КОМП'ЮТЕРНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

УДК 629.341 **DOI: 10.30977/BUL.2219-5548.2022.96.0.7**

ІНФОРМАЦІЙНА СИСТЕМА СПОСТЕРЕЖЕННЯ ЗА ПОВЕДІНКОЮ СТУДЕНТІВ ПІД ЧАС ПРОВЕДЕННЯ Е-ТЕСТІВ

Аксак Н. Г. [1], Татарников А. О. [2]
[1]Харківський національний економічний університет ім. С. Кузнеця
[2]Харківський національний університет радіоелектроніки

Анотація. *У статі запропоновано клієнт-серверну систему спостереження та аналізу поведінки студентів під час проведення е-тестування, яка дозволяє максимально ефективно використовувати ресурси шляхом розроблення та реалізації методів оброблення та аналізу зібраних за час спостереження даних із використанням можливостей комп'ютерного зору. Пропонується комплекс взаємопов'язаних блоків моніторингу та стеження, що містить такі основні можливості: відстеження активних URL, отримання переліку мережних підключень, стеження в режимі реального часу буд-яких змін параметрів (ширини та висоти) активного вікна, створення знімків робочої ділянки екрана комп'ютера та фотознімків за допомогою вебкамери, автоматичне зчитування вмісту буфера обміну та ведення логу натиснутих клавіш, розпізнавання обличчя студента та стеження за його положенням щодо вебкамери. У разі виявлення порушень отримані повідомлення записуються та відправляються на сервер.*

Ключові слова: *електроне тестування, система моніторингу, розпізнавання обличчя, клієнт-серверна архітектура.*

Вступ

У зв'язку з переведенням навчального процесу на принципово новий рівень істотно зростає роль самостійної роботи студентів, яка стала основною формою отримання знань. Тому все більше стає актуальним комп'ютерне тестування як засіб контролю успішності засвоєння студентами навчальних матеріалів. Це дозволяє прискорити процес контролю та забезпечує його об'єктивність. Однак процес тестування є вразливим до можливих зловживань із боку студентів – використання допоміжних засобів, допомога сторонніх тощо.

Аналіз наявних рішень

Для забезпечення контролю процесу проведення комп'ютерного тестування найчастіше використовуються клієнт-серверні архітектури, серед яких можна визначити однорівневі, дворівневі та трирівневі архітектури, що належать до інструментів цифрового прокторингу. Деякі інструменти можуть отримувати доступ до віддаленого комп'ютера користувача, створювати знімки екрана, забезпечувати одночасну роботу з декількома пристроями в режимі реального часу тощо.

Так, наприклад, програмний застосунок *Екзамус* призначений для аналізу та контролю поведінки користувачів за допомогою AI-алгоритмів. Перед початком тестування учень має пройти спеціальну процедуру ідентифікації за фотографією, і тільки після цього може перейти до тестування [1].

Програма *ExamCookie* відстежує комп'ютерну активність студентів під час іспиту [2]. Програма гарантує, що студенти не можуть використовувати сторонні засоби під час іспиту. Основним недоліком цієї програми є те, що учень знає про моніторинг процесу тестування та може втручатися в роботу самої системи, наприклад ввести дані іншої особи.

Система прокторингу *ProctorEdu* забезпечує онлайн-спостереження, протоколювання та оцінювання поведінки користувачів під час проходження значних онлайн-заходів [3]. Ця система може бути підключена до необхідної платформи тестування (ПК, телефон, планшет), як у автоматичному режимі, так і з безпосередньою участю проктора (представника Центру тестування, який адмініструє проведення іспиту в конкретній аудиторії закладу вищої освіти відповідно до встановленого порядку проведення іспиту).

Exam Monitor – це IT-система, розроблена для цифрового іспиту, щоб відбити бажання від шахрайства [4]. До недоліків цієї програми належать: обмежений функціонал; відсу-

тність можливості контролю в режимі реального часу; необхідність постійного вивантаження інформації для подальшого ознайомлення з результатами спостережень.

Proctortrack – гібридна модель, яка в режимі реального часу об'єднує віддалених людей-прокторів із розширеним штучним інтелектом. Вона дозволяє втручатися у випадку підозрілої поведінки, шахрайства чи допомоги студентові. Результати тестування цілісності аналізуються за допомогою ШІ [5]. Основним недоліком Proctortrack є те, що для роботи цього ПЗ необхідно придбати ліцензію.

Microsoft Remote Desktop (MRD) – програма для віддаленого доступу та керування комп'ютером Windows. Клієнти існують практично для всіх версій Windows (а також із Linux, Mac OS X, iOS, Android) [6]. Недоліками є обов'язкова наявність на підключеному комп'ютері Windows Pro і вище та неможливість підключення в одній локальній мережі пристроїв до віддаленого робочого стола без додаткових налаштувань.

Remote Utilities – програмне забезпечення для віддаленого робочого стола, яке дозволяє користувачеві віддалено керувати іншим комп'ютером через власний протокол і бачити робочий стіл віддаленого комп'ютера, керувати його клавіатурою та мишею [7].

UltraVNC – це потужна, проста у використанні та безкоштовна програма для віддаленого доступу до ПК, що може відображати екран іншого комп'ютера (через інтернет або мережу) на власному екрані [8] за допомогою протоколу Remote Frame Buffer (RFB). Основним же недоліком цієї програми є її складність у роботі та налаштуванні.

AeroAdmin – це програма, що забезпечує віддалений доступ до комп'ютера через інтернет або в локальній мережі та працює за принципом «все в одному». AeroAdmin дозволяє віддалено керувати комп'ютером через інтернет, зокрема неконтрольований доступ, паралельні сесії, передачу файлів та можливість підключатися до необмеженої кількості віддалених комп'ютерів [9]. Незалежно від того, чи підключений користувацький комп'ютер або серверний, користувач може встановлювати обмеження та права доступу: тільки перегляд, захоплення клавіатури та миші, файловий менеджер, синхронізація буфера обміну. Зі свого боку адміністратор може додатково встановлювати права доступу для кожного учасника робочої сесії.

DameWare – метод віддаленого контролю за комп'ютером користувача, який використовується як засіб технічної підтримки користувачів [10]. Серед недоліків можна виділити обмежений базовий фунціонал у процесі використання безкоштовної версії.

Chrome Remote Desktop (CRD) є програмним забезпеченням для віддаленого підключення до комп'ютера користувача за допомогою браузера Chrome або пристроїв Chromebook [11]. Основним недоліком Chrome Remote Desktop є те, що його робота не можлива без встановлення та використання веббраузера Google Chrome.

AnyDesk – програма для віддаленого робочого стола із закритим вихідним кодом, що поширюється AnyDesk Software GmbH [12]. Програмне забезпечення забезпечує незалежний від платформи віддалений доступ до персональних комп'ютерів та інших пристроїв, на яких запущено хост-програму. До ключових недоліків програми AnyDesk можна віднести обмежений функціонал та збільшене навантаження на викладача через необхідність постійного спостереження за великою кількістю комп'ютерів.

Отже, кожна з розглянутих систем має свої переваги й такі недоліки:

− у процесі тестування студент знає про процес моніторингу, система не має можливості підтримки процесу контролю в режимі реального часу;

− обмежений для використання функціонал під час використання безкоштовної ліцензії;

− відсутність для студентів можливості ознайомлення із зібраними в процесі спостереження результатами після закінчення самого тестування;

− платна ліцензія.

Мета та постановка завдання

Метою є створення клієнт-серверної системи аналізу поведінки студентів під час проведення е-тестування на основі розпізнавання обличчя.

Для досягнення поставленої мети необхідно розробити модель процесу спостереження за поведінкою студентів (ССПС) під час проведення е-тестів, її складових компонентів та їхню взаємодію.

Модель процесу спостереження за поведінкою студентів під час проведення е-тестів

Процес спостереження за поведінкою студентів під час проведення е-тестів може бути поданий трирівневою структурою, що опису-

ється моделлю (1), яка містить такі компоненти: клієнтська частина, блок синхронізації із сервером та серверна частина (рис. 1).

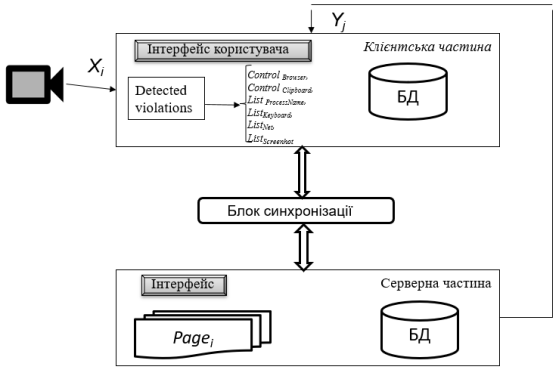

Рис. 1. Модель процесу спостереження за поведінкою студентів під час проведення е-тестів

Модель процесу спостереження за поведінкою студентів під час проведення е-тестів виражається як перетворення вхідних значень *X* у вихідні величини *Y*:

$$Z \subset X \times Y, \qquad (1)$$

де $X = \{X_j(i)\}$, ($i = \overline{1,N}$, N – кількість користувачів, що проходять тестування) – вхідні дані з комп'ютера користувача: X_1 – вміст буфера обміну; X_2 – активні URL у різних типах браузерів; X_3 – список мережних процесів; X_4 – скриншоти всієї ділянки екрана комп'ютера; X_5 – натиснуті клавіші на клавіатурі; X_6 – розміри вікна; X_7 – список активних процесів; X_8 – знімки з вебкамери; Y – виявлені порушення [13].

Для вихідної величини *Y* побудована множина завдань, розв'язання яких належить множині $D_Z=\{D_{user}, D_{sin}, D_{server}\}$: тут $D_{user}=\{Cont_{Browser}, Cont_{Clipboard}, List_{ProcessName}, List_{Keyboard}, List_{Net}, List_{Screenhot}\}$, де $Cont_{Browser}$ – завдання визначення активного URL у різних типах браузерів; $Cont_{Clipboard}$ – завдання відстеження змін вмісту буфера обміну; $List_{ProcessName}$ – завдання отримання списку активних процесів на комп'ютері користувача; $List_{Keyboard}$ – завдання оброблення та логування натиснутих користувачем клавіш на клавіатурі; $List_{Net}$ – завдання отримання списку мережних процесів; $List_{Screenhot}$ – завдання створення скриншотів всієї ділянки екрана комп'ютера та знімків за допомогою вебкамери [14] за умови зміни активного вікна програми або його розмірів; D_{sin} – завдання синхронізації клієнтської та серверної частини; $D_{server} = \{Page_k\}$, k = 1, 2, 3, де $Page_1$ – завдання створення сторінки, де відображається інформація про всі дії, які відбуваються на клієнтських машинах; $Page_2$ – завдання створення сторінки роботи з текстовими даними, що дозволяє ознайомитися з текстовими результати, які були зібрані за час спостереження; $Page_3$ – завдання створення сторінки роботи із зображеннями, яке дозволяє користувачеві ознайомитися з фотознімками та скриншотами, що були зроблені під час тестування.

Відображення $T: X \to Y$ дозволяє для кожного $X(i)$ знайти таке $Y_m \in Y$ ($m = \overline{1,Q}$, Q – кількість порушень), що є розв'язкою завдання D_Z.

Значення $Y_m \in Y$ використовуються для прийняття рішення про порушення та вироблення подальшої тактики поведінки.

Користувацький інтерфейс є програмою, де на вхід подається зчитаний із сервера набір інструкцій, за допомогою яких активуються необхідні для спостереження функції. Зчитана інформація – це функції детектування та пошуку порушень. Функція детектування порівнює зчитані результати з попередніми та еталонними. Вихідним результатом є згрупований масив, що містить: ідентифікатор користувача, повідомлення про порушення, таблиці з інформацію про порушення на комп'ютері студента та на сервері. Процес прийняття рішення про порушення є «порівнянням» отриманих проміжних результатів із попередніми та з еталонними значеннями, що були передані із сервера перед початком тестування. На сервері після отримання повідомлення можна ознайомитися із самим порушенням, для цього реалізований спеціальний web-інтерфейс. Та в разі необхідності проктор може відправити повідомлення з попередженням конкретному студентові, або якщо потрібно змінити налаштування клієнтських застосунків.

Для оцінювання якості процесу функціонування пропонованої системи використовується сукупність критеріїв оцінки ефективності $K = \{k_1, k_2\}$, що дозволяє визначити реакцію детектування порушень.

Першим критерієм є час детектування порушень k_1: $\tau = \min(\tau^{Br}, \tau^{Cl}, \tau^{Pr}, \tau^{key}, \tau^{Net}, \tau^{Scr})$,
де τ^{Br} – час визначення активного URL у різних типах браузерів; τ^{Cl} – час відстеження

змін вмісту буфера обміну; τ^{Pr} – час отримання списку активних процесів на комп'ютері користувача; τ^{key} – час оброблення та логування натиснутих користувачем клавіш на клавіатурі; τ^{Net} – час отримання списку мережних процесів; τ^{Scr} – час створення скриншотів усієї ділянки екрана комп'ютера та знімків за допомогою вебкамери за умови зміни активного вікна програми або його розмірів.

Співвідношення множини реально створених сторінок $Page_k$ *(i=1,2,3)* визначається такими характеристиками: $\psi_k = \{\rho_k, e_k\}$, де $\rho_k = \frac{a}{a+c}$ – коефіцієнт повноти, що характеризує інформацію про дії на клієнтській машині, яка відображена на побудованих сторінках $Page_k$ до загальної кількості дій; $e_k = \frac{b}{a+b}$ – коефіцієнт шуму, що характеризує частку наданої інформації без порушень; a – кількість зафіксованих порушень; b – кількість поведінки без порушень; c – кількість загальних порушень.

Для ефективного функціонування пропонованої моделі необхідно, щоб система відповідала таким вимогам:

$$k_2 : \forall\left(D_j \in D_Z\right)\left[\left(\tau < \tau^{\max}\right)\&\left(\rho_k \to 1\right)\& \right. \\ \left.\&\left(e_k \to 0\right)\right] \Rightarrow Y, \quad (2)$$

де D_j – підзавдання загального завдання D_Z; τ^{max} – максимально допустимий час вирішення завдання.

Моделювання системи спостереження за поведінкою студентів під час е-тестування

Серверна та клієнтська частина програми для моніторингу тестування студентів призначена для використання на ПК під управлінням ОС Windows 10 (32/64-розрядні версії). В ОС має бути встановлений Connector/NET 8.0.20 або вище, та NET Framework версії 4.5 або вище.

Розроблюваний програмний застосунок має змогу працювати в повністю автономному режимі, за винятком ситуацій, коли необхідно виконати зміну порту підключення до серверного застосунку [15]. Також ПЗ має змогу віддаленого контролю за всіма підключеними користувачами та дозволяє відправляти конкретним користувачам повідомлення про порушення.

Програмне рішення, створене в Visual Studio, складається з десяти проєктів (рис. 2).

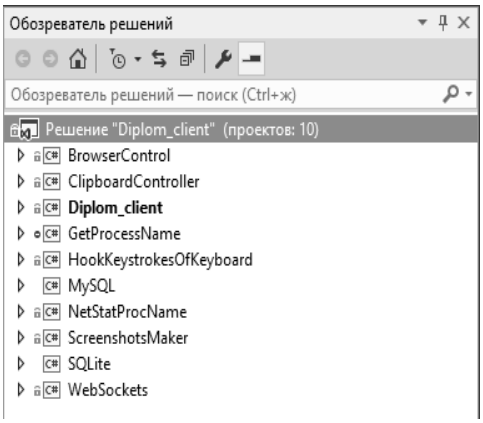

Рис. 2. Файлова структура проєкту

У випадку з використанням декількох проєктів поділ вихідного коду реалізується за допомогою поділу функціоналу програмного рішення на окремі складники, що реалізуються в різних за призначенням проєктах. Це дозволяє розбити функціонал клієнтського застосунку на окремі функціональні блоки. Такий підхід спрощує, прискорює роботу над кожним окремим елементом та дозволяє спростити процес оптимізації кожної окремої частини й усього проєкту загалом.

Файлова структура серверної частини відповідає стандартній структурі застосунків, що розробляються з використанням можливостей CMS WordPress. Файлова структура вихідного коду вебзастосунку складається з великої кількості стандартних файлів самої CMS та файлів розробленої «теми», в якій описані функціональні можливості програми, зокрема інтерфейс користувача, логіка доступу та оброблення даних, робота з користувачами тощо.

Для роботи CMS WordPress використовує трирівневу архітектуру MVC, тому що це найбільш широко використовуваний та надійний шаблон проєктування. Ця архітектура не накладає жодних обмежень на спосіб реалізації інтерфейсу користувача та забезпечує поділ відповідальності між функціональними блоками.

Вебзастосунок (серверна частина програми) містить три основні сторінки, з якими може працювати користувач.

На основній сторінці відображається інформація про всі дії, що відбуваються на клієнтських машинах, та є змога надсилати повідомлення клієнтам у процесі тестування. Сторінка роботи з текстовими даними дозво-

ляє ознайомитися з текстовими результати, що були зібрані за час спостереження. Сторінка роботи із зображеннями дозволяє користувачеві ознайомитися з фотознімками та скриншотами, що були зроблені під час тестування.

Для роботи з різними браузерами використовуються класи Chorme, Firefox, Opera, Microsoft Edge, кожен з яких реалізує функціонал отримання активного URL.

Для початку детектування необхідно спочатку становити на комп'ютері користувача приховану програму «клієнт». Перед початком тестування із сервера відправляється команда про увімкнення спостереження.

Користувач не може взаємодіяти з інтерфейсом клієнтського застосунку, і тому не знає що за ним спостерігають. У цей час із сервера на клієнтські застосунки відправляється команда, «стеження за натиснутими клавішами».

Через 5 секунд на сервер починають надходити повідомлення про детектування порушень із боку студентів. Проктор може бачити ідентифікатор користувача, час детектування та тип порушення. На рис. 3 зображено вікно детектування порушень у режимі реального часу. Система надіслала повідомлення, що користувач у процесі тестування користується клавіатурою, що є недопустимим.

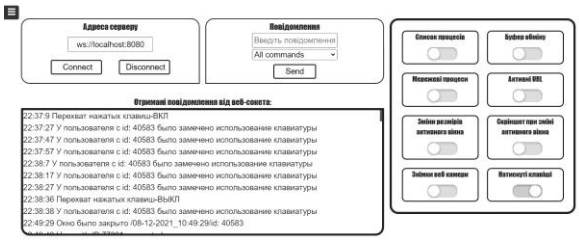

Рис. 3. Детектування порушень у режимі реального часу

Для того щоб відправити попередження конкретному студенту, проктору необхідно ввести ідентифікатор користувача й повідомлення в поле блоку «Повідомлення» та натиснути кнопку «Send». Після чого студент отримає попередження (рис. 4).

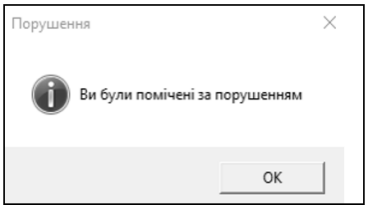

Рис. 4. Попередження про порушення

Після отримання такого повідомлення можна продовжити спостереження або завершити тестування.

Тестування програми проводилося на двох різних комп'ютерах із різними версіями Visual Studio, а саме в Visual Studio 2017, Visual Studio 2015 та Visual Studio 2010, де була перевірена основна функціональність програми й здійснена перевірка на сумісність з іншими версіями компілятора Visual Studio.

Для моніторингу за користувачами застосунок автоматично підключається до віддаленого сервера.

Під час проведення експериментів був проведений моніторинг 100 студентів із використанням пропонованої системи та декількома популярними аналогами, а саме AnyDesk, CRD, MRD. Були зафіксовані 12 порушень: (1) *створення нового вікна браузера*; (2) *відкриття нової вкладки браузера;* (3) *зміна розмірів головного вікна*; (4) *увімкнення сторонніх утиліт*; (5) *підключення через віддалений робочий стіл;* (6) *пошук відповіді за запитом в інтернеті*; (7) *використання іншого монітора*; (8) *використання телефона;* (9) *допомога сторонньої особи*; (10) *поворот голови від екрана*; (11) *переключення між різними програми або браузерами*; (12) *вимкнення системи моніторингу*. Однак не всі системи змогли детектувати порушення (3), (4), (5), (8), (9), (10). Результати детектування порушень подано в табл. 1.

Таблиця 1 – Порівняльний аналіз детектування найскладніших порушень

Порушення	Кількість порушень (детектованих / загальна)			
	CRD	MRD	AnyDesk	**ССПС**
(3)	0/19	0/19	0/19	**19/19**
(4)	0/3	0/3	0/3	**2/3**
(5)	0/2	0/2	0/2	**1/2**
(8)	0/9	0/9	0/9	**9/9**
(9)	0/5	0/5	0/5	**5/5**
(10)	0/11	0/11	0/11	**10/11**

На відміну від аналогів, розроблена система спостереження за поведінкою студентів може виявляти значно більшу кількість порушень. Крім того, для оцінювання ефективності роботи пропонованої системи ССПС використовується сукупність критеріїв, що дозволяють визначити реакцію детектування

порушень. Основним критерієм, завдяки якому досягається високий рівень точності детектування, є час детектування порушень. На відміну від аналогів, у розробленій системі є можливість налаштування часових інтервалів для функцій детектування.

Висновки

Запропоновано модель процесу спостереження за поведінкою студентів на основі розпізнавання обличчя, яка дозволяє в режимі реального часу відстежувати порушення під час проведення е-тестування. Для реалізації системи моніторингу на основі запропонованої моделі використана мова програмування C# на платформі .NET (клієнтська частина) та CMS WordPress (серверна частина). Інтерфейс користувача та організація взаємодії з ним реалізується за допомогою основних можливостей HTML, CSS, JavaScript, PHP та C#.

Описано принцип роботи розробленої системи та зроблений порівняльний аналіз з наявними аналогами.

Література

1. Экзамус. URL: https://ru.examus.net/ (дата звернення: 14.12.2021).
2. ExamCookie. URL: https://www.exam-cookie.dk (дата звернення: 15.12.2021).
3. ProctorEdu. URL: https://proctoredu.ru/ (дата звернення: 15.12.2021).
4. Exam Monitor. URL: https://sdu. exammonitor.dk/ (дата звернення: 16.12.2021).
5. Proctortrack. URL: https://www.proctortrack.com/proctorlive-ai/ (дата звернення: 16.12.2021).
6. Microsoft Remote Desktop. URL: https://www.microsoft.com/en-us/p/microsoft-remote-desktop/9wzdncrfj3ps (дата звернення: 17.12.2021).
7. Remote Utilities. URL: https://www.remoteutilities.com/ (дата звернення: 17.12.2021).
8. UltraVNC. URL: https://uvnc.com/ (дата звернення: 18.12.2021).
9. AeroAdmin. URL: https://www.aeroadmin.com/ru/ (дата звернення: 18.12.2021).
10. DameWare. URL: https://www.dameware. com/ (дата звернення: 19.12.2021).
11. Chrome Remote Desktop. URL: https://remotedesktop.google.com/ (дата звернення: 19.12.2021).
12. AnyDesk. URL: https://anydesk.com/(дата звернення: 20.12.2021).
13. Cloud-fog-dew Architecture for Personalized Service-oriented Systems / N. Axak, D. Rosinskiy, O. Barkovska, I. Novoseltsev // The 9th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2018, Kyiv, Ukraine, 2018. P. 80–84.
14. Татарников А.О., Аксак Н.Г. Дослідження методів обробки та класифікації емоцій людини при розробці систем комп'ютерного зору. *Теорія і практика сучасної науки*: Матеріали II міжнародної науково-теоретичної конференції (м. Краків, 12 листопада 2021). Краків, 2021. С. 91–92.
15. Іващенко Г.С., Татарников А.О. Організація взаємодії та синхронізації даних між sqlite і mysql. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління:* Матеріали десятої міжнародної науково-технічної конференції (м. Харків, 9–10 квітня 2020). Харків, 2020. С. 70.

References

1. Examus. URL: https://ru.examus.net/ (accessed: 14.12.2021).
2. ExamCookie. URL: https://www.exam-cookie.dk (accessed: 15.12.2021).
3. ProctorEdu. URL: https://proctoredu.ru/ (accessed: 15.12.2021).
4. Exam Monitor. URL: https://sdu. exammonitor.dk/ (accessed: 16.12.2021).
5. Proctortrack. URL: https://www.proctortrack.com/proctorlive-ai/ (accessed: 16.12.2021).
6. Microsoft Remote Desktop. URL: https://www.microsoft.com/en-us/p/microsoft-remote-desktop/9wzdncrfj3ps (accessed: 17.12.2021).
7. Remote Utilities. URL: https://www.remoteutilities.com/ (accessed: 17.12.2021).
8. UltraVNC. URL: https://uvnc.com/ (accessed: 18.12.2021).
9. AeroAdmin. URL: https://www.aeroadmin.com/ru/ (accessed: 18.12.2021).
10. DameWare. URL: https://www.dameware. com/ (accessed: 19.12.2021).
11. Chrome Remote Desktop. URL: https://remotedesktop.google.com/ (accessed: 19.12.2021).
12. AnyDesk. URL: https://anydesk.com/ (accessed: 20.12.2021).
13. Cloud-fog-dew Architecture for Personalized Service-oriented Systems / N. Axak, D. Rosinskiy, O. Barkovska, I. Novoseltsev // The 9th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2018, Kyiv, Ukraine, 2018. P. 80–84.
14. Tatarnykov A.O., Axak N.G. The research of methods of processing and classification of the human emotions in the development of the computer vision systems. *Theory and practice of modern science*: Proceedings of the II International Scientific and Theoretical Conference (Krakow, November 12 2021). Krakow, 2021. P. 91–92 [in Ukrainian].

15. Ivashchenko H.S., Tatarnykov A.O. Organization of interaction and synchronization of data between sqlite and mysql. *Modern directions of development of information and communication technologies and means of management*:. Proceedings of the tenth international scientific and technical conference (Kharkiv, April 9–10 2020). Kharkiv, 2020. С. 70. [in Ukrainian].

Аксак Наталія Георгіївна, д.т.н., проф. каф. інформаційних систем, Харківський національний економічний університет ім. С. Кузнеця, nataliia.axak@nure.ua, тел. +38 050-142-18-80,
Татарников Андрій Олександрович, магістрант, Харківський національний університет радіоелектроніки, andrii.tatarnykov@nure.ua, тел. +38 050-766-18-85.

The system of monitoring students' behavior during e-tests

Abstract. Problem. *In connection with the transfer of the educational process to a fundamentally new level, the role of independent work of students, which has become the main form of the knowledge, is growing significantly. Therefore, the computer testing is becoming increasingly important as a means of monitoring the students' learning materials. These speeds up the control process and ensures its objectivity. However, the testing process is vulnerable to possible students' abuse – use of aids, outside help, etc.* ***Goal.*** *The goal is developing a client-server system for analyzing students' behavior during e-testing.* ***Methodology.*** *Solution of the set tasks involved application of the set-theoretical approach in order to build up the model of an observation process for students' behavior and their component organization as well as their interaction, image processing methods.* ***Results.*** *A model of the monitoring process of the students' behavior is proposed, which allows real-time monitoring of violations during e-testing based on face recognition. The principle of operation of the developed system also described comparative analysis with existing analogues.* ***Originality.*** *A monitoring system is proposed, which includes the following features: tracking the active URLs, obtaining the list of network connections, real-time tracking of any changes in parameters (width and height) of the active window, creation of screenshots of the working area of the computer screen and photographs using a web camera, automatic reading of the contents of the clipboard and keeping a log of pressed keys, taking screenshots of the working area of the computer screen and photographs using a web camera, recognizing the student's face and tracking their position regarding the webcam. If violations are detected, the received messages are recorded and sent to the server.* ***Practical value.*** *The use of the proposed monitoring system for students' behavior will allow to remotely identify a larger number of violations that were not previously detected.*

Key words: *electronic testing, monitoring system, recognition.*

Axak Natalia, professor, Doct. of Science, Department of information systems, Simon Kuznets Kharkiv National University of Economics, nataliia.axak@nure.ua tel. +38 050-142-18-80,
Tatarnykov Andrii, undergraduate, Kharkiv National University of Radioelectronics, tel. 38 050-766-18-85, andrii.tatarnykov@nure.ua.