страна могат да анализират съдържащата се в него информация и да вземат решение за адекватно действие.

ГЛАВА 3. АРХИТЕКТУРА НА ГЕНЕТИЧЕН ПРЕПОДАВАТЕЛСКИ ПОМОЩНИК

В тази глава се разглежда основната концепция за създаване на генетичен преподавателски помощник, както и различните етапи, през които е преминало разработването на архитектурата му.

Първи етап - модул изграден от един агент

Първата версия на архитектурата на ГПО е изградена от един агент. Този агент е прозрачен за преподавателите и обучаемите, като контактува с тях посредством техните персонални асистенти.

Основни функции на агента

Според това на кой предоставя информацията от анализа и каква е тя можем да обособим две групи функции:

- Да информира асистента на преподавателя (АП)за: средна оценка от проведения изпит; проблемни за обучаемите раздели от учебния материал – предоставя на АП информация за резултатите от различните дялове на изпита, като подрежда тези секции във възходящ ред в зависимост от обобщеното представяне на обучаемите; проблемни за обучаемите въпроси от проведения тест – предоставя списък на въпросите, на които обучаемите най-често грешат.
- Да информира асистента на обучаемия (АО) за: оценката от изпита; проблемни раздели от учебния материал – предоставя на съответния АО информация за резултатите от различните дялове на изпита, като подрежда тези секции във възходящ ред в зависимост от обобщеното представяне на обучаемите.

Наличието на електронни тестове, които не са анализирани, са първоначалните вярвания (beliefs) на агента за анализ. Желанията (desires) на агента се генерират в зависимост от междинните резултати от анализа. Могат да бъдат различни условия, при които агентът трябва да се намеси и да стартира допълнителни анализи. В зависимост от случаят едно от желанията се трансформира в цел. Целта определя какво да бъде съответното действие на агента (intention). След като завърши с пълното анализиране на всички тестове от изпита предоставя съответните обобщени резултати на АП и АО. Те от своя страна уведомяват преподавателя и обучаемите за постигнатите резултати.

Втори етап - модул като мултиагентна система

След първоначалната реализация на генетичния преподавателски помощник, състоящ се от един агент се премина към подобрение на архитектурата. Тя включва разделяне на агента на група от агенти, образуващи мултиагентана система, където всеки един агент изпълнява строго специфични операции.

Модулът се състои от 5 агента:

- Агент 1 - Агент за анализ на успеваемост на изпит – ESAA - Анализира теста като цяло. Пресмята средна оценка от изпита и вдига флаг при необходимост за допълнителен анализ на определен тест.

- Агент 2 - Анализира маркираните тестове за допълнителен анализ по секции и предоставя обобщен резултат за секциите от тестовете на изпита като ги сортира от секция с най – слаб резултат към секция с най добри резултати. Агентът също така вдига флаг за необходимостта от анализ на въпросите в определена секция.

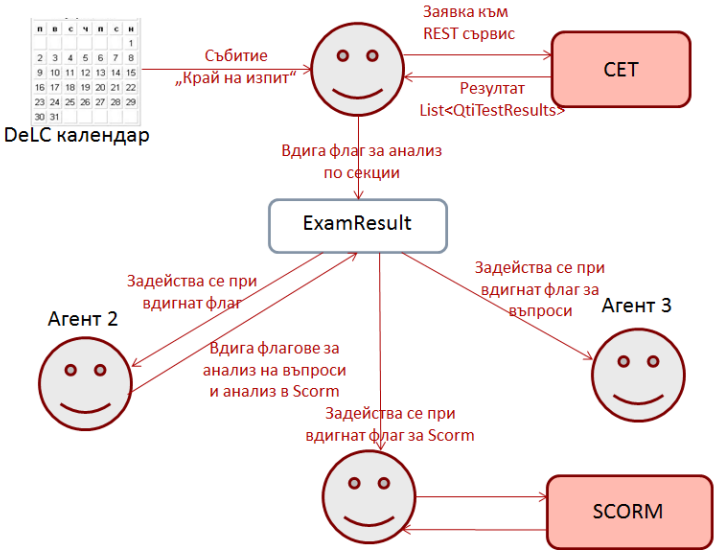
- Агент 3 - Анализира отделните въпроси за маркираните секции на определен AssessmentResult и предоставя обобщен резултат за проведения изпит, представляващ списък с въпросите, на които се допускат най – много грешки.

- Агент 4 - Проверява информация в SCORM – проверява се информацията за активността на определени студенти с цел да се разбере дали студентите са преминали успешно през материала и тестовете в SCORM плеъра.

- Агент 5 - Обобщава резултатите, информира ПАС и ПАП. Този агент се грижи за систематизиране и оформяне на получената от анализите информация и предоставянето и на съответния ПАП и ПАС.

Начин на работа на мултиагентната система

На Фигура 1 е представен начина на работа на мултиагентната система. При настъпване на събитие край на изпит, агент 1 се задейства, изпраща заявка към СЕТ и в резултат получава списък със AssessmentResult. Започва да обработва резултатите от тестовете, като преди това инициализира нов списък със ExamResult, където към всеки AssessmentResult се добавят флагове, необходими за работата на останалите агенти. В този първоначален и общ анализ се прави оценка за необходимостта от допълнителни анализи, като при необходимост А1 променя флага за анализ по секции от ExamResult.



Фигура 1. Начин на работа на мултиагентната система

Този флаг, заедно с AssessmentResult са първоначално вярване на А2 и при промяна той се активира. При необходимост А2 вдига флаговете за анализ на отделни въпроси и за анализ на активността на съответния студент в SCORM плейъра. Тези флагове са част от първоначалната вяра на А3 и А4 и при тяхната променя съответните агенти се задействат. Агентите синхронизират работата си, като отбелязват в ExamResult дали са свършили своята работа. По този начин при вдигнат флаг за анализ на въпросите в дадена секция, агент 3 започва работа едва когато е отбелязано, че агент 2 е завършил своя анализ на всички секции. Резултатите от работата на агенти 1,2,3,4 биват обобщени от А5 и необходимите данни се изпращат към ПАП и Grade book – респективно всеки ПАС на явилите се на теста.

Формализация на Агент за анализ на успеваемост на изпит – ESAA (Exam's Success Analytic Agent)

1. Events –
   - Край на изпит - e = EndExam (EventCourseExam)
   Happens (e, t) => Happens (EndExam (EventCourseExam), t)
2. Beliefs
   - Има свършил изпит
   Believe (a, b1); a = ESSA, b1= Finished (Exam)
   - Списък от непроверени тестове
   Believe (a, b2); a = ESSA, b2 = Exist (ExamResult) ∧ ¬ Happens (Analize (ExamResult), t),
   - Оценката от теста е отлична
   Believe (a, b3); a = ESSA, b3 = Excellent (AssessmentResult)
   AssessmentResult = TestResult ∧ ItemResult ∧ SectionResult
3. Goals - Goal(a, g)

- Да създаде списъка с ExamResults и да го зареди с AssessmentResult от CET.

g1 = Initialize (ExamResults)

Goal (ESSA, g1)

- Да маркира тестове за допълнителна обработка g2 = Mark (AssessmentResult)

Goal (ESSA, g2)

– Да изчисли средната оценка от изпита g3 = Calculate (AverageExamGrade)

Goal (ESSA, g3)

4. Plans –Plan (a, g, p)

- Взема резултатите от изпита от CET и инициализира списъка с Examresults

p1 = [GetResults(SET), Initialize(ExamResutList)]

- Маркира за допълнителен анализ тези тестове, които са с оценка, различна от отлична

P2 = Mark (AssessmentResult)

- калкулира средната оценка от теста и изпраща резултата на A5

p3 = Calculate (AverageExamGrade)

Интеграция на мултиагентната система с QTIWorks

Работата по изграждането на Генетичния преподавателски помощник основно е насочена към A-подпространството или т.нар. Аналитично ниво. Неговата архитектура, както и архитектурата на компонентите, Към настоящият момент към пространството се разработва система за електронно тестване. За целта в D-подпространството е добавен QTIWorks като основен инструмент за създаване и използване на тестово съдържание (Test Engine). QTIWorks [18, 19] е инструмент с отворен код за предоставяне на елементи и тестове за оценка, разработен изцяло върху стандарта QTI 2.1. Тя е разработена като част от проекта JTIC QTIDI на Училището по физика и астрономия в университета в Единбург. За интегриране на системата се използва спецификацията Learning Tools Interoperability–LTI версия 1.1 [20], разработена от IMS Global Learning Consortium. Основната концепция на LTI е да се създаде стандартен начин за интегриране на богати учебни приложения с различни платформи като системи за управление на обучението, портали, хранилища за учебни предмети и др. В LTI тези учебни приложения се наричат Инструменти (предоставени от доставчиците на инструменти –Tool providers), а платформите се наричат Потребители на инструменти (Learning management systems –LMS или Tool Consumers)[21].

За осъществяване на достъп до теста VES имплементира стандарта LTI в частта му за Tool Consumer. Съответно се използват, подписани с протокола Oauth 1.0a съобщения.

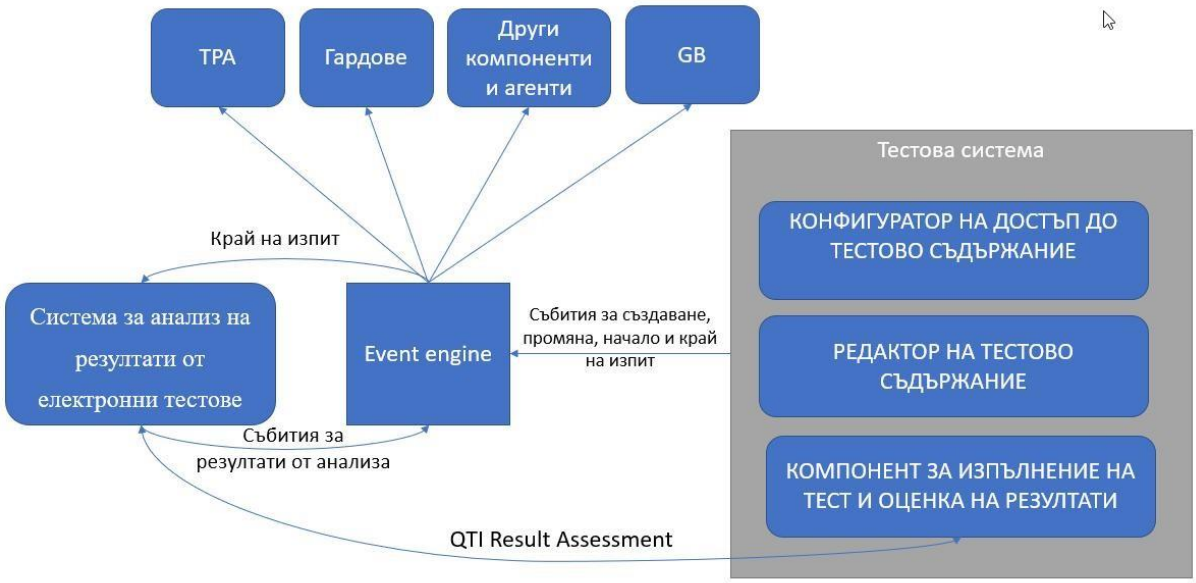
Когато студент отговаря на всички изисквания в конфигурацията за достъп до тестово съдържание, ще има възможност да започне тест

След като приключи даден изпит, ТС изпраща информация до компонента Преподавателски бележник в Аналитичното ниво, като част от него е системата за анализ на резултати от електронни тестове [22]. Системата започва анализ на резултатите, като при нужда взима необходимите Result Assessments от ТР. След като завърши анализа, системата предоставя резултатите на компонентите Студентска книжка и Преподавателски бележник

Трети етап – модул като IoT компонент

По време на интеграцията на QTIWorks се откроi необходимост от допълнителни промени в архитектурата на генетичния преподавателски помощник, с цел пълна съвместимост с референтния модел на ВИПС. Необходимо е генетичният преподавателски помощник да комуникира със околната си среда посредством Event Engine модула.

На Фигура 2 е показана околната среда на мултиагентната система – съответно взаимоотношенията и с тестовата система, преподавателския бележник, студентската книжка, гардове и други компоненти посредством Event Engine.

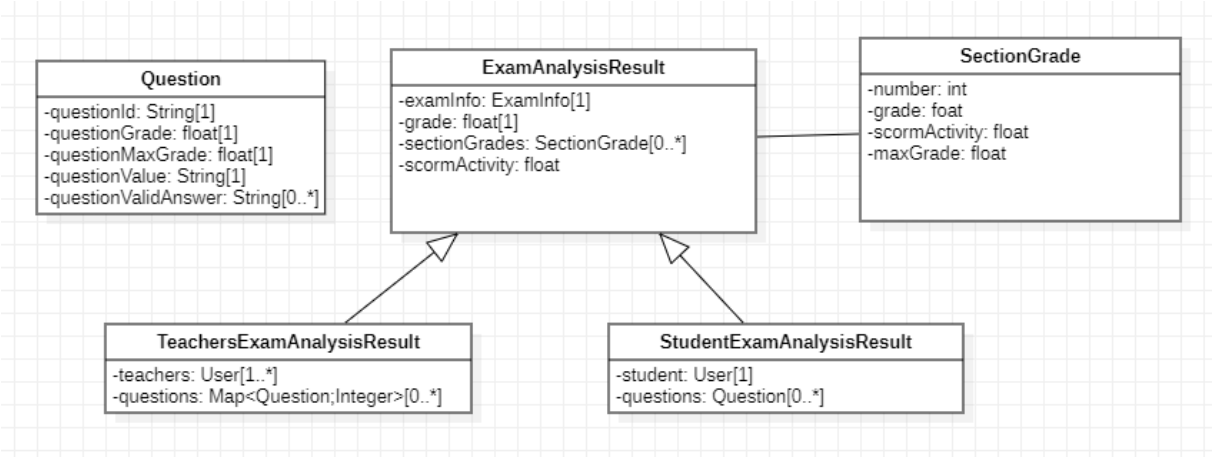


Фигура 2. Околна среда на мултиагентната система при трети етап.

За реализацията на съответната комуникация на мултиагентната система посредством Event Engine с външния свят, са реализирани два

генератора на събития – Събитие за завършен анализ за преподавателя и Събитие за завършен анализ за студент. Генераторите се използват от агент 5, за да може да информира Преподавателския бележник (TPA) и Студентската книжка (GB) – съответно всеки ПАС на студент явил се на изпит и всеки ПАП на преподавателите организирали изпита.

Моделът на данните на събитието „Завършен анализ за преподавателя“ е класът TeachersExamAnalysisResult, представен на Фигура. 3.



Фигура 3. Модел на данните на събитие за завършен анализ на резултати от електронни тестове за преподавател и студент.

Друга задача на агент 5 е да информира GB – съответно ПАС на всеки студент за резултатите от изпита. За целта агентът използва разработения генератор на събития за завършен анализ на резултати от електронни тестове за студент. Моделът на данните на събитието е класът StudentExamAnalysisResult, представен на Фигура. 3.

ГЛАВА 4. ТЕСТОВА СИСТЕМА

Основният компонент в околната среда на системата за анализ на резултатите от електронни тестове е тестовата система. За да имаме пълен контрол върху този компонент, е необходимо да се разработи собствена VES тестова система.

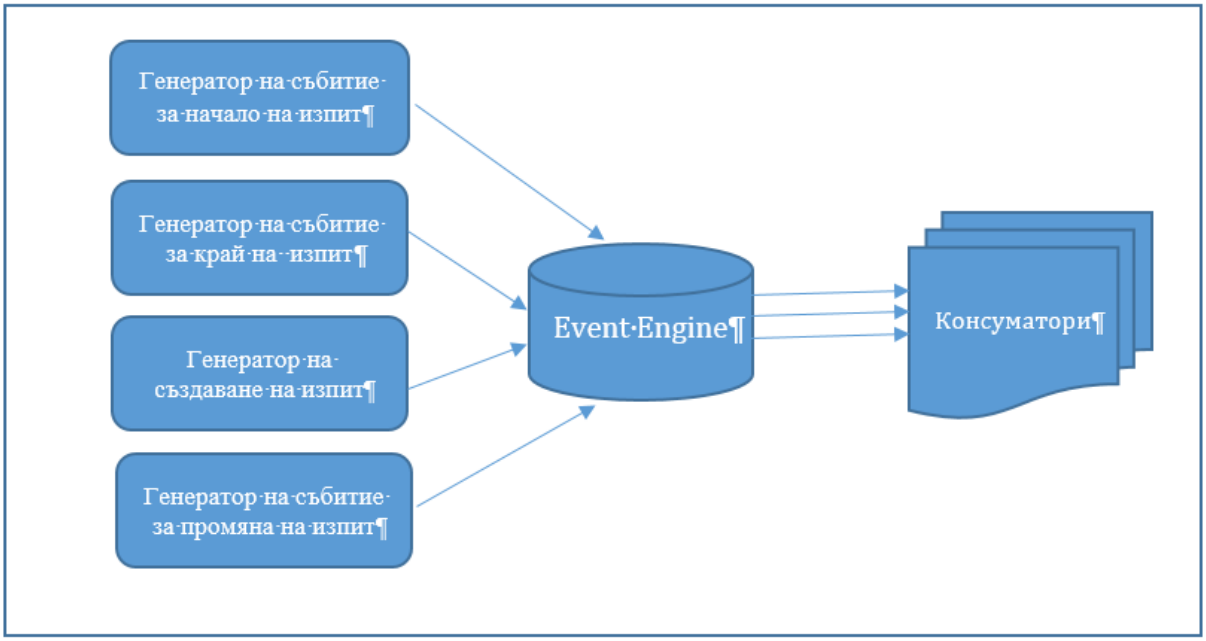
Тестовата система е изградена от три основни компонента, предоставящи специфична независима централна функционалност. Всеки компонент е независим и работи с тестово съдържание, отговарящо на QTI v.2.1 стандарта.

<u>Конфигуратор на достъп до тестово съдържание</u>

Това е компонент за управление на конкретното използване на тестово съдържание от крайния клиент – този който ще решават тест. Този компонент отговаря за това кой, кога, къде и за какво време може да използва определено тестово съдържание. Модулът предоставя две основни функционалности. Функционалност за управление на конфигурации за достъп до тестови ресурси и функционалност за използване на тези конфигурации

Генератор на събития

Системата за конфигуриране на достъп до тестово съдържание обвързва две групи потребители – преподаватели и обучаеми, с конкретно тестово съдържание, като дефинира период, в който теста може да се стартира от обучаемите. От друга страна персоналните асистенти и гардове отговарят за предприемане на конкретни действия при настъпване на конкретна промяна в околната им среда (настъпване на конкретно събитие). Важно условие е системата да е проактивна и да информира всички заинтересовани страни за настъпване на четири важни събития – създаване и промяна на конфигурация за изпит и начало, край на изпит (Фиг. 5), под формата на събитие до “Event Engine”, а от своя страна ЕЕ ще информира абонираните персонални асистенти и гардове.



Фигура 5. Генерирани събития в Конфигуратор на достъп до тестово съдържание

За целта са имплементирани два генератора на събития – Генератор на събитие за начало и край на изпит и генератор за създаване и промяна на изпит.

Редактор на тестово съдържание

“Редактора на тестово съдържание” е WEB базиран компонент отговарящ за процеса по управление на тестово съдържание. Компонента е независима система, като достъп до нея имат само оторизирани потребители.

Основните функционалности предоставяни от системата са: Създаване, промяна и изтриване на тестови ресурси; Организиране на тестови ресурси; Експортиране на тестови ресурси; Споделяне на тестови ресурси; Копиране на тестови ресурси.

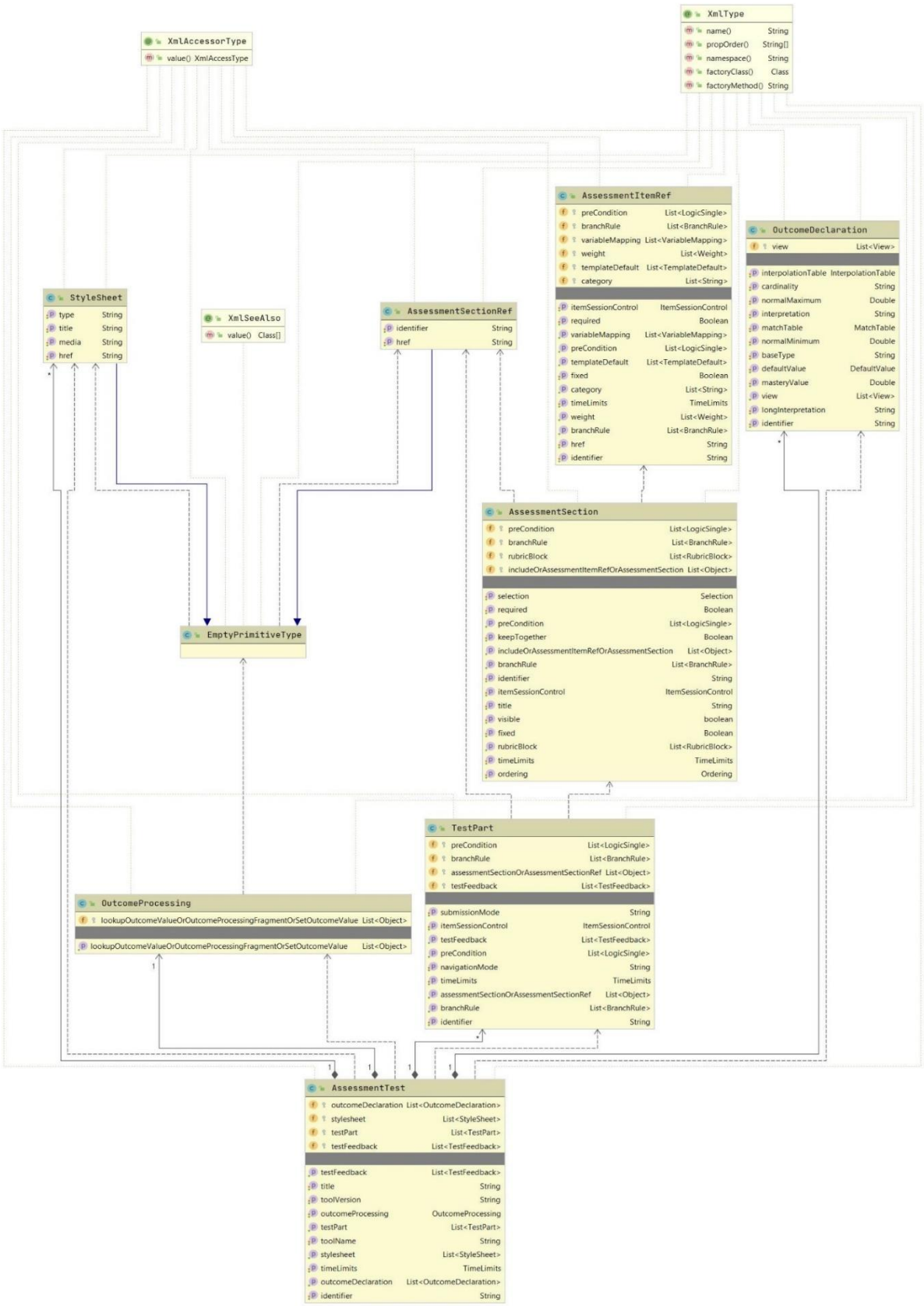
Модели на тестови ресурси

Следвайки стандарта QTI v2.1 е имплементиран обектния модел необходим за създаването на тестовото съдържание. Тестовите ресурси по спецификация представляват XML ресурси. За целта моделите включват необходимата поддръжка за сериализирането на обектите във файлове с XML съдържание и десериализирането на тези файлове в обекти. Процеса по създаване на тестово съдържание се валидира чрез XSD модели за валидация, предоставени от IMS GLOBAL. Благодарение на това се гарантира, че създадените ресурси са съвместими със всяка система, имплементираща съответния стандарт.

- Обектен Модел на тест - AssessmentTest

Тестът е група от тестови елементи (въпроси) със свързан набор от правила, които определят, кои от елементите вижда кандидатът, в какъв ред и по какъв начин кандидатът взаимодейства с тях. Правилата описват валидните пътища през теста, кога отговорите се изпращат за автоматична обработка, след което за проверка и оценка от преподавател. Правилата включват и случаите когато трябва или не требя да се даде обратна връзка.

На Фигура 6 е представен обектен модел на тест (AssessmentTest) и свързаните обекти.



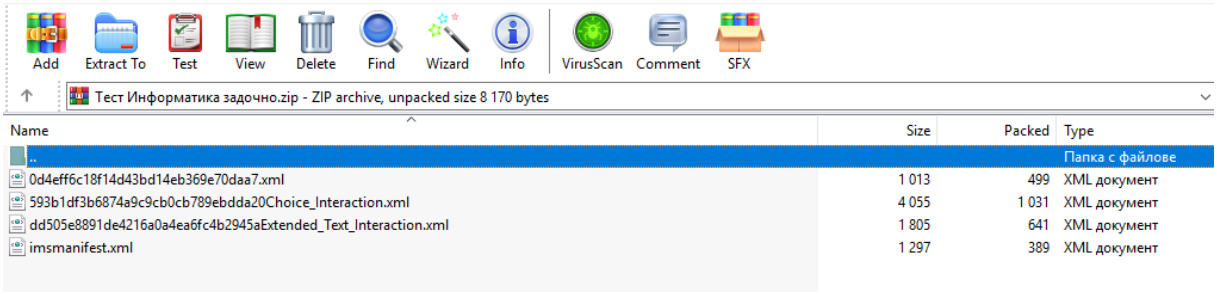
Фигура 6. Обектен модел на тест (AssessmentTest)

- Обектен Модел на тестов въпрос - AssessmentItem

В спецификацията на стандарта QTI v2.1 тестовите въпроси включват информацията, която се представя на един кандидат и информация за това как да оценят отговора. Точкуването се извършва, когато отговорите на кандидатите се трансформират в резултати от правилата за обработка на отговорите.

Пакетиране на тестове, въпроси и свързани ресурси

Всеки въпрос и тест се съхраняват в отделен XML файл. В този смисъл един тест с 10 въпроса представляват минимум 11 файла. Тази съвкупност от файлове е необходимо да се доставят заедно за ползване. За целта всички свързани тестови ресурси се пакетират в ZIP файл. На Фигура 7 е представено съдържанието на пакетиран тест с два въпроса



Фигура 7. Съдържание на пакетиран тест с два въпроса

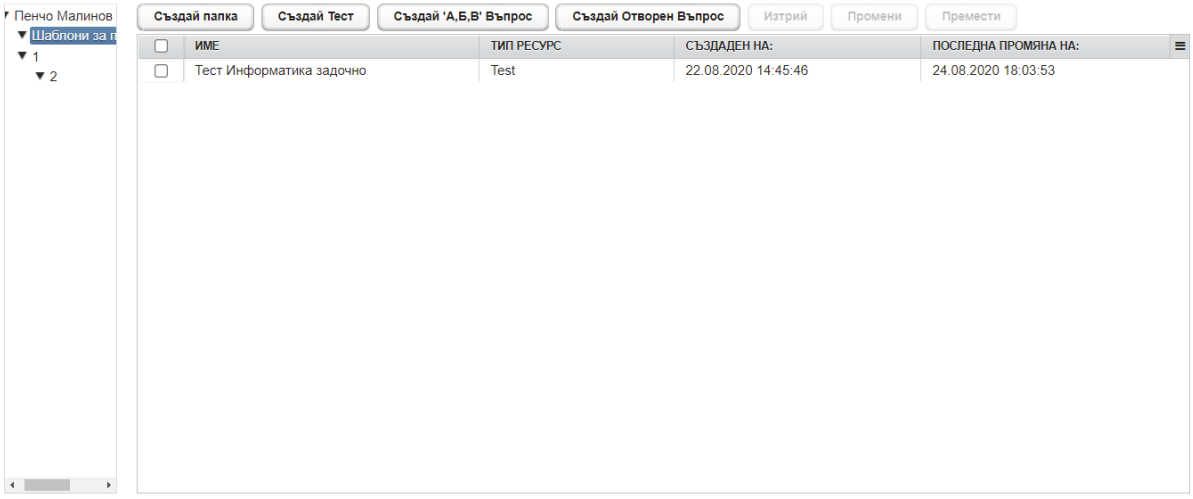
За описанието на структурата в архива се използва **imsmanifest.xml** файл. Той представлява контейнер за структури от данни, чието съдържание описва семантично завършен екземпляр на информационния модел за опаковане на съдържание на IMS структури. Манифестът може да съдържа и да препраща към дъщерни елементи на манифест в същия документ на IMS Manifest. Основният елемент на манифеста дефинира цял IMS пакет. На Фигура 8 е представено съдържанието на **imsmanifest.xml** от архив с тест с два въпроса.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<manifest xmlns="http://www.imsglobal.org/xsd/imscp_v1p1" xmlns:ns2="http://www.imsglobal.org/xsd/imscp_extensionv1p2"
xmlns:ns3="http://www.w3.org/1999/xlink" xmlns:ns4="http://ltsc.ieee.org/xsd/LOM" identifier="0d4eff6c18f14d43bd14eb369e70daa7_manifest">
    <resources>
        <resource identifier="0d4eff6c18f14d43bd14eb369e70daa7" type="imsqti_test_xmlv2p1" href="0d4eff6c18f14d43bd14eb369e70daa7.xml">
            <file href="0d4eff6c18f14d43bd14eb369e70daa7.xml"/>
            <dependency identifierref="dd505e8891de4216a0a4ea6fc4b2945aExtended_Text_Interaction"/>
            <dependency identifierref="593b1df3b6874a9c9cb0cb789ebdda20Choice_Interaction"/>
        </resource>
        <resource identifier="dd505e8891de4216a0a4ea6fc4b2945aExtended_Text_Interaction" type="imsqti_item_xmlv2p1"
        href="dd505e8891de4216a0a4ea6fc4b2945aExtended_Text_Interaction.xml">
            <file href="0d4eff6c18f14d43bd14eb369e70daa7.xml"/>
        </resource>
        <resource identifier="593b1df3b6874a9c9cb0cb789ebdda20Choice_Interaction" type="imsqti_item_xmlv2p1"
        href="593b1df3b6874a9c9cb0cb789ebdda20Choice_Interaction.xml">
            <file href="0d4eff6c18f14d43bd14eb369e70daa7.xml"/>
        </resource>
    </resources>
</manifest>
```

Фигура 8. XML репрезентацията на IMS Manifest

Функционалности и реализация

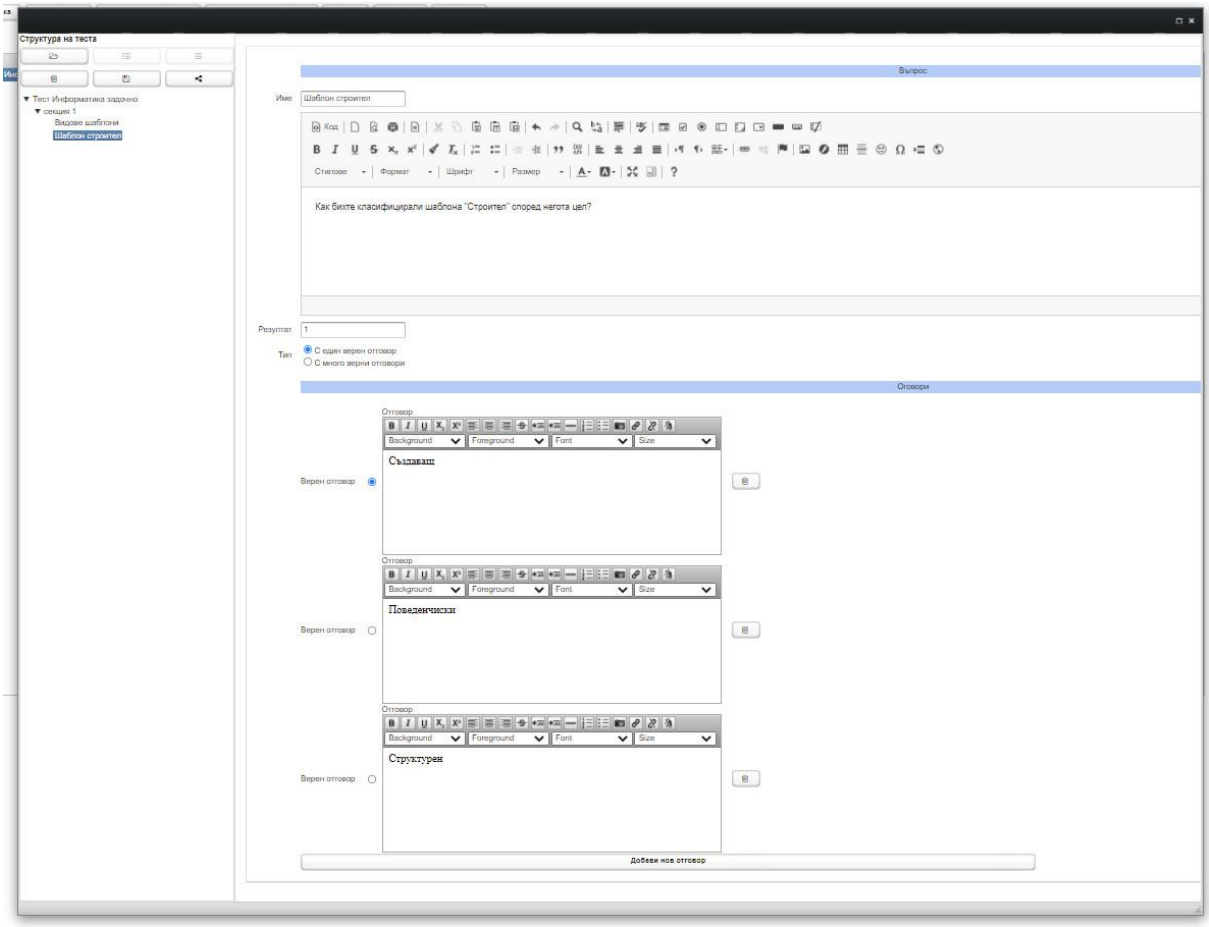
Когато потребител за първи път използва системата, се създава основната му конфигурация с основна папка с неговото име. В тази папка се съхранява създаденото от потребителя тестово съдържание, като допълнително може да създава под папки, в които да го организира. Основният екран на дизайнера, представен на Фигура 9 е разделен на 2 части – навигатор на структурата от папки и панел за управление на съдържанието.



Фигура 9. Основен екран на Редактор на тестово съдържание

Разработени са два екрана за създаване и промяна на въпроси. Първият е за въпрос от отворен тип. Преподавателят може да въведе име на въпроса, съдържание и резултат. За създаване на съдържанието на потребителя се позволява да използва група от форматиращи инструменти, както и вмъкване на таблици, блокове от код, специални символи. Вторият екран е за въпрос с възможност за избор на един или няколко отговора. На

Фигура 10 е представен екранът за създаване и промяна на този тип въпроси, в рамките на тест.



Фигура 10 Диалогов прозорец за създаване на въпрос с един или няколко възможни избора за отговор

Секцията „Споделяне за разработка“ предоставя функционалност преподавателите да споделят конкретния тестови ресурс с други потребители, като те ще могат да го променят и споделят също.

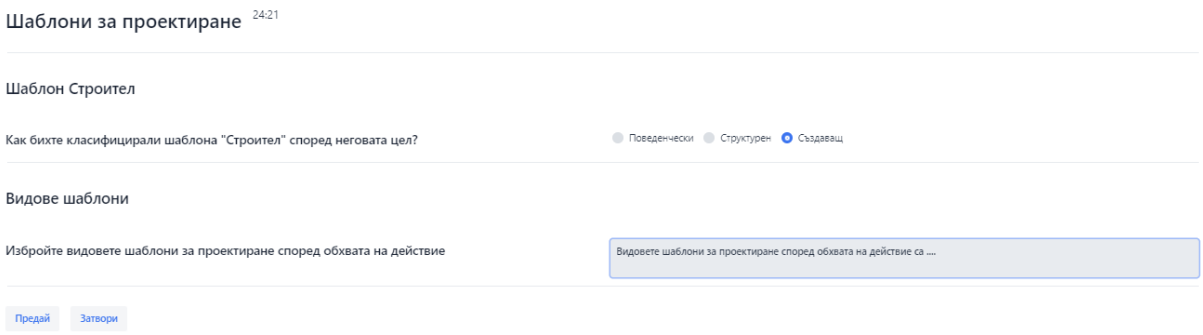
Компонент за изпълнение на тест и оценка на резултати

Компонента предоставя две основни функционалности. Изпълнение на тестово съдържание за даден кандидат и функционалност за оценка на предадените тестове.

Изпълнение на тестово съдържание

Всеки едни тестови ресурс в системата разполага с генерирани URL за изпълнение, споделена тайна и ключ за достъп. Чрез тях и външни системи могат да предоставят достъп за провеждане на тест. За да се стартира изпитът, системата споделяща ресурса трябва да изпрати и пълна

информация за потребителя, явяващ сена изпита. Това е информация за дисциплината, курса, групата и подгрупата, или друга информация свързана със самия тест. На базата ба тази информация се създава йерархия от папки в които се копира ZIP файлът с теста. В същата папка се създава и самия резултат от провеждането на изпита. След това на потребителя се отваря прозорец със съдържанието на теста (Фиг. 11)



Фигура 11. Изпълнение на тест

Студентът има възможност да отговори на въпросите. При стартиране на теста се стартира таймер – видим до заглавието на теста, отброяващ оставащото време за приключване на теста. Всяко действие от потребителя се отбелязва на момента в Result Assessment.

Студентът може да приключи с теста, когато прецени в рамките на предоставеното му време. В случай, че не го направи, тестът автоматично ще бъде приключен. В резултат на това ще се генерира завършен Result Assessment XML файл.

Оценка на резултатите на проведени тестове

След приключването на теста от всички участници, преподавателите могат да видят резултатите от изпита и да започнат оценяване, ако е нужно. Тестовете могат да се оценят автоматично, само ако не съдържат отворени въпроси, в противен случай преподавателският екип трябва да ги оцени

Реализирани консуматори и генератори на събития

Като част от VES, основен начин за комуникация с останалите модули и компоненти се осъществява чрез събития изпращани и получавани от Event Engine. За да може да се добави проактивно поведение, в системата са реализирани група от консуматори и генератори на събития

Консуматорът на събития „Начало на изпит“ използва информацията от събитието „Начало на изпит“ за да ограничи достъпа на неоторизирани

потребители, както и да филтрира достъпа в рамките на списък с конкретни IP адреси, в зависимост от конфигурираната стая.

Консуматор на събитие „Край на изпит“ се грижи да ограничи достъпа за стартиране на тестовия ресурс. .

Освен консуматори на събития системата е и генератор на събития. Генерират се събития за това, че е предадена и последната работа за даден изпит, като то настъпва след последния предаден тест.

ГЛАВА 5. EVENT ENGINE

След оценка на преимуществата и недостатъците на различните решение се взе решение „EE“ да бъде изграден с Kafka и Kafka Streams. За да може да се гарантира независимост на интелигентните агенти и системи една от друга и тяхната проактивност на базата на събитийни модели, се изгради универсално средство за създаване и управление на събития “Event Manager“ (EM), като част от EE. По този начин EE не само служи, като разпределителен център на събития, но и има възможност да изпълнява различни акумулиращи операции върху данните от събитията и резултатът да бъде представен като ново събитие.

За целта се създаде универсално средство за описание на събитийни модели – EventCore библиотека и инструмента „EM“. За да се използва EM е необходимо всеки разработчик да използва библиотеката за да опише като Java класове своите модели.

EventCore

EventCore е библиотека съставена от анотации и интерфейс, с чиято помощ се описват събития като Java обекти. Според естеството на събитието, то може да съдържа разнородни данни. Съответно, данните може да се представят под формата на прости типове в Java – int, long, booelan и т.н., но също така може да се представят и като съставни данни – описани като отделни Java класове.

Анотациите са:

@KafkaResource – основна анотация, служеща за идентификация на Събитийните модели. Всеки клас, анотиран със съответната анотация, ще бъде в областта на действие на EE

@KafkaResourceField – анотацията се поставя върху полетата в Java класа, репрезентиращ събитие. Полетата в класа служат за описание на информацията за събитието. Анотацията позволява конфигурирането на говорещо име за всяко поле. Името се използва в EE UI.

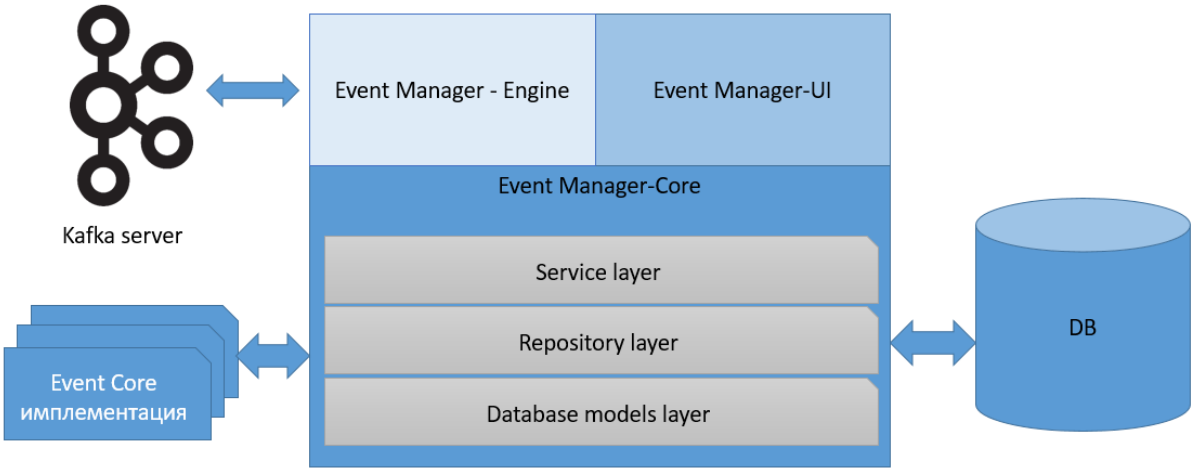
@ KafkaComplexResourceField – с тази анотация се отбелязват полета, които са съставни – обекти съдържащи група от полета анотирани с @KafkaResourceField или @KafkaComplexResourceField. Благодарение на тази анотация може да се описват събития със сложна структура.

С анотациите се описват класовете, но за работата на ЕЕ е необходимо да се създадат инстанции от всеки клас. За целта е необходимо всеки клас да имплементира интерфейса „KafkaBean“. ЕЕ проследява всички имплементации на „KafkaBean“, и създава съответните инстанции.

Event Manager

„Event Manager“ е WEB базирано приложение за създаване на събития на базата на агрегиращи операции приложени върху определен тип събития в определен период от време. Изграден е от няколко компонента (Фигура 12):

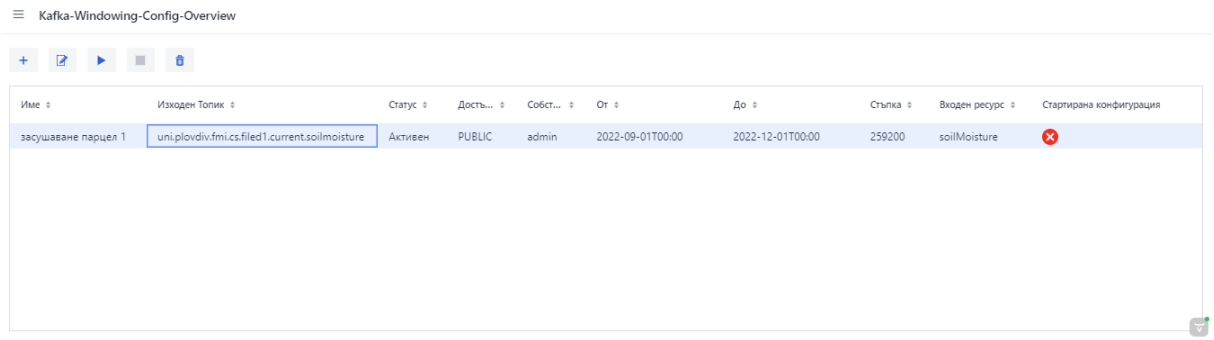
- *„Event Manager-Core“* – включва необходимите модели на данните и услуги за тяхното управление.
- *„Event Manager-UI“* представлява клиентската част на приложението предоставяща уеб базирано решение за управление и създаване на конфигурации за генериране на събития
- *“Event Manager - Engine”* компонент за реалното управление на конфигурациите, като се грижи за създаване и управление на Kafka streams топологии.



Фигура 12. Структура на Event Manager

Event Manager-UI

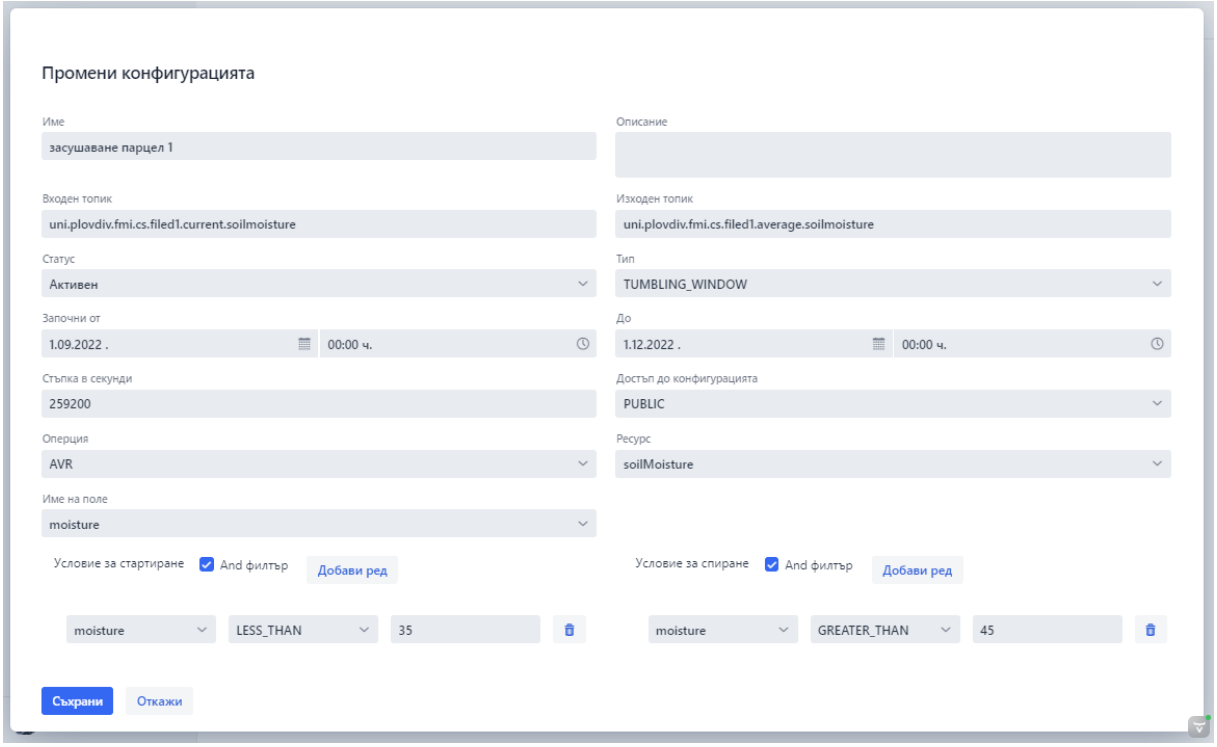
Event Manager-UI е WEB базиран компонент, с който лесно може да се създаде и управлява конфигурация за Kafka Stream топология. За вход в системата е необходимо потребителя предварително да има профил. Всеки потребител има достъп до основния табличен екран визуализиращ списък с конфигурациите до които потребителя има достъп (Фиг. 13). От този екран потребителите имат достъп да създават променят изтриват конфигурации. Също така могат да стартират Kafka Stream топологии на базата на конфигурацията



Фигура 13. Основен екран на Event Manager-UI

Приложението разполага с централен контейнер, пазещ връзка между конфигурациите и стартираните топологии. В колоната „стартирана конфигурация“ се визуализира икона за това стартирана ли е топология или не. При изтриване на конфигурация, автоматично се спира кореспондиращата топология, ако има такава. При промяна на конфигурацията и при налична стартирана Kafka Stream топология, тя се спира и се стартира нова на базата на новата версия на конфигурацията

При създаване или промяна на конфигурация (Фиг. 14) потребителите трябва да попълнят валидни данни за входен и изходен топик. По този начин се определя от къде идват данните и къде ще се изпращащ резултатите от топологият. Потребителите могат да избират типа на прозореца в рамките на който ще се извършва определена агрегираща функция, както и да избере самата нея. Също така могат да определят достъпа до конфигурация и периода за активност на самата конфигурация. Необходимо е да се избере ресурс, който да бъде клас използващ инструментите за описване на събитийни модели на „Event Core“ библиотеката.



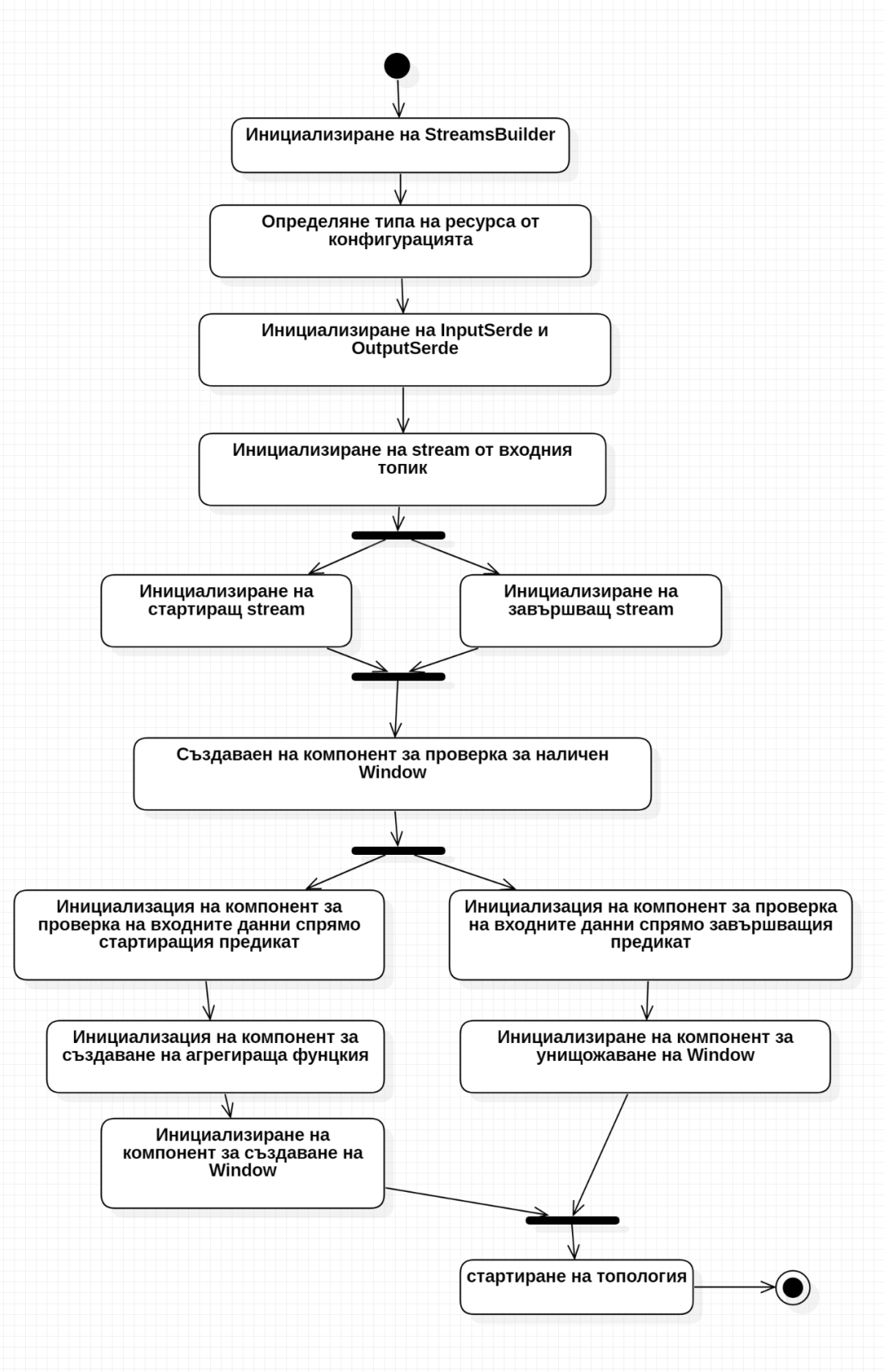
Фигура 14. Диалогов прозорец за промяна на конфигурация за топология

Event Manager – Engine

Ролята на модулът „Event Manager – Engine“ е да създава и управлява Kafka Stream топологии. Той е изграден от няколко основни компонента - Контейнер за Kafka stream; Генератор на предикати; Генератор на топологии;

– Контейнер за Kafka stream – при стартиране на Kafka stream топология на базата на конфигурация, контейнерът я съхранява, като не позволява на връзката между топологията и конфигурацията да се разпадне.

– Генератор на топологии – компонентът се грижи да създаде топология на базата на конфигурация. конкретни действия (Фиг. 15)



Фигура 15. Activity диаграма - Създаване на Kafka Stream топология.

ЗАКЛЮЧЕНИЕ

В дисертацията са изследвани различни стандарти за електронно обучение, доставка и управление на тестово съдържание. Базирайки се на референтната архитектура на ViPS са реализирани редица компоненти, които да предоставят възможност за реализацията на генетичен преподавателски помощник, опериращ като IoT система. Обоснова се необходимостта от реализиране на тестова система базирана на стандарта QTI, която да подпомогне процеса на обучение.

След анализиране на референтната ViPS архитектура се разработи генетичен преподавателски помощник, подпомагаща работата на преподавателя, чрез предоставяне на първичен анализ на резултатите от електронни тестове. В процеса на реализация на архитектурата и прототипа на преподавателския помощник се премина през няколко етапа, описани в дисертационния труд. В резултат на това се разработи преподавателски помощник, представляващ мултиагентна система, опериращa за външния сват като IoT система. Създаден бе модул Event Engine, възприемащ и анализиращ състоянието на физическия свят – използван като основен инфраструктурен компонент, предоставящ унифициран начина за комуникация между всички агенти, гардове, системи и компоненти във ViPS.

ПОСТИГНАТИ РЕЗУЛТАТИ ПО ПОСТАВЕНИТЕ ЗАДАЧИ

Във връзка с изпълнението на целите на дисертацията, са постигнати следните резултати по определените задачи:

Задача 1: Да се направи анализ на средите за доставка на тестово съдържание. За разработката на преподавателския помощник и тестовата система е направен задълбочен анализ на стандартите за доставка на тестово съдържание. Благодарение на този анализ е разработен имплементационния модел на тестовите ресурси, архитектурата на тестовата система и архитектурата и моделите на компонента за пакетиране на тестово съдържание.

Задача 2: Да се разработи базова версия на преподавателски помощник. Разработена е архитектура и прототип интелигентен модул за анализ на резултати от електронни тестове (генетичен преподавателски помощник) – в дисертационния труд е представена архитектура и прототип на генетичен преподавателски помощник. В процеса на разработката се премина през 3 етапа, като всеки етап надграждаше и разширяваше архитектурата и възможностите на прототипа на преподавателския

помощник. Предложената архитектура е базирана на референтната ViPS архитектура, като разширява аналитичното ниво.

Задача 3: Да се създаде тестова система на базата на IMS QTI стандарт. При реализирането на генетичния преподавателския помощник, се потвърди необходимостта от разработване на тестова система, имплементираща стандарта QTI. Създадоха се три отделни независими модула – „ Конфигуратор на достъп до тестово съдържание“, „Редактор на тестово съдържание“ и „Компонент за изпълнени на тест и оценка на резултати“. Базирани на референтната архитектура на ViPS, компонентите използват Event Engine, за да информират външния свят за настъпилите промени и събития в рамките на системата.

Задача 4: Да се разработи допълнителен модул, възприемащ и анализиращ състоянието на физическия свят. Един от основните проблеми за разрешаване за реализацията на преподавателския помощник е взаимодействието му с външния свят. Разработването на Event Engine модул разрешава този проблем, предоставяйки унифициран начин за комуникация между всички участници в референтната ViPS архитектура. Освен да служи за комуникационен клъстър Event Engine има възможност да възприема и анализира събитията от физическия свят. Благодарение на това модулът служи като източник на данни и активен участник за ViPS.

ПРЕДЛОЖЕНИЕ ЗА ПРОДЪЛЖАВАНЕ НА ИЗСЛЕДВАНЕТО

Могат да бъдат предложени различни възможности за продължаване на изследването, резултатите от което са представени в настоящата дисертация.

Основно направление на бъдещи изследвания може да бъде свързано с приложение на Event Engine в реални сценарии в различни приложни области, като напр. интелигентно селско стопанство, електронно обучение, дигитализация на културно-историческото наследство и персонални туристически екскурзоводи, интелигентни градове. За целта е необходимо подготовка и провеждане на тестове с различни домейн-събития. Един значим проблем е установяване на необходимите признаци (данни) за идентифициране и локализиране на различните видове домейн-събития. Най-вероятно е тези необходими данни да трябва да се добиват от различни източници, а не само от сензорните мрежи. Един задълбочен анализ на отделните домейн-събития за избрана приложна област би дал отговор на този въпрос. Съответно, ще бъде необходим реинженеринг на архитектурата на Event Engine, така че да включва също компонент(и) предлагащ(и) решение на този проблем.

Друг проблем, свързан с усъвършенстване на Event Engine, е този, че в една реална ситуация домейн-събитията могат да се случват едновременно. В случая трябва да се мисли за създаване на някакъв вид паралелна Event Engine.

Трети проблем, които може да стане обект на бъдещи изследвания, касае взаимодействието на Event Engine с персоналните асистенти. В крайна сметка компонентът Event Engine е планиран и реализиран като помощно средство на един персонален асистент – Event Engine трябва да подпомага персоналния асистент да идентифицира така наречените инцидентни домейн-събития.

ПРИНОСИ

Приносите в дисертационното изследване могат да се определят като:

Научни:

Създаване на архитектура на генетичен преподавателски помощник опериращ като IoT система във виртуално-физическо пространство;

Научно-приложни:

2. Създаване на прототип на генетичен преподавателски помощник на базата на представената архитектура;

3. Реализиран модел на данните на тестови ресурси и тяхното пакетиране, на базата на стандарта QTI v2.1 и вграждането му в тестовата система.;

4. Реализирана е компонент за унифициран подход за описание на събития и вграждането му в компонента Event Engine;

Приложни:

5. Разработка на Тестова система изградена от три независими модула използващи тестови ресурси имплементиращи стандарта QTI v.2.1;

6. Разработен е компонент за управление и създаване на конфигурации за генериране на събития реализиран в рамките на модула Event Engine.

Връзката между приносите, задачите, структурата на дисертацията и публикациите са представени в следната таблица.

Принос	**Задача**	**Глава**	**Публикации**
1	2	1, 2, 3	1,2,3,4

2	2	3	1,2,3,4
3	1	1, 4	2
4	4	5	
5	3	4	3,4
6	4	5	

Таблица 1. Връзка между приносите, задачите, структурата на дисертацията и публикациите на автора

ПУБЛИКАЦИИ ПО ДИСЕРТАЦИОННИЯ ТРУД

1. Kehayova, I., Malinov P., Stoyanov S., "Intelligent personal assistants in a virtual learning space", International Conference "From DeLC to VelSpace", Plovdiv, Bulgaria, ISBN: 0-9545660-2-5, pp. 175–182, 2014.

2. Malinov P., Kehayova I., "Intelligent agents supporting eLearning by analyzing electronic test result", Scientific Conference, "Innovative ICT: Research, Development and Application in Business and Education", Hisar, Bulgaria, ISBN: 978-954-8852-56-7, pp. 45-54, 2015.

3. Kehayova I., Malinov P., Valkanov V., “Analitycal Level ofVelSpace”, Списание „Компютърни науки и комуникации”, Том 5, брой 4, ISSN: 1314-7846, pp. 23–28, 2016.

4. Kehayova I., Malinov P., Valkanov V., Doychev E., “Architecture of a Module for Analyzing Electronic Test Results”, International Conference “IEEE Intelligent Systems IS’16”, Sofia, Bulgaria, ISBN 978-1-5090-1353-1, pp. 784–788, 2016.

БИБЛИОГРАФИЯ

[1] ADL Consortium http://www.adlnet.orgRetrievedon

[2] IMS Global Learning Consortium https://www.imsglobal.org/aboutims.html

[3] Stoyanov S., Popchev I., Ganchev S., O'droma M., "From CBT to e-Learning", Information Technologies and Control, vol. 4, pp. 2-10, 2005.

[4] Stoyanov S., Ganchev I., Popchev I., O'Droma M., "An approach for the development of InfoStation-Based eLearning architecture", Compt. Rend. Acad. Bulg. Sci., vol. 62, no. 9, pp. 1189-1198, 2008.

[5] Stoyanov S., Popchev I., Doychev E., Mitev D., Valkanova V., Stoyanova-Doycheva A., Valkanov V., Minov I., "Educational portal", Cybernetics and Information Technologies (CIT), vol. 10, no. 3, pp. 49-69, 2010.

[6] S. Stoyanov, I. Ganchev, I. Popchev, M. O'Droma, R. Venkov, "DeLC - Distributed eLearning Center", 1st Balkan Conference in Informatics, Thessaloniki, Greece, ISBN: 960-287-045-1, pp. 327-336, 2003.

[7] Боряна Делийска, Николай Хаджидимитров, Съвременни образователни стратегии и технологии за преподаване, Ръководство 2014

[8] P. Maes, “Agents that reduce work and information overload,” Communications of the ACM, Vol. 37, No. 7, July 1994, 30-40.

[9] M. Wooldridge, An Introduction to MultiAgent Systems, John Wiley & Sons Ltd, 2009.

[10]S. Stoyanov, Context-Aware and Adaptable eLearning Systems, PhD Thesis, STRL, De Montfort University, Leicester, UK, 2012.

[11]S. Stoyanov, T. Glushkova, E. Doychev, A. Stoyanova-Doycheva, V. Ivanova, Cyber-Physical-Social Systems and Applications. Part I: Reference Architecture,, LAP LAMBERT Academic Publishing, 2019.

[12]T. Glushkova, A. Stoyanova-Doycheva, V. Ivanova, S. Stoyanov, E. Doychev, Cyber-Physical-Social Systems and Applications. Part II: Applications, LAP LAMBERT Academic Publishing, 2019.

[13] S. Stoyanov, D. Rusev, Integration of virtual and physical worlds in ViPS, Big Data, Knowledge and Control System Engineering (BdKCSE’19), IEEE Conference, 21-22 November, 2019, Sofia.

[14] S. Stoyanov, A Virtual Space Supporting eLearning, Proceedings of the Forty Fifth Spring Conference of the Union of Bulgarian Mathematicians Pleven, April 6–10, 2016, 72-82

[15] S. Stoyanov, A. Stoyanova-Doycheva, V. Ivanova, V. Tabakova-Komsalova, An Event Model for Smart Agriculture, 2021 IEEE International Conference Automatics and Informatics (ICAI), 30 September - 2 October 2021. Varna

[16] Z. Guglev, S. Stoyanov, I. Popchev, B. Toskov, Events Proactive Management in Virtual-Physical Space, Information Technologies and Control, 2/2019, Online ISSN: 2367-5357, DOI: 10.7546/itc-2019-0006, 2-9.

[17] Z. Guglev, S. Stoyanov, Hybrid approach for manipulation of events in the Virtual Referent Space, Компютърни науки и комуникации, Vol 7 № 1 (2018), 79-85.

[18] MCKAIN, David. QTIWorks. 2013.

[19] QTIWorks https://webapps.ph.ed.ac.uk/qtiworks/ last visited 07.2018г

[20] LTI Specification https://www.imsglobal.org/activity/learning-tools-interoperability - - last visited 14.06.2017г.

[21] LTI Implementation Guide https://www.imsglobal.org/specs/ltiv1p1/implementation-guide – last visited 15.06.2017г.

[22] Kehayova I., Malinov P., Valkanov V., Doychev E., “Architecture of a Module for Analyzing Electronic Test Results”, International Conference “IEEE Intelligent Systems IS’16”, Sofia, Bulgaria, ISBN 978-1-5090-1353-1, pp. 784–788, 2016.