ПЛОВДИВСКИ УНИВЕРСИТЕТ „ПАИСИЙ ХИЛЕНДАРСКИ“
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА
КАТЕДРА „КОМПЮТЪРНА ИНФОРМАТИКА“

ГЕОРГИ ПЕТРОВ ПАШЕВ

ДИНАМИЧНО ГЕНЕРИРАНЕ И ОПТИМАЛНО УПРАВЛЕНИЕ НА ПОТОЦИ ОТ ДЕЙНОСТИ И РЕСУРСИ ЗА ПРОВЕЖДАНЕ НА ЕЛЕКТРОННОТО ОБУЧЕНИЕ

АВТОРЕФЕРАТ

НА ДИСЕРТАЦИОНЕН ТРУД ЗА ПРИСЪЖДАНЕ
НА ОБРАЗОВАТЕЛНА И НАУЧНА СТЕПЕН „ДОКТОР“

по област на висше образование
4. Природни науки, математика и информатика
професионално направление
4.6. Информатика и компютърни науки
докторска програма „Информатика“

Научен ръководител:
проф. д.м.н. Георги Атанасов Тотков

Рецензенти:
проф. д-р Кънчо Йорданов Иванов
проф. д-р Иван Койчев

Пловдив
2017 година

Дисертационният труд е обсъден и насочен за защита пред научно жури, на заседание на катедра „Компютърна информатика“ при Факултета по математика и информатика на Пловдивския университет „Паисий Хилендарски“, на 27.10.2016 г.

Материалите по защитата са на разположение на интересуващите се в Деканата на Факултета по математика и информатика, Нова сграда на ПУ „Паисий Хилендарски“ (бул. „България“ №236), всеки работен ден от 8:30 до 17 часа.

Защитата на дисертационния труд ще се състои 16.02.2017 г. от 12:30 часа в Заседателната зала на ПУ „Паисий Хилендарски“ (Ректорат, ул. „Цар Асен“ №24).

Автор: Георги Петров Пашев

Заглавие: ***ДИНАМИЧНО ГЕНЕРИРАНЕ И ОПТИМАЛНО УПРАВЛЕНИЕ НА ПОТОЦИ ОТ ДЕЙНОСТИ И РЕСУРСИ ЗА ПРОВЕЖДАНЕ НА ЕЛЕКТРОННОТО ОБУЧЕНИЕ***

GEORGI PETROV PASHEV

DYNAMIC GENERATION AND OPTIMAL MANAGEMENT OF FLOWS OF ACTIVITIES AND RECOURCES FOR E-LEARNING

In the context of e-learning the level of personalization is a measurement for learning quality and effectiveness. In other words: e-learning paths should vary, depending on learning goals, knowledge, motivation and individual learning styles.

A student can pass the course through one or more learning paths. Personalized learning content is more suitable and comprehensible in a context of personal learning goals and achievements.

Support of different aspects of personalization and adaptations of learning instruments is not in the required abstraction level: too often depends on distinct methods of combining LOs in order to build Learning Paths (LPs) for "passing through" an e-course and achieving of Learning Goals. The selection of LOs quite often is built on rather simplified principles, doesn't meet the requirements for modelling and following more complex e-learning strategies.

Goals of the dissertation research

The primary goal of the research is to research, **propose** and test models, methodologies and software instruments, suitable for the creation of adaptive e-learning systems (Ae-lSs) with a large set of possible applications.

The following **research questions** fulfill the primary goal:

Question 1. Research on methodologies and tools for providing adaptive e-learning, including comparative analysis of Ae-ISs,

Question 2. Creation of a generalized model, architectire and a prototype of a software system which provides adaptive e-learning (within an institutional informational infrastructure), as well as a methodology for their application;

Question 3. Introduction and testing of conceptual and computer models for As-IS with different applications;

Question 4. Design, creation and testing of software instruments for adaptive e-learning.

СЪДЪРЖАНИЕ

Списък на използваните съкращения

XML ..... Extendable Markup Language
АМП ..... Аспект за многомерно пространство
АОС ..... Автоматизирана система за обучение
АСЕО ..... Адаптивна система за електронно обучение
АТ ..... Акумулативен тест
АТЕ ..... Акумулативна тестова единица
ГПИ ..... Графичен потребителски интерфейс
ЗО ..... Заявка за обучение
ИПО ..... Изучавана предметна област
МП ..... Мрежа на Петри
ПО ..... Предметна област
СеО ..... Система за Електронно обучение
УД ..... Учебна дейност
УО ..... Учебен обект
УП ..... Учебен път
УЦ ..... Учебна цел
SA ..... Метазнание

Увод

В контекста на е-обучението, равнището на персонализация е мярка за качество и ефективност за обучението. Казано по друг начин, учебните пътища е целесъобразно да се различават един от друг на базата на поставени учебни цели, познания, мотивация и индивидуални стилове на учене на обучаваните. На базата на тези персонални характеристики, обучаван може да „премине“ курса по един или повече учебни пътища. Ако е персонализирано, учебното съдържание е по-подходящо и разбираемо за обучавания в контекста на неговите лични цели и постижения. Поддръжката на различни аспекти на персонализация и адаптация на съответните инструменти не е на желаното равнище на абстракция – много често зависи от конкретния начин, по който се предполага, че учебните обекти (УО) ще се комбинират за построяване на учебен път (УП) за „преминаване“ през е-курс, или за постигане на учебните цели. Подборът на УО за включване в УП обикновено почива на прекалено опростени принципи, не е съобразен с необходимостта от моделиране или следване на по-сложни стратегии за обучение.

Цел и задачи

Основна цел на дисертационното изследване е да се предложат, изследват и апробират модели, методи и софтуерни инструменти, подходящи за създаване на адаптивни системи за електронно обучение (АСеО) с различно предназначение.

Изпълнението на поставената цел предполага решаване на **четири задачи**:

Задача 1. Проучване на концепции и средства за осигуряване на адаптивно е-обучение, вкл. провеждане на сравнителен анализ на конкретни АСеО;

Задача 2. Създаване на общ модел, архитектура и прототип на софтуерна система за осигуряване на адаптивно е-обучение (в рамките на институционална информационна инфраструктура), както и на методика за тяхното прилагане;

Задача 3. Въвеждане и апробиране на концептуални и компютърни модели за АСеО с различно предназначение;

Задача 4. Проектиране, реализация и тестване на софтуерни инструменти за адаптивно е-обучение.

Структура на дисертацията

В **Глава 1. Обзор** са разгледани **Основни понятия; Методики и модели свързани с адаптивното е-обучение; Съществуващи реализации на АСЕО*,*** на които е приведена класификация. В края на **гл. 1** са формирани ***цели за АСЕО***.

В **Глава 2.** е въведен **Модел на адаптивен процес,** дефинирани са релевантни модели като модели в Изучавана предметна област (ИПО), Модел на работен поток, инстанция на работен поток, съобщение; Модел на аспект и пр. модели. Представена е и **Методика за изграждане на АСЕО,** базирана на **Модел на АСЕО.**

В **Глава 3.** са представени ***логически изглед*** и ***технически изглед*** на **Обща архитектурна рамка на АСЕО.** Моделирани са архитектурно и важни компоненти от *логическия изглед* като модули „Обучаван в курс“, „Преподавател“, „Учебни дейности“, „Администрация“, „Администриране на курс“, „Генератор на персонализирани оптимални учебни пътища“, „Генератор на тестови единици“, „Рамка за работни потоци“.

В **Глава 4.** е представен програмен код от реализацията на рамка АСЕО.

В **Глава 5.** се представя с *елементарни примери* за създаване на адаптивен курс върху рамката АСЕО от техническа гледна точка.

Благодарности

Искрено благодаря на моя научен ръководител – проф. д.м.н. Георги Тотков, за доверието и напътствията при провеждане на дисертационно изследване в областта. Несъмнено, творческият процес беше ползотворен и двустранен, което позволи създаване на нестандартни идеи и тяхната еволюция и прилагане в конкретни резултати на това изследване.

Благодаря също на колегите от Университетския информационен център за оказаната подкрепа, разбиране и съдействие и възможността да участвам в създаване на ползотворна творческа среда при зачитане на добри идеи от всички колеги.

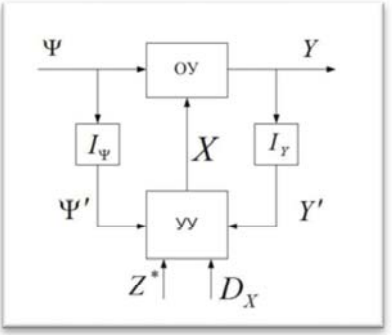
Глава 1. Обзор

1.1 Същност на адаптивното е-обучение

На базата на разбирането за същността на е-обучението, дадена в [Тотков Г. (2016)], може да се заключи, че: Същността на адаптивното е-обучение се заключава в *моделиране на адаптивно обучение като информационен процес с използване на ИКТ*.

В процеса на адаптивно е-обучение участват субекти, информационни ресурси, дейности/подпроцеси, материални ресурси, метазнания и ресурси, управление на процеса, ограничения. Субектите в адаптивното е-обучение са: обучаван, преподавател, автор, експерт в предметната област (ПО), администратор и др. За всеки от субектите има определен набор от важни атрибути, които го характеризират. Информационните ресурси в процеса на е-обучението могат да бъдат например: бази знания за ПО, учебни и тестови единици, е-портфолиа на субекти, цифровизирани дейности и др. [Тотков Г. (2016)]. Дейности/подпроцеси в адаптивното е-обучение могат да бъдат от различни типове: учебни, административни, управленски, комуникационни, други. Като част от управленските дейности можем да посочим: Планиране, Провеждане, Управление на различни форми на обучение, Контрол и др. Материални ресурси, релевантни към процеса на адаптивно е-обучение, са например: зали, компютърни системи, обем/вид на зали (лабораторни, семинарни, компютърни, други). Метазнанията и ресурсите в адаптивното е-обучение включват: учебни цели, описание на ПО, методики за адаптивно обучение, различни връзки между същностите в субектите, различни атрибути на тези субекти. Метазнанието включва метаданни за субекти, информационни ресурси, дейности и др. които не са част от ИПО. Управлението на процеса на адаптивното е-обучение включва управление на гореизброените управленчески дейности. Ограниченията в процеса на адаптивно обучение могат да бъдат например свързани с *време*, *място,* стойност и др.

Задачата за обучение може да се формулира като задача за управление [(Л.А, 1981)]. В случая Адаптивната система за обучение (АОС) изпълнява функциите на управляващо устройство (фиг. 1). На фигурата има следните обозначения: Ψ – състояние на външната среда; Y – състояние на ученика; I_{ψ}, I_Y – съответстващи измерители; Y' и Ψ' – резултатите от измерването; X – управляващи (обучаващи и контролиращи) въздействия; D_x – ресурси (ограничения на управлението); Z^* - цели на управлението, необходими за преход на обучаван в състояние Y^*.



Фигура 1 Обща структура на АОС

Подсистемата за управление на обучението трябва да включва **пет вериги на адаптация**:

- **Верига на корекция на параметрите**. В тази верига адаптацията се свежда до корекция на значението на параметрите на моделите на обучавания, така че да се добие максимално съответствие на моделите на еволюиращия обучаван.
- **Верига на параметричната идентификация**. Може да се окаже, че с помощта на корекции на параметри на модели не може да се придобие приемливо съответствие между модели и обучаван. В тези случаи е целесъобразно използване на процедура по параметрична идентификация на моделите. Естествено, че допълнителните управляващи въздействия (обучаващи и контролиращи) в такъв случай трябва да бъдат минимални, понеже тези въздействия нарушават процеса за постигане на поставените цели на управлението;
- **Верига на структурната идентификация**. Възможна е ситуация, при която и параметричната идентификация е недостатъчна – моделът в предната ситуация не представя обучавания адекватно. Възможно решение е изменението на класа на функциите, отразяващи грешките в измененията на величините Ψ и X и нестационарните характеристики на обучавания;
- **Верига на корекция на ограниченията**. Ако по-горните вериги на корекция не са достатъчно адекватни за моделирането на обучавания, възможно е разширяване на множеството на допустимите управляващи въздействия да се окаже полезно, т.е. разширяването на можествата D_x;
- **Верига на корекция на целите на управлението**. Ако по-горните корекции се окажат недостатъчни за адекватно моделиране на обучавания, тогава може да се премине към корекция на компонентите на целите на управлението Z^* [(Карпенко А. П., 2011)].

1.2 Изводи

Бизнес процесите са обикновено адаптивни към промени. Нотацията на бизнес процесите от гледна точка на експлицитната им нотация е много по-адаптивна, отколкото написани процедури и политики. [(Eibenova, 2015)]

На базата на направения обзор, можем да идентифицираме следните направления за моделиране и създаване АСЕО:

- Създаване на адекваятни модели на:
  - субекти, информационни ресурси, материални ресурси, процеси, инстанции на процеси, съобщения, метазнание (SA);
  - работна рамка АСЕО като процес на адаптивно обучение като по-общ случай на процес на управление на различни субекти/процеси/инстанции на процеси, а не само от гледна точка на управление върху обучавания, както е направено в [(Л.А, 1981)];
  - всеки SA както виртуално, така и като реален обект, така и чрез ГПИ.
- Функционалност за:
  - задаване на управленски цели;
  - описание и съобразяване с модела на предметната област;
  - гъвкавост при определяне на съответстващите измерители I Ψ, IY и начина на измерване;
  - лесна интеграция със съществуващи софтуерни системи;
  - голямо разнообразие на видовете УД.
- Обратна връзка:
  - за гъвкава корекция на параметрите на субекти и техни свойства в АСЕО;
  - адаптивна идентификация на тези параметри по време на експлоатация на системата;
  - структурна идентификация, в т.ч. и автоматизирано събиране на метаданни за обекти и подпроцеси в процеса на адаптивно е-обучение;
  - корекция на учебните цели.

Глава 2. Модели на процеси, методики и АСЕО

2.1 Основни понятия

Определение 1. Множеството T е ***кореново дърво***, (по-нататък ще казваме само ***дърво),*** ако е празното множество, или ако едновременно са изпълнени следните условия:

- Съдържа единствен елемент t, наречен корен на дървото, който ще означаваме с $root(T)$.
- Останалите елементи са (с изключение на корена) са разделени на m ($m \geq 0$) не пресичащи се непразни множества $\mathrm{T}_1, \mathrm{T}_2, \ldots, \mathrm{T}_\mathrm{м}$, всяко от което е дърво.

Корените t_1 на T_1, …, t_m на $\mathrm{T}_\mathrm{м}$ ще наричаме преки наследници на t. Броят на преките наследници на даден връх ще наричаме негова *степен.* Ако степента на връх е 0, той се нарича *листо.*

Път в дърво е последователност от върхове $t_1, t_2, \ldots., t_n$ без повторение, като за всеки два последователно върха t_{i-1}, t_i е изпълнено точно едно от двете: t_i е наследник на t_{i-1} или t_{i-1} е наследник на t_i.

Множеството от всички възможни пътища в дърво Т, които водят до листа, ще бележим с $Paths(T)$.

Нека е дадена функцията $findLeaves$, която връща редица $dest$ от всички листа на T. Тогава, нейната дефиниция в псевдо код би изглеждала така:

```
1  void findLeaves(tree  T, List<tree> & dest)
2  {
3      if (!T) return;
4      foreach( itm in T.ancestors ){
5          if(T.ancestors.Length==0){
6              dest.push(T);
7          }else{
8              findLeaves(itm, dest);
9          }
10     }
11 }
```

Фигура 2. Псевдо код на дефиниция на функция, която намира всички листа на дърво

На ред 8. на фиг. 2 функцията се извиква рекурсивно, при условие, че текущият връх на кореновото дърво не е листо. На ред 4. има обхождане на всички наследници на T. Нека $valLeaf$ е функция, която връща стойност на листо. ва $valLeaf: Paths(T) \longrightarrow Val$. Нека $ValTree = Val \cup T$.

Определение 2. Обект-екземпляр за обект-клас $o = \{attr_1, attr_2, \dots, attr_l\}$ е всяко наредено множество от двойки $\{< attr_1, val_1 >, < attr_2, val_2 >, \dots, < attr_l, val_l >\}$, за което $val_i \in ValTree, i \in \overline{1, l}$.

Измерение D е всяко крайно непразно множество от стойности $D \in set\ (Val)$. Многомерно пространство M с измерения $D_1, D_2, \dots D_n$, $n \in \mathbb{N}$, $n \geq 1$ е декартовото произведение $D_1 \times D_2 \times \dots \times D_n$.

Нека $\tau_i \in set(D_i),\ i = \overline{1, n}$. Декартовото произведение $H = \tau_1 \times \dots \times \tau_n$ е правоъгълна област в многомерно пространство. За $\tau_1, \dots, \tau_n$ съществува изискването да са не пресичащи се множества.

Нека Rep е функция, за която $Rep: O \times H \longrightarrow H$. Проекция по Rep на обект $o \in O$ в H ще наричаме $Rep(o, H)$. В текста по-долу ще бъде наричана и *представяне*.

Възможен алгоритъм за изграждането на представянето Rep е следният:

За всеки val_i във всяка наредена двойка в обект екземпляр:

Стъпка 1: Ако $val_i \in Val \setminus Strings$, се създава измерение D_i, ако още не е създадено, и се добавя стойност за него такова, че $D_i = val_i$;

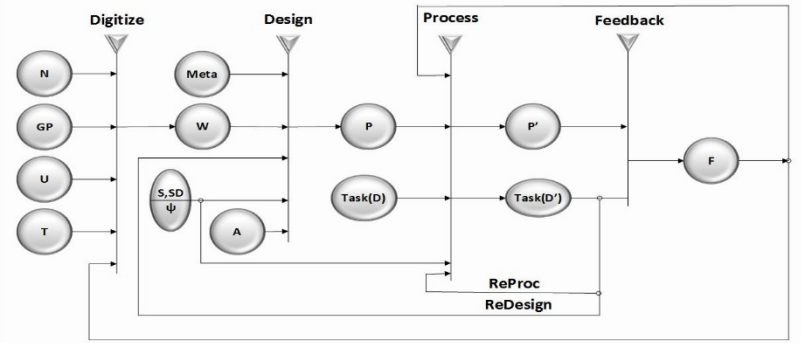
Стъпка 2: Ако $val_i \in Strings$, на val_i се съпоставя целочислена стойност: разстояние на Хеминг [Hamming] $D_i = hamming(val_i, emptyStr_i)$ където $empty\ Str_i$ е низ със същата дължина като val_i, но с базова стойност за всеки символ.

Стъпка 3: Ако $val_i \in T$, за всеки път до листо в дървото $Paths_j(T)$ създаваме $D_i = valLeaf_j$, такова, че $valLeaf_j$ е стойността в листото, до което води пътя $Paths_j(T)$. Изпълняваме последователно *Стъпка 1* и *Стъпка 2* за val_i = $valLeaf_j$ за всяко $valLeaf_j$.

2.2 Модел на адаптивен процес за обучение

Анализът, проведен в Глава 1., позволява да се направи заключение за това, кои са елементите и подпроцесите, свързани с общия модел на процес за адаптивно обучение. Тези елементи и подпроцеси, заедно с техните означения, които се използват в текста на дисертацията, са следните:

- S – процес на адаптивно обучение (виж обект $o \in O$);
- SD – предметна област на S;
- N – множество от нормативни документи и критерии, включително спецификации и стандарти за оценяване в SD;
- GP – набор от добри практики за ЕО в SD;
- U – изисквания на потребителите към адаптивното обучение в SD;
- W – цифрово хранилище (warehouse) с информационни ресурси в конвенционален и електронен вариант за S и неговия контекст C;
- $Digitize$ – процес на цифровизиране и разполагане в W на електронни копия на документи, свързани с контекста C;
- $Design$ – процес на проектиране и създаване на M в контекста на C;
- $Proc$ – процес на компютърно моделиране на така създадените концептуални модели в процеса $Design$;
- A – множеството от аспекти, относно които се разглежда S и неговите елементи;
- Ψ – външната среда, в която протича S;
- D – ограниченията върху заявката за осъществяване на процес S (вкл. времеви, пространствени и логически ограничения за провеждането на S);
- Task(D) е заявка за адаптивно обучение;
- С тилда (~) над дадено означение на компонент е указано, че това е негово следващо състояние;
- Wint означава частта от CeO - компютърен модел на S и неговите елементи (виж прил. 1);
- Wext - частта от CeO, отнасяща се до компютърните модели на елементи, които не са част от S (например метазнания и ресурси за S, предишен опит относно S - преди поредното осъществяване на процеса, и т.н.);
- $Feedback$ – процес на създаване на обратни връзки, съответно за актуализация на информацията в хранилището (чрез $Digitize$), актуализация на състоянията на компютърните модели (обозначени с тилда (~));
- $ReProc$ – процес за актуализация на модела на D, както и на заявката за обучение
- *ReDesign – процес за актуализация на концептуалните и виртуалните (компютърни) модели Wint, Wext , както и моделите на техните пораждащо обекти, в т.ч. S, Ψ, A;*



Фигура 3 Модел на адаптивен процес на обучение

Примери за други процеси S в е-обучението могат да бъдат и процеси за оценяване на качеството [Doneva'16]. Такива процеси по естествен път се включват в настоящия модел на процес за адаптивно обучение.

2.3 Основни модели

Модел на Работен поток

Работният поток (или Работен процес) p е граф – наредена двойка $p = (\mathrm{N}, \mathrm{E})$, съставена от следните две компоненти:

А) Множество възли $\mathrm{N} = \{\mathrm{N_i};\ \mathrm{i} \in \overline{1,\ \mathrm{n}}\}$, $\mathrm{N_i} = (\mathrm{R}_i, \mathrm{D}_i, \mathrm{VR}_i,\ User_i, \mathrm{F}_i, \Gamma_i, p_i, SR_i, SD_i)$, където: $R_i = \{\mathrm{R_{ij}};\ \mathrm{j} \in \overline{1,\ \mathrm{s}}\}$; $\mathrm{s} \in \mathbb{N}$ е множество от обекти – входни данни (наричане още сигнали), т.е. $R_i \in set(O)$; $\mathrm{D}_i = \{\mathrm{D}_{il};\ \mathrm{l} \in \overline{1,\ \mathrm{k}};\}$, $\mathrm{k} \in \mathbb{N}$ – множество от изходни обекти (наричани още слотове), т.е. $D_i \in set(O)$; $\mathrm{VR}_i = \{\mathrm{VR}_{il};\ \mathrm{l} \in \overline{1,\ \mathrm{k}};\}, \mathrm{k} \in \mathbb{N}$ – множество от входни обекти, които потребител може да въвежда ръчно (наричани още потребителски сигнали), т.е. $VR_i \in set(O)$; $\mathrm{VR}_i \subseteq D_i$; $User_i = \{\mathrm{User}_{il};\ \mathrm{l} \in \overline{1,\ \mathrm{k}};\}, \mathrm{k} \in \mathbb{N}$ - множество от потребители, които имат право да изпълняват текущ възел от Работен поток; $\mathrm{F}_i = \{\mathrm{F}_{ip};\ \mathrm{p} \in \overline{1,\ \varepsilon}\}, \varepsilon \in \mathbb{N}$ – множество от релации за създаване на изходни данни (наричани още генериращи функции), такива, че F_{ip}: $R_i \longrightarrow \mathrm{D}_i$; $\Gamma_i = \{\Gamma_{i\phi};\ \phi \in \overline{1,\ \psi}\}$, $\psi \in \mathbb{N}$ – множество от разрешаващи функции, такива, че $\Gamma_{i\phi}$: $R_i \longrightarrow \{true, false\}$; $p_i = \{\mathrm{p}_{ip};\ \mathrm{p} \in \overline{1,\ \varepsilon}\}, \varepsilon \in \mathbb{N}$; $p_i \subset P$; - множество от работни потоци (подпроцеси), които да се стартират автоматично след изпълнението на текущ възел (стъпка), $SR_i = \{\mathrm{SR_{ij}};\ \mathrm{j} \in \overline{1,\ \mathrm{s}}\}$; $\mathrm{s} \in \mathbb{N}$ е множество от обекти – входни данни, които се зареждат по стойност автоматично от сесийно пространство (наричане още сесийни сигнали), т.е. $SR_i \in set(O)$; $SR_i \subset R_i$; $SD_i = \{\mathrm{SD_{ij}};\ \mathrm{j} \in \overline{1,\ \mathrm{s}}\}$; $\mathrm{s} \in \mathbb{N}$ е множество от обекти – изходни данни, които се запзават по стойност автоматично в сесийно пространство (наричане още сесийни слотове), т.е. $SD_i \in set(O)$; $SD_i \subset D_i$; Б. Множество ребра $\mathrm{E} \subseteq N \times N$.

Множеството от всички процеси p ще бележим с P. Инстанция на работен поток ще наричаме наредената четворка $i_p = (p, \Lambda, Y, H)$, където:

- p е работният поток, за който е създадена инстанцията
- Λ е множество от входни параметри, които определят уникално инстанцията, като за всеки обект $\lambda \in \Lambda$ е изпълнено $\lambda \in set(O)$.

• Υ е множество от изходни параметри, които инстанцията на поток връща като резултат от изпълнението си, като за всяко $\upsilon \in \Upsilon$ е изпълнено $\upsilon \in set(O)$.

• H е пространство от обекти, запаметени във вътрешно сесийно пространство за инстанцията от обекти, като за всяко $\eta \in$ H е изпълнено $\eta \in set(O)$.

В Λ и H има специализирани обекти, които обезпечават правилото протичане на инстанциите на работните потоци, като например:

• Обект - идентификатор на текущия потребител, за който се изпълнява инстанцията

• Обект, съдържащ името на работния поток, по който е създадена инстанцията

• Обект – списък, указващ идентификатори на възли, през които дотук е преминало изпълнението на инстанцията на потока

• Обект – списък на активните в инстанцията на потока транзакции

• Множеството I_p е от всички налични инстанции на работен поток p. Функция за избор на наличните инстанции на работен поток ще бележим с: $wfInstances: P \longrightarrow \mathrm{I_p}$.

• Релация $outp: I_P \longrightarrow \Upsilon$ е релация, която връща изходните параметри на инстанция на работен поток $i_p \in I_P$.

Инстанция на съобщение ще наричаме $MessageI = (MT, UF, UsersTo,\ TM,\ As, CourseGoals, M_C, PMess, Message, MO)$, където: MT е заглавие на съобщението; UF е множество от потребители, от които е изпратено съобщението; $UsersTo$ е множество от потребители, от които е получено съобщението; TMе форматиран текст на съобщението; $As \subset All(LA)$ е множество от учебни дейности, за които се отнася съобщението; $CourseGoals \subset\ All(C_G)$ е множество от цели в заявка за обучение M_C, за които се отнася съобщението; $AttachedFiles$ е множество от прикачени файлове към съобщението; $PMess \in All(Message) \cup \varnothing$ е идентификатор на съобщение, за което това съобщение представлява отговор; $Message$ е съобщението на което $MessageI$ е инстанция; $MO = (i_p, N_i,\ MOS)$ индикатор за изпращане на данни в инстанция на работен поток i_p в стъпка , N_i на работен поток p, като атрибутите на обект $MOS \in set(O)$ се предават чрез съответствие по име на сигналите R_i. Трябва да се отбележи, че част от инстанциите на съобщения са автоматично генерирани от АСЕО. Примери за такива автоматично генерирани съобщения са: Съобщение до преподавател за наличие на обучаван с нулева Перспектива в курс; Съобщение до преподавател за наличие на непостижима цел в курс от група обучавани и др.;

Определение 3*: Модел на заявка за обучение* е наредената 5-орка $\mathbf{M_C} = < \mathbf{C_N}, \mathbf{C_G}, \mathbf{Teach}, \mathbf{Stude}, \mathbf{StGr} >$**, където:**

• C_N: име на курса;

• $C_G = \{C_{G_i}: i \in \mathbb{N}, i = \overline{1,o}\},\ o, i \in \mathbb{N}$ е множество от дървета $C_{G_i} \subset T$, за които за всеки възел $t_{c1}, \dots, t_{cn}$ имаме дефинирана функция за съответствие на инстанция на работен поток $getProc: T_C \longrightarrow \mathrm{I_p} \cup \emptyset$

• $Teach = \{Teacher_k: k = \overline{1,q}: k, q \in \mathbb{N}\}$ е множеството от преподаватели,

• $Stude = \{Student_l: l = \overline{1,r}; l, r \in \mathbb{N}\}$ е множеството от обучавани в курса

• $StGr = \{StGr_1, \dots, StGr_m\}, StGr_k \subseteq Students\ 1 \leq k \leq m,$ където $StGr$ е множество от групи от обучавани.

Аспект за многомерно пространство

Определение 4: Аспект за многомерно пространство (АМП) е наредената тройка $\boldsymbol{A} =< \boldsymbol{M}, \boldsymbol{Rep}, \boldsymbol{Dist} >$, където: $\boldsymbol{M}$ е многомерно пространство; $\boldsymbol{Rep}$ е конкретна имплементация на функция – представяне на обект в правоъгълна област със сигнатура: $\boldsymbol{Rep}: \boldsymbol{O} \times \boldsymbol{H} \longrightarrow \boldsymbol{H}$; $\boldsymbol{Dist}$ е функция за разстояние, респективно удовлетворяваща критериите за това.

Предимства на така дефинирания АМП:

- Позволява при появата на нова дидактична таксономия УО да бъдат разглеждани в контекста на нови многомерни пространства, вкл. с различни представяния на тези обекти в новото многомерно пространство и различна функция за разстояние за всеки нов аспект.
- Чрез минимални или нулеви промени в реализацията на рамката на АСЕО да се адаптират нови дидактични таксономии;
- Едни и същи учебни обекти да се разглеждат от различни гледни точки , правила и методики за разглеждането им, което разширява обхвата на възможностите на рамката за АСЕО.

Модел на обучаван в курс

Определение 5: Модел на обучаван в курс $\boldsymbol{M_S}$ е наредената двойка: $\boldsymbol{M_S} =< \boldsymbol{Student_l}, \boldsymbol{T_S} >$, където $\boldsymbol{Student_l}$ е конкретният обучаван, за който се отнася множество от коренови дървета , $\boldsymbol{T_S}$, за които са възможни следните наредени двойки (корен, наследник):

- $(\boldsymbol{M_c}, \boldsymbol{A})$: налична заявка за обучение, аспект, в който се разглеждат обектите, в т.ч. и $Student_l$;
- $(\boldsymbol{A}, \textbf{prospective}(\ Student_l, \boldsymbol{M_c},\ \mathrm{C_G},\ \mathrm{A}))$: аспект и наличната перспектива от **достижими цели** на $Student_l$ за този аспект и тази заявка за обучение $\boldsymbol{M_c}$. Релацията за перспектива на обучаван в курс, аспект и дадени цели в този курс има следната сигнатура: $\boldsymbol{prospective}: Students \times \boldsymbol{M_c} \times \mathrm{C_G} \times \mathrm{A} \longrightarrow \mathrm{C_G}$. Тя връща тези цели $\mathrm{C_G}$, които са понастоящем достижими за обучавания $Student_l$.;
- $(\boldsymbol{A}, accomplished(M_C, A, Student_l))$: аспект и постигнати цели. Релация за постигнати цели на обучаван, в контекста на заявка M_C и аспект A е със следната сигнатура: $accomplished: M_C \times \mathrm{A} \times \mathrm{Students} \longrightarrow \mathrm{set}(\{(\mathrm{C_{G_1}}, i_{p_1}), \ldots, (C_{G_n}, i_{p_n})\})$. Резултатът е множество от наредени двойки (C_{G_k}, i_{p_k}), чиито първи елемент е постигната цел, а вторият елемент е инстанцията на процеса, с който е постигната;
- $(\boldsymbol{M_c}, \boldsymbol{Message1})$: заявка за обучение, както и налична инстанция на съобщение, за която обучаваният е изходящ или входящ потребител;
- $(Message1_1, \boldsymbol{Message1_2})$: налична инстанция на съобщение, за която обучаваният е входящ или изходящ потребител, както и инстанция на съобщение отговор.

Множеството от дървета за всеки обучаван се изгражда динамично, по време на изпълнението на работните потоци по изграждане на оптимални пътища в заявка за обучение, както и по време на работните процеси по акумулиране на метаданни за учебни дейности и оценяване на обучаваните.

Модел на учебни материали и дейности

Преподавател може по всяко време да регистрира в множество от курсове УО. Ако желае, може да въведе и метаданни за тях. По този начин тази учебна единица

може да започне да се включва в ЗО, предлагани на обучаваните, стига метаданните да се съгласуват с учебните цели, зададени от преподавателя. В по-обща ситуация обаче, в заявка за провеждане на обучение преподавателят може да включи учебната единица, без тя да е съпроводена от „свои“ метаданни. В този случай е необходимо да е интегриран специфичен механизъм за автоматизирано генериране на необходимите метаданни.

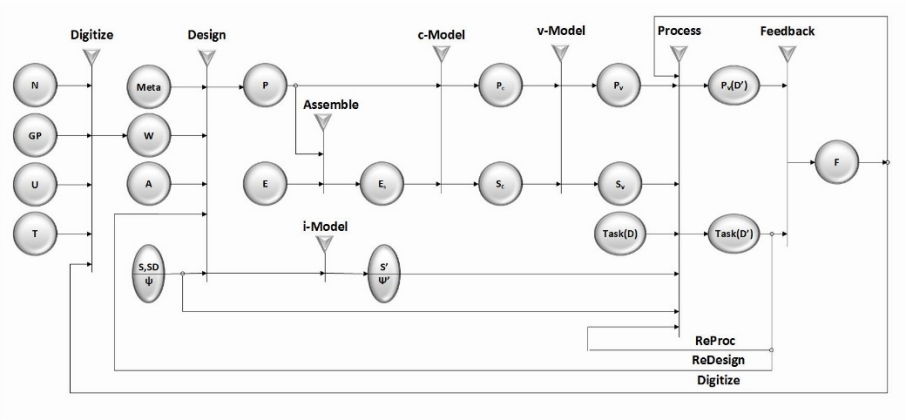
При включване на учебен обект в заявка за обучение, същият получава предимство за включване в УП с цел, предоставяне и изучаване от обучавани, със следващо генериране на специален УО от т. нар. „акумулативен тестов тип“. Акумулативният УО по същество е задание за анотация от обучавани на предявен УО от гледна точка на неговото учебно съдържание, и по същество предоставят на обучаваните ролята на експерти в ИПО, които имат за задача определяне на необходимите метаданни за УО. На базата на въведените (по специален въпросник) екземпляри от метаданни, преподавателят (или експерт в ИПО) може не само автоматизирано да определи експертния набор от метаданни, но и да оцени степента на познаване на същността на УО от обучаваните.

Определение 6: Модел на преподавател е наредената двойка $\boldsymbol{Teacher} = (\boldsymbol{Teacher\_obj}, \boldsymbol{T_{Teacher}})$,където $\boldsymbol{Teacher\_obj} \in \boldsymbol{set}(\boldsymbol{O})$ е обект на конкретен преподавател, за който се отнася динамичното множество от динамични коренови дървета $\boldsymbol{T_{Teacher}}$, за които са възможни следните наредени двойки (корен, наследник):

- $(\boldsymbol{M_c}, Messagel)$: налична заявка за обучение, както и инстанция на съобщение от или към потребител - преподавател
- $(Messagel_1, Messagel_2)$: инстанция на съобщение и отговор на инстанция на съобщение
- $(\boldsymbol{M_c}, Editable)$: налична заявка за обучение, както и булев параметър $Editable \in \{true, false\}$, който указва, дали преподавателят има право да редактира текущия курс, за който се отнася текущото $T_{Teacher}$.
- На най-висшият корен на кореново дърво винаги е заявката за обучение $\boldsymbol{M_c}$, за който се отнася дървото.

2.4 Методика за изграждане на рамка на АСЕО

На фиг. 4 е представен Модел на АСЕО.



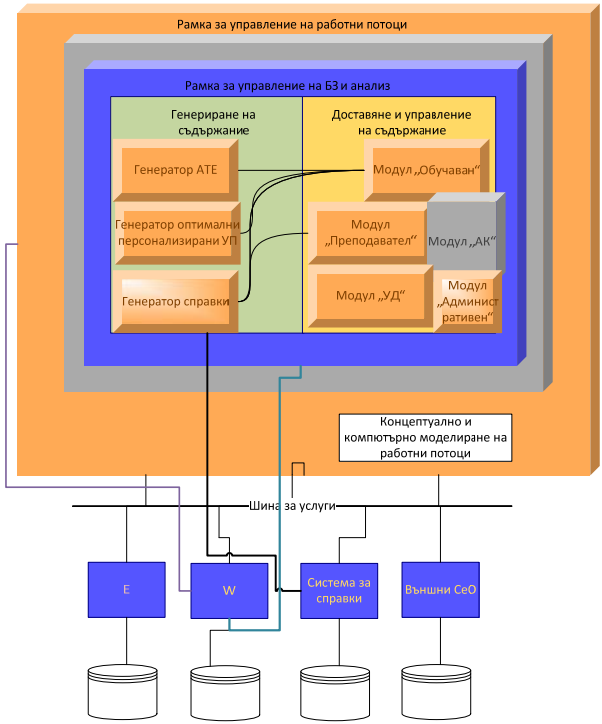
Фигура 4 Модел на АСЕО

Методиката е съвкупност от следните процеси, някои от които вървят паралелно във времето:

- Digitize – Обобщаващ процес, който обобщава Нормативните документи N, добрите практики P и изискванията за потребителя U;
- Design – Процес на моделиране на извличане на метаданните Meta, които са продукт от предишния процес и филтрирането им спрямо предметната област SD; В този процес получените метаданни се филтрират спрямо дадена гледна точка (Аспект) и се получава процес/обект SA за зададен аспект A. При този процес всички външни системи E(S), касаещи този процес/обект S се обработват спрямо разглеждания аспект и стават EA(S). Аналогичното се случва и с външното въздействие ΨA(S);
- C-Model – процес на проектиране и създаване на M в контекста на C. При този процес се създават концептуални модели;
- V-Model – процес на виртуализация на концептуалните модели, получени в предишния процес. В ролята на SV(A) могат да влизат всякакви виртуализирани обекти, както и потоци и инстанции на работни потоци;
- Process – процес на изпълнение на така създадените компютърни модели от предишния процес.

Глава 3. Архитектура на рамка АСЕО

3.1. Обща архитектура на рамка АСЕО



Фигура 5 Обща архитектурна рамка на АСЕО (Логически изглед)

Генераторът на справки се използва както от модул „Преподавател“, така и от модул „Обучаван“.

В Модул „Обучаван“ обучаваните имат възможност да управляват своите заявки за обучение, да преследват цели с тях, като получават генерирани учебни пътища, да провеждат учебни дейности, да ги анотират и да отговарят на автоматизирано генерирани акумулативни тестови единици (АТЕ) за тях, да задават въпроси за всеки един от тези компоненти на наличните преподаватели в контекста на дадена заявка за обучение и др.

Модул „Преподавател“ е основната входна точка на преподавателите в АСЕО. Там те имат възможност да проверяват отговори от анотации или тестове, предадени от студенти и да отговарят на въпроси, зададени от тях за конкретна УД, избрана цел в избрана заявка за обучение (ЗО), да преглеждат автоматично генерирани инстанции на съобщения от различни работни потоци в системата, като например съобщение за наличие на непостижими цели от дадена група от обучавани.

Модул „Администриране на курсове“ функционира за тези, които са посочени за администратори на конкретни ЗО. Там преподаватели имат възможност да задават метаданни за цели в ЗО, както и да добавят задължителни или избираеми цели в ЗО за определени групи от обучавани (кохорти), както и задължителни и избираеми цели за всички обучавани в ЗО. Този модул е достъпен и за администраторите на системата.

Модул „Административен“ е достъпен за администраторите на АСЕО и той им дава възможност да администрират ЗО, потребители, техните роли (администратор, обучаван, преподавател, гост), да причисляват обучаваните в различни групи (кохорти) и др.

Модул „Учебна дейност“ (УД) дава възможност на администраторите на АСЕО да регистрират и импортират УД в АСЕО и в последствие да причисляват УДи към определени ЗО, за да може в последствие да се използват пълноценно в ЗО, включително и от eLP_GA да ги подбира при генерирането на оптимални персонализирани учебни пътища.

3.2. Техническо-приложен изглед на архитектурна рамка на АСЕО

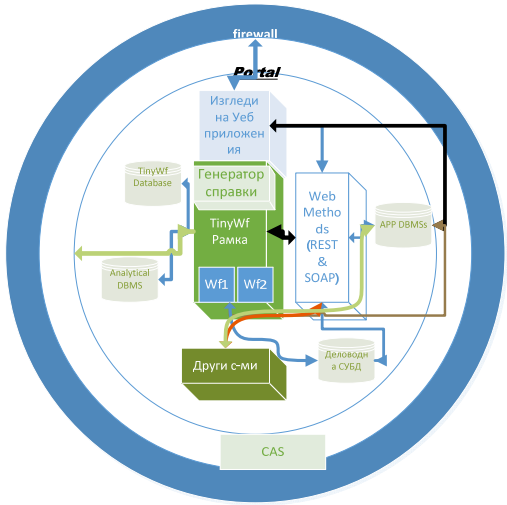
На фиг. 5 е представена архитектурна схема на приложението, където може да се види как са интегрирани различните компоненти в схемата , както идейно, така и в текущата експериментална постановка. Достъпът на потребителите става чрез портала PeuPortal, който е uPortal базиран. В него потребителят може да получи достъп със своя единен университетски акаунт чрез услуга за централна автентификация (CAS Server).

В цялостната архитектурна рамка, интерфейсът на повечето от приложенията е реализиран, чрез автоматичното му генериране от системата за изпълнение на работни потоци TinyWf, която е резултат от текущото изследване [Пашев, Г. (2015)]. Все пак, допустимо е използването на изгледи от уеб приложения, които не са реализирани с рамката за работни потоци TinyWf, включително и такива, реализирани с други рамки за изпълнение на работни потоци. В инстанциите от работните потоци I_P (напр. Wf1, Wf2, …) в изходните обекти Υ се добавят и обекти, които в слоя за визуализация на изхода от работни процеси се трансформират в подходящи входно – изходни екрани за визуализация на потребител.

Jasper Reports Server се достъпва чрез библиотечни разширения в TinyWf, Moodle REST API – директно от TinyWf, а различните REST услуги, консумирани от TinyWf достъпват СУБД на приложните системи, за които са изградени [(Георги Пашев А. Т., 2015)]. Jasper разполага с мощно средство за създаване и извеждане на шаблони на справки в различни формати (pdf, xls, doc, html и др.). Шаблоните могат да бъдат съхранявани на Jasper Server и да се достъпват отдалечено.

Архитектурата на приложението е 4-слойна и включва:

- **Слой 1.** Релационна база данни, съдържаща референции, съхранени процедури, изгледи и функции. Заявките към базата данни, изпратени от по-горни слоеве, са разположени тук, с цел постигане на възможно най-висока скорост при изпълнение на заявка. Приложен е принципът за най-близко разполагане на данни и средства за тяхната обработка, използван при алгоритми за оптимизация;
- **Слой 2.** Транспортен слой (за връзка между слой 1. и слой 2.);
- **Слой3.** Дефиниране на работни процеси, вкл. за генериране на интерфейс чрез REST или SOAP протоколи. За дефиниция на възлите на бизнес процеса се използват URL шаблони;
- **Слой 4.** Потребителски слой, който предоставя достъп на работни процеси (в зависимост от потребителски роли, права и лични настройки) до процеси на самата система.



Фигура 6 Техническо-приложен изглед на архитектура на рамка на АСЕО

За реализация на слой 1. е избран релационен модел. За улеснение на процеса на генериране на интерфейса в слой 3. е приета конвенция за именуване на обектите в базата данни. Името на полето, което представлява външен ключ към дадена таблица започва с „ID_“ и след това се изписва името на свързаната таблица. Потребителските действия, свързани с данните в базата данни, са капсулирани в съхранени процедури, изгледи и функции. Голяма част от бизнес-логиката е съхранена по подо-

бен начин. Това позволява на по-горните слоеве да се фокусират върху типове дейности, които са по-подходящи за тяхното разположение.

Транспортен слой 2. по същество е доставчик на услуги. Уеб услугата е метод за комуникация между две електронни устройства в мрежа, която на практика предоставя функционалност, достъпна на мрежов адрес в уеб пространството. W3C дефинира уеб услугата като софтуерна система, създадена да подпомага комуникацията между оперативно съвместими машини по мрежата. Поддържат се протоколи като REST, SOAP и др. Слой 2. е важна част от интеграцията, защото предоставя на Слой 3. уеб услуги за консумация. Последните могат да бъдат от разнотипни приложения, което е в основата на тяхната интеграция.

Слой 3. Слой за генериране на потребителски интерфейси

Всяко интегрирано приложение включва набор от работни процеси. За проектиране и създаване на интегрирано приложение е необходимо да се реализират множество от работни процеси, съпроводени от съответни потребителски интерфейси. За автоматизация на тези дейности е приет следният подход: На базата на *формален модел на работен поток* се предоставят средства за описание на конкретни работни процеси и в слой 3. при необходимост се генерират потребителски интерфейси за съпровождане на бизнес процесите.

Слой 4. Потребителски слой

За осъществяване на слой 4. на архитектурата се изисква наличие на уеб портал със следните минимални функционалности: едновременен достъп на даден потребител до разнотипни системи в зависимост от неговите роли и права в тях; общ изглед към достъпните функционалности, респективно работни процеси в електронно портфолио на потребителя; поддържане на заявки към LDAP и релационни бази данни; предоставяне на потребителски услуги, компоненти и работни рамки за свързване на по-сложни системи; гъвкава система за управление на потребителски групи, вкл. реализация на вложени групи; възможности за използване на API и др.

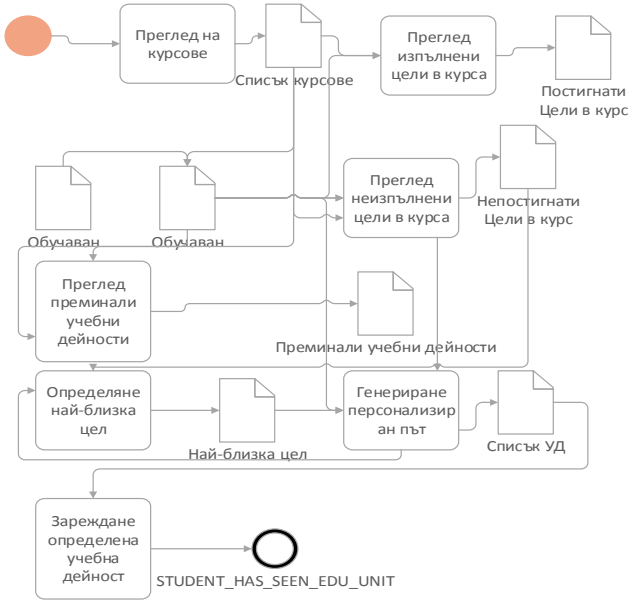
На базата на проучване и анализ на различни уеб портали, за експерименталната реализация на слой 4. е избран uPortal [uPortal]. Освен че притежава горните характеристики, uPortal се отличава и с други полезни характеристики: базиран е на отворени стандарти, Java, XML, XSLT, JSP и J2EE; ползва се от милиони потребители; разполага с възможности за идентификация на потребители, вкл. LDAP и база данни; предлага връзки с библиотеки и средства за интеграция като CAS; single sign–on и др.

TinyWf (от Tiny Workflow Framework) е софтуерна рамка, реализирана на PHP [(Георги Пашев А. Т., 2015)], на базата на която могат да се проектират приложения, интегриращи функционалности на хетерогенни системи, и ползващи формално дефинирани бизнес процеси.

3.3. Модул „Обучаван в курс“

При избор на конкретна цел, се преминава към генериране на персонализиран учебен път за обучавания (съответният алгоритъм, в термините на многомерното пространство е скициран на фиг. 7.). Генерираният персонализиран учебен път представлява списък от нови учебни обекти, които обучаваният трябва да „премине“ успешно, за да постигне учебната цел. Всяка една от учебните обекти в УП може да е съпроводена (или не) с достатъчно изчерпателни метаданни, които определят местоположението й в Пространството. Ако учебната единица е наскоро добавена се пред-

полага, че акумулираните метаданни, вероятно, все още не са адекватни на нейното съдържание.



Фигура 7 Работен поток на обучаван

За определяне на условието за достатъчността на наличните метаданни се използва метрика: брой проведени генериращи акумулативни тестове за съответната учебна единица. При сравнително малък брой, на обучаваният се предлага автоматично генериран акумулативен тест. След като обучаваният изпълни теста, се изпраща сигнал към преподавател, че е предложено ново решение, което трябва да бъде проверено и оценено. При оценяване на отговор от преподавателя като правилен, автоматично се генерира функция/предикат (и метаданни), с която съответната учебна единица ще продължи да участва в е-обучението и при автоматичното генериране на учебни пътища.

3.4. Генератор на пресонализирани учебни пътища

Алгоритъмът за генериране на персонализирани учебни пътища (*eLP_GA*) може да се дефинира формално като функция със следната сигнатура:

$$eLP_{GA}: Student \longrightarrow [(\zeta, \varsigma, \upsilon)] \quad (3)$$

Входният параметър на функцията е обучаван $Student_l$. Функцията връща списък от наредени тройки $[(\zeta, \varsigma, \upsilon)]$, които отговарят на учебните дейности, където: ζ – е URL, който уникално идентифицира УД, ς – е разстояние между УД и състоянието на текущия обучаван M_S и представянето на това състояние в многомерно пространство (МП), а υ – е кратко име на УД.[(Pashev & Totkov, 29-May-2014)], [(Георги Пашев Г. Т., 2016)].

В това, което следва нека да обозначим:

- *RVec(Obj) – представяне на обект Obj* като радиус – вектор в *МП;*

• $V_L(RVec(Obj),\ (\omega,[slot]))$ – върната стойност за даден *RVec(Obj)* за измерение, което се идентифицира с глагол ω и списък с входни параметри $[slot]$;

• $search$: $BasePoint\ x\ reachRadius\ x\ searchFilter\ x\ n \longrightarrow [[(\omega,[slot],val)]]$ – функция, която намира *n* най-близки точки към *BasePoint*, достижими в рамките на радиус *reachRadius* и отговарят на филтъра за търсене $searchFilter$
$searchFilter$: $booleanExpression \longrightarrow \rho$: $\rho \subseteq P_L$;

• $searchGE$: $BasePoint\ x\ reachRadius\ x\ searchFilter\ x\ n \longrightarrow$ $[[(\omega,[slot],val)]]$ – функция, която (за разлика от $search$) връща списък от точки, които имат абсолютни стойности, които са по-големи или равни на абсолютната стойност на *BasePoint*. Списъкът има максимален размер от *n* елемента, които отговарят на търсещия филтър *searchFilter;*

• $orderTo$: $BasePoint\ x\ [[(\omega,[slot],val)]] \longrightarrow [[(\omega,[slot],val)]]$ – подрежда списък от точки в МП по близост до *BasePoint.*

• $isGE(RVec(objWhat),RVec(objToWhat))$ – булева функция (предикат), която проверява дали абсолютната стойност на $RVec(objWhat)$ е по-голяма от тази на $RVec(objToWhat)$ и ако е, връща *true.*

```
1  studPoint = RVec(SO); {locate student point in MDS }
2  maxReachRadius=((a=V_L (studPoint,[(maxReach,[SO])]))===null)
3  ?MAX_RADIUS_VAL:a;
4
5  uncompCGs=orderTo(studPoint,searchGE(studPoint, maxReachRadius, "isCourseGoal = 1 AND
   CompletedBY("+SO.ID+") <> 1"))
6
7  if(incompCGs.length===0) exit("No course goals available right now.");
8
9  selCG=uncompCGs[0];
10
11 retLAs=[];
12
13 V_L(selCG,[(obligLA,[:id_CG])]).foreach(function(dimObject){retLAs.push(getLA(dimObject.val))});
14
15 V_L (selCG,[(obligLA,[:id_CG ])]).
16 foreach(function(dimObject){dimObject=null});
17     while(1){
18
19         foreach(curRetLA  in
           retLAs){V_L(curRetLA).foreach(function(dimObject){studPoint.addOrUpdate(dimObject)})}
20
21         potentialLAs=orderTo(studPoint,searchGE(studPoint, maxReachRadius, "isLA = 1 AND
           prevPresentedTo("+SO.ID+")<3"));
22         C: /*Check if studPoint has all dimension values greater or equal than those of selCG,
           i.e. studPoint has accomplished selCG;*/
23         if(isGE(studPoint, selCG)){
24             exit("successful path generation", retLAs);
25         }
26         if(potentialLAs.length===0) exit("Error: no suitable LAs found");
27         curLA = potentialLAs[0];
28         potentialLAs.shift();
29         curLA.push(curLA);
30         V_L(curLA).foreach(function(dimObject){studPoint.addOrUpdate(dimObject)});
31 goto C;
32 }
33
```

Фигура 8 Algorithm for Personalized eLP Generation

Фиг. 8. представя eLP_GA. Алгоритъмът за генериране на персонализирани учебни пътища (eLP_GA) започва с инициализация на някои променливи като $studPoint$ (текущото представяне на обучаван в МП). За този обучаван се инициализира стойност на максимален радиус на достижимост $maxReachRadius$, като проверява дали за обучавания вече има динамично изчислен такъв, а ако няма, слага стойност по подразбиране (ред 2-3 от фиг. 8). Редове 5-7 извършват намиране на непостигнати цели за текущия обучаван в рамките на текущата заявка за обучение. Ако такива цели не са намерени, се излиза от алгоритъма със съответното съобщение.

При наличие на непостигнати учебни цели, първата от тях става текуща (ред 9). Освен това, се инициализира (на ред 11) празен списък от УД, който евентуално след приключването на алгоритъма ще съдържа УД, с които обучавания би могъл да постигне текущата УЦ. На ред 13 се извършва проверка дали в избраната учебна цел се съдържат предикати, които указват наличие на задължителна учебна дейност за постигане на тази учебна цел, и ако се съдържат такива, те се добавят в списъка с УД. На ред 15 във временният обект на УЦ в паметта се премахват тези измерения, които съдържат тези предикати, за да може алгоритъмът да работи правилно след това. Всяка от УД, които вече се намират в списъка, могат да съдържат допълнителни предикати/функции в измеренията, такива, че при съответното преминаване на обучавания през тях, неговият радиус-вектор в МП да постига УЦ. Именно затова, неговата точка се „премества“ в динамичната памет на МП, така, все едно е преминала през наличните УД (ред 19). На ред 21 в МП се търсят такива УД, които биха били потенциални за постигането на УЦ: същевременно са подредени по близост до УЦ и се намират в сферата на достижимост, дефинирана с радиуса на достижимост и централна точка: точката на обучавания в МП. Търсенето се оптимизира още повече, като се добавя във филтъра и критерий учебната дейност да не се предлага на обучавания повече от 2 пъти при генериране на различни учебни пътища. В ред 23-25 се прави проверка дали точката на обучавания в МП не се преместила в МП така, че вече да е постигната УЦ. Ако е така, се излиза от алгоритъма със съобщение за успех и с наличния списък от УД. На ред 26 се прави проверка дали списъка с потенциални УД е празен и ако е така, влизаме в хипотеза на невъзможност за генериране на списък от УД и се излиза от алгоритъма с такова съобщение.

При наличие на УД в списъка с потенциални, на принципа на опашката се изважда първата в списъка и се въвежда в списъка с УД, с който би трябвало да постигне УЦ. В последствие точката на обучавания в МП се премества така, все едно е преминала и през тази УД и се повтаря нова итерация на алгоритъма.

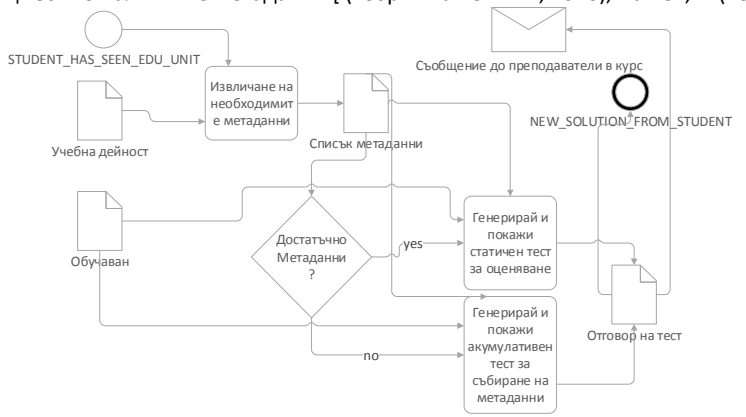
Трябва да се отбележи, че се постига известна оптимизация на генерирания списък на УЦ, понеже списъкът с потенциални УД е подреден по близост на УД до УЦ. Друга оптимизация е свързана с проверка преди влизане в цикъла на алгоритъма дали в сферата на достижимост на обучавания няма поне 1 УД, която е максимално близко до УЦ, но е с по-големи или равни координати за измеренията от УЦ (проверка за постигане на УЦ само с 1 УД).

3.5. Генератор на тестови единици

Важен въпрос при адаптивните системи, които генерират учебни пътища е – с какви тестове да се оценяват знанията на обучаваните. Първоначално тези тестове не присъстват в описанието на курса, ако той се описва абстрактно – с учебни цели за постигане.

В тази посока е предложена методика и система за реализация на подпроцес „автоматизирано генериране на метаданни“, която интегрира специфична двумерна рамка по разширената таксономия на Блум (наричана по-нататък „Рамката“). На всяка метаданна по тази методика отговаря еднозначно определена клетка в Рамката, с конкретен тип отворен въпрос. При вече известна стойност на предикат, оценена като вярна от преподавател, се знае и верният/те отговор/и на въпроса. Следователно, от наличните метаданни за учебна единица може да се генерира тест, с който да

се проверят знанията/уменията на обучавания на различни равнища по таксономията на Блум (Запаметяване, Разбиране, Приложение, Анализ и т.н.). Тази особеност се използва от адаптивната система, която (при условие, че има достатъчно метаданни за тази учебна единица) стартира модул за генериране на акумулативни тестови единици на различни равнища по разширената таксономия на Блум, който генерира подходящ тест по наличните метаданни [(Георги Пашев Г. Т., 2016), Пашев, Г. (2014)].



Фигура 9 Подпроцес „Автоматизирано генериране на метаданни“

Понеже верните отговори са на разположение, оценката може веднага да се генерира. При преподавателска реакция и получена окончателна оценка, текущото състояние на обучавания се актуализира като позиция в Пространството.

3.6. Рамка за работни потоци

Резултатите от изпълнението на i_P под формата на Υ за визуализация от Генератора на интерфейс. Самите работни потоци могат да бъдат така имплементирани, че да използват уеб услуги за комуникация с различни приложения, което осигурява естествена методология и инструментариум за интеграция на разнотипни системи чрез изпълнение на работни потоци. [(Георги Пашев А. Т., 2015)]

Част от работните потоци записват данни в Аналитична СУБД, която предоставя удобен инструментариум за различни типове обобщаващи заявки и заявки за подобие/близост на обекти по дадена заявка за търсене. В контекста на АСЕО, такива заявки биха били например: намери тези учебни учебни дейности, които са най-близо до тази учебна цел.

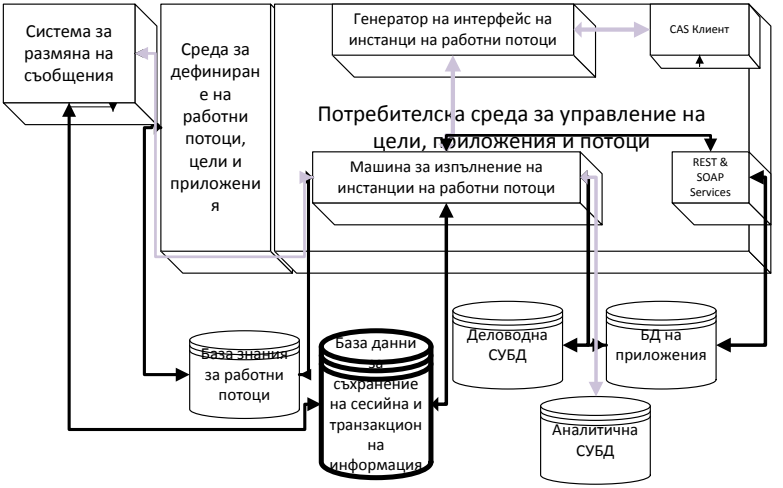
Деловодна СУБД се използва от някои работни потоци за съхранение и извличане на различна деловодна история за обучавани и преподаватели.

За правилното управление на която и да е инстанция на работен поток i_P е необходимо използването на БД, в която да се съхранява, актуализира, извлича сесийната и транзакционната информация в работен поток p.

Системата за размяна на съобщения се използва за размяна на инстанции на съобщения $MessageI$ между потребители/инстанции на работни потоци и потребители/инстанции на работни потоци. Тя използва БД за сесийна и транзакционна информация, за да управлява съобщенията и инстанциите на съобщения.

Всеки конкретен работен поток може да използва директно или чрез обръщение към уеб методи в REST или SOAP уеб услуги БД на различни други приложения. В контекста на АСЕО, примери за други такива системи биха били:

- Централна деловодна система за студенти/обучавани;
- Регистър на преподаватели;
- Външни системи за управление на електронни курсове (напр. Moodle).



Фигура 10 Архитектура на Рамка за изпълнение на Работни потоци TinyWf

Глава 4. Реализация на рамка АСЕО

4.1. Рамка за изпълнение на работни потоци

Всяко състояние на даден бизнес процес се представя с конкретен възел от съответното множество N и се определя от стойностите на своите входни и изходни атрибути (данни). Изходни (както и входни) атрибути на дадено състояние се проектират като сесийни променливи на входни атрибути на други състояния. Всяко състояние на работния процес е резултат на поне едно извикване на уеб метод. Бизнес процесите се задават „в рамките на“ отделните потребителски групи (възможни са конфигурации на различни потребителски групи). Входните данни (атрибути) на състояние се задават „ръчно“ (въвеждане от оторизиран потребител) или автоматично – като „проекция“ на сесийна променливи на изходни атрибути на предходно състояние на процеса. Изходните данни (атрибути) на състояние се определят след като потребител се получи или избере резултат от изпълнение на уеб метода. Част от възможните следващи състояния се филтрират на базата на условия, които могат да се зададат при описанието на съответния бизнес процес.

Парадигмата, реализирана в TINYWF2, следва стратегия за постигане на цели, които могат да бъдат осъществени на конкретни стъпки на отделните бизнес процеси, и с участието на точно определени потребителски групи. Всяка стъпка на бизнес процес се стартира („активира“) и се изпълнява при изпълнение на определена група от условия, между които:

- право за изпълнение от текущ потребител –член на оторизирана потребителска група;
- достъпни и осигурени са всички данни за стартиране на поредната стъпка;
- данни, необходими за активиране на стъпката не са налице, но е допустимо да бъдат „ръчно“ въведени от потребител;
- изпълнени са предусловия, които разрешават паралелно изпълнение на функции, генериращи изходни данни (слотове) и т.н.

Изпълнението на стъпка на конкретен бизнес процес може да се опише по следния начин:

- в сесийното пространство се зареждат променливите, **видими за инстанцията на процеса**;
- проверяват се условията от по-горе;
- при отсъствие на всички сигнали в сесийното пространство – предпоставка за стартиране на съответната стъпка, се проверява дали потребителят може да ги зададе ръчно. Ако е допустимо, на потребителя се осигурява възможност за задаване на съответните стойности в открития за целта диалогов прозорец;
- отново се проверява дали са изпълнени необходимите условия;
- ако условията са изпълнени се преминава към изпълнение на функциите, генериращи изходни данни (слотове) за възлите;
- при успешно генериране на слотове, някои от тях (за които има подобно указание) се запазват в сесийното пространство на инстанция на текущия, или на друг процес (ако това е указано в неговото описание).

Изпълнението на инстанция на работен процес протича по следния начин:

- определя се уникалния идентификатор на инстанцията като редица seq (име на процес, име на потребител, променлива 1, ..., променлива n);
- проверява се дали в базата данни на TINYWF2 е записан контекста на инстанцията, включващ списък на вече „изпълнени“ възли, както и запазеното обкръжение от променливи за текущата инстанция, и дали е в състояние „действащ“;
- генерира се списък от стъпки на процеса, които все още не са приключили като се отчита и факта – дали текущият потребител е член на потребителска група, която има право да ги изпълнява;
- стъпките се изпълняват по представения по-горе начин, като това продължава до изпълнение на едно от следните условия – изпразване на списъка от възможни следващи стъпки на процеса, приключване на процеса, невъзможност за продължаване по-нататък;
- сесийния контекст и историята на изпълнение на отделните стъпки на процесната инстанция, се запазват, ако това е указано в общите настройки на процеса.

След всяко изпълнение на отделна стъпка, автоматично се запазва следа от изпълнението („отпечатък“ на стъпката), вкл. стойности на сигнали, слотове, постигнати цели и др. По-късно подобни „отпечатъци“ се използват за формиране на обобща-

ващи аналитични заявки в Elasticsearch относно процеси, които протичат в университетската инфоструктура.

От своя страна, отпечатък от изпълнението на всяка стъпка на даден процес (ако това изрично е указано в настройките) е елемент от JSON масив в пространството на контекстите на процеси, в които се съхраняват всички контексти и статуси на процесите в исторически план.

За работните потоци, които са маркирани като запомнящи се, се възстановява сесийната информация, която е била записана от предишно изпълнение на инстанцията на работен поток или от други инстанции на други работни потоци, които са изпращали съобщение към настоящата инстанция. За целта се прочитат сесийни файлове, чиито имена са функция от името на потребителя и името на изпълнявания работен поток. Тези файлове са в json формат и съсдържат целия сесиен контекст на съответната инстанция на работен поток. В асоциативния масив _SESSION се записват всички тези сесийни променливи. Преди това този масив е зачистен от всякакви други сесийни променливи, освен тези, които са свързани с информация за потребител и текущия изпълняван работен поток.

Преди изпълнението на инстанцията на работен поток трябва да се определи от кой възел да се стартира/продължи изпълнението на инстанцията на работен поток. Ако в сесийния контекст има записана сесийна променлива $_SESSION ['FSM_STARTING_NODE'.$sessfile], от нея се зарежда идентификатора на стартовия възел, от който се продължава изпълнението. Трябва да се отбележи, че самият работен поток може програмно да промени стартовия възел, ако логиката му го изисква чрез директно манипулиране на стойността на глобалната променлива FSM_starting_node. Чрез манипулиране на FSM_cur_state_id може да се смени текущия възел при запазване на информацията за стартов възел. Това позволява при определени случаи да се наруши строгото описание на работния поток, където ясно и точно е описано кой възел след кого следва.

В течение на изпълнението на раборн поток, текущата стъпка може да е маркирана като „autosubmit“, което означава, че входните данни за стъпката се предават автоматично и стъпката се изпълнява автоматично, без да има нужда от потребителска интеракция. Асоциативния масив FSM_states съдържа инфорамция за работния поток, а $FSM_states[$FSM_cur_state_id] е асоциативен масив, съдържащ информация за текущата изпълнявана стъпка. В текущата имплементация има няколко вида режима на autosubmit:

- autosubmit: автоматично предаване на входни данни на състояние
- autosubmit1: автоматично предаване на изходни данни на стъпка към следваща стъпка
- autosubmitIfOneRecord: автоматично предаване на изходни данни на стъпка към следваща стъпка, ако резултата от изпълнението на текущата стъпка е само един като брой (и реално няма смисъл от потребителски избор на резултат).

В описанието на работния поток в различни стъпки има дефинирани различни callback функции, които следва да бъдат изпълнени. Една от тези функции е например viewFunctionInputDisplay, която се изпълнява веднъж преди създаването на формата за въвеждане на входни данни от потребител. Тя е подходяща за инициализация на различни променливи или за показване на допълнителна информация на потребите-

ля за стъпката. Отсъствието на $_REQUEST['exec_operation'] показва, че трябва тепърва да се изчвртае формата, с която се изпълнява текущата стъпка. Информацията, която се намира в тази форма е и информацията, необходима за изпълнението на стъпката. Наличието на флаг $FSM_states[$FSM_cur_state_id]["infoStep"] показва, че текущо изпълняваната стъпка е само информационна, което значи, че тя само показва на екрана някакви данни, без да изпълнява някакъв уеб метод от мрежовия слой с услуги. Асоциативният масив $FSM_states[$FSM_cur_state_id]["disabledFields"] съдържа имената на входните полета, които са забранени за вход от потребител. Асоциативният масив $FSM_states[$FSM_cur_state_id]["hiddenInputFields"] съдържа информация за входните полета, които са скрити за потребителя. Асоциативният масив $FSM_states [$FSM_cur_state_id["datePickerFields"] съдържа информация за тези входни полета, за които трябва да се изчертава контрол за избор на дата, ако входната данна има смисъл на дата. Много важен механизъм при изпълнението на работния поток е проектиране на данни от и към сесийното пространство към вход/изход на стъпка. На Фиг. 9 е показана пример за употреба на механизъм, който проектира тези сесийни променливи, които са посочени в работния поток като автоматично проектиращи се във входни данни на стъпка.

```
$get_from_projection_input=array(
    1=>array("node"=>4, "field"=>"ID_Student",
"cookieName"=>"ChosenStudent_ID"),
);
```

Фигура 11 Информация за проектиране от сесийно пространство в стъпка

На фиг. 11 е дадено примерно описание на правила за автоматично прехвърляне на стойност от сесийно пространство от сесийна променлива ChosenStudent_ID към входно поле ID_Student на възел 4.

След като входните данни на стъпката са въведени от потребител или програмно от изпълнение на работен поток, е необходимо те да бъдат валидирани. В рамката за изпълнение на работни потоци има вграден механизъм, който позволява в работния поток да се дефинират валидиращи функции.

При наличие на валидираща функция, която има име от вида X _validateInput_Y, където X е името на работния поток, а Y е идентификатор на текущата стъпка, тя се възприема от рамката за валидираща функция, която може да приеме или да отхвърли изпълнението на стъпката. Ако такава функция съществува и тя върне Boolean false, тогава се приема, че входните данни са невалидни и изпълнението на стъпката не се разрешава.

Наличието на флаг $FSM_states[$FSM_cur_state_id]["rememberLastState"] автоматично стартира този процес. В последствие работния поток може да използва тази информация, за да проследи какво за последно е въвеждал потребителя при изпълнението на някакви предишни стъпки, както и да разбере дали тези стъпки изобщо са били изпълнявани някога. Освен това, този механизъм може да се използва за автоматизирано събиране на метаданни на някакви обекти (например документи), които са резултат от изпълнението на текущата инстанция на работен поток, като самата история на изпълнение на работния поток дава уникален отпечатък за метаданните на генерирания документ и в какъв точно контекст е бил генериран.

```
$array_obj=array();
if(!isset($FSM_states[$FSM_cur_state_id]["infoStep"])){
        if($isValidatedInput)
        getResultforUrlSufix($url_suffix, $array_obj,
isset($FSM_states[$FSM_cur_state_id]["httpMethod"])?($FSM_states[$FSM
 cur state id]["httpMethod"]):"POST",
isset($FSM_states[$FSM_cur_state_id]["templateGetPart"])?($FSM_states
[$FSM_cur_state_id]["templateGetPart"]):"", $variableWithvalues);
      }
```

Фигура 12 Изпълнение на уеб метод на стъпка

Ако текущата стъпка не е маркирана като информационна, се извиква уеб метода от мрежовия слой, който отговаря на текущата стъпка и резултатът от изпълнение, който е релация се трупа във вид на масив от обекти в масива $array_obj.

Всяка стъпка се включва в работния поток като член на асоциативния масив $FSM_states. Чрез "template" се генерира линнк към уеб метода, който трябва да се изпълни, при изпълнението на текущата стъпка. В този шаблон полетата се описват във фигурни скоби, например {ID_Student;Студент}. httpMethod указва какъв е типът на HTTP заявката, която трябва да се изпълни. viewFunctionOutputDisplay е callback функция, която се изпълнява след като се изчертае таблицата с резултат от изпълненние на стъпката.

Рамката за изпълнение на работни потоци очаква да получи JSON асоциативен масив от уеб метода, който стъпката извиква при изпълнението си. Има вграден механизъм за автоматична замяна на имената на атрибутите от JSON масива с налични термини в речник, който е достъпен за всичко работни потоци. По този начин, при създаване на работен поток, ако се използват едни и същи термини за именуване на атрибутите на заявките, няма да има нужда те да се въвеждат наново в езиков файл, а директно ще бъдат извлечени от речника с имена на атрибути и техните съответствия като имена за пред потребител. За речник се използва асоциативния масив $aliases.

Рамката има и вграден механизъм за филтриране на изходните данни по стойности на някой/и от атрибутите.

$FSM_states[$FSM_cur_state_id]["showFilter"] съдържа името на изходен атрибут, по който се създава автоматизирано филтриране. filterDisplayVal от заявка съдържа стойносттана данната, по която се филтрира, а filterDefaultVal е стойността по подразбиране.

За по-голямо удобство на потребителя е предвиден механизъм за вграждани на потребителски контроли в изходната таблична форма на стъпка. При наличие на callback функция $FSM_states[$FSM_cur_state_id]["instantLineEditField"], тя се извиква и създава специфичен HTML, който да се вгради на всеки ред от колоната на $FSM_states[$FSM_cur_state_id]["instantLineEditFieldFieldName"]. Това позволява разработчикът на работния поток да управлява както желае начинът по който изходното съдържание се показва и/или манипулира за всяка колона на изходната таблица.

```
            if(substr($keyvals[$j], 0, 5)!="ID_N_"){
                echo $cur_res->{$keyvals[$j]};
            }else{
                $aaa=null;
                                        echo
cache searchInNomenclCache(substr($keyvals[$j], 5), $cur res-
>{$keyvals[$j]});
            }
```

Фигура 2 Автоматизирано изграждане на изглед за номенклатурни полета

На фиг 13 е показана автоматизация при работа с номенклатурни атрибути на изходна релация от изпълнение на стъпка на работния процес. По конвенция е прието, че всички номенлатурни атрибути имат име от вида ID_N_Table, където N_Table е някаква номенклатурна релация в БД на приложението на съответния работен поток, която има задължителни атрибути ID и Name. При наличие на такива номенклатурни атрибути в изходната релация, техните ID стойности автоматично се заменят в изгледа с говорящи за потребителя Name стойности. Понеже това е свързано с голям брой заявки към мрежовия слой за сравнително кратко време (за изчертаване на всеки ред от изходната таблица), е вграден механизъм за кеширане на вече известните номенклатурни стойности.

4.2. Модул „Обучаван в курс“

В текущата реализация Модул „Обучаван в курс“ е реализиран като работен поток. Тук ще представим по-важната част от кода на работния поток и на имплементациите на някои от възлите му. Спазва се четирислойната архитектурна рамка, разгледана в гл. 3, в която има слой с БД съхранени процедури, мрежови слой, слой за генериране на ГПИ чрез изпълнение на работни потоци и потребителски портал.

В контекста на TinyWf работната рамка, е необходимо да се посочат някои настройки в работния поток, като например базов URL адрес на мрежовия слой, който подсигурява уеб методите, които ще изпълняват възлите в работните потоци.

```
13    include_once dirname(__FILE__) ."/LIB.php";
14
15    $SERVICE_PREFIX="http://127.0.0.1/PUAdapt/service.php";
16
17
18    $DEFAULT_REACH_RADIUS=5;
19
```

Фигура 14 Базови настройки на работен поток

На ред 15 на фиг. 14 е показана настройката за URL адрес на мрежовия слой, а настройката на ред 18 е свързана с имплементацията на алгоритма за генериране на оптимални учебни пътища. Това по същество е радиусът на достижимост на обекта на обучавания, за който се генерира учебния път, ако за неговия обект няма зададен такъв радиус.

Тук ще покажем описанието на някои от по-важните стъпки в работния поток.

```
22    1 =>array("name"=>"SelectMenu", "caption"=>"Следващи действия", "infoStep"=>true,
23        "nextTransitionView"=>true,
24        "viewFunctionOutputDisplay"=>function(){
25
26        },
27
28    "viewFunctionInputDisplay"=>function(){
29
30        },
31    "hiddenInputFields"=>array(),
32    "hiddenOutputFields"=>array(),
33     "template"=>"",
34                "httpMethod"=>"GET",
```

Фигура 35 Стъпка "Следващи действия"

На фиг 15 е показана дефиницията на стъпка, която играе ролята на начално меню за обучавания. Флагът infoStep на ред 22 показва, че текущата стъпка е чисто

информационна и не е свързана с автоматизирано изпълнение на уеб метод, вградено като механизъм в самия TinyWf. При този изглед ще се покажат преходите, които са дефинирани от тази стъпка към изходящите и стъпки, чиято дефиниция ще покажем по-долу.

Една от важните стъпки в работния поток е показването на целите в адаптивната заявка за обучение, които обучавания има да постига.

На тази стъпка обучавания трябва да избере учебна цел от тези, която би желал да постига и за която на следващата стъпа ще мъ се генерира автоматизирано персонален учебен път от учебни дейности.

На фиг. 16 е показана имплементацията на стъпка GenerateLearningPath, която извиква уеб метод със същото име. Това е стъпката, която създава и като резултат от изпълнението си показва на екрана учебните дейности, които участват в генерирания учебен път или съобщение за грешка, ако по някаква причина генерираният път не е възможно да бъде създаден.

```
5 =>array("name"=>"GenerateLearningPath", "caption"=>"Генериран на учебен
път",

"template"=>"?procName={procName;Процедура;.;GenerateLearningPath}&ID_Studen
t={ID_Student;Студент}&ID_Course={ID_Course;Курс}&ID_CG={ID_CG;Учебна цел}",
    "autosubmit"=>true, "rememberLastState"=>true, "autosubmitNoLastState"
=> true,
    "httpMethod"=>"GET",
    "hiddenInputFields"=>array("procName"),
    ),
```

Фигура 46 Имплементация на стъпка "GenerateLearningPath"

В уеб метода със същото име е реализацията на алгоритъма *eLP_GA* (виж гл. 3).

4.3. Генератор на персонализирани учебни пътища

Генераторът на персонализирани учебни пътища е имплементиран чрез стъпка в главен работен процес на Модул „Обучаван в курс“, която извиква уеб метод в мрежовия слой с име GenerateLearningPath. Алгоритъмът е подробно описан в гл. 3, а в текущата точка ще дадем имплементацията му като PHP уеб метод мрежовия слой.

В следващите редове ще използваме понятията „точка“ и „радиус-вектор“ взаимозаменяемо.

Сесийната променлива $_SESSION['vector_coords'] се използва за постепенно акумулиране на имената на измеренията, за които предстои да се правят векторните изчисления в многомерното пространство. Това помага за „разгъването“ на радиус-векторите на точките на всички обекти, като на тези от тях, на които им липсват инициализирани измерения от списъка с всички измерения, находящ се в $_SESSION ['vector_coords'], им се добавят тези измерения с нулеви стойности. В последствие векторните изчисления се изчисляват с точки, намиращи се в многомерно пространство с инициализирани всички тези измерения.

Ако за учебната дейност няма генериран обект, то този обект се създава. И да има обект и да няма, резултат от изпълнението на съхранената процедура е актуален идентификатор на обекта на учебната цел.

Следващият етап от реализацията на GenerateLearningPath е инициализацията на обекта на текущия обучаван, който става по начин, аналогичен на показания.

Инициализацията на обекти на учебни дейности става чрез първоначално избиране на идентификаторите на учебните дейности, които са записани като достъпни в настоящата заявка за обучение. При липса на такива учебни дейности, изпълнението на програмата ще излезе от уеб метода със съобщение за грешка, че няма налични учебни дейности в адаптивния курс. Инициализират се асоциативните масиви ID_OBJ_LAs и map_ID_LA_to_LA, които съдържат информация за учебните дейности, като първия масив съдържа техни идентификатори, а втория: техни обекти, отговарящи на структурата им в заявката, по която са извлечени от БД.

Следваща важна стъпка от инициализацията е инициализиране на известни координати за всеки от тези извлечени обекти в многомерното пространство.

Във foreach цикъла се инициализира векторно пространство vectorLAs, което съдържа векторите, отговарящи на обектите на вече извлечените учебни дейности. За целта се използва помощната функция GetObjCoordinates .

Следващата стъпка е премахване на някои координати от учебната цел, които са свързани с посочване на задължителни учебни дейности (ако има такива за нея). Отделно от това, ако има задължителни учебни дейности, те се добавят в генерирания учебен път selectedLAs.

По конвенция е възприето, че при наличие на измерение в радиус-вектора на учебната цел, чието име е от вида OBLIG_LA_X, където X е идентификатор на учебна дейност, тогава тя е задължителна за тази учебна цел. С unset($vector_ID_OBJ_CG [$key]) се премахва това измерение от вектора на текущата учебна цел, като целта е то да не взема участие в по-нататъшната част от алгоритъма, понеже евентуалното му участие само би объркало генерирането на оптималния път от учебни дейности.

Трябва да се отбележи, че списъкът от измерения на всеки обект, в пространството от учебни дейности и текущата учебна цел се филтрира от всички измерения, чиито имена не започват с “gen__”, което за алгоритъма е знак, че те участват в генерирането.

```
ExtendObjectCoordinates($vector_ID_OBJ_CG);
ExtendObjectCoordinates($vector_ID_OBJ_STUD);
foreach ($vectorLAs as $key => $value) {
    ExtendObjectCoordinates($vectorLAs[$key]);
}
$_SESSION['CUR_CG']=$vector_ID_OBJ_CG;
```

Фигура 17 „Нормализиране“ на измеренията

На фиг. 17 е показано т.нар. „нормализиране“ на измеренията, като на всеки радиус вектор се добавят измеренията, които той все още няма инициализирани, като липсващо измерение на радиус-вектор се добавя с името на съответното измерение и стойност 0 за него.

За целта се използва функцията ExtendObjectCoordinates . След като векторното пространство е вече напълно инициализирането, се преминава към същинската част на реализацията на алгоритъма.

При условие, че в предишен етап на изпълнение на алгоритъма е избрана задължителна учебна дейност и тя вече се намира в пространството на избраните учебни дейности selectedLAs, то трябва да се провери дали текущата учебна цел вече не е напълно постигната с тях. Проверява се чрез IsInReachRadius($vector_ID_OBJ_STUD, $selectedLAs[$key], $DEFAULT_REACH_RADIUS) дали наличната УД е в радиуса на достижимост на обучавания, а чрез IsGreaterOrEqual($selectedLAs[$key], $vector_ID_OBJ_CG)

се проверява дали целта се достига с някоя от тези избрани УД. Трябва да отбележим, че не е задължително при налична задължителна УД в учебна цел, тя да се постига само чрез нея. Може в УЦ да са записани други функции/предикати, които да отговарят на измерения, свързани с дидактическо описание, което да не е постигнато от тази УД.

Следващ етап от алгоритъма е да се провери дали случайно в радиуса на достижимост на студента няма поне една УД, която сама по себе си (без да има нужда от генериране на цял път от УД) постига тази учебна цел.

Ако този по-елементарен случай е изпълнен, изобщо не се преминава към същинската част на генериране на цял път от УД, а директно се приключва изпълнението на GenerateLearningPath с това, което е налично дотук в selectedLAs.

Ако никой от горните по-прости частни случаи не е изпълнен, се преминава към същинското генериране на учебен път от множество УД. Трябва да се отбележи, че чрез uasort($vectorLAs, "cmpVectors"), УД в пространството от налични УД vectorLAs в адаптивния курс се преподреждат в асоциативния масив по близост до целта за постигане. Това дава възможност да се подбират с предимство такива УД, които са по-близо до търсената цел, което е един вид оптимизиране в алгоритъма.

```
    //start moving current student radius-vector until reaching
desired LAs list
    while(true){
        //uasort($vectorLAs, "cmpVectors");
        //$curLA=  array shift($vectorLAs);
        $curIDX LA=0;
        $Vector_curLA=NULL;
        $hadAtLeastOneInReachRadius=false;
        foreach ($vectorLAs as $key => $value) {
            $curIDX LA=$key;
            $Vector curLA=$value;
            if(IsInReachRadius($vector_ID_OBJ_STUD, $Vector_curLA,
$DEFAULT_REACH_RADIUS)){
                MoveTo($vector_ID_OBJ_STUD, $Vector_curLA);

$selectedLAs[$ptr selLAs++]=$map ID LA to LA[$curIDX LA];
                MarkLAAsUsed($ID Student, $curIDX LA);
                unset($vectorLAs[$curIDX_LA]);
                $hadAtLeastOneInReachRadius=true;
                break;
            }
        }
        if(!$hadAtLeastOneInReachRadius){
            //няма достатъчно учебни дейности, излизай
            $rv=[(object)["error"=>"Няма достатъчно налични учебни
дейности. (Безкрайно амбициозна цел.) Изход."]];
            return $rv;
        }
        //избрана поредна учебна дейност. Виж дали си постигнал целта
        if(IsGreaterOrEqual($vector_ID_OBJ_STUD, $vector_ID_OBJ_CG)){
            //целта постигната и пътя генериран
            return $selectedLAs;
        }
    }
```

Фигура 58 Основната част от генерирането на оптимален учебен път

Безкрайният цикъл while(true) позволява безкрайно зацикляне, докато не се излезе от цикъла и от уеб метода чрез достигане на секция от код с return. В този цикъл: Първоначално с IsInReachRadius($vector_ID_OBJ_STUD, $Vector_curLA, $DEFAULT_ REACH_RADIUS) се проверява дали изобщ има поне 1 УД, която се намира в сферата на достижимост на обучавания и ако има, тя се премахва от vectorLAs и се добавя в selectedLAs. Ако няма, това е признак за невъзможност за генериране на учебен път със съобщение за грешка, показващо наличие на безкрайно амбициозна цел. Освен това с MoveTo($vector_ID_OBJ_STUD, $Vector_curLA), точката на текущия обучаван в многомерното пространство се премества така, все едно обучавания е прочел и разбрал напълно тази УД. По този начин и неговата сфера на достижимост се премества в многомерното пространство; С IsGreaterOrEqual($vector_ID_OBJ_STUD, $vector_ID_OBJ_CG) се проверява дали след поредното преместване на точката на обучавания в многомерното пространство и след поредното добавяне на УД в selectedLAs и премахване на вектора и от vectorLAs не е постигната вече Учебната цел. Ако е постигната, се излиза от GenerateLearningPath с генерирания списък от УД. Ако винаги не е, тогава все някога ще се влезе в хипотезата if(!$hadAtLeastOneInReachRadius) (Безкрайно амбициозна цел).

Преди изпълнението на алгоритъма, трябва да се провери дали няма дефиниран работен поток, който може да се използва за генерирането на списъка от УД. Ако има такъв работен поток, се използва той, вместо този алгоритъм.

```
if(existsProcess($ID_CG)){
    $rv listLAs=  invokeProcess($ID CG, $arr);
    return $rv_listLAs;
}
```

Фигура 19 Автоматизирана замяна на* eLP_GA *с работен поток

При наличие на подходящ работен поток за тази цел, който е регистриран в TinyWf, се извиква негова инстанция, вместо да се изпълнява алгоритъма *eLP_GA*. Като конвенция във фреймуърка е прието, че ако поток има име от вида CG_X, където X е идентификатор на учебната цел в рамките на фреймуърка, тогава той се изпълнява автоматично от GenerateLearningPath. При това като входни параметри на инстанцията на работен поток се предават същите входни параметри, които са били предадени чрез асоциативния масив $arr на GenerateLearningPath.

Оптималността на генерираните персонализирани учебни пътища се гарантира от след-ното: Подреждане на УД в курса по близост до целта и предлагането им в такъв ред; Търсене на частни случаи, като наличие на точно една УД, с която да се постига УЦ или дали при наличие на задължителна УД в целта; нейните метаданни покриват всички останали метаданни (координати) на целта и е безсмислено търсенето на още УД.

Глава 5. Изграждане на прототипи върху работна рамка ACEO

В настоящата глава ще се спрем на създаване на прототипи на адаптивни курсове върху работната рамка на ACEO, като паралелно с това разгледаме пример за създаването на адаптивен курс по дисциплината „Въведение в компютърните науки“. Реализацията е съвкупност от етапи на планиране, софтуерна разработка и експлоатация на прототип върху ACEO.

За изпълнението на всяка от стъпките има удобен ГПИ, автоматично генериран от фреймуърка за изпълнение на работни потоци TinyWf, но за целите на компактността описваме последователности от изпълнения на стъпки чрез псевдокод. Отделно от това, създаване на скриптове във формат подобен на формата на псевдокода позволява създаването на ***импортиращи/експортиращи скриптове*** на адаптивни курсове между отделни инстанции на АСЕО.

Чрез механизъм, описан в гл. 4, точка „Генериране на персонализирани учебни пътища“, разработчикът на заявка за обучение би могъл да дефинира работен поток, който да има специфично поведение, което да е по-различно от работата на алгоритъма *eLP_GA*. Рамката за изпълнение TinyWf позволява дефиниране на работни потоци както в PHP структури от данни, демонстрирани широко в гл. 4 при дефинирането на работни потоци за различните модули, така и в NodeJS Javascript формат, които работни потоци са ориентирани за автоматизирано изпълнение, а не толкова за интеракция с потребителя. Тук ще демонстрираме дефиниция на такъв работен поток, който прави нещо сравнително елементарно, а именно предлагане на всички налични УД в адаптивния курс като път за постигане на УД. С цел опростяване на примера, тези УД не се филтрират по никакъв начин, освен определянето на контекста (курс и цел в курса), но в конкретна ситуация допълнително филтриране би било полезно.

Работния поток в този случай се състои от един единствен възел `GenOutput`. Входните параметри на възела, които са дошли от сесийното пространство чрез описанието на `SessionVarsToInputs`, което е било автоматично запълнено и с входните параметри на инстанцията на работен поток, автоматично се зареждат като атрибути на Javascript обекта `inputs`. Callback функциите в масива `makeOutputFunctions` се изпълняват паралелно и генерират елементи на масива `resVariable`, който масив при приключване на изпълнението на инстанцията на работен поток автоматично се сериализира в JSON формат и се предава автоматично като резултат от изпълнението на инстанцията на работния поток. По този начин PHP функцията `invokeProcess`, която се извиква от Генератора на персонализирани учебни пътища получава генерирания от работния поток учебен път и го връща така, все едно самият той го е генерирал. В случая, елементарният пример връща всички учебни дейности, регистрирани в текущия адаптивен курс, но в реална ситуация по-подходяща имплементация би била по-полезна.

Заключение

В рамките на дисертационното изследване са решени поставените задачи и е **постигната основната цел** на дисертационното изследване – да се предложат, изследват и апробират модели, методи и софтуерни инструменти, подходящи за създаване на АСеО с различно предназначение.

Основните **приноси** на дисертационния труд могат да се характеризират като научни, научно-приложни и приложни.

Научни приноси на дисертационното изследване са:

Н1. Създадени са множество модели за осигуряване на адаптивно е-обучение, вкл. модели на работни потоци от дейности; потребителски гледни точки (аспекти в многомерно пространство); обучавани; преподаватели; учебни материали и дейности,

Н2. Създаден е общ модел за осигуряване на адаптивно е-обучение;

Н3. Предложена е методика за създаване на АСеО с различно предназначение.

Научно-приложни приноси на дисертационното изследване са:

НП1. Създадена е архитектура, подходяща за изграждане на АСеО с различно предназначение и обхват;

НП2. Реализиран е прототип на софтуерна система (в рамките на институционална информационна инфраструктура) за осигуряване на адаптивно е-обучение.

Приложни приноси на дисертационното изследване са:

П1. Създадени и тествани са *компютърни модели* на общите модели от Н1;

П2. Проведени са конкретни експерименти по създаване на АСеО с различно предназначение, вкл. за разширяване на популярната СеО Moodle до АСеО.

Перспективи за развитие

Получените резултати дават основа за следващо развитие на някои перспективни направления в областта на дисертационното изследване, като:

- Създаване на **GUI** за описание на работни потоци от дейности в е-обучението;
- Осигуряване на **независимост** от конкретно СУБД и от синтаксиса на съхранените в него процедури чрез допълнителен слой на абстрахиране от СУБД (Възможно са различни подходи, включително и употреба на Object-Relational Mapping (ORM).).
- Усъвършенстване на **стратегиите и методиките за акумулиране на метаданни** за учебни дейности;
- Проектиране и създаване на **интелигентни персонални съветници** – за обучавани (при избор на специалност или персонализиран учебен път), за преподаватели (при създаване на на нови специалности и съобразяване с кандидатстудентски предпочитания) и др.;
- Създаване на **обща работна рамка** за преследване и постигане на учебни цели чрез работни потоци от различни по тип (не само учебни) дейности и процедури;
- **Кластеризация на учебни цели** по интереси на обучавани или на потенциални бъдещи работодатели с идея за по-добра реализация на пазара на труда;
- **Автоматизирана идентификация на непостижими или ресурсоемки учебни цели** с ползване на наличните УД както и на обучавани със специфични проблеми в своята успеваемост и д**р.**

Апробация

Резултатите от дисертационното изследване са представени в **10 (десет) публикации**; една от които в престижна международна конференция CompSysTech'16; четири - докладвани на специализирани международни и национални конференции; както и две глави в 1 книга:

1.Pashev G., G. Totkov, Hr. Kostadinova, Hr. Indzhov, Personalized Educational Paths through Self-Modifying Learning Objects. От 17-th International Conference on Computer Systems and Technologies CompSysTech'16. ACM (In Print).

2.Пашев Г., Автоматизирано генериране на Адаптивен план за обучение. От Научни трудове на Съюза на учените в България – Пловдив Серия В. Техника и технологии, том XIII., Съюз на учените, сесия 5 - 6 ноември 2015, стр. 181-186;

3.Пашев Г., Г. Тотков. Автоматизирано генериране на персонализирани учебни пътища чрез аспекти в многомерни пространства . От Деветата Национална конференция "Образованието и изследванията в информационното общество" . АРИО, ИМИ-БАН, ПУ "Паисий Хилендарски", Пловдив 2016;

4.Pashev G, E. Alendarova, G. Totkov Automated Assessment Through Integration of Heterogeneous Systems with a Workflow Engine. От Proceedings of the National Conference on "Education and Research in the Information Society", Plovdiv, May, 2015. Institute of Mathematics and Informatics Bulgarian Academy of Sciences, Association for the Development of the Information Society, 119p-128p;

5.Пашев Г.; АВТОМАТИЗИРАНИ C++ КОМПИЛАЦИЯ, ИЗПЪЛНЕНИЕ И ОЦЕНЯВАНЕ НА РЕШЕНИЕТО НА БАЗАТА НА СРАВНЕНИЕ НА ИЗХОДИТЕ . От Научни трудове на Съюза на учените в България–Пловдив Серия В. Техника и технологии, том XII.,Съюз на учените сесия 31октомври - 1ноември 2014 ISSN 1311-9419. СУБ – Пловдив, стр. 219-222;

6.Pashev G., G. Totkov; Dynamic Determination of Personalized Educational Paths. От Proceedings of the National Conference on "Education and Research in the Information Society", Plovdiv, May, 2014. Institute of Mathematics and Informatics Bulgarian Academy of Sciences, Association for the Development of the Information Society, 161p-170p;

7.Pashev Georgi**,** Среда за управление на работните потоци EMS и Програмен Език и парадигма. От Научни трудове на Съюза на учените Пловдив, Серия В: Техника и технологии. Пловдив, 2014, стр. 223 - 226.

8.Pashev Georgi, Ivan Kodinov, Georgi Totkov**,** Process Definition and Control in EMSG Complex Work-flow Management System Using Process Graphs and Data Addressing in a File with Flow Identifier Operator . От Proceedings of "Days of Science 2013" Union of Scientists in Bulgaria - Plovdiv. Plovdiv: Union of Scientists Plovdiv, p. 138-142;

9.Тотков и др., Адаптивни системи за е-обучение, Съвременни тенденции в е-обучението; стр. 80-104; изд. „Ракурси“ ООД; ISBN 978-954-8852-46-3; Пловдив, 2014г.

10. Тотков и др., Експериментални СеО, Съвременни тенденции в е-обучението; стр. 146-181; изд. „Ракурси“ ООД; ISBN 978-954-8852-46-3; Пловдив, 2014г.

Част от резултатите на изследването са апробирани в **2 (два) европейски проекта**:

- BG051PO001-4.3.04-0064 „Пловдивски електронен университет: национален еталон за провеждане на качествено е-обучение в системата на висшето образование“ (2012-2014), финансиран от ЕСФ;
- BG051PO001-3.1.08-0041 „Стандартизиране и интегриране на разнотипни информационни и управленски университетски системи“ (2013-2014), финансиран от ЕСФ.

Известни са 5 (пет) цитирания на публикации по дисертационната тема.

В съществуващата информационна инфраструктура на ПУ са „вградени“ и се ползват резултати на изследването, както следва:

- Рамка за изпълнение на работни потоци TinyWf (версия 1.0) – за реализация на софтуерни модули за управление на университетските такси и стипендии (вкл. онлайн кандидатстване);
- Рамки за изпълнение на работни потоци TinyWf (версии 1.0 и 2.0) – за моделиране и реализация на работни потоци от различни университетски дейности, като част от интегрираната информационна система ПеУ [Тотков. (2014)];
- Модул за провеждане на адаптивно е-обучение в университетска СеО Moodle;
- Организиран и проведен адаптивен е-курс по дисциплината „Въведение в компютърните науки“ (уч. 2015/2016 г.) – за студенти от 1-ви курс (редовно и задочно обучение) на специалности „Информатика“ и „Бизнес информационни технологии“ към ФМИ, както и на спец. „Телематика“ към Физическия факултет, и др.

Цитирания

Забелязани са 5 (пет) цитирания по темата на дисертационния труд:

- Пашев Г., Е. Алендарова, Г. Тотков, Проверяване на знанията и автоматично оценяване чрез интегриране на разнотипни системи с работни процеси, Сборник на 8-ма Нац. конференция „Образованието и изследванията в информационното общество“ (ред. Г. Тотков и Ив. Койчев), 28 май – 29 май 2015 г., Асоциация „Развитие на информационното общество“,Пловдив, ISSN 1314-0753, 119 – 128;
- Пашев Г., А. Трайков, Е. Алендарова, Г. Тотков, Интегриране на функционалности от разнотипни системи в Moodle, Scientific Works of the Union of Scientists in Bulgaria – Plovdiv, Series B. Natural Sciences and the Humanities, ISSN 1311-9192, Vol. XVII, 2015, 161-164;
- Пашев Г., Г. Тотков, Динамично определяне на персонализирани учебни пътища, Сборник на 6-та Нац. Конференция „Образованието в информационното общество“ (ред. Г. Тотков и Ив. Койчев), 30 май – 31 май 2014 г., Пловдив, Асоциация „Развитие на информационното общество“, 161 – 170;
- Пашев Г., Г. Тотков, Автоматизирано генериране на персонализирани учебни пътища чрез аспекти в многомерни пространства, 9-та Национална конференция „Образованието и изследванията в информационното общество“. Пловдив, 26-27 май 2016 г., Асоциация „Развитие на информационното общество“, Ракурси ООД, ISSN 13140752, 43-52;
- Pashev G., G. Totkov, H. Kostadinova, Hr. Indzhov, Personalized Educational Paths through Self-Modifying Learning Objects, Proc. of the International Conference on Computer Systems and Technologies - CompSysTech’16, Palermo, Italy (in print).

са цитирани от:

Гафтанджиева Силвия, Модел и система за динамично оценяване на качеството във висшето образование, Дисертация за присъждане на образователна и научна степен „доктор“, 2016г, гр. Пловдив.

Библиография

Bayne, S. (2004). Smoothness and striation in digital learning spaces. От E-Learning and Digital Media (стр. 302--316). SAGE Publications.

BRUSILOVSKY, P. &. (2012). Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK. От ACM Journal on Educational Resources in Computing,.

Brusilovsky, P. (1994). Student model centered architecture for intelligent learning environments. От Proc. of Fourth International Conference on User Modeling,(Hyannis, MA, 15-19 August 1994), MITRE.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. От User modeling and user-adapted interaction (стр. 87--129). Springer.

Brusilovsky, P. (2000). Adaptive hypermedia: From intelligent tutoring systems to Web-based education. От Intelligent tutoring systems: Lecture notes in computer science, vol. 1839, pp. 1-7.

Brusilovsky, P. a. (1997). Distributed intelligent tutoring on the Web. От Artificial Intelligence in Education: Knowledge and Media in Learning Systems. IOS, Amsterdam (стр. 489).

Carrier, J. L. (1988). A fast adaptive multipole algorithm for particle simulations. От SIAM journal on scientific and statistical computing 9.4.

Chow, G. T. (2010). Mobile Handheld Devices as Information Management Tools in Higher Education. Heidelberg Press.

Clocksin, W. F., & Mellish, C. S. (2003). Programming in Prolog. Berlin ; New York: Springer-Verlag.

Costa, P. T., & McCrae, R. R. (1992). Normal personality assessment in clinical practice: The NEO Personality Inventory. От Psychological assessment. American Psychological Association.

Cronbach, L. J. (1957). he two disciplines of scientific psychology. От American psychologist 12.11 .

De Bra, P. (2000). Pros and cons of adaptive hypermedia in Web-based education. От Journal on CyberPsychology and Behavior, vol. 3, pp. 71-77.

De Bra, P. a. (2006). The design of AHA. От Proceedings of the seventeenth conference on Hypertext and hypermedia (стр. 133--134). ACM.

Doneva R., S. Gaftandzhieva, G. Totkov, FETCH Project: The Maturity of Quality Management Through Dynamic Evaluation of the Project Progress, International Journal on Information Technologies & Security, ISSN 1313-8251, Issue 3 (vol. 8, 2016, pp. 29-38.

Бончев Боян, Д. В. (2010). Преглед в областта на адаптивните модели на обучение. София: СУ "Св. Климент Охридски".

Георги Пашев, А. Т. (2015). Интегриране на функционалности от разнотипни системи в Moodle . От Научни работи на Съюза на учените, Пловдив, Серия Б. Пловдив: СУБ Пловдив.

Георги Пашев, Г. Т. (2016). Автоматизирано генериране на персонализирани учебни пътища чрез аспекти в многомерни пространства . От Деветата Национална конференция "Образованието и изследванията в информационното общество" .
Георги Пашев, Е. А. (н.д.). Automated Assessment Through Integration of Heterogeneous Systems with a Workflow Engine. От Proceedings of the National Conference on "Education and Research in the Information Society", Plovdiv, May, 2015, 119p-128p. Institute of Mathematics and Informatics Bulgarian Academy of Sciences, Association for the Development of the Information Society.
Ивлева. (2004). Разработка и исследование интеллектуальных контролирующих систем с настраиваемой нечеткой экспертной подсистемой выставления оценок. Рязан.
Карпенко А. П., Д. А. (2011). Модельное обеспечение автоматизированных обучающих систем. Обзор. http://technomag.edu.ru.
Л.А, Р. (1981). Адаптация сложных систем. Методы и приложения. Рига: Зинатне.
М.Н., П. (2003). Разработка Concept Tree представления и контроля знаний, обеспечивающий заданный уровень функционирования человеко-машинных систем управления:. М.Н. Пущин.- Москва, 184 с.
Пашев. (2015). Автоматизирано генериране на Адаптивен план за обучение. От Научни трудове на Съюза на учените в България – Пловдив Серия В. Техника и технологии, том XIII., Съюз на учените, сесия 5 - 6 ноември 2015.
Пашев, Г. (2014). АВТОМАТИЗИРАНИ C++ КОМПИЛАЦИЯ, ИЗПЪЛНЕНИЕ И ОЦЕНЯВАНЕ НА РЕШЕНИЕТО НА БАЗАТА НА СРАВНЕНИЕ НА ИЗХОДИТЕ . От Научни трудове на Съюза на учените в България–Пловдив Серия В. Техника и технологии, том XII.,Съюз на учените сесия 31октомври - 1ноември 2014 ISSN 1311-9419. СУБ - Пловдив.
Пашев, Г. (2015). Среда за управление на работните потоци EMS и Програмен Език и парадигма. От Научни трудове на Съюза на учените Пловдив, Серия В: Техника и технологии. Пловдив.
Тотков. (2014). Пловдивски е-университет. Пловдив: Ракурси.
Тотков. (2014). Увод в е-обучението. От Т. и. др. Пловдив: Ракурси ООД.
Тотков, Г. (2014). Съвременни направления на е-обучението. Пловдив: Ракурси. doi:ISBN 978-954-8852-46-3
Тотков, Г. (2016). За е-обучението, Сборник научни доклади на VI-та Национална конференция по електронно обучение във висшите училища. Kiten: Университетско издателство „Св. Кл. Охридски“, София, Китен, 2-5.6. 2016, ISSBN 978-954-07-4114-7.

БЕЛЕЖКИ