International Conference
FROM DELC TO VELSPACE
Plovdiv, 26–28 March 2014

ФОРМАЛЕН МОДЕЛ НА ВИРТУАЛНО ОБРАЗОВАТЕЛНО ПРОСТРАНСТВО

Станимир Стоянов

Резюме. *За подпомагане на обучението посредством използване на съвременни информационни и комуникационни технологии във ФМИ на Пловдивския университет се разработва Разпределен център за електронно обучение. В съответствие с развитието на Интернет и уеб центърът се трансформира във виртуално образователно пространство. В публикацията е представен теоретичният модел на това пространство. Моделът ще служи като основа за изграждане на пространството като разпределена контекстно-зависима и адаптивна софтуерна среда.*

Ключови думи: Virtual Education Space, DeLC, Formal Models, Intelligent Agents, eLearning, Reference Architectures.
Mathematics Subject Classification 2010: 68-06, 68T42.

1. ВЪВЕДЕНИЕ

За подпомагане на обучението посредством използване на съвременни информационни и комуникационни технологии, във ФМИ на Пловдивския университет се разработва инфраструктура, наречена Разпределен център за електронно обучение, познат като DeLC [1]. Предназначението на центъра е да доставя по един контекстно-зависим, адаптивен и персонализиран начин електронни образователни услуги и електронно учебно съдържание, разположени върху физически разделени сървъри [2]. Цялостната концепция за DeLC, като контекстно-зависима и адаптивна инфраструктура за електронно обучение е представена в [3]. В съответствие с нея, следващ етап е изграждане на Виртуално Образователно Пространство (ВОП). В пространството, контекстно-зависимостта на образователните услуги ще се поддържа от автономни интелигентни компоненти с реактивно, интерактивно и проактивно поведение. Съществуващите образователни клъстера поетапно ще се трансформират в такива компоненти. Архитектурата на пространството ще бъде напълно децентрализирана. Трансформацията на DeLC във ВОП се предприема поради въздействието върху начина на опериране на Интернет-базирани приложения (каквото е DeLC), което ще имат в недалечното бъдеще две глобални тенденции – Интернет, като мрежа от компютърни мрежи,

постепенно се трансформира в мрежа на нещата (Internet of Things) [4, 5] и в сегашния, синтактичен уеб възниква следващия семантичен уеб [6, 7]. Първите идеи за създаване на пространството са дадени в [8, 9]. Възможностите за трансформацията на DeLC във ВОП са разгледани в [10].

В публикацията е представен теоретичният модел на виртуалното образователно пространство. Моделът ще служи като основа за изграждане на пространството като разпределена контекстно-зависима и адаптивна софтуерна среда. Във втора точка на публикацията се дискутират съществуващите в специализираната литература понятия за контекстно-зависимост. Актуалната версия на формалният модел на виртуалното пространство е разгледан в трета точка. Предложената референтна архитектура на опериращите в пространството агенти е представена в четвъртата точка. В последната точка накратко се дискутира осигуряването на контекстно-зависимост на архитектурно ниво.

2. КОНТЕКСТНО-ЗАВИСИМОСТ

Първите опити за задоволителна дефиниция на понятието ***контекстно-зависимост*** (context-awareness) са свързани с възможностите на мобилните устройства да отчитат местоположението на използващите ги потребители. В [11] понятието се използва за описание на възможностите на софтуера да отчита позицията на потребителя. В [12] се въвежда понятието „компютинг на разположението“ (situated computing) като способност на компютърните устройства да откриват, интерпретират и реагират на различни аспекти на локалната околна среда на потребителите. Контекстно-зависимостта е дефинирана като способност на програма или устройство да улавя различни собствени състояния и такива на заобикалящата го среда в [13].

Дей [14] критикува тези дефиниции в два аспекта:

- контекст се дефинирани чрез примери, т.е. изреждане на различни случаи;
- контекст се дефинира чрез синоними, основно като околна среда или ситуация.

Според Дей, тези дефиниции биха затруднявали разработването на контекстно-зависими приложения. По тази причина той предлага една по-обща работеща дефиниция. Според [15] контекст е всяка информация, която може да бъде използвана за характеризиране ситуацията на една идентичност. Под „идентичност“ може да се разбира човек, местоположение или обект, разглеждани като съществени за взаимодействието между потребител и приложение, в което се включват и самите те. В съответствие с дефиницията за контекст Дей определя, че една система е контекстно-зависима, ако тя използва контексти за доставка съществена информация и/или услуги, като съществеността зависи от задачите на потребителя [14, 15].

Подобно на контекста и контекстно-зависимостта съществуват различни дефиниции за адаптивността. Предложената в [16] дефиниция е инспирирана от биологическите системи, като характеризира автономния софтуер в термините на свойства като напр., само-конфигурация, само-оптимизация, самолечение, самозащита или с общото означение само*-свойства (self-* properties). В [17] автономният софтуер (в run-time) може да поддържа собственото си състояние или да адаптира различно от началното си поведение, като следствие от промени в потребностите на потребителя или в оперативната среда. В съответствие с [18] една динамична адаптивна система е компютърна система с възможности за разпознаване, че околната среда, с която има поделя общ интерфейс е променена и в следствие от това да модифицира собственото си поведение за адаптиране към тези променени условия. В [19] само-адаптивните системи са способни да се адаптират към променящите се потребности на потребителите и изискванията за ресурси в run-time. Спрямо [20] само-адаптивните системи се адаптира към промените в околната среда, доставяйки надеждност, мощност и ефективност без намеса на потребителя. Друга дефиниция, предложена в [21], е че само-адаптивният софтуер оценява собственото си поведение и го променя, когато оценката индикира, че не изпълнява предвиденото предназначение или ако са възможни по-добри функционалност или производителност. В [22] само-адаптивният софтуер модифицира поведението си в отговор на промените в оперативната му среда. Под оперативна среда се разбира всяко обозримо от софтуерната система нещо, като напр., въвеждания на крайни потребители, външни хардуерни устройства и сензори или програмируеми устройства.

В специализираната литература по различен начин са поставени също отношенията между контекстно-зависимостта и адаптивността. В много случаи понятията контекстно-зависимост и адаптивност се включват взаимно. Така напр., представената в [23] йерархия на адаптивност включва контекстно-зависимост като елемент на по-долно ниво. В [24] е адаптивната система усеща контекста си, който е оперативната околна среда. Обратно, в архитектурата CSOA (Context-aware Service Oriented Architecture) адаптацията е подчинен на контекстно-зависимостта аспект [25].

3. ФОРМАЛЕН МОДЕЛ

3.1 ОБЩ ПРЕГЛЕД НА ПРОСТРАНСТВОТО

Основно свойство на пространството е неговата контекстно-зависимост. В нашия случай **контекстно-зависимост** е способността на една система да открива, идентифицира и интерпретира промените (събитията) в околната си среда и в зависимост от тяхното естество да предприема компенсиращи действия. Основни компенсиращи дейности (атрибути на контекстно-зависимостта) са персонализация и адаптация. Персонализацията е способност

на системата да се пригоди към индивидуалните особености, желания, намерения, цели на потребителите. Адаптацията е способност на системата да се пригоди към останалите контекстни особености, като напр. област на познанието, учебна дисциплина, вид на използваните от крайните потребители устройства.

Основа за изграждане на пространството е теоретичен модел, наречен C3A (Context-Aware Architecture). Необходимостта от теоретичен модел може да се мотивира от следните три гледни точки:

- *Представяне на „общата картина“* – теоретичният модел дава възможност да се представи „общата картина“ на пространството. Понеже изграждането на виртуално образователно пространство е комплексна и многоаспектна задача лесно можем да се „загубим“ в детайлите. За тази цел в контекста на „общата картина“ е сравнително лесно да се специфицират проблемите за решаване;
- *Синтезирано представяне* – с помощта на теоретичния модел характерните особености и основните механизми на пространството могат да се представят синтезирано, така че да могат да се използват за неговата реализация;
- *Генетични структури* – теоретичният модел предоставя генетични структури (еталони, шаблони) – те ще служат като основа за разработване на конкретните компоненти на пространството.

C3A моделът се разработва поетапно, представен като отделни версии:

- **Версия 0.1**[26] – представя обща дефиниция на пространството. Освен това, като основни компоненти на модела са специфицирани агенти с ограничена рационалност. Характеризирани са два типа агенти – персистентни и оперативни. Даден е общ жизнен цикъл на контекстно-зависима архитектура;
- **Версия 0.2** [27] – в тази версия са прецизирани основните понятия на модела. Въведени са специалните подпространства, наречени „обсег“ на агент и „зона на случване“ на събитие, Подчертана е съществената роля на събитията в модела;
- **Версия 0.3 –** това е актуална версия, която ще бъде представена в настоящата публикация.

Обща дефиниция на пространството

Фундаментално понятие в модела е ***smart space***, дефинирано като SmartSpace = {SubSpaces, Assistants, Events, CADI}, където:

- SubSpaces = $\{SP_i \mid i = 1, \ldots, n\}$ е множество на подпространства;
- Assistants = $\{as_i \mid i = 1, \ldots, m\}$ е множество на асистенти;
- Events = $\{e_i \mid i = 1, \ldots, k\}$ е множество на събития;
- CADI (Context-Aware Data Infrastructure) – инфраструктура на данните на пространството.

Подпространства

Логическата инфраструктура на виртуалното пространство се състои от три функционални слоя:

- *Виртуално Образователно Пространство* (ВОП) – състои се от различни видове софтуерни компоненти за планиране, подготовка, организиране и доставяне на споделяеми, контекстно-зависими и персонализирани електронни образователни услуги и елктронно учебно съдържание;
- *Интегрирана Технологична Платформа* (ИТП) – предоставя интегрирана платформа върху която ще се изгражда ВОП;
- *Компютърна и Комуникационна Инфраструктура* (ККИ) – доставя необходимата за опериране на ВОП хардуерна инфраструктура.

Подпространствата представят разпределената конфигурацията на ККИ и ИТП. Всяко подпространство моделира един изчислителен възел от тази конфигурация.

Формално, SmartSpace = $\bigcup_{i=1}^{p} SP_i$, $p \leq n$. За две подпространства $SP_i \in$ SubSpaces и $SP_j \in$ SubSpaces въвеждаме следните дефиниции:

- *Empty subspace*, if $SP_i = \varnothing$;
- *Overlapping subspaces*, if $SP_i \cap SP_j \neq \varnothing$;
- *Identical (or completely overlapping) subspaces*, if $SP_i \equiv SP_j$;
- *Disjunctive subspaces*, if $SP_i \cap SP_j = \varnothing$ – в общия случай изискваме две подпространства да бъдат дизюнктивни.

CADI

Инфраструктурата на данните предоставя общата околна среда на опериращите в пространството агенти с ограничена рационалност (вж. т. 5.). Тук могат да възникват различни събития, което прави средата своеобразен интерфейс между физическия свят на реалното образование и виртуалния свят на електронното обучение. CADI осигурява също контекстно-зависимост и интелигентно поведение на ниво данни. Интелигентно поведение не може да се постигне само с наличието на интелигентни агенти. Необходими са и „интелигентни“ (или интелигентно структурирани) данни. Под „интелигентни“ разбираме подходящо структурирани данни, които могат да бъдат разпределено съхранявани и при необходимост (изпълнение на комплексни заявки) лесно интегрирани [7]. От функционална гледна точка моделът на данните ще включва две комплексни структурни хранилища на данни – дигитални библиотеки и административни база данни. Дигиталните *библиотеки* се използват основно за съхраняване на учебно съдържание. За добрата структурираност на информационните ресурси във ВОП се използват два стандарта. Първият стандарт SCORM 2004 [28] е за структуриране на учебното съдържание. Вторият стандарт QTI 2.0 е за структуриране на електронните тестове [29]. В *административните бази данни* се съхранява цялата необходима помощна информация за планиране, организиране,

протоколиране и документиране на учебния процес, като напр. учебни планове, програми и разписания, протоколи от изпити, дневници, студентски книжки, ученически и преподавателски бележници.

Недостатък на стандартите е отсъствието на семантика. Това е типично за всички решения, базиращи се единствено на метаданни (XML аналогични подходи). Комбинирането на материали от различни автори може да бъде много трудно. Търсенето и извличането може да не бъде поддържано по оптимален начин. Търсенето, извличането и организацията на образователни ресурси обикновено трябва да се прави ръчно – вместо това може да бъде правено от съответен агент. Приноси на семантичните технологии:

- Фокусирано върху обучаемия – учебният материал, възможно от различни автори, може да бъде свързан в общо съгласувани онтологии. Могат да бъдат разработени персонализирани курсове посредством семантични заявки. Учебният материал може да бъде търсен и извличан в контекста на актуални проблеми, в съответствие с решението на обучаемия;
- Гъвкав достъп – знанията могат да бъдат достъпни във всякакъв ред, пожелан от обучаемия. Предпоставките ще бъдат определяни от подходящи семантични анотации. Поддръжка на нелинеен достъп;
- Интеграция в единна платформа за бизнес-процесите на организациите. Образователните дейности могат да бъдат интегрирани в тези процеси.

Необходим е някакъв механизъм за установяване на споделено разбиране: онтологии. При електронното обучение е целесъобразно изграждане на следните хранилища на знания (онтологии) [30]: онтологии на учебно съдържание, педагогически онтологии, структурни онтологии.

Входни точки на пространството

SpaceEntries = {port*, pa_1, …, pa_l}, където съществува една специална входна точка – образователния портал (MyDeLC, OpenDeLC, RDeLC).

- Ще въведем трета група агенти: ПА, които:
  - Не са персистентни и оперативни, понеже те ще бъдат
  - Самогенериращи се агенти – могат да бъдат постоянни или временни (можем да създаваме временни входни точки в пространството).
- Ясно представяне различията между трите вида агенти
- Да разработим отделни жизнени цикли за трите вида агенти.

От гледна точка на изчислимостта, съществени за моделиране на пространството са два компонента – асистенти и събития.

3.2 АСИСТЕНТИ

Асистентите са автономни компоненти, „населяващи“ пространството, за които изискваме да бъдат интерактивни, проактивни и социални (т.е. с

възможности за взаимодействие помежду си). В съответствие с тези изисквания дефинираме множеството на асистентите

$$\text{Assistants} = \text{Agents} \times \text{Services},$$

където

- $\text{Agents} = \{ a_i \mid i = 1, \ldots, r \}$ е множество на агенти;
- $\text{Services} = \{ s_i \mid i = 1, \ldots, q \} \cup \{ \text{nilservice} \}$ е множество на електронни услуги, доставящи минималната функционалност на пространството, включващо също услуга, която няма функционалност (празна услуга).

В съответствие с дефиницията един асистент може да бъде само самостоятелен агент или агент, обслужващ дадена услуга. Услугите не могат да бъдат самостоятелни компоненти в пространството. В общия случай, агентите на пространството са „практически разсъждаващи“ с „ограничена рационалност“, т.е.:

- жизненият им цикъл включва две основни стъпки – „разсъждение“ (определя се целта на агента) и „планиране“ (определя се как да се постигне тази цел);
- оперират с ограничени ресурси и имат частичен контрол и влияние върху SmartSpace.

Агентите с „ограничена рационалност“ притежават „обсег на действие“, който ще означаваме като $scope(a_i)$. Ако $a_i, a_j \in \text{Agents}$, тогава:

- $scope(a_i) \cap scope(a_j) \equiv \varnothing$ – агентите нямат общ обсег;
- $scope(a_i) \cap scope(a_j) \neq \varnothing$ – агентите са с общ обсег.

Освен това изискваме:

- Множеството на услугите да се променя динамично;
- Да съществува „минимална функционалност“ (минимално множество от услуги), която да осигурява нормално опериране на пространството (без екстри).

В пространството различаваме различни типове асистенти, като

$$\text{Assistants} = \text{PA} \cup \text{SA} \cup \text{Gards},$$

където PA, SA, Gards са дизюнктивни множества и:

- PA – множество на персоналните асистенти, които подпомагат потребителите си и доставят „входните точки” на ВОП;
- SA – асистенти (ще ги наричаме специалисти), които предоставят специализирана помощ при изпълнение заявките на потребителите към пространството. Обикновено са разположени върху сървърните възли на ВОП, взаимодействат с електронните услуги и са прозрачни за потребителите.
- Gards – специализирани асистенти, които гарантират сигурно и ефективно изпълнение на образователните сценарии при извънредни условия или внезапни събития. Обикновено те са част от интерфейса между физическия и виртуалния свят. Активирането на гардовете обикновено инициира смяна на образователните с emergence сценарии.

3.3 СЪБИТИЯ

Множеството на събитията Events = $\{e_i \mid i = 1, \ldots, s\}$ е един от най-съществените елементи на модела. Събитията в модела не са атомарни примитиви, а по-скоро се представят като комплексни структури с атрибути, които характеризират различни аспекти – структурни, информационни, темпорални, каузални, пространствени и експериментални [28].

Върху множеството на събития могат да се дефинират интервали като последователности от събития, подредени по определен признак. Например, EvInt = $[e_1, \ldots e_{pr-1}, \mathbf{e_{pr}}, e_{pr+1}, \ldots, e_e]$, наредени във времето събития, където актуалното събитие e_{pr} се интерпретира като настоящето, подинтервалът $[e_1, \ldots e_{pr-1}]$ е минало, респективно $[e_{pr+1}, \ldots, e_e]$ – бъдеще.

4. РЕФЕРЕНТНИ АРХИТЕКТУРИ НА АГЕНТИТЕ

Агентите, които оперират във виртуалното пространство, са с „ограничена рационалност", търсещи смислени решения при ограничен контрол върху околната среда и целесъобразно използващи наличните ресурси. Мащабът на действие на един рационален агент може да се определи като оптимален успех в съответствие с неговите актуални знания и способности, ограничени ресурси и ограничено време. За вземане на решение рационалните агенти използват един модел, познат като „практически разсъждения". Практическите разсъждения са насочени към действието, като рационалните агенти извършват две неща:

- обмисляне – на този етап агентите решават какво да правят (каква цел искат да постигнат), използвайки менталните си състояния;
- планиране (means-ends reasoning) – на този етап агентите решават как да постигнат набелязаната цел.

Оперирането на рационалните агенти се представя като "*sense-think-act*" жизнен цикъл, в който обмислянето и планирането е окомплектовано със сензорите (възприятията) и ефекторите (въздействието върху околната среда) на агентите.

Класическата архитектура за разработване на рационални агенти е BDI (Belief-Desire-Intention) архитектурата [29, 30, 31], използваща модел на човешката дейност за представяне на „ограничената рационалност", където:

- Вяра (beliefs) – както и при агентите с цели представя приеманията на агента за околната среда (модел на околната среда);
- Желания (desires) – общи желания или задачи на агента, които още не са трансформирани в конкретни намерения;
- Намерения (intentions) – намеренията на агента са еквивалентни на съществуващите му ангажименти, вкл. и към самия себе си.

Жизненият цикъл на една абстрактна (референтна) BDI архитектура е представен на (Фигура 1.). Трите атрибута (познати също като *ментални състояния*) са съществени за осигуряване проактивно поведение на агентите, като изборът на желанията се извършва от множество на съществуващи опции, а изборът на намерения – от досегашни намерения и нови желания. Когато една опция успешно премине през функцията-филтър и е избрана от агента като намерение, казваме, че агентът е приел *ангажимент* към тази опция. Ангажиментите означават темпорална продължителност – след като едно намерение вече е възприето, то не може да изчезне веднага. В този архитектурен модел ангажиментите се декомпозират на c = [d, i, pl], c $\in$ Commitments, Commitments = (Options x Options x Plans), където:

- d, i са множества на опции;
- Options – тук е множество от множества на опции, т.е. Options $\subseteq 2^{Opt}$;
- Plans – тук е множество от множества на планове, т.е. Plans $\subseteq 2^{Pla}$. Плановете могат да бъдат образователни сценарии или отделни електронни услуги.

repeat
p := sense(x);
b_{new} := update(b_{old}, p);
d_{new} := evaluate(b_{new}, d_{old});
i_{new} := filter(b_{new}, d_{new} , i_{old});
pl_{new} := means-ends(b, i_{new}, pl_{old});
a := act(pl_{new});
forever
evaluate: Beliefs x Options → Options ;
filter: Beliefs x Options x Options → Options ;
means-ends: Beliefs x Options x Scenarios → Scenarios.
act: Plans → WorkFlow.

Фигура 1. BDI архитектура

Функцията means-ends представя процеса за решаване как да бъде постигната една цел, използвайки налични средства (познат също като планиране). Функцията приема като вход представяне на:

- цел – намерение на агента;
- актуално състояние на околната среда – вяра на агента;
- възможни действия на агента.

Means-ends функцията генерира план като изход. В нашата референтна архитектура плановете са предварително зададени и се съхраняват под формата на образователни сценарии в дигитални библиотеки. Контрол върху тези библиотеки имат специализираните агенти.

Основната задача на персоналните асистенти е да идентифицират кореспондиращите за конкретните случаи образователни сценарии и да осигуряват необходимата персонализация. Персонализацията се извършва на

ниво ментални състояния, като основна роля играят желанията (desires) на агентите. Желанията представят индивидуални календари на обслужваните потребители. Един индивидуален календар се дефинира като множество от събития, очаквани да се случат във времето. Индивидуалните календари могат да бъдат автоматично генерирани от учебните разписания за определен срок. От учебните разписания могат да се вземат свързани с непосредствената учебна дейност на ученика събития, като напр. лекции (уроци), семинарни упражнения, изпити. В последствие календарът може да бъде допълнен със събития от личен характер, като напр., рожден ден, среща с приятели, хоби.

В определени случай някои намерения се трансформират в цели (intentions) на агента. Целите са основа за идентифициране на определен образователен сценарии. За изпълнението на актуалния сценарий, агентите взаимодействат със специализираните агенти, обслужващи дигиталните библиотеки в пространството.

5. АРХИТЕКТУРНА КОНТЕКСТНО-ЗАВИСИМОСТ

За осигуряване на контекстно-зависимост на архитектурно ниво, различаваме два типа агенти – персистентни и оперативни. Персистентните агенти изпълняват две различни функции. Първата е свързана с доставка на минималната функционалност на пространството, която се реализира от наличните електронни услуги. Втората осигурява контекстно-зависимото поведение на пространството, като агентите идентифицират промените в тяхната околна среда и в зависимост от естеството им динамично генерират подходящи оперативни агенти (като реакция на съответната промяна). След като оперативният агент извърши необходимото компесаторно действие, той се премахва (от персистентния агент или се самопремахва).

В модела, двете операции формално се представя както следва:

- генериране на оперативни агенти: $\forall a_i \in PerA\ (\exists\ a_j \in OperA, (\exists\ e_k \in E \wedge e_k \in scope(a_i)) \rightarrow generate(a_i,e_k)=a_j))$, където $Agents = PerA \cup OperA \wedge PerA \cap OperA = \varnothing$;
- премахване на оперативни агенти: $\forall a_i \in PerA\ (\exists\ a_j \in OperA \wedge$ if $complete(a_j) \rightarrow (remove(a_i,a_j) \vee selfremove(a_j))$, където $Agents = PerA \cup OperA \wedge PerA \cap OperA = \varnothing$.

6. ЗАКЛЮЧЕНИЕ

В публикацията е представен актуалният теоретичен модел на виртуалното образователно пространство, което се изгражда като наследник на DeLC. Моделът се използва като формална рамка и основа за реализирането на пространството като разпределена софтуерна среда. Моделът се развива и детайлизира непрекъснато. В момента се разработват първите прототипи на компонентите на пространството, специфицирани в модела.

БЛАГОДАРНОСТИ

Изследването частично е подкрепено от проект НИ13 ФМИ-02, финансиран от НПД на Пловдивския университет „Паисий Хилендарски”.

ЛИТЕРАТУРА

[1] Stoyanov, S., I. Popchev, E. Doychev, D. Mitev, V. Valkanov, A. Stoyanova-Doycheva, V. Valkanova and I. Minov, DeLC Educational Portal, *Cybernetics and Information Technologies (CIT),* Vol. 10, No. 3, 2010, 49–69.

[2] Stoyanov, S., I. Ganchev, I. Popchev and M. O'Droma, An Approach for the Development of a Context-Aware and Adaptive eLearning Middleware, *Intelligent Systems: From Theory to Practice*, *V. Sgurev et al., Ed. Berlin Heidelberg: Springer-Verlag*, 2010, 519–535, ISBN: 978-3-642-13427-2.

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

[4] Ashton, K., *That 'Internet of Things' Thing*, RFID Journal, 2009.

[5] Dieter, U., H. Mark and M. Florian, *Architecting the Internet of Things*, Springer, 2011, ISBN: 978-3-642-19156-5.

[6] Berners-Lee, T., J. Handler and O. Lassila, The Semantic Web, *Scientific American*, No. 284, May 2001, 34–43.

[7] Allemang, D. and J. Hendler, *Semantic Web for the Working Ontologist*, Elsevier, 2011, ISBN: 978-0-12-385965-5.

[8] Вълканов, В., *Изследване за ВОП в средното училище*, ИИКТ, София, 2013.

[9] Орозова, Д., С. Стоянов и И. Попчев, Виртуално образователно пространство, *Научна конференция с международно участие „Знанието – традиции, иновации, перспективи*”, Бургас, 2013.

[10] Дойчев, Е., *Среда за електронни образователни услуги*, Пловдив, Пловдивски университет „П. Хилендарски“, 2013.

[11] Schilit, B. and M. Theimer, Dissemination Active Map Information to Mobile Hosts, *IEEE Network*, 8 (5), 1994, 22–32.

[12] Hull, R., P. Neaves and J. Bedford-Roberts, Towards Situated Computing, *1st International Symposium on Wearable Computers*, 1997, 146–153.

[13] Pascoe, J., Adding Generic Contextual Capabilities to Wearable Computers, *2nd International Symposium on Wearable Computers*, 1998, 92–99.

[14] Dey, A. and G. Abowd, Towards a better understanding of context and context-awareness, *Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness*, New York, ACM Press, 2000.

[15] Dey, A., Understanding and Using Context, *Personal and Ubiquitous Computing Journal*, Vol. 5, No. 1, 2001, 4–7.

[16] Kephart, J. and D. Chess, The vision of autonomic computing, *IEEE Computer*, 36 (1), 2003, 41–50.
[17] Qureshi, N. and A. Perini, An Agent-based Middleware for Adaptive Systems, *The Eighth International Conference on Quality Software*, 2008 IEEE, 423–428.
[18] Berry, D., B. Cheng and J. Zhang, The four levels of requirements engineering for and in dynamic adaptive systems, *Proceedings of the 11th International Workshop on Requirements Engineering: Foundation for Saftware Quality (REFSQ05)*, 2005, 95–100.
[19] Floch, J., S. Hallsteinsen, E. Stav, F. Eliassen, K. Lund and E. Gjorven, Using architecture models for runtime adaptability, *IEEE Software*, 23 (2), 2006, 62–70.
[20] Gjorven, E., F. Eliassen, K. Lund, V. S. W. Eide and R. Staehli, Self-adaptive systems: A middleware managed approach, *A. Keller, J.-P. Martin-Flatin (editors) Self-Man, Vol. 3996 of Lecture Notes in Computer Science*, Springer, 2006, 15–27.
[21] Laddaga, R., Self-adaptive software, Tech. Rep. 98–12, DARPA BAA, 1997.
[22] Oreizy, P., M. Gorlick, R. Taylor, D. Heimbigner, G. Johnson, N. Medvidovic, A. Quilici, D. Rosenblum and A. Wolf, An architecture-based approach to self-adaptive software, *IEEE Intel. Syst.*, 14, 3, 1999, 54–62.
[23] Salehie, M. And L. Tahvildari, Self-Adaptive Software: Landscape and Research Challenges, *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 4, No. 2, 2009, 14:1–14:42.
[24] Liaskos, S., A. Lapouchnian, Y. Yu, E. Yu and J. Mylopoulos, On goal-based variability acquisition and analysis, *Proceedings of the 14th IEEE International Conference on Requirements Engineering (RE'06)*, IEEE Computer Society, September 2006.
[25] Vale, S. and S. Hammoudi, Model Driven Development of Context-aware Service Oriented Architecture, *The 11th IEEE International Conference on Computational Science and Engineering*, 412–418.
[26] Вълканов, В., *Контекстно-ориентирано управление на електронни услуги*, дисертация, ИИКТ – БАН, 2013.
[27] Stoyanov, S., V. Valkanov, I. Popchev, A. Stoyanova-Doycheva and E. Doychev, A Model of Context-Aware Agent Architecture, *Compt. Rend. Acad. Bulg. Sci.*, 67, 4/2014.
[28] SCORM 2004 Specifications, http://www.adlnet.gov/scorm/scorm-2004-4th/.
[29] IMS Question and Test Interoperability Specification, http://www.imsglobal.org/question/.
[30] Antoniou, G. and F. van Harmelen, *Semantic Web Primer. Cambridge: MIT Press*, 2004.
[31] Jain, R., EventWeb: Developing a Human-Centered Computing System, *Computer*, February 2008, 42–50.
[32] Rao, S. and M. Gerogeff, An Abstract Architecture for Rational Agents, *Principles of Knowledge Representation and Reasoning*, 1992, 439–449.

[33] Rao, S. and M. Georgeff, Modeling Rational Agents within a BDI-Architecture, Principles of Knowledge Representation and Reasoning, 1991, 473–484.
[34] Rao, A. and M. Georgeff, BDI Agents: from theory to practice, *First International Conference on Multi-Agent Systems ICMAS-95*, June 1995, 312–319.

Faculty of Mathematics and Informatics
Paisii Hilendarski University of Plovdiv
236, Bulgaria Blvd.,
4003 Plovdiv, Bulgaria
stani@uni-plovdiv.bg

A FORMAL MODEL OF VIRTUAL EDUCATION SPACE

Stanimir Stoyanov

Abstract. *In order to support training through the use of modern information and communication technologies at FMI University of Plovdiv, a Distributed e-Learning Center (DeLC) is developing. In accordance with the current evolution of Internet and Web the DeLC is transformed into a virtual education space as next step. In the paper, the theoretical model of this space is presented. The model will serve as a basis for the construction of the virtual education space as a distributed context-aware and adaptable software environment.*