МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

КОМП'ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ:
ОСВІТА, НАУКА, ВИРОБНИЦТВО

НАУКОВИЙ
ЖУРНАЛ

Відповідальний редактор – проф.,д-р техн.наук В. Д. Рудь

№23 2016

м. Луцьк
Видавництво Луцького національного технічного університету

Адреса редколегії:
Луцький національний технічний університет,
кафедра комп'ютерної інженерії.
вул.Львівська 75, ауд.141
м.Луцьк, 43018
тел. (0332) 74-61-15
E-mail: cit@lntu.edu.ua,
ekaterinamelnik@gmail.com
сайт журналу: **ki.lutsk-ntu.com.ua**

КОМП'ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ: ОСВІТА, НАУКА, ВИРОБНИЦТВО

№23 2016р.

Журнал засновано у грудні 2010 р.
Свідоцтво про реєстрацію КВ № 16705–5277 Р.
Засновник: Луцький національний технічний університет
Рекомендовано до друку Вченою радою Луцького національного технічного університету (протокол №12 засідання від 24.05.2016)
Журнал рішенням МОН України **наказом №515 від 16.05.2016р,** включено в перелік наукових фахових видань.
Журнал має російський індекс наукового цитування (РІНЦ)

ISSN 2524-0560 (Online)
ISSN 2524-0552 (Print)

ЗМІСТ

УДК 004.93: 621.313
Аль-Джасри Г.Х.М., Болтенков В.А. к.т.н., Червоненко П.П. к.т.н.
Одесский национальный политехнический университет

ЛОКАЛЬНО-КОГЕРЕНТНАЯ ОБРАБОТКА ИЗМЕРИТЕЛЬНОЙ ИНФОРМАЦИИ В СИСТЕМАХ АКУСТИЧЕСКОГО МОНИТОРИНГА ТЕЧЕЙ ТЕПЛОНОСИТЕЛЯ

Аль-Джасрі Г.Х.М., Болтьонков В.О., Червоненко П.П. Локально-когерентна обробка вимірювальної інформації в системах акустичного моніторингу течій теплоносія. Запропоновано метод підвищення точності локалізації течі теплоносія акустичною мікрофонною системою. Метод дозволяє виділити області локальної когерентності на парах мікрофонів в умовах багаторазового відбиття акустичного сигналу всередині приміщень. При оцінці координат течі за TDOA технологією пропонується враховувати тільки ті оцінки TDOA, для яких квадрат модуля когерентності вище порогового. Запропонований метод дозволяє зменшити помилку оцінки координат течі в 2-4 рази.
Ключові слова: моніторинг течій теплоносія, TDOA технології, квадрат модуля когерентності, локально-когерентна обробка, форм-фактор кореляційної функції.

Аль-Джасри Г.Х.М., Болтенков В.А., Червоненко П.П. Локально-когерентная обработка измерительной информации в системах акустического мониторинга течей теплоносителя. Предложен метод повышения точности локализации течи теплоносителя при помощи акустической микрофонной системы. Метод позволяет выделить области локальной когерентности на парах микрофонов в условиях многократных отражений акустического сигнала внутри помещений. При оценке координат течи по TDOA технологии предлагается учитывать только те оценки TDOA, для которых квадрат модуля когерентности выше порогового. Предложенный метод позволяет уменьшить ошибку оценки координат течи в 2-4 раза.
Ключевые слова: мониторинг течей теплоносителя, TDOA технологии, квадрат модуля когерентности, локально-когерентная обработка, форм-фактор корреляционной функции.

Al-Jasri G. Kh. M.,Boltenkov V.A., Chervonenko P.P. Locally-coherent Processing of the Measuring Mnformation in the Acoustic Water Leak Monitoring Systems. A method for improving the localization accuracy of coolant leaks with the acoustic microphone systems. The method allows to distinguish the areas of local coherence on the microphone pairs in multiple reflections conditions of the acoustic signal indoors. In assessing the origin of a leak on the TDOA technology It is proposed to take into account only those TDOA estimation, for which square coherence module is above coherence threshold. The proposed method gives possibility to reduce the error of leak coordinate estimation in 2-4 times.
Keywords: water leaks monitoring, TDOA technology, square coherency module, locally-coherent processing, correlation function form factor.

Постановка научной проблемы. Проблема создания систем мониторинга течей теплоносителя в энергетическом оборудовании весьма актуальна как с точки зрения экономии энергоресурсов, так и в плане повышения сохранности основного металла оборудования теплоэнергетических агрегатов, повреждаемого при возникновении и развитии течей. Несмотря на практическую важность проблемы на сегодняшний день не существует надежных и в то же время достаточно информативных систем мониторинга течей. Одним из путей решения проблемы являются акустические бесконтактные системы мониторинга течей [1]. Принцип действия таких систем основан на регистрации звукового сигнала, возникающего при истечении перегретого теплоносителя через дефект, пространственно разнесенной системой акустических датчиков (микрофонов), установленных в контролируемом технологическом помещении – акустической сенсорной сетью. При истечении теплоносителя происходят достаточно сложные процессы звукообразования, акустический сигнал от течи является широкополосным и занимает частотную полосу (2-45) кГц при общем уровне звукового давления до 95 дБ [2].

Основными задачами системы акустического мониторинга течей являются:

– обнаружения факта течи,

– локализация течи, т.е. оценка координат дефекта, через который происходит истечение теплоносителя,

– контроль состояния и работоспособности акустических сенсоров (микрофонов).

Задача локализации течи является наиболее сложной в связи с тем, что в технологических помещениях акустический сигнал испытывает затухание, многократные переотражения от объектов, находящихся в помещении и его стен. Задачу локализации предлагается решать на основе TDOA технологии, т.е. путем оценивания разностей времен прихода (англ. Time Differences Of Arrivals – TDOA) сигнала на разнесенные в пространстве микрофоны [3].

Целью данного исследования является разработка метода, позволяющего повысить точность оценивания TDOA и соответственно точность оценивания координат течи в сложных условиях образования звукового поля в технологическом помещении.

Пусть сигнал широкополосного источника звука (течи) регистрируется сетью из N пространственно разнесенных микрофонов (координаты их полагаются произвольными, но известными). Технология обработки измерительной информации такова. Для каждой пары микрофонов *(i,j)* (а таких пар при N микрофонах в помещении существует $N(N-2)/2$) оценивается обобщенная взаимно-корреляционная функция (ВКФ) по Кнеппу-Картеру $\hat{R}_{ij}(\tau)$ [4]. Положение максимума ВКФ позволяет оценить разности времен прихода (Time Differences Of Arrivals – TDOA) $\hat{\tau}_{ij}$ на i-й и j-й микрофоны ($i=1,...,N,\ j=1,...,N, i\neq j$):

$$\hat{\tau}_{ij} = \arg\max_{\tau\in T} \hat{R}_{ij}(\tau), \tag{1}$$

где T – интервал анализа.

Для каждого пространственного тетраэдра, образованного любыми четырьмя микрофонами можно составить систему TDOA-уравнений:

$$\begin{aligned}
&\sqrt{(x_0-x_1)^2+(y_0-y_1)^2+(z_0-z_1)^2}-\sqrt{(x_0-x_2)^2+(y_0-y_2)^2+(z_0-z_2)^2}=c\hat{\tau}_{12},\\
&\sqrt{(x_0-x_1)^2+(y_0-y_1)^2+(z_0-z_1)^2}-\sqrt{(x_0-x_3)^2+(y_0-y_3)^2+(z_0-z_3)^2}=c\hat{\tau}_{13},\\
&\sqrt{(x_0-x_1)^2+(y_0-y_1)^2+(z_0-z_1)^2}-\sqrt{(x_0-x_4)^2+(y_0-y_4)^2+(z_0-z_4)^2}=c\hat{\tau}_{14},
\end{aligned} \tag{2}$$

где (x_0, y_0, z_0) - координаты источника звука, $(x_k, y_k, z_k), k=1,...,4$ - координаты четырех микрофонов из всего их набора, $\hat{\tau}_{ij,}\ i=1,\ j=2,3,4$ - соответствующие оценки TDOA, c - скорость звука для условий измерения. Приведенная выше система уравнений описывает три гиперболоида вращения, которые являются поверхностями положения, т.к. на поверхностях гиперболоидов $\hat{\tau}_{ij}=const$. Решение системы уравнений дает оценку местоположения источника $(\hat{x}_0, \hat{y}_0, \hat{z}_0)_i$. Оценки координат источника передаются на следующий уровень обработки. На следующем уровне окончательная оценка местоположения источника определяется по какой-нибудь из многочисленных известных стратегий [3] , например, как среднее по всем оценкам, полученным на предыдущем уровне.

Точность оценивания координат течи в первую очередь определяется точностью оценивания значений TDOA для каждой пары микрофонов $\hat{\tau}_{ij}$, которая в свою очередь определяется степенью когерентности сигналов, регистрируемых каждой парой микрофонов [5]. Количественной мерой когерентности двух случайных сигналов $x(t)$ и $y(t)$, принимаемых парой микрофонов, является комплексная функция когерентности, равная взаимному спектру мощности $C_{xy}(f)$, нормированному к корню квадратному из произведения собственных спектров мощности этих сигналов:

$$\gamma_{xy}(f) = \frac{G_{xy}(f)}{\sqrt{G_{xx}(f)G_{yy}(f)}}, \tag{3}$$

где f - частота. На практике обычно используется квадрат модуля когерентности (КМК) $\gamma^2_{xy}(f)$. По физическому смыслу, как функция когерентности, так и КМК – это коэффициент корреляции (или линейной связи) пары сигналов на каждой частоте f анализируемого частотного диапазона [5,6] На практике удобнее использовать КМК, как функцию по определению нормированную к единице: в частности, если $\gamma^2_{xy}(f)=1$ на всех частотах исследуемого частотного диапазоне сигналы считаются полностью когерентными, при $\gamma^2_{xy}(f)=0$ сигналы некогерентны. На практике при обработке широкополосных сигналов $0<\gamma^2_{xy}(f)<1$, причем порогом когерентности, выше которого когерентность считается существенной для широкополосных сигналов, служит величина (0,75-0,8) [6]. Изложенная выше схема оценивания координат широкополосного источника не предполагает учета свойств взаимной когерентности сигналов. Целью исследования, результаты которого приведены в статье, является повышение точности оценивания TDOA, а соответственно и оценивания координат течи, путем учета степени взаимной когерентности сигналов, принимаемых разнесенными парами микрофонов.

Исследования проведены на многоканальных цифровых записях сигналов течей, полученных в экспериментах поставленных в НИЛ «Атомспецавтоматика» Одесского национального политехнического университета под руководством д.т.н., проф. А.В. Королева [7]. Общая схема

эксперимента такова. Специальный теплофизический стенд позволяет имитировать сигналы от различных видов дефектов в элементах теплотехнического оборудования на реальных рабочих давлениях и температурах. Генерируемый при модельной течи акустический сигнал регистрируется группой из 8-ми пространственно разнесенных измерительных конденсаторных микрофонов МФК–003М с улучшенными характеристиками (рабочая амплитудно-частотная характеристика 20-50000 Гц при неравномерности не более 5 дБ). В частности, на рис.1 приведена спектральная плотность мощности (СПМ) акустического сигнала, зарегистрированного микрофоном №1 от имитировавшейся течи через дефект уплотненной прокладки фланцевого соединения размером 0,2 мм, рассчитанная методом периодограмм Уэлча [8] .

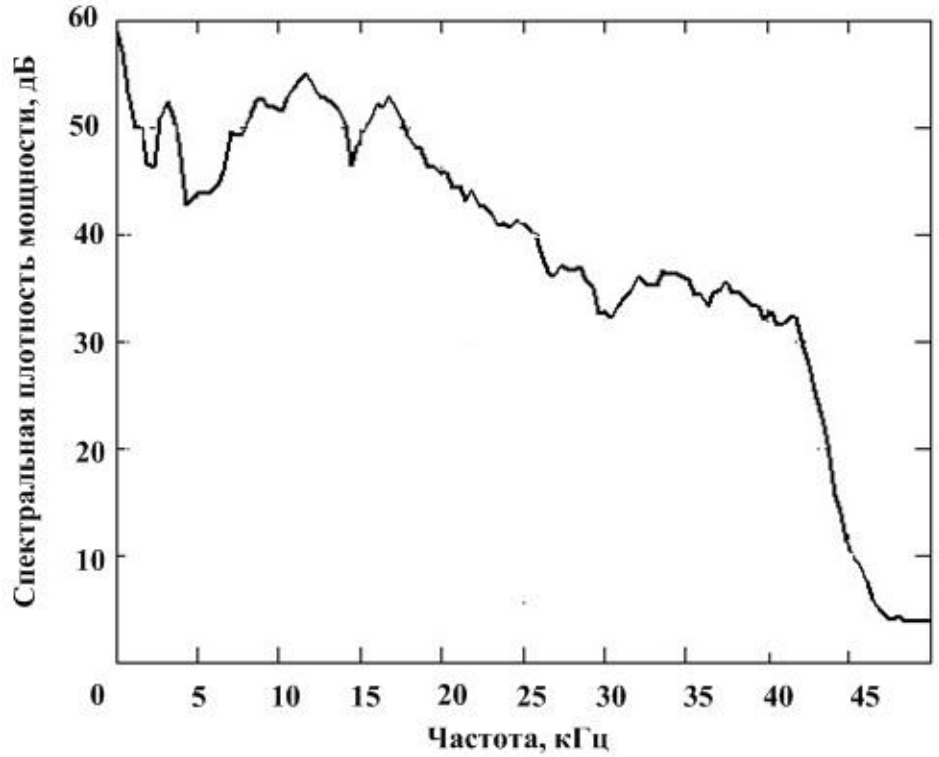


Рис.1. Спектральная плотность мощности для сигнала протечки.

Функция КМК двух сигналов от той же течи, зарегистрированных на паре микрофонов №1 и №2, разнесенных на 4,6 м, приведена на рис.2.

Анализ вида функции когерентности показывает, что в некоторых областях анализируемого частотного диапазона когерентность высока, а в некоторых областях частотного диапазона она практически незначительна. Среди причин снижения когерентности сигналов, принимаемых разнесенными микрофонами, выделим следующие.

1) Процесс локализации течи происходит, как правило, в помещении, содержащем оборудование, трубопроводы и другие отражающие и поглощающие объекты. В результате возникает многолучевое распространение сигнала (реверберация). В результате этого эффекта в точке приема складываются как сигнал, пришедший по прямому пути, так и его многократно отраженные реплики. Интерференция частотных составляющих широкополосного сигнала в точке приема приводит к существенной потере когерентности в отдельных участках частотного диапазона. В частности, эксперимент проводился в прямоугольном помещении с размерами 9,4 м * 11,5 м * 5,5 м. Кроме отражающих поверхностей (стен, потолка, застекленных оконных проемов) в помещении находилось различное оборудование и трубопроводы из различного материала и различных размеров. Все перечисленные элементы играли роль системы хаотически расположенных отражателей.

2) Коэффициент поглощения звука при распространении является частотно зависимой функцией, пропорциональной квадрату частоты [7]. Это также в совокупности с эффектом многолучевого распространения вносит вклад в потерю когерентности сигналов.

3) Хотя при эксперименте контролировалась скорость звука c путем учета ее температурной зависимости

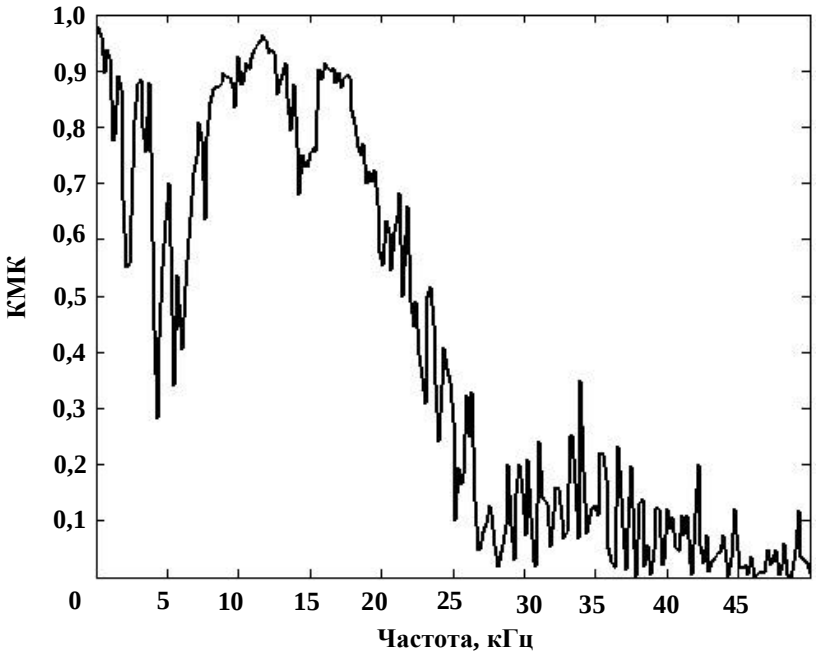


Рис. 2. Квадрат модуля когерентности для пары сигналов,
зарегистрированных микрофонами, разнесенными на расстояние 4,6 м.

$$c = 331 + 0.6t^0 \text{ (м / сек)}, \quad (4)$$

где t^0 - температура воздуха в помещении (в градусах Цельсия), вблизи зоны течи существуют области высокой температуры (до 150^0C и выше), что вызывает фазовые флуктуации частотных компонент широкополосного сигнала, пропорциональные четвертой степени температуры.

Все перечисленные факторы приводят к формированию сложной картины взаимной когерентности сигналов, принимаемых парой разнесенных микрофонов.

Анализ картины КМК позволяет установить, что в отдельных частотных диапазонах когерентность сигналов достаточно высока ($\gamma_{xy}^2(f) > 0{,}8$). Очевидно, при оценке ВКФ, а затем и TDOA по положению максимума ВКФ именно области высокой когерентности вносят основной вклад в формирование эффективной статистической оценки. Назовем эти области областями локальной когерентности (Local Coherence Area – LCA) или «окнами когерентности» по аналогии с «окнами прозрачности» в атмосферной оптике и распространении радиоволн [9].

Сформулируем алгоритм локально-когерентной обработки сигналов для оценки TDOA.

Шаг 1. Для каждой пары микрофонных датчиков рассчитать функцию КМК $\gamma_{xy}^2(f)$ во всем анализируемом диапазоне частот.

Шаг 2. Оценить $\max\limits_f \gamma_{xy}^2(f)$. Если $\max\limits_f \gamma_{xy}^2(f) < 0{,}8$, то Выход. В случае общей низкой когерентности на паре микрофонов, эту пару следует исключить из дальнейшего рассмотрения, поскольку от нее нельзя ожидать качественной оценки TDOA.

Шаг 3. Если $\max\limits_f \gamma_{xy}^2(f) > 0{,}8$, перейти к шагу 4.

Шаг 4. Для функции КМК определить области локальной когерентности («окна когерентности»), т.е. частотные области $(fн, fв)_k$, где выполняется условие $\gamma_{xy}^2(f) > 0{,}8$, $k = 1, \ldots, N_{LCA}$, где k - номер окна когерентности, N_{LCA} - общее число окон когерентности в анализируем частотном диапазоне, $(fн, fв)_k$ - соответственно нижняя и верхняя частотная граница k -го окна.

Шаг 5. Если $(fв - fн)_k < 200$ Гц, исключить окно с номером k из дальнейшего рассмотрения, ввиду его малой ширины.

Шаг 6. Для всех выявленных N_{LCA} окон когерентности синтезировать полосовые фильтры с частотами среза $(fн, fв)_k$. Поскольку для оценки TDOA фильтр должен обладать максимально линейной

фазовой характеристикой, удобно синтезировать фильтры Селестника-Ланга-Барруса (ФСЛБ) [10], которые обеспечивают постоянство групповой задержки при фильтрации и одновременно обеспечивают несколько требуемых полос пропускания. Практический опыт показывает, что достаточно синтезировать цифровые ФСЛБ 20-25-го порядка.

Шаг 7. Пропустить сигналы $x(t)$ и $y(t)$, принимаемых парой микрофонов, через синтезированные полосовые фильтры, соответствующие выявленным окнам когерентности.

Шаг 8. Для каждого из профильтрованных сигналов вычислить ВКФ и оценку TDOA $\hat{\tau}_{ij}^{(k)} = \arg\max_{\tau \in T} \hat{R}_{ij}(\tau)$.

Шаг 9. Усреднить N_{LCA} полученных оценок РВП.

Шаг 10. Выход.

Проиллюстрируем эффективность предложенного алгоритма на экспериментальных данных, полученных в условиях, описанных выше. На рис.4 приведена ВКФ для указанной пары микрофонов.

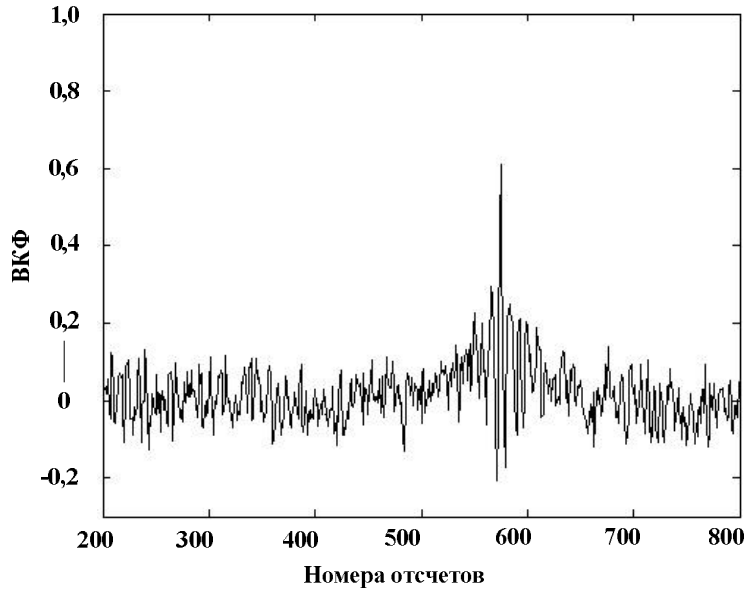


Рис.3. Взаимно-корреляционная функция для пары сигналов без учета их когерентности.

Качество корреляционной обработки будем количественно оценивать двумя параметрами: смещением оценки TDOA $bias(\Delta\tau_{ij})$ относительно истинной (которая в данном случае известна из геометрии эксперимента и скорости звука для условий измерения) $\Delta\tau_{ij}$ и форм-фактором взаимно-корреляционной функции, под которым будем понимать отношение площади главного лепестка модуля ВКФ к общей площади модуля ВКФ на всем интервале анализа:

$$FF = \frac{S(abs(R_{ij}(\tau)))_{main\_lobe}}{\underset{T}{S}(abs(R_{ij}(\tau)))}. \quad (5)$$

Практика оценивания TDOA показывает, что рост введенного форм-фактора ВКФ существенно уменьшает дисперсию оценки $\Delta\tau_{ij}$. Для приведенного случая $bias(\Delta\tau_{ij})$=584-498=86 (истинное значение задержки равно $\Delta\tau_{ij}$ =498 отсчетов) и FF =0,6%.

Применим предложенный алгоритм локально-когерентного анализа. На рис.4 показаны окна когерентности для данного случая.

В описываемой акустической ситуации выявлено два окна когерентности: LCA1(7960-14150)Гц и LCA2(15400-17950)Гц.

После фильтрации исходных сигналов ФСЛБ 25-го порядка в окне LCA1(7960-14150)Гц получена ВКФ, приведенная на рис.5.

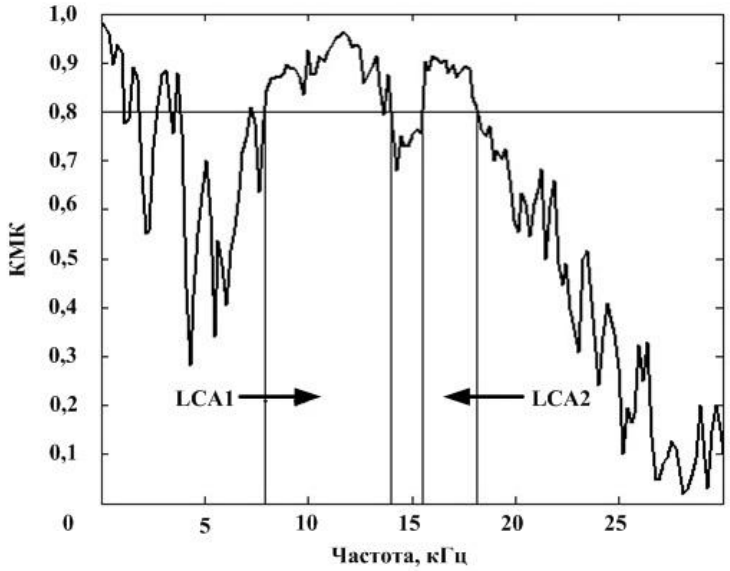


Рис.4. Области локальной когерентности для экспериментального примера.

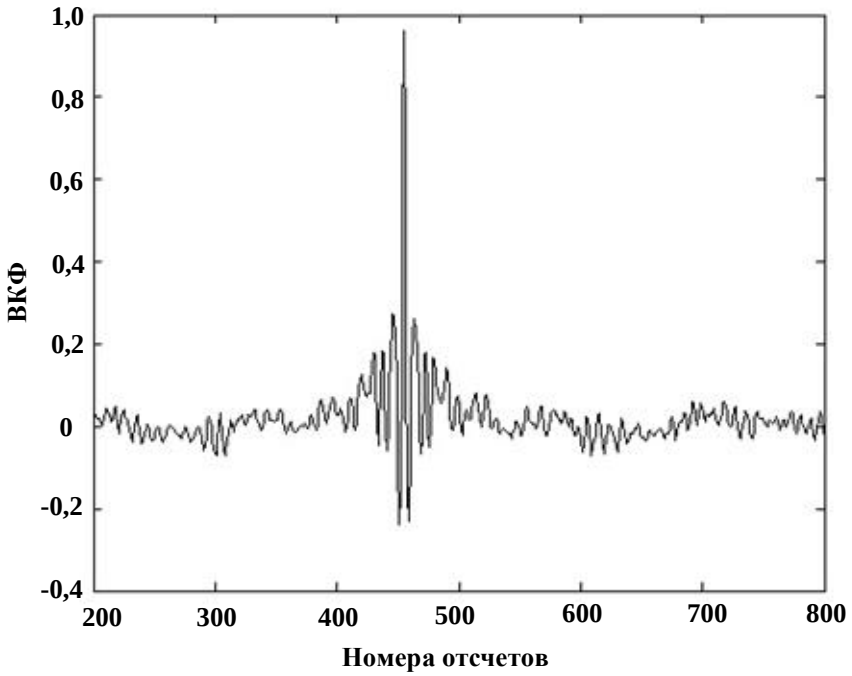


Рис.5. Взаимно-корреляционная функция после фильтрации сигналов в окне когерентности LCA1(7960-14150)Гц.

После фильтрации параметры в первом окне прозрачности показатели качества оценки ВКФ стали такими: $bias(\Delta\tau_{ij})$ =452-498=-46 и FF =2,11%. Усредненная по двум окнам когерентности оценка TDOA $\bar{\tau}_{ij}$ =518 (напомним, что истинное значение задержки равно $\Delta\tau_{ij}$ =498 отсчетов). Предложенная локально-когерентная обработка проведена для всех пар, образованных восемью микрофонами. При этом число окон когерентности изменялось от двух (для пар наиболее удаленных микрофонов) до пяти (для пар близко расположенных микрофонов). В таблице 1 приведены оценки координат течи, полученные прямым путем и с применением локально-когерентной обработки.

Таблица 1
Сравнительный анализ точности оценивания координат течи

Координаты течи	Истинные значения	Оценки, полученные прямой обработкой	Ошибки оценивания при прямой обработке	Оценки, полученные локально-когерентной обработкой	Ошибки оценивания при локально-когерентной обработке
x	6,55 м	7,03 м	7,32%	6,67 м	1,83%
y	7,22 м	6,84 м	5,26%	7,04 м	2,49%
z	0,60 м	0,82 м	3,67%	0,61 м	1,61%

Анализ таблицы 1 показывает, что точность оценивания координат течи при локально-когерентной обработке существенно повышается: так относительная ошибка оценивания координат течи уменьшается в 2-4 раза по сравнению с некогерентной обработкой.

Выводы и перспективы. В работе предложен метод локально-когерентной обработки широкополосных звуковых сигналов при локализации течей в теплотехническом оборудовании. Метод, основанный на оценке частотных диапазонов, в которых взаимная когерентность акустических сигналов на паре микрофонов выше установленного порога (окон когерентности) позволил существенно уменьшить смещение и дисперсию оценок TDOA тем самым повысить точность оценивания координат течи.

При формулировке алгоритма предполагался стационарный характер течи и соответственно уровня генерируемого звукового сигнала. В таких условиях предложенный алгоритм легко реализуется в реальном масштабе времени (в частности с применением систем программирования Matlab, Scilab, LabView). Однако при течах с большим расходом может наблюдаться интенсивное во времени развитие течи и предложенная процедура может не уложиться в темп нарастания уровня звукового сигнала и изменения его спектра. Поэтому в дальнейших исследованиях предполагается усовершенствование предложенного алгоритма с реализацией его как адаптивного и перестройкой передаточных функций фильтров в реальном масштабе времени.

1. В.А. Болтенков, Г.Х.М. Аль-Джасри. Исследование акустических систем мониторинга течей теплоносителя // – 2015. – Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2015. – Вип. 20. – С. 16-22.
2. Болтенков В.А. Алгоритмы обработки информации при акустическом бесконтактном поиске протечек на верхнем блоке реактора ВВЭР-1000 / В. А. Болтенков, А. В. Королев, М. В. Максимов, О. В. Маслов. // Известия высших учебных заведений и энергетических объединений СНГ: Энергетика – 2009. – N 3. – С. 67-72.
3. Huseynov J., Baliga S, . Dillencourt M. et al. Gas-leak Localization Using Distributed Ultrasonic Sensors // Proc. – 2009. – 7293, Smart Sensor Phenomena, Technology, Networks, and Systems. – 72930Z. –DOI:10.1117/12.812058.
4. Knapp C.H., Carter G.C. The Generalized Correlation Method for Estimation of Time Delay// IEEE Trans. Acoust., Spech, Signal Process. – 1976. – Vol. 24, №4 – P.320-327.
5. Картер Г.К. Оценивание когерентности и временной задержки // ТИИЭР. – 1987. – Том 75, №2 – С.64-84.
6. Бендатт Дж, Пирсол А. Прикладной анализ случайных данных. М.: Мир, 1989. – 540 с.
7. Kozick R.J., Sadler B.M. Algorithms for Localization and Tracking of Acoustic Sources with Widely Separated Sensors // Proc. of 2000 Meeting of the IRIS Specialty Group on Battlefield Acoustics and Seismics. – Laurel,MD, 2000. — P.35-76.
8. Королев А.В. Экспериментальное исследование особенностей акустических сигналов при истечении пара и газа через различные насадки / Королев А.В., Литвин А.Н. // Ядерная и радиационная безопасность. – 2002. – Т. 5, Вып. 4. – С. 38-40.
9. Сергиенко А. Б. Цифровая обработка сигналов. – СПб.: БХВ-Петербург, 2011. – 768 с.
10. Зуев В.Е., Наац И.Э. Обратные задачи оптики атмосферы. Л.: Гидрометеоиздат, 1989. – 286 с.
11. Selesnick, I. W., M. Lang, and C. S. Burrus. Constrained Least Square Design of FIR Filters without Specified Transition Bands // IEEE Transactions on Signal Processing – 1996 – Vol. 44, No. 8 – P. 1879–1892.

UDC 519.865:658.8.012.12
H. M. Hubal
Lutsk national technical university

SOME LINEAR MATHEMATICAL MODELS IN ECONOMICS

Hubal H. M. Some linear mathematical models in economics. A linear mathematical model of a diversified economy and a linear mathematical model of exchange are explored in this paper. Mathematical analysis of these models is carried out.
Keywords: model of a diversified economy, model of exchange, technological matrix, matrix of trade.
Bibl. 3.

Губаль Г. М. Деякі лінійні математичні моделі в економіці. У статті досліджено лінійну математичну модель багатогалузевої економіки і лінійну математичну модель обміну. Здійснено математичний аналіз цих моделей.
Ключові слова: модель багатогалузевої економіки, модель обміну, технологічна матриця, матриця торгівлі.
Літ. 3.

Губаль Г. Н. Некоторые линейные математические модели в экономике. В статье исследовано линейную математическую модель многоотраслевой экономики и линейную математическую модель обмена. Проведено математический анализ этих моделей.
Ключевые слова: модель многоотраслевой экономики, модель обмена, технологическая матрица, матрица торговли.
Лит. 3.

Introduction. Linear mathematical models of a diversified economy and exchange are explored in this paper. We use mathematical apparatus of the theory of matrices to construct and to solve systems of linear algebraic equations.

The main part. All linear mathematical models have the properties of additivity and homogeneity. Additivity means that if the variable x_1 creates the effect γ_1 on its single use, the variables x_1, x_2 create the effect $\gamma_1 + \gamma_2$ on their joint use. Homogeneity means that if the variable x_1 creates the effect γ_1, then, for an arbitrary real number λ, the variable λx_1 creates the effect $\lambda\gamma_1$.

A linear mathematical model of a diversified economy. Consider a simplified economic mathematical model of inerindustry balance [1, 2]. A linear mathematical model in an economy asuumes that an economy consists of a number of interacting industries each consuming the products, including its own, and producing another, and that all these industries are connected with the final demand for final consumption goods. This assumption is a simplified reflection of the labour division between successive phases of production, such as production and processing of raw materials, transportation of the finished product to the market. By using the linear equations and the coefficients of proportionality a_{ij} depending on industry technology we can define how many products every industry has to produce to satisfy its demand, the demand of another industries and population that are supposed to be known. Connection between industries is displayed in the tables of interindustry balance, a mathematical model that allows to analyze them being recommended to connect separate industries with consumer demand, to determine the optimal prices and to calculate the maximum profit.

We use the matrix method for solving systems of linear algebraic equations in a mathematical model of a diversified economy.

The purpose of balance analysis is to answer the question that arises in macroeconomics and connected with the efficiency of conducting the diversified economy: what ought the production balance of each of *n* industries to be for all needs for products of the industry given to be satisfied? However, every industry act, on the one hand, as a manufacturer of products and, on the other hand, as a consumer of its products and products produced by other industries. For example, the automotive industry by the steel in the steel undustry, the tires in the rubber industry, the electric power in the electric power industry, etc.

Consider a static mathematical model of the economy (the static means that in the review period we do not take into account any change in time). The model is linear. This allows us to determine the amount of products that is necessary to satisfy the market demand and to determine the prices for it through added magnitudes.

Assume that *n* types of products are are produced, purchased, consumed and invested, *n* types of industry each producing their products are considered in an economic system. One part of products is for consumption by this industry and other ones, the other part being for sale (consumption) in non-production sphere.

Let an economic production system consist of *n* industries, i.e. produce *n* types of products.

Consider the production process for a certain time period, for example, for a year.

We display the connection between industries by the scheme of production and distribution interindustry

balance shown in table 1.

Industry production	Production distribution between industries						Sum for production needs $\sum$	The volume of final production	The gross volume of production
	1	2	…	j	…	n			
1	x_{11}	x_{12}	…	x_{1j}	…	x_{1n}	$\sum_{j=1}^{n} x_{1j}$	Y_1	X_1
2	x_{21}	x_{22}	…	x_{2j}	…	x_{2n}	$\sum_{j=1}^{n} x_{2j}$	Y_2	X_2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
i	x_{i1}	x_{i2}	…	x_{ij}	…	x_{in}	$\sum_{j=1}^{n} x_{ij}$	Y_i	X_i
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
n	x_{n1}	x_{n2}	…	x_{nj}	…	x_{nn}	$\sum_{j=1}^{n} x_{nj}$	Y_n	X_n

Table 1

Denote the total (gross) volume of the *i*-th industry production by X_i, the volume of the *i*-th industry products ($i=\overline{1,n}$) consumed (wasted) by the *j*-th industry in production process by x_{ij} ($i,\ j=\overline{1,n}$), the volume of the final products of the *i*-th industry for non-production consumption by Y_i ($i=\overline{1,n}$).

Note that the gross volume of production is breafly called the gross product. The volume of the final product for non-production consumption is breafly called the friendly product.

Indicators given above, namely X_i, x_{ij}, Y_i ($i, j=\overline{1,n}$), can be expressed in terms of natural units (thing, ton, litre, barrel, etc.), as well as value units. Depending on this there are natural or value interindustry balance. From an economic point of view, interindustry balance is more important in terms of value. In particular, it allows us to combine the industries ino groups or subgroups (for example, the oil industry and the gas industry combining into the oil-and-gas industry) that facilitates the preparation of product balances.

Thus, if the *i*-th indutry produces just the necessary amount of product, we obtain the balance equation for the *i*-th industry:

$$X_i = x_{i1} + x_{i2} + \cdots + x_{in} + Y_i = \sum_{j=1}^{n} x_{ij} + Y_i,$$

namely the gross volume of the *i*-th industry production is equal to the total volume of the products consumed by *n* industries and the final product. This equation shows the possibility that the *i*-th undustry uses some of its products (i.e. x_{ii}).

Thus, balance principle of communication in various industries is that the gross volume of any *i*-th industry production is equal to the sum of the consumprion volumes in production and non-production spheres, i.e.:

$$\begin{cases} X_1 = x_{11} + x_{12} + \cdots + x_{1n} + Y_1, \\ X_2 = x_{21} + x_{22} + \cdots + x_{2n} + Y_2, \\ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \\ X_i = x_{i1} + x_{i2} + \cdots + x_{in} + Y_i, \\ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \\ X_n = x_{n1} + x_{n2} + \cdots + x_{nn} + Y_n, \end{cases}$$

or

$$X_i = \sum_{j=1}^{n} x_{ij} + Y_i, \quad i = \overline{1, n}. \tag{1}$$

Let us consider value interindustry balance with all the magnituted involved in its relations have the value expression.

Let us introduce the coefficients of direct material costs. If the *j*-th industry plans to produce X_j production units, it is necessary to know how many production units the *i*-th industry will need. It is evident that the answer depends on the technology of the industry given. We assume that the volume of the *i*-th industry production required for manufactiring X_j production units is directly proportional to X_j, namely $x_{ij} = a_{ij} X_j$, whence the coefficient of proportionality

$$a_{ij} = \frac{x_{ij}}{X_j}, \quad i, j = \overline{1, n}, \tag{2}$$

where a_{ij} is the coefficient of direct maerial costs of the *i*-th industry production per unit of the gross volume of the *j*-th industry production. a_{ij} depends on the *j*-th industry technology. This means linear dependence of material costs x_{ij} on the gross volume of the products X_j. Therefore, the model of interindustry balance built on this basis is linear.

These coefficients form a square matrix of coefficients of direct material costs (the technological matrix):

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ . & . & . & . \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}.$$

The matrix A describes a technology in a single intensities of all industries. It is evident that the *j*-th industry works with intensity X_j, $j = \overline{1, n}$. We assume that the matrix A is a constant – technology is considered to be unchanged for some period (for example, for a year).

Taking into account (2), we rewrite the system of equations (1) in the form:

$$X_i = \sum_{j=1}^{n} a_{ij} X_j + Y_i, \quad i = \overline{1, n}$$

or in the matrix form:

$$X = AX + Y, \tag{3}$$

where *A* is the matrix of direct material costs, $X = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix}$ is the single-column matrix of the gross volumes of products, $Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$ is the single-column matrix of the volumes of final products.

The equation (3) is the equation of linear interindustry balance.

The main task of interindustry balance is to find the single-column matrix of the gross volumes of products *X* that provides the given single-column matrix of the volumes of final products *Y* if the matrix of direct material costs *A* is known.

We write the equation (3) in the form:

$$(E - A)X = Y, \qquad (4)$$

where E is the identity matrix.

If the matrix $E - A$ is non-singular, we can write the equation (4) in the form:

$$X = (E - A)^{-1}Y, \qquad (5)$$

where $(E - A)^{-1} = S$ is the matrix of total material costs with each element s_{ij} that is the gross volume of the *i*-th industry production required to provide producing of the *j*-th industry final production unitsi ($i,\ j = \overline{1, n}$).

According to the economic content of the problem, the values of X_i ought to be non-negative if the values of Y_i and a_{ij} ($i,\ j = \overline{1, n}$) are non-negative.

The interindustry balance equation can be also used in the case when the single-column matrix of the gross volumes of the products X and the matrix of direct material costs A are known; we have to find the single-column matrix of the volumes of final products Y.

The same type of analysis can be applied to determine prices. The technological coefficient a_{ij} can be considered as the number of the *i*-th industry production units reqiured to produce the *j*-th industry production units. Let p_j be the price of the *j*-th industry production unit. Then the cost of material inputs required to manufacture the *j*-th industry production unit is equal to $a_{1j}p_1 + \cdots + a_{nj}p_n$. The difference r_j between the price of the *j*-th industry production unit and the cost of material inputs required to manufacture the *j*-th industry production unit is the added value of the *j*-th industry. Thus, $r_j = p_j - \sum_{i=1}^{n} a_{ij}p_i$. The added value can include the cost of labour, depreciation, taxation, profit, etc.

If, for any non-negative single-column matrix Y and for a non-negative matrix A, there exists a non-negative solution *X* of the equation (4), then the non-negative matrix *A* is productive.

Note that the matrix *A* is non-negative if all its elements are non-negative.

If the maximum of the sums of column elements of the matrix *A* is not greater than one, at least for one of columns, the sum of elements is strictly less than one, then the matrix *A* is productive.

The interindustry balance equation is used to plan and forecast production.

The interindustry balance analysis is widely used by all leading economically developed countries.

For example, we consider a conventional production system that consists of three industries producing one type of products of volume X_i ($i = \overline{1, 3}$) units, i.e. the single-column matrix of gross volumes of production is $X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix}$. To ensure production, every industry uses a part of products produced by itself and by related industries. Let us find the single-colum matrix of the gross volumes of production *X* if

$$Y = \begin{pmatrix} 60 \\ 70 \\ 30 \end{pmatrix}, \quad A = \begin{pmatrix} 0{,}05 & 0{,}35 & 0{,}4 \\ 0{,}1 & 0{,}1 & 0{,}4 \\ 0{,}2 & 0{,}1 & 0{,}2 \end{pmatrix},$$

where *Y* is the single-column matrix of the volumes of final production, *A* is the technological matrix.

To use the formula (5), we find the matrix $(E - A)^{-1}$.

$$E - A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} 0{,}05 & 0{,}35 & 0{,}4 \\ 0{,}1 & 0{,}1 & 0{,}4 \\ 0{,}2 & 0{,}1 & 0{,}2 \end{pmatrix} = \begin{pmatrix} 0{,}95 & -0{,}35 & -0{,}4 \\ -0{,}1 & 0{,}9 & -0{,}4 \\ -0{,}2 & -0{,}1 & 0{,}8 \end{pmatrix}.$$

We calculate the determinant of the matrix $E - A$:

$$\det(E-A)=0,95\cdot 0,68+0,1\cdot(-0,28-0,04)-0,2\cdot(0,14+0,36)=$$
$$=0,646-0,032-0,1=0,514.$$

The determinant of the matrix $E-A$ being non-zero, the matrix $E-A$ is non-singular. Therefore, there exists a unique reciprocal of $E-A$.

Let us find the reciprocal of $E-A$:

$$(E-A)^{-1}=\frac{1}{0,514}\begin{pmatrix}\begin{vmatrix}0,9 & -0,4\\ -0,1 & 0,8\end{vmatrix} & -\begin{vmatrix}-0,35 & -0,4\\ -0,1 & 0,8\end{vmatrix} & \begin{vmatrix}-0,35 & -0,4\\ 0,9 & -0,4\end{vmatrix}\\ -\begin{vmatrix}-0,1 & -0,4\\ -0,2 & 0,8\end{vmatrix} & \begin{vmatrix}0,95 & -0,4\\ -0,2 & 0,8\end{vmatrix} & -\begin{vmatrix}0,95 & -0,4\\ -0,1 & -0,4\end{vmatrix}\\ \begin{vmatrix}-0,1 & 0,9\\ -0,2 & -0,1\end{vmatrix} & -\begin{vmatrix}0,95 & -0,35\\ -0,2 & -0,1\end{vmatrix} & \begin{vmatrix}0,95 & -0,35\\ -0,1 & 0,9\end{vmatrix}\end{pmatrix}=$$
$$=\frac{1}{0,514}\begin{pmatrix}0,68 & 0,32 & 0,5\\ 0,16 & 0,68 & 0,42\\ 0,19 & 0,165 & 0,82\end{pmatrix}.$$

Thus the single-column matrix of the gross volumes of production has the form

$$X=\frac{1}{0,514}\begin{pmatrix}0,68 & 0,32 & 0,5\\ 0,16 & 0,68 & 0,42\\ 0,19 & 0,165 & 0,82\end{pmatrix}\begin{pmatrix}60\\ 70\\ 30\end{pmatrix}=$$
$$=\frac{1}{0,514}\begin{pmatrix}40,8+22,4+15\\ 9,6+47,6+12,6\\ 11,4+11,55+24,6\end{pmatrix}\approx\begin{pmatrix}152,1\\ 135,8\\ 92,5\end{pmatrix}.$$

A linear mathematical model of exchange (a mathematical model of international trade). Let us consider a linear mathematical model of exchange which is interpreted as a model of international trade that allows us to determine trade income of countries (or their ratio) for balanced trade [3]. Let $K_1, K_2, \ldots, K_n$ be the group of *n* countries that are trading. We denote by Z_j the trading income (the national income) of the *j*-th country generated from sale of own goods in both domestic and foreign markets. The structure of trade relations between the countries is considered to be known: the share q_{ij} of the trading income Z_j that the *j*-th country spends on the purchase of goods (imports) from the *i*-th country is a constant; in particular, q_{ij} does not depend on the value of Z_j. This hypothesis is an assumption about linearity of the model.

Let us consider the structural matrix of trade (the matrix of exchange):

$$Q=\begin{pmatrix}q_{11} & q_{12} & \cdots & q_{1n}\\ q_{21} & q_{22} & \cdots & q_{2n}\\ . & . & . & .\\ q_{n1} & q_{n2} & \cdots & q_{nn}\end{pmatrix}.$$

Starting trading according to the matrix of exchange *Q*, countries will have the trading income with magnitude described by the single-column matrix *QZ* after one round.

We consider that all the trading income is spent either on the purchase of goods on its territory or on imports from other countries, namely the sum of elements of any column of the matrix *Q* is equal to unity:

$$\sum_{i=1}^{n}q_{ij}=1,\quad j=\overline{1,n}.$$

For the country K_i, the income from domestic and foreign trade is

$$Z_i=\sum_{j=1}^{n}q_{ij}Z_j,\quad i=\overline{1,n}.$$

For the balanced trade, it is necessary to find such a matrix of the trading income

$$Z = \begin{pmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{pmatrix},$$

for which the matrix equation:

$$Z = QZ, \qquad (6)$$

is fulfilled. This equation can be solved for *Z*.

It follows from mathematical analysis of the model that if the system operates *k* rounds of the trade with the matrix of exchange Q , after each round, we have the single-column matrices of the trading income: $QZ, Q^2Z, \ldots, Q^kZ$. Indeed, substituting QZ for the one-round trade for Z in the right-hand side of the formula (6), we obtain $Q(QZ) = (QQ)Z = Q^2Z$ for the two-round trade; substituting Q^2Z for the two-round trade for Z in the right-hand side of the formula (6), we obtain $Q(Q^2Z) = Q^3Z$ for the three-round trade, etc. For example, for the three-round trade, we substitute the matrix of the trading income for the two-round trade in the formula (6).

For example, we take three countries (denoting them by K_1, K_2, K_3), participants in the trade with the trading income respectively Z_1, Z_2, Z_3. Let the country K_1 spends a half of the trading income on the purchase of goods on its territory, a quarter of the trading income on the purchase of goods from the country K_2 and a quarter on the purchase of goods from the country K_3. The country K_2 equally spends the trading income on the purchase of goods from the country K_1 on its territory and from the country K_3. The country K_3 spends a half of the trading income on the purchase of goods from the country K_1, it spends the other half of the trading income on the purchase of goods from the country K_2 and does not buy anything on its territory. Let us find the national income of the countries that would satisfy the balanced trade without a deficite if the amount of their national income is 9000 conditional monetary units a year.

Let us write the structural matrix of trade:

$$Q = \begin{array}{c} \\ K_1 \\ K_2 \\ K_3 \end{array} \begin{array}{c} \begin{array}{ccc} K_1 & K_2 & K_3 \end{array} \\ \begin{pmatrix} \frac{1}{2} & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{3} & 0 \end{pmatrix} \end{array}.$$

Let q_{ij} be the share of the trading income that the *j*-th country spends on the purchase of the *i*-th country's goods. Note that the sum of elements in every column of the matrix Q is equal to unit.

After summarizing the results of trading for the year, the *i*-th country gets the income (the one-round trade doing for the year):

$$Z_i = \sum_{j=1}^{3} q_{ij} Z_j, \quad i = \overline{1,3}.$$

Let us write the matrix equation to find the matrix *Z*:

$$Z = QZ \quad \text{або} \quad (Q - E)Z = O,$$

i.e.

$$\begin{pmatrix} -\frac{1}{2} & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{4} & -\frac{2}{3} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{3} & -1 \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

The solution of the system of equations:

$$Z_1 = 2Z_3, \quad Z_2 = \frac{3}{2}Z_3, \quad Z_3 \in \mathrm{R}.$$

The result obtained means that the given countries' trade balance is achieved at the ratio of their national incomes $2:\frac{3}{2}:1$.

Let us find the countries' national incomes for the year that would satisfy the balanced trade without a deficite under the condition that the sum of incomes is equal to $Z_1 + Z_2 + Z_3 = 9000$ conditional monetary units. Substitute the values $Z_1 = 2C,\ Z_2 = \frac{3}{2}C,\ Z_3 = C$ in this equality, where $C = \text{const}$. Then we obtain

$$2C + \frac{3}{2}C + C = 9000,$$

whence $C = 2000$. Thus, $Z_1 = 4000,\ Z_2 = 3000,\ Z_3 = 2000$ conditional monetary units.

Note that here are simplified versions of a mathematical model of interindustry balance and foreign trade.

Conclusions. A linear mathematical model of a diversified economy and a linear mathematical model of exchange are explored in this paper. Mathematical analysis of these models is carried out.

1. Baimukhamedov M. F. Optimal control model diversified economy taking into account delay investment / M.F. Baimukhamedov, G.S. Baimukhamedova // Russian journal of agricultural research. – 2014. – Vol. 8.
2. Schneider B.R. A comparative political economy of diversified business groups / B.R. Schneider // Review of international political economy. – 2016.
3. Shmurev V.I. Generalized linear exchange model / V.I. Shmurev // Journal of applied and industrial mathematics. – 2008. – Vol. 2.

УДК 539.3
Делявський В.М., Грінченко Л.Г.
Луцький національний технічний університет

НАПРУЖЕНО-ДЕФОРМОВАНИЙ СТАН ПРЯМОКУТНОЇ ІЗОТРОПНОЇ ПЛИТИ СЕРЕДНЬОЇ ТОВЩИНИ ПІД ЗОСЕРЕДЖЕНИМ НАВАНТАЖЕННЯМ

Делявський В.М., Грінченко Л.Г. Напружено-деформований стан прямокутної ізотропної плити середньої товщини під зосередженим навантаженням. Плити є основним конструкційним елементом проїжджої частини мостів. Для довільного навантаження напружено-деформований стан плити при конкретних граничних умовах можна визначити досить просто, якщо для такої плити відомо фундаментальний розв'язок, який описує залежність прогину плити від положення зосередженої сили, прикладеної в її длвільній точці.
Ключові слова: ізотропна плита, навантаження.

Делявский В.Н., Гринченко Л.Г. Напряженно-деформированное состояние прямоугольной изотропной плиты средней толщины под сосредоточенной нагрузкой. Плиты являются основным конструкционным элементом проезжей части мостов. Для произвольной нагрузки напряженно деформированного состояния плиты при конкретных предельных условиях можно определить достаточно просто, если для такой плиты известно фундаментальное решение, которое описывает зависимость прогиба плиты от положения сосредоточенной силы, прилагаемой в ее длвильний точке.
Ключевые слова: изотропная плита,нагрузка.

Delyavskiy V., Grinchenko L. Tensely deformed the state of rectangular isotropic flag of middle thickness under a point load. Flags are the basic construction element of trafficway of bridges. For the arbitrary loading tensely deformed the state of flag at concrete maximum terms it is possible to define sufficiently simple, if for such flag a fundamental decision, which describes dependence of bending of flag on position of the concentrated force, enclosed in its to the dlvil'niy point, is known.
Keywords: izotropic flag, loading.

В даній статті, в рамках моделі плити середньої товщини, запропонованої І.О.Прусовим [1] методом рядів побудовано фундаментальний розв'язок для прямокутної ізотропної плити середньої товщини вільноопертої на всіх краях.

Математична модель плити.

Згідно моделі І.О.Прусова [1] складові вектора переміщень $\{u_1, u_2\}$ визначаються через три невідомі функції w, F, Φ

$$u_1 = -\left[x_3 \frac{\partial w}{\partial x_1} + \lambda_0(x_3) \frac{\partial F}{\partial x_1} \right] + \lambda_1(x_3) \frac{\partial \Phi}{\partial x_2},$$
$$u_2 = -\left[x_3 \frac{\partial w}{\partial x_2} + \lambda_0(x_3) \frac{\partial F}{\partial x_2} \right] - \lambda_1(x_3) \frac{\partial \Phi}{\partial x_1} \qquad (1)$$

де F і Φ довільні функції змінних x_1, x_2; а λ_0, λ_1 є непарні функції змінної x_3, що задовольняють умовам $\lambda_j^{(0)}(0) = 0$; $\lambda_j'(h) = \lambda_j'(-h) = 0$ $(j = 0,1)$ і забезпечують умови рівності нулю дотичних напружень $\sigma_{\alpha 3}$, $\alpha = 1,2$ на поверхнях плити. Приймається, що прогин плити w не змінюється по товщині

$$w(x_1, x_2, x_3) = w(x_1, x_2) . \qquad (2)$$

Функція $\Phi(x_1, x_2)$ є розв'язком рівняння Гельмгольца а $N(x_1, x_2)$ є оператор Гельмгольц

$$\left(\frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} - K^2 \right) \Phi = 0 \qquad (3)$$

де

$$K^2 = \frac{4G\lambda_1(h)}{D(1-\nu)k_1}, \tag{4}$$

Тут G - зсуву, D – циліндрична жорсткість плити, ν - коефіцієнт Пуассона; k_1 є моментом функції $\lambda_1(x_3)$, а $\lambda_1(h)$ її значення на поверхнях плити.

Функція F зв'язана з узагальненим прогином $\overline{w}$ залежністю

$$F = \varepsilon^2 \nabla^2 \overline{w}, \tag{5}$$

де

$$\varepsilon^2 = \frac{D}{2G\lambda_0(h)}. \tag{6}$$

Функція $\overline{w}$ є розв'язком неоднорідного бігармонійного рівняння

$$L\overline{w} = q/D, \tag{7}$$

де $L(x_1, x_2)$ бігармонійний оператор

$$L(x_1, x_2) = \frac{\partial^4}{\partial x_1^4} + 2\frac{\partial^4}{\partial x_1^2 \partial x_2^2} + \frac{\partial^4}{\partial x_2^4}; \tag{8}$$

$q(x_1, x_2)$ поперечне навантаження, прикладене до верхньої поверхні плити;

Розв'язок системи диференціальних рівнянь рівноваги

Систему диференціальних рівнянь (3),(7) розв'язуємо методом розділення змінних. Згідно праці [2] виберемо функцію Гельмгольца $\Phi(x_1, x_2)$ як непарну функцію змінних x_1, x_2.

$$\Phi(x_1, x_2) = \sum_{k=1}^{\infty} \left\{ \phi_k^{[1]}(x_1) \sin \delta_k^{*[2]} x_2 + \phi_k^{[2]}(x_2) \sin \delta_k^{*[1]} x_1 \right\} \tag{9}$$

Тут

$$\delta_k^{*[j]} = \frac{(2k-1)\pi}{2a_j}, \quad j = \overline{1,2} \tag{10}$$

де $a_j\,(j=1,2)$ - розміри плити в плані; $\phi_k^{[j]}(x_j)$ невідомі функції змінних x_j. Розв'язок неоднорідного диференціального рівняння (7) вибираємо у вигляді суми двох парних функцій:

$$\overline{w} = w_o + w_*, \tag{11}$$

загального розв'язку w_0 однорідного рівняння

$$L\overline{w} = 0 \tag{12}$$

і часткового розв'язку w_* неоднорідного рівняння (7)

Загальний розв'язок однорідного рівняння шукаємо у вигляді

$$w_* = \sum_{k=1}^{\infty} \left\{ f_k^{[1]}(x_1) \cos \delta_k^{*[2]} x_2 + f_k^{[2]}(x_2) \cos \delta_k^{*[1]} x_1 \right\}. \tag{13}$$

Тут $f_k^{[j]}(x_j)$ і $\phi_k^{[j]}(x_j)$ невідомі функції змінних x_j, які задовольняють диференціальним рівнянням

$$f_k^{[j](IV)}(x_j) - 2\delta_k^{*[3-j]^2} f_k^{[j]''}(x_j) + \delta_k^{*[3-j]^4} f_k^{[j]}(x_j) = 0, \tag{14}$$

$$\phi_k^{[j]''}(x_j) - \left(K^2 + \delta_k^{*[3-j]^2}\right)\phi_k^{[j]}(x_j) = 0, \tag{15}$$

Розв'язок тих рівнянь має вигляд

$$f_k^{[j]}(x_j) = R_{1(k)}^{[j]*} ch\delta_k^{*[3-j]} x_j + R_{2(k)}^{[j]*} x_j sh\delta_k^{*[3-j]} x_j,$$
$$\phi_k^{[j]}(x_j) = R_{3(k)}^{[j]*} sh\gamma^{*[3-j]} x_j, \tag{16}$$

де $\gamma_k^{*[j]} = \sqrt{K^2 + \delta_k^{*[3-j]^2}}$.

Частковий розв'язок неоднорідного рівняння (7) визначаємо методом рядів Фур'є. Для цього навантаження $q(x_1, x_2)$ розкладаємо в ряд Фур'є (по синусах)

$$q(x_1, x_2) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} q_{mn} \sin \delta_m^{[1]} x_1 \sin \delta_n^{[2]} x_2 \,, \qquad (17)$$

де

$$q_{mn} = \frac{4}{a_1 a_2} \int_0^{a_1}\int_0^{a_2} q(x_1, x_2) \sin \delta_m^{[1]} x_1 \sin \delta_n^{[2]} x_2 dx_1 dx_2 \qquad (18)$$

коефіцієнти ряду Фур'є.

Щоб задовольнити бігармонічне рівняння, частковий розв'язок записуємо у вигляді

$$w_* = \frac{1}{D}\sum_{m=1}^{\infty}\sum_{n=1}^{\infty} \frac{q_{mn}}{\left(\delta_m^{[1]^2} + \delta_m^{[2]^2}\right)^2} \sin \delta_m^{[1]} x_1 \sin \delta_n^{[2]} x_2 \,. \qquad (19)$$

Обчислити коефіцієнти q_{mn} для зосередженої сили. Для цього зосереджену силу P замінюємо навантаженням $q(x_1, x_2)$ рівномірно розподіленим на нескінчено малому квадратному елементі з розмірами $\delta x \delta$

$$q(x_1, x_2) = \frac{P}{\delta^2}. \qquad (20)$$

Навантаження $q(x_1, x_2)$ рівне нулю всюди крім елемента обмеженого прямими: $\{x_1 = \xi_1, x_1 = \xi_1 + \delta, x_2 = \xi_2,\ x_2 = \xi_2 + \delta\}$, показаними на рисунку 1.

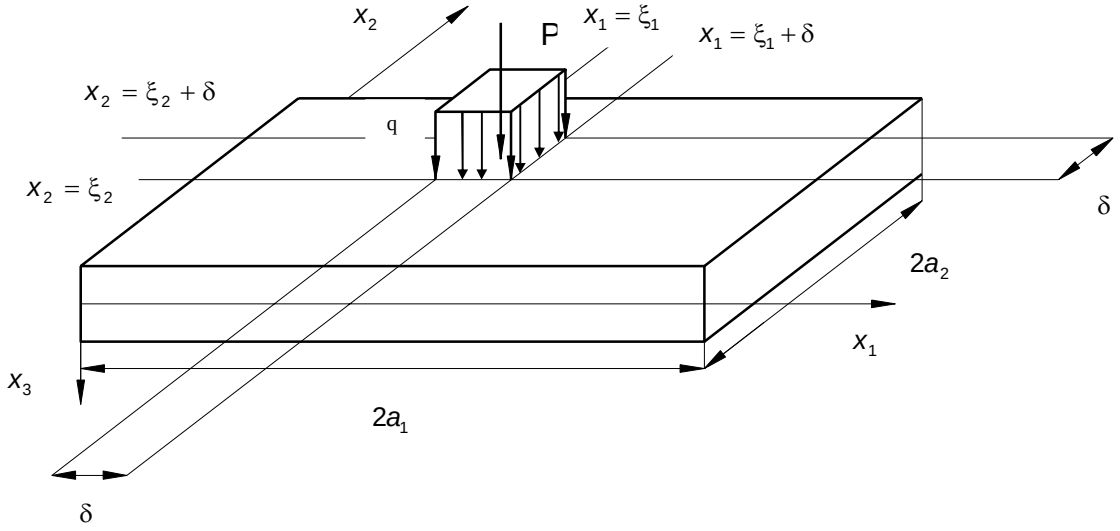


Рис. 1. Схема навантаження плити зосередженою силою.

Коефіцієнти q_{mn} виражаються так:

$$q_{mn} = \frac{4}{a_1 a_2} \int_0^{a_1}\int_0^{a_2} \frac{P}{\delta^2} \sin \delta_m^{[1]} x_1 \sin \delta_n^{[2]} x_2 dx_1 dx_2 =$$

$$= \frac{4P}{\delta^2 a_1 a_2} \int_{\xi_1}^{\xi_1+\delta}\int_{\xi_2}^{\xi_2+\delta} \frac{P}{\delta^2} \sin \delta_m^{[1]} x_1 \sin \delta_n^{[2]} x_2 dx_1 dx_2 = \qquad (21)$$

$$\frac{4P}{\delta^2 a_1 a_2}\left[\cos \delta_m^{[1]}(\xi_1 + \delta) - \cos \delta_m^{[1]} \xi_1\right]\left[\cos \delta_n^{[2]}(\xi_2 + \delta) - \cos \delta_n^{[2]} \xi_2\right].$$

враховуючи, що

$$\cos \alpha - \cos \beta = -2 \sin \frac{\alpha + \beta}{2} \sin \frac{\alpha - \beta}{2},$$

отримуємо

$$\cos\left[\delta_m^{[1]}(\xi_1+\delta)\right]-\cos\left[\left(\delta_m^{[1]}\xi_1\right)\right]=-2\sin\left[\delta_m^{[1]}\left(\xi_1+\frac{\delta}{2}\right)\right]\sin\left[\delta_m^{[1]}\frac{\delta}{2}\right],$$

$$\cos\left[\delta_n^{[2]}(\xi_2+\delta)\right]-\cos\left[\left(\delta_n^{[2]}\xi_2\right)\right]=-2\sin\left[\delta_n^{[2]}\left(\xi_2+\frac{\delta}{2}\right)\right]\sin\left[\delta_n^{[2]}\frac{\delta}{2}\right],$$

$$q_{mn}=\frac{4P}{a_1a_2\delta_m^{[1]}\delta_n^{[2]}\varepsilon^2}4\sin\left[\delta_m^{[1]}\left(\xi_1+\frac{\delta}{2}\right)\right]\sin\left[\delta_m^{[1]}\frac{\delta}{2}\right]\sin\left[\delta_n^{[2]}\left(\xi_2+\frac{\delta}{2}\right)\right]\sin\left[\delta_n^{[2]}\frac{\delta}{2}\right].$$

(22)

При $\varepsilon\to 0$ отримуємо:

$$\sin\left(\delta_m^{[1]}\frac{\delta}{2}\right)\approx\delta_m^{[1]}\frac{\delta}{2};\ \sin\left(\delta_n^{[2]}\frac{\delta}{2}\right)\approx\delta_n^{[2]}\frac{\delta}{2}.$$

Остаточно коефіцієнти q_{mn} набувають вигляду:

$$q_{mn}=\frac{4P}{a_1a_2}\sin\left(\delta_m^{[1]}\xi_1\right)\sin\left(\delta_n^{[2]}\xi_2\right). \qquad (23)$$

Дане співвідношення отримано в системі координат з початком в лівому нижньому куті плити. Частковий розв'язок w_* записаний в системі координат з початком в центрі плити описується виразом

$$w_*=\frac{4P}{a_1a_2}\frac{1}{D}\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\frac{\sin\left(\delta_m^{[1]}\xi_1\right)\sin\left(\delta_n^{[2]}\xi_2\right)}{\left(\delta_m^{[1]^2}+\delta_m^{[2]^2}\right)^2}\sin\delta_m^{[1]}(x_1+a_1)\sin\delta_n^{[2]}(x_2+a_2). \qquad (24)$$

Для ілюстрації запропонованого методу виконано розрахунок плити з такими даними:
товщина плити є стала і рівна $2h$=0,7 м. Інші дані:

$E=2{,}07\cdot10^5$ MPa, ν=0,3, $q_{11}=0{,}4$ MPa.

Прийнято, що зосереджена сила діє P діє в середині плити і виносить P=400 kN.

На рисунках (2) – (3) показано зміну прогину w і згинного моменту в перерізі $x_2=0$ для плити сталої ширини $2a_1=3$ m і різних довжин

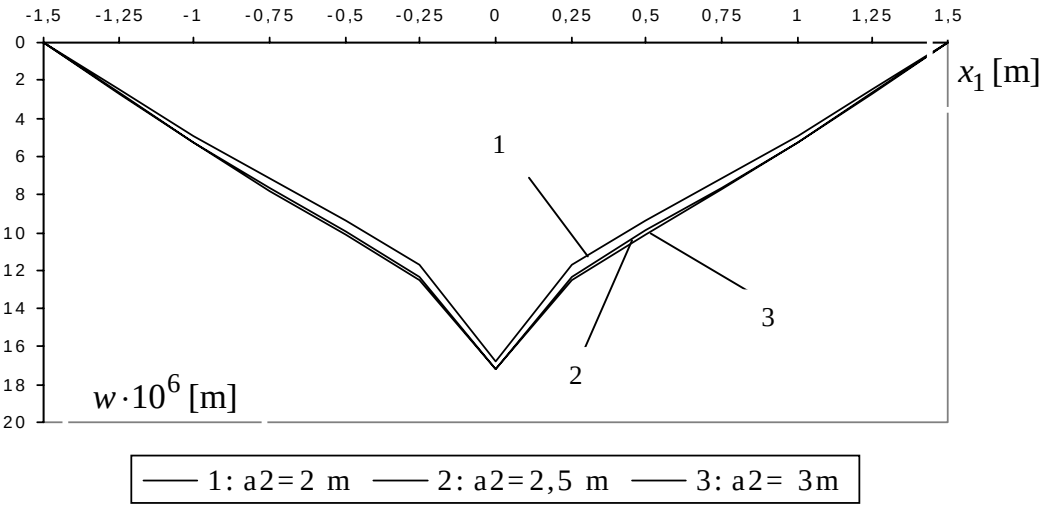


Рис. 2. Прогин w в перетині $x_2=0$ при постійній ширині $2a_1=3$ m і різній довжині $2a_2$.

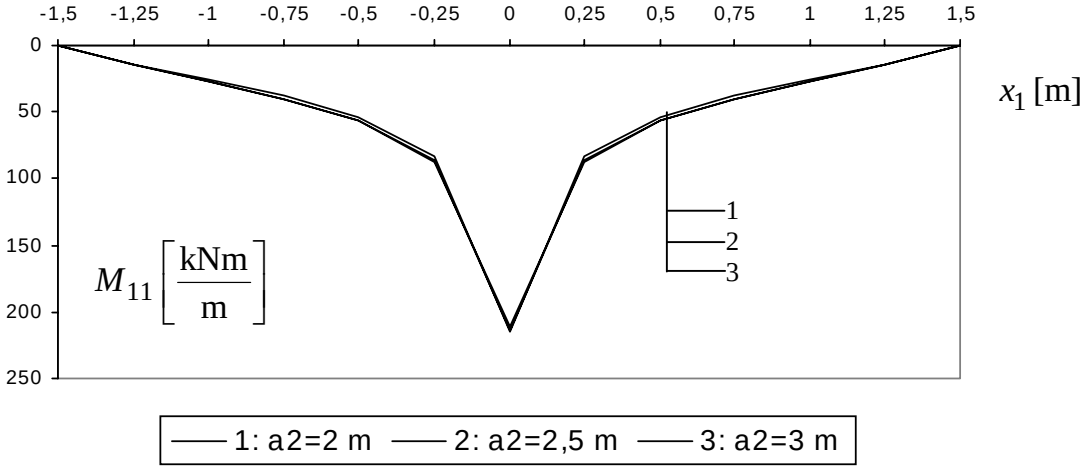


Рис. 3. Графіки згинного моменту M_{11} в перерізі $x_2 = 0$ для сталої ширини $2a_1 = 3\,\text{m}$ і різних довжин $2a_2$.

Як видно з рисунків (2) і (3) прогин *w* і згинний момент M_{11} рівні нулю на краях плити, в силу задоволення граничних умов. В центрі плити ті величини досягають максимальних значень і зростають з ростом довжини плити. В точці прикладання зосередженої сили всі статичні і геометричні величини отримують нескінчено великі значення. То зв'язано з тим, що чим більше вибираємо розкладів *m* і *n* ряду Фур'є тим отримуємо точніший закон розкладеного навантаження *q*. В нашому випадку навантаження *q* є зосереджена сила розкладена на нескінчено малому елементі, що очевидно відповідає нескінчено великим числовим значенням. В дійсності не існує ідеально зосереджених сил або навантаження розподіленого на нескінчено малому елементі. Тому результати в точці прикладання зосередженої сили не можна трактувати як реальні і їх слід відкинути.

На рисунках (4) – (5) подано зміну переміщення u_1 по товщині плити в середині сторони a_2 для різних відношень $\frac{2h}{2a_1}$. На рисунку (4) бачимо зміну переміщення u_1 по товщині плити для відношення $2h/2a_1$ =0,166, що відповідає тонкій плиті, в той час як рисунок (5) представляє плиту середньої товщини.

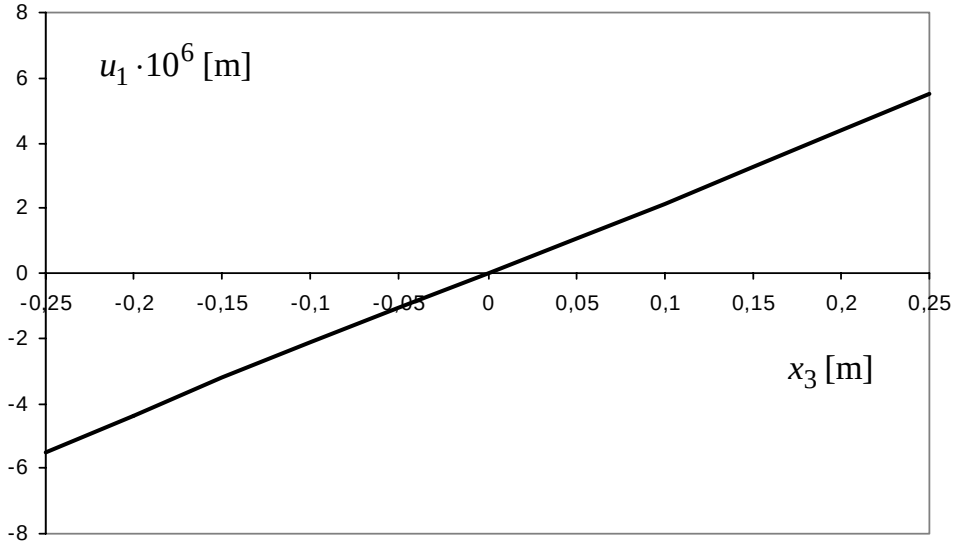


Рис. 4. Зміна переміщення u_1 по товщині для відношення $2h/2a_1$ =0,166.

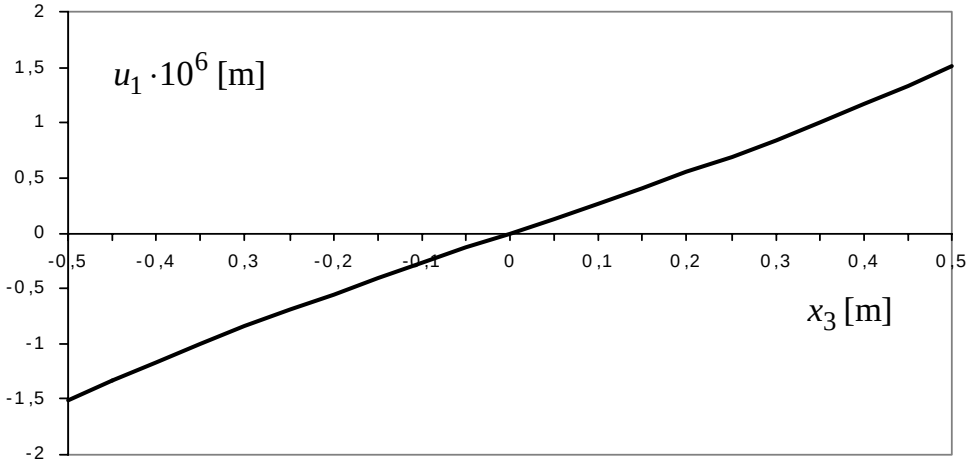


Рис. 5. Зміна переміщення u_1 по товщині для відношення $2h/2a_1$ =0,333.

Summary. The solution of problem about of the state of stresses and strains in rectangular isotropic moderate thickness plate acted by the concentrated force aplied in its arbitrariry point has been obtained. The problem was solved in the frames of the plate theory sugested by I. Prusov with help of separating variables method using the Fourie's series method. As a example the rectangular plate free supported at all edges was considered. The boundary conditions was satisfied by means of collocation method.

1. Прусов И.А. Метод сопряжения в теории плит.-Минск: Изд-во Белорус. ун-та, 1975.- 256 с.
2. Delyavskyy M., Gołaś J., Podhorecka A.: O pewnym podejściu do rozwiązywania płyt wielowarstwowych, XLV Konf. Nauk. Krynica 99, s.63-70.
3. Делявский М.В. Расчет напряженного состояния в толстой ортотропной плите под действием изгибающей нагрузки // Пробл. прочности .- № 11-12.- С. 117-123.
4. Kączkowski Z. Płyty. Obliczenia statyczne.- Warszawa: Arkady, 1980.-
5. Nowacki W.: Dzwigary powierzchniowe, Warszawa, PWN, 1980.-
6. Подгорный А.Н. Марченко Г.А., Пустынников В.И. Основы и методы прикладной теории упругостиё- Киев: Вища школа, 1981.-328с.

УДК 681.515.8
Здолбіцька Н.В., Здолбіцький А.П., Сопіжук Р.В., Супронюк В.В.
Луцький національний технічний університет

DC-AC ПЕРЕТВОРЮВАЧ З МІКРОКОНТРОЛЕРНИМ КЕРУВАННЯМ ЧАСТОТИ ІНВЕРТОРА

Н.В. Здолбіцька, А.П. Здолбіцький, Р.В. Сопіжук, В.В. Супронюк. DC-AC перетворювач з мікроконтролерним керуванням частоти інвертора. Розглянуто перетворювачі електричної енергії, їх схемотехнічні рішення та основні відмінності апаратної та програмної частин. Спроектовано підвищувальний імпульсний перетворювач постійної напруги із використанням недорогих сучасних електронних компонентів, наведено структурну та принципову схеми інвертора, дано рекомендації по використанню, вказано переваги та недоліки проектованого рішення.
Ключові слова: інвертор, мікроконтролер, ATmega328P, трансформатор, транзистори, потенціометр, частота 50-200Гц.
Рис. 4. Літ. 10.

Н.В. Здолбицкая, А.П. Здолбицкий, Р.В. Сопижук, В.В. Супрунюк. DC-AC преобразователь с микроконтроллерным управлением частоты инвертора. Рассмотрены преобразователи электрической энергии, их схемотехнические решения и основные различия аппаратной и программной частей. Спроектирован повышающий импульсный преобразователь постоянного напряжения с использованием недорогих современных электронных компонентов, приведена структурная и принципиальную схемы инвертора, даны рекомендации по использованию, указано преимущества и недостатки проектируемого решения.
Ключевые слова: инвертор, микроконтроллер, ATmega328, трансформатор, транзисторы, потенциометр, частота 50-200Гц.

N.V. Zdolbitska, A.P. Zdolbitskyy, R.V. Sopizhuk, V.V. Suprunyuk. DC-AC converter with microcontroller -controlled frequency inverter. Considered electric power converters, circuit solutions and their main differences between the hardware and software parts. Designed buoyant DC pulse converter using inexpensive modern electronic components are structural and schematic diagram of the inverter, Recommendations on use, given the advantages and disadvantages of the designed solution.
Keywords: invertor, microcontroller, ATmega328, transformer, transistors, potentiometer, frequency 50-200Hts.

Перетворювачем електричної енергії є пристрій, який пов'язує дві (або більше) електричні системи з відмінними один від одного параметрами і дозволяє по заданому закону змінювати ці параметри, забезпечуючи обмін електричною енергією між зв'язуваними системами [1]. Напівпровідникові перетворювачі напруги (електронні трансформатори), що зв'язують системи змінного і постійного струму, а саме перетворювачі постійної напруги в змінну, називаються інверторами.

Історично першими виникли перетворювачі енергії на базі електромеханічних пристроїв. Електричний двигун перетворював електричну енергію в механічну, яка за допомогою генератора перетворювалась в електричну з потрібними параметрами [2]. Сучасні пристрої перетворення енергії не містять рухомих елементів, тому їх називають «статичними перетворювачами», зокрема інвертори DC-AC перетворюють, наприклад, постійну напругу 12 В у змінну напругу 220 В. Інвертор значно дешевший за міні-електростанцію, є мініатюрним і легким [3]. Спільно з одним, або декількома акумуляторами він може працювати як автономне джерело безперебійного живлення для будинку, котельної, пожежних і охоронних систем. Якщо є мережева напруга 220 Вольт, він просто пропускає його "крізь" себе і, при необхідності, заряджає акумулятори. Якщо напруга в мережі зникла, інвертор миттєво починає генерувати змінну напругу 220 Вольт від акумуляторів [4]. Час автономної роботи залежить від потужності навантаження і ємності акумуляторів. Так, наприклад, чотирьох акумуляторів по 190 А/г вистачить приблизно на 16 годин автономної роботи при постійному навантаженні 0,5 КВт [5]. При появі мережевої напруги прилад автоматично перемкнеться в початковий стан очікування і зарядить акумулятори.

Аналоги та схемотехнічні рішення. Поширеними схемотехнічними рішеннями для інверторів DC-AC стали класичні схеми мостових інверторів та їх модифікації. Наприклад, функціональну схему інвертора серії DA, що перетворює нестабільну напругу постійного струму в стабілізовану (в тому числі і по частоті) однофазну напругу змінного струму, наведено в [6]. На вході силового двохступеневого фільтра каскад формує багатоімпульсні послідовності. Силовий трансформатор, що входить до складу перетворювача постійної напруги, живиться від мостового каскаду транзисторів MOSFET. При такому варіанті побудови інвертора, незважаючи на додаткове перетворення енергії в імпульсному перетворювачі напруги (440 В) при раціональному проектуванні перетворювача постійної напруги (в

даному випадку робоча частота дорівнює 80 кГц), досягаються високі енергетичні і масо-габаритні характеристики.Двотактний мостовий каскад виконаний на польових транзисторах з ізольованим затвором MOSFET (в 200-ватному перетворювачі напруги застосовуються польові транзистори фірми International Rectifier IRFP23N50L: TR2, TR3, TR4, TR5). Мостовий каскад і силовий фільтр охоплені негативним зворотним зв'язком, який забезпечує високі енергетичні характеристики інвертора. Для забезпечення необхідної форми синусоїдальної вихідної напруги використовується дволанковий LC-фільтр змінного струму (дросель виконаний на кільцевому осерді з МО-пермалою).

Однією із сфер застосувань інверторів є альтернативна енергетика. Наприклад, американська фірма Texas Instruments пропонує рішення C2000™ Solar DC/AC Single Phase Inverter на базі своїх нових мікроконтролерів C2000™ F28M35H52C. У пристрої конструктивно реалізовано одна фаза змінної напруги. В основі інверторного каскаду перетворення лежить повний міст DC/AC. Цифровий контроль дозволяє реалізувати інтерфейси віддаленого моніторингу та контролю пристрою. Сумарний коефіцієнт нелінійних спотворень (THD) менший ніж на 5 відсотків, а повний ККД навантаження більший ніж 96%. Єдиним мінусом є його ціна, яка досягає 450$. [7].

До світових лідерів в галузі силової електроніки входить також американська компанія Xantrex [8]. Компанія була заснована більше 20-ти років тому і в результаті злиття виробників інверторів перетворилася на великий промисловий холдинг. Продукцію компанії відрізняє висока відмовостійкість, в конструктиві вже закладено грозовий захист, захист від екстрених перевантажень, захист від некоректної інсталяції. Останні 10-років компанія активно розвиває напрямок керуючих систем для вітрогенераторів, сонячних батарей і автоматизації систем автономного живлення. Ціна на продукцію Xantrex дуже висока, але в деяких випадках (універсальні контролери для сонячних батарей) конкурентні альтернативи відсутні.

Одним із лідерів азіатського ринку є компанія CyberPower Systems [9]. Це тайванський виробник ББЖ та інверторів. Його продукція відрізняється відповідністю заявленим характеристикам, чистим синусом на виході інверторів, високою довговічністю.

Щодо елементної бази, то безперечним лідером виробником мікросхем обв'язки залишається компанія MAXIM™. Широко застосовуються її перетворювачі DC-AC на базі ШІМ-регулювання (наприклад серії MAX 1739, 1839) для регулювання яскравості флюоресцентних ламп [10].

Мета та завдання розробки. Метою роботи є розробка недорогого пристрою інвертора DC-AC на сучасній елементній базі. Основні технічні вимоги до проектованої системи:

- доступність елементної бази;
- напруга на вході 12 В (акумулятори, або напруга бортової мережі автомобіля);
- змінна напруга на виході інвертора 220 В;
- максимальне навантаження на інвертор 250 Ват;
- можливість контролю параметрів (напруга, частота, контроль режимів заряду та розряду акумуляторів).

Структурна схема та принцип дії проектованої системи. Структурна схема пристрою зображена на рисунку 1.

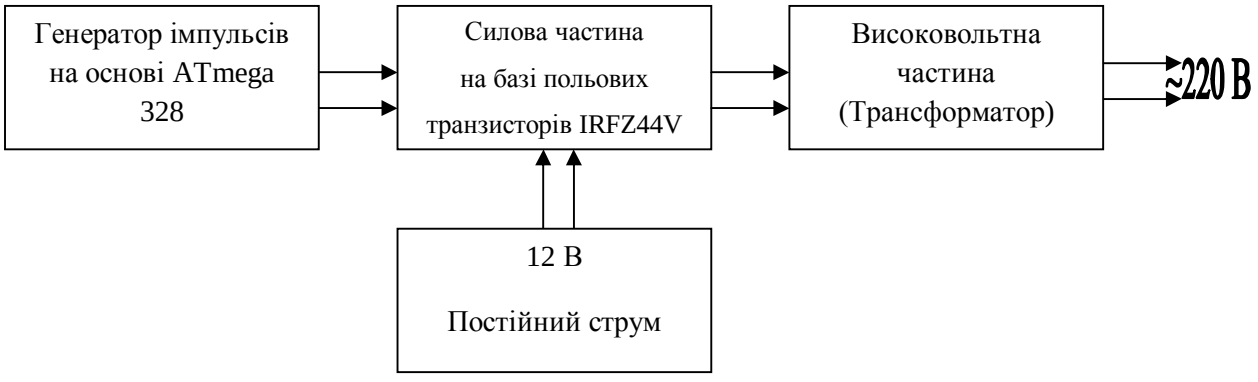


Рис. 1. Структурна схема пристрою

Джерело енергії постійного струму, в найпоширенішому випадку акумулятор 12В, підключається до трансформатора через трипозиційний комутатор. Комутатор є набором електронних ключів, що забезпечує 3 стани: до первинної обмотки трансформатора підключено джерело живлення позитивної полярності, до первинної обмотки трансформатора підключене джерело живлення

негативної полярності та стан коли первинна обмотка короткозамкнена. Послідовно перемикаючи ці стани, на первинній обмотці формується змінна напруга частотою 50Гц і амплітудою 12В. На вторинній обмотці трансформатора при цьому формується напруга з тією ж частотою і формою, проте ефективна напруга складає 220В. Ідеалізовані графіки напруги на трансформаторі показано на рисунку 2. Вихідна напруга знімається з вторинної обмотки, тому має аналогічні параметри.

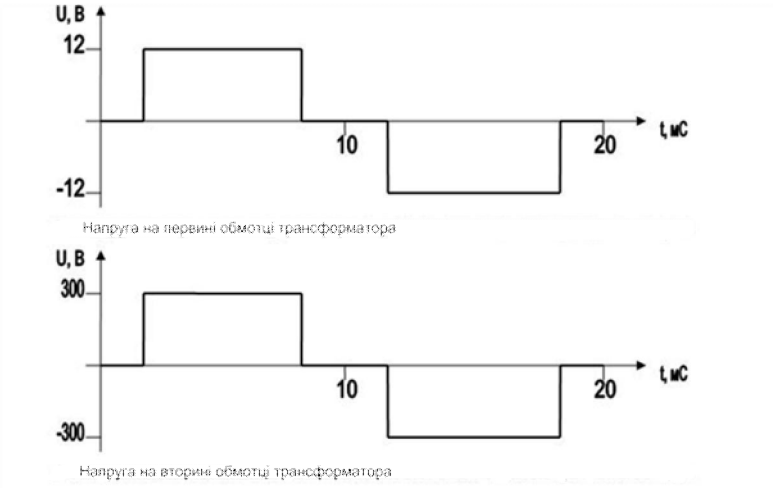


Рис. 2. Графіки напруги типу "модифікована синусоїда" на трансформаторі

Дана форма напруги називається "модифікована синусоїда" і широко застосовується в інверторах для мережі 50Гц. Взагалі параметри, що задають форму модифікованої синусоїди, це амплітуда вихідної напруги і коефіцієнт заповнення, що показує відношення тривалості імпульсу до періоду сигналу. Ці параметри задаються при конструюванні інверторів. З тих міркувань, що інвертор повинен замінювати мережу 220В/50Гц, зазвичай вибирається амплітудне значення напруги модифікованої синусоїди таке ж, як і в мережі, тобто 311В. При цьому, аби забезпечити ефективну напругу 220В, таку ж як і в мережі, коефіцієнт заповнення виходить 0.5. Проте в інверторі цього типу амплітуда вихідної напруги виходить залежною прямопропорційно від напруги джерела. Якщо як джерело енергії використовується акумулятор, а це найпоширеніший випадок, то його напруга при розряджанні знижується, і амплітуда модифікованої синусоїди на виході перетворювача також знижується, відповідно знижується і ефективне значення напруги на виході перетворювача. Для того, щоб поліпшити якість енергії на виході перетворювача в цих умовах часто застосовують схеми керування, які змінюють коефіцієнт заповнення вихідної напруги так, щоб підтримувати ефективну напругу незмінною.

На рисунку 1 зображено блок-схему роботи пристрою, алгоритм роботи якого полягає в почерговому вмиканні і вимиканні 17 і 18 пінів мікроконтролера ATmega 328P. Відповідні сигнали надходять на затвори польових транзисторів, які в свою чергу керують напругою і струмом на трансформаторі, що піднімає напругу з 12 В до 220 В.

Наприклад, інвертор, розрахований на напругу джерела 12В, працює від розрядженого акумулятора з напругою 10В. При цьому амплітуда напруги на виході знижується пропорційно до 259В. Схема управління змінює коефіцієнт заповнення вихідної напруги до 0.72, при цьому ефективна напруга залишається рівною 220В. Проте форма напруги та її амплітуда змінюється, а це може бути недопустимо для деяких навантажень, що буде показано далі.

Програмне забезпечення. Фрагмент программного коду, який відповідає за роботу згідно наведенного вище алгоритму:

```
digitalWrite(N,HIGH);
delayMicroseconds(k);
digitalWrite(N,LOW);
delayMicroseconds(k);
digitalWrite(M,HIGH);
delayMicroseconds(k);
```

```
digitalWrite(M,LOW);
//----
#define N 11
#define M 12
int h=5000;
int k=5000;
//----
```

Даний елемент коду забезпечує генерування прямокутних імпульсів напруги, що мають затримку від 5000 до 1250 наносекунд. Генерація імпульсів здійснюється на виводах МК 11 та 12. Затримка між імпульсами обов'язкова, оскільки необхідно дати часовий інтервал для завершення перехідних процесів транзисторів. Можна також змінити скважність імпульсів, що допоможе знімати більшу потужність із вихідного трансформатора, але це потягне за собою в довгостроковій перспективі такі негативні наслідки, як перегрів транзисторів та зміну форми вихідного сигналу.

Для зміни частоти було використано потенціометр номіналом 20кОм, який підключений на вхід МК АЦП 0.

Реалізація АЦП із значенням від 0 -1023 здійснюється наступним фрагментом програмного коду:

```
k = map(analogRead(0),0,1023,5000,1250);
```

Функція map виконує провідну роль, адже саме вона перетворює дані (маппінг) діапазону 0-1023 на діапазон значень 5000-1250.

Для відображення даних був використаний дворядковий LCD монітор розміром 16x2 на базі контролера HD44780.

```
lcd.setCursor(0,0);
lcd.print("Hz=");
lcd.setCursor(3,0);
lcd.print(Hz);
int v = (analogRead(1)/1023.0) * 20;
delayMicroseconds(k);
lcd.setCursor(0,1);
lcd.print("v=");
lcd.setCursor(2,1);
lcd.print(v);
```

Для роботи з цим монітором було використано стандартну бібліотеку LiquidCrystal. Дисплей підключений до ніжок МК 4,5,6,7,8,9,10. На ньому інтерактивно відображається поточна частота в герцах та стан заряду акумулятора у відсотках.

Для отримання даних про стан заряду використано аналоговий вхід АЦП 1. Для роботи АЦП із підвищеною напругою акумулятора було використано резистивний поділювач напруги. Наступний фрагмент коду показує перетворення отриманих даних у вольти і у відсотки рівня заряду акумулятора:

```
int v = (analogRead(1)/1023.0) * 20;
```

Керуючий мікроконтролер працює на частоті 16 Мегагерц із зовнішнім кварцовим резонатором, що дозволяє отримати прийнятний рівень точності генерації імпульсів.

Прототип та моделювання системи. Даний перетворювач напруги побудований на основі мікроконтролера ATmega328P, який запрограмований на частоту 50Гц на мові програмування Arduino.

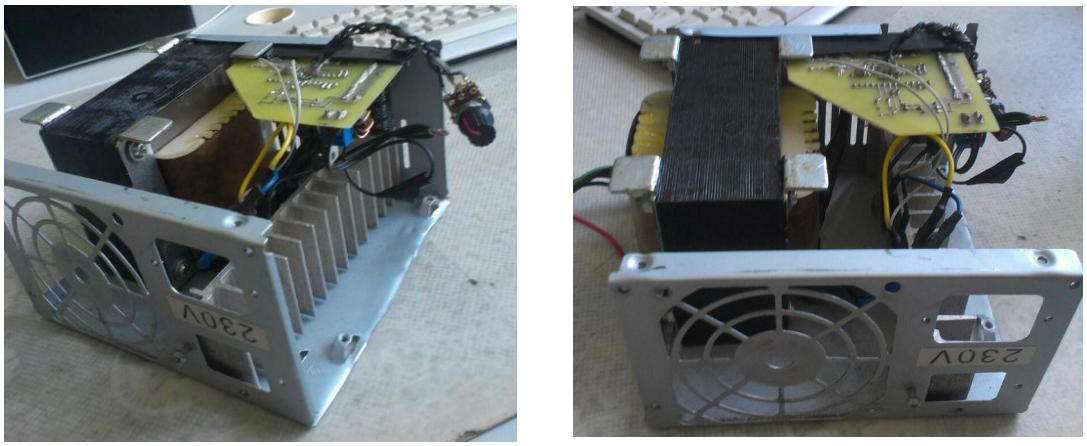


Рис. 3. Зовнішній вигляд прототипу перетворювача напруги

Даний інвертор є низькочастотним. Виміряний коефіцієнт корисної дії прототипу цього пристрою досягає 75%. Оскільки використовується трансформатор максимальною потужністю 350Вт, то існує можливість отримати більшу потужність за рахунок збільшення частоти та зниження загального ККД.

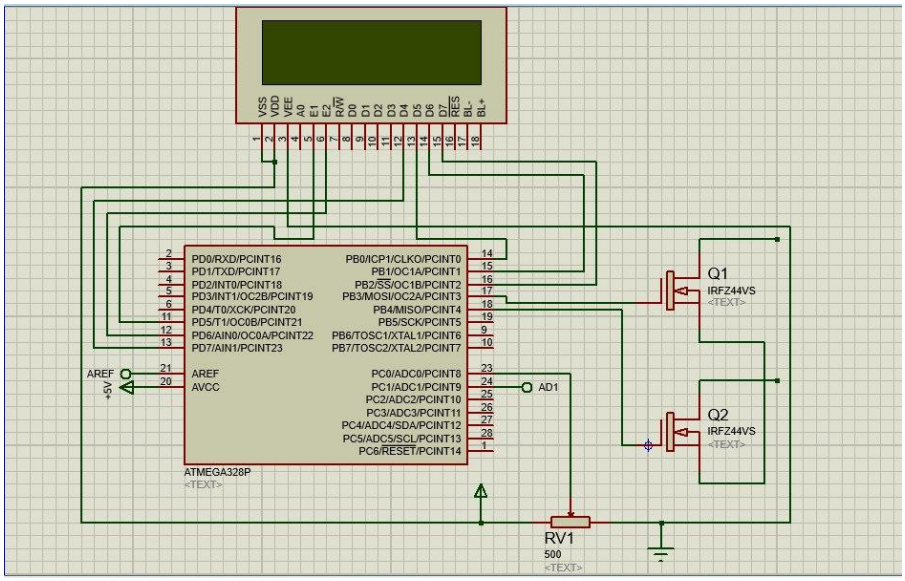


Рис. 4. Структурна схема перетворювача у програмі Proteus

В системі живлення електронної частини інвертора використано стабілізатор LM7805 та низькочастотний фільтр на базі котушки індуктивності і конденсатора великої ємності, які частково зменшують вплив змінної напруги, яка протікає у контурі трансформатора, на електронну частину. Під час роботи перетворювача напруги відбувається значне тепловиділення від польових транзисторів, тому було встановлено вентилятор із автоматичним регулюванням по температурі. При досягнення температури 60^{o}C включається вентилятор для охолодження пристрою. При зниженні температури до 40 ^{o}C вентилятор вимикається. Захистом від перегріву служить термореле. Коли температура сягає більше 100 ^{o}C автоматично вимикається живлення мікроконтролера і припиняється робота приладу. Також встановлено захисний діод від переполюсування.

Для зняття необхідної інформації та моніторингу інвертора використано давач струму, вольтметр, термодавач та LCD монітор який виводить всю необхідну інформацію. Також є RGB-діод, який інформує відповідним кольором про режим роботи перетворювача. Зелений колір – нормальний режим, червоний – перенавантаження.

Особливості експлуатації приладу. Даний пристрій здебільшого використовується в польових умовах, там де немає доступу до мережі 220 В. Він містить вбудований потенціометр, який запрограмований на зміну частоти пристрою. Слід зазначити, що не всі пристрої можна підключати до мережі з частотою більшою ніж 60 Гц. Зокрема це стосується таких пристроїв як двигуни. Швидкість обертів збільшиться і вони вийдуть з штатного режиму роботи. Але в переважній більшості електронні пристрої легко працюють на більшій частоті. При збільшені частоти збільшується потужність пристрою, але тим самим значно навантажуються польові транзистори. Оскільки в розробленому пристрої є індикатор рівня заряду акумулятора, це дозволяє моніторити його стан. Коли рівень заряду нижчий за 11,5 В, припиняється генерація імпульсів.

Висновки

Результатом проведеної роботи є пристрій інвертор DC-AC на базі мікроконтролера ATmega328p, силового низькочастотного трансформатора та силових ключів. Було розроблена структурна схема пристрою, визначені технічні вимоги. Також в процесі налагодження та експлуатації було досягнуто відповідних технічних показників, які задовольняють вимоги технічного завдання, усунуто виявлені недоліки. Рівень якості виробу відповідає загальноприйнятим стандартам.

До переваг розробленого пристрою можна віднести:

- простоту схеми та доступність складових електронних компонентів;
- можливість змінювати частоту;
- коефіцієнт корисної дії досягає 80%;
- моніторинг частоти і заряду акумулятора;
- керування польовими транзисторами без спеціалізованого драйвера.

Недоліки:

- досить значні для даного класу пристроїв масо-габаритні показники, зумовлені наявністю низькочастотного трансформатора;
- порівняно з імпульсними перетворювачами напруги менша потужність;
- висока ціна трансформатора.

Можливим є подальше удосконалення пристрою, наприклад, дозволити змінювати частоту в залежності від навантаження. Існує можливість додатково підключати різні давачі для побудови чистої синусоїди за допомогою ШІМ модуляції і електролітичних конденсаторів низької ємності на затворах польових транзисторів. Подальші модифікації пристрою дозволять підвищити експлуатаційні характеристики пристрою.

1. Крогерис А., Рашевиц К., Рутманис Л. и др. Полупроводниковые преобразователи электрической энергии / Под ред. А. Крогериса. Рига: Зинатне. 1969.
2. Ирвинг М., Готтлиб. Источники питания. Инверторы, конверторы, линейные и импульсные стабилизаторы. – 2-е изд. – М.: Постмаркет, 2002. – 544 с.
3. Преобразователи напряжения и автомобильные инверторы напряжения (12-220В, 220-12В, 24-12В) [Електронний ресурс]. – Режим доступу: http://auto-line.net/ru/shop/car-electrics/car-voltage-converters.
4. «Польовий» перетворювач напруги [Електронний ресурс]. – Режим доступу: http://www.radiosvoboda.org/content/article/26936954.html.
5. Схема преобразователя 12\220 [Електронний ресурс]. – Режим доступу: http://monitor.espec.ws/section44/printview127136.html.
6. Транзисторные DC-AC преобразователи напряжения: характеристики, структурные схемы, рекомендации по применению // "Силовая электроника" № 2, 2004.
7. C2000™ Solar DC/AC Single Phase Inverter // http://www.ti.com/tool/TIDM-SOLAR-ONEPHINV#descriptionArea
8. http://www.xantrex.com/power-products/power-inverters/overview.aspx
9. https://www.cyberpowersystems.com/support/knowledge-center/general-literature
10. https://datasheets.maximintegrated.com/en/ds/MAX1739-MAX1839.pdf

УДК 004.056.55
Красиленко В.Г., к.т.н., с.н.с., доцент, професор; Нікітович Д.В., науковий співробітник
Вінницький інститут Університету "Україна"

МОДЕЛЮВАННЯ КРИПТОГРАФІЧНИХ ПЕРЕТВОРЕНЬ КОЛЬОРОВИХ ЗОБРАЖЕНЬ НА ОСНОВІ МАТРИЧНИХ МОДЕЛЕЙ ПЕРЕСТАНОВОК ЗІ СПЕКТРАЛЬНОЮ ТА БІТОВО-ЗРІЗОВОЮ ДЕКОМПОЗИЦІЯМИ

Красиленко В.Г., Нікітович Д.В. Моделювання криптографічних перетворень кольорових зображень на основі матричних моделей перестановок зі спектральною та бітово-зрізовою декомпозиціями. В роботі представлені результати моделювання криптографічних перетворень кольорових зображень на основі їх декомпозиції на спектральні складові і матричні бітові зрізи і застосування модифікованих матричних моделей перестановок для перемішування елементів масивів і забезпечення можливості верифікації цілісності створених криптограм. Процеси прямого і зворотного криптографічних перетворень зводяться до матрично-матричних процедур. В якості матричних ключів використовуються одна або дві матриці перестановок або їх ступені. Показано як використання декомпозиції і конкатенації зображень дозволяє виконувати не тільки самі перетворення, але і контролювати цілісність криптограм. Експерименти в середовищі Mathcad з кольоровим зображенням (128*128 елементів) з метою їх зашифрування і розшифрування за допомогою запропонованих моделей підтвердили адекватність і переваги останніх.
Ключові слова: криптографічні перетворення зображень, матричні моделі, декомпозиція, конкатенація, матриця перестановок, ступінь матриці, матрично - матрична процедура, матрично- бітово-зрізове (bit-plane) кодування, криптограма, стійкість криптографічної системи, матричні ключі.

Красиленко В.Г., Никитович Д.В. Моделирование криптографических преобразований цветных изображений на основе матричных моделей перестановок со спектральной и битово-срезовой декомпозициями. В работе представлены результаты моделирования криптографических преобразований цветных изображений на основе их декомпозиции на спектральные составляющие и матричные битовые срезы и применения модифицированных матричных моделей перестановок для перемешивания элементов массивов и обеспечения возможности верификации целостности созданных криптограмм. Процессы прямого и обратного криптографических преобразований сводятся к матрично-матричным процедурам. В качестве матричных ключей используются одна или две матрицы перестановок или их степени. Показано как использование декомпозиции и конкатенации изображений позволяет выполнять не только сами преобразования, но и контролировать целостность криптограмм. Эксперименты в среде Mathcad с цветным изображением (128*128 элементов) с целью их шифрования и расшифровки с помощью предложенных моделей подтвердили адекватность и преимущества последних.
Ключевые слова: криптографические преобразования изображений, матричные модели, декомпозиция, конкатенация, матрица перестановок, степень матрицы, матрично-матричная процедура, матрично-битово-срезовое (bit-plane) кодирование, криптограмма, стойкость криптографической системы, матричные ключи.

Krasilenko V.G., Nikitovich D.V. Simulation of cryptographic transformations of color images based on matrix models of permutations with spectral and bit-plane decompositions. The results of the simulation of cryptographic transformations of color images on the basis of their decomposition into spectral and bit slices and use of modified permutation matrix models for mixing elements of arrays and enable verification of the integrity of the established cryptogram. The direct and inverse cryptographic transformations are reduced to the matrix-matrix procedure. As matrix keys used one or two of permutation matrix or their degree. It is shown how to use decomposition and concatenation of images allows you to perform not only the transformation, but also to control the integrity of cryptograms. Experiments in Mathcad with color multilevel images (128 * 128 elements) with a view to encrypt and decrypt using the proposed models have confirmed the relevance and benefits of the latter.
Keywords: cryptographic transformation of color images, matrix models, matrix key, decomposition, matrix-bit-slice (bit-plane) encoding, cryptogram, cryptographic system stability, a permutation matrix, the degree of the matrix, the matrix-matrix procedure.

Вступ. Постійне збільшення обсягів та значимості інформаційних потоків з розвитком телекомунікаційних мереж, електронних комунікацій, широке застосування інформаційних технологій потребують, особливо для захищених систем управління, надійного та ефективного захисту цілісності інформаційних об'єктів (ІО) та стійкості до потенційних загроз. Суттєво зросла доля специфічних текстово-графічних документів (ТГД) у вигляді табличних даних, малюнків, діаграм, підписів, віз, резолюцій, тощо, які є зображеннями і які необхідно опрацьовувати та передавати каналами з обмеженим чи закритим доступом, засвідчувати їх цифровими підписами. Особливе місце серед відомих методів захисту ІО від несанкціонованого доступу займають криптографічні методи, що спираються на властивості самих ІО, а не матеріальних носіїв ІО та пристроїв їх обробки, передачі та зберігання. Більшість методів та засобів криптографічних перетворень (КП) ІО чи зображень, процедур і протоколів формування ключів та їх обміну орієнтовані на послідовну скалярну обробку блоків ТГД. Навіть для використовуваних найкращих симетричних алгоритмів (на основі діючого стандарту AES, IDEA, тощо) довжини блоків та ключів не перевищують 256 бітів, за винятком хіба-що FEAL, RC6 та інших нових, де ці довжини можуть обмежуватись 1К-2К бітами [1]. Програмні засоби реалізації КП у порівнянні з апаратними системами, хоч і мають ряд переваг, є ненадійними та слабкими до атак з точки зору їх «хакерських» зламувань чи завідомо введених вставок для контролю спецслужбами. Збільшення

складності задач та об'ємів ІО, що перероблюються в реальному часі, обумовило створення паралельних комп'ютерів, але зростання швидкості виконання операцій та складності процесорів призводить до збільшення збоїв та помилок, тому ще одним важливим завданням стає і контроль цілісності ІО, виявлення та виправлення помилок. Поява паралельних алгоритмів, а особливо матричних процесорів, потребує переорієнтації КП на ці засоби, та створення моделей матричного типу (МТ) [2-6]. Тому пошук нових матричних моделей (ММ) та засобів виконання КП ІО у вигляді ТГД (зображень), що краще відображаються на матричні процесори та забезпечують перевірку цілісності зашифрованих ІО, є актуальним завданням.

Аналіз останніх досліджень і публікацій. У роботі [2] були продемонстровані переваги КП матричними алгоритмами на основі більш узагальнених матричних афінних шифрів, в тому числі при створенні сліпих цифрових підписів на ТГД [3]. Ще більш узагальнені матричні афінно-перестановочні шифри були запропоновані та досліджені в [4], базовою операцією яких є матричні моделі перестановок (ММ_П), які мають наочну простоту. Проте, як показано в [6], КП на їх основі без додаткових операцій не змінюють гістограми зображень, а запропоновані в ній модифіковані моделі з декомпозицією бітових зрізів, хоч і усувають цей недолік, потребують у деяких випадках крім двох матричних ключів (МК) ще й двох векторних (ВК). В той же час всі вищезгадані моделі не дозволяють перевіряти цілісність криптограми та наявність перекручувань. У роботах [7,8] були розглянуті модифіковані матричні моделі КП з верифікацією цілісності криптограм лише для чорно-білих зображень, а в [9] хоч і розглядались КП кольорових, але без аспектів верифікації цілісності, тому є потреба розширити узагальнити ці моделі на випадок кольорових зображень, враховуючи специфіку їх форматів та розширень.

Постановка наукової проблеми. Таким чином є необхідною і актуальною спроба подальшої модифікації та узагальнення ММ для КП саме кольорових зображень з метою покращення їх характеристик і функціональних можливостей за рахунок верифікації цілісності. Моделювання та перевірка функціонування створених моделей на реальних ІО дозволить оцінити їх показники, можливості та особливості застосувань.

Тому метою даної роботи є узагальнення ММ_П на випадок КП кольорових зображень з перевіркою цілісності ІО та використанням спектральної та бітово-зрізової декомпозиції та надлишкового зрізу-підпису, їх моделювання у програмному середовищі Mathcad та оцінювання.

Виклад основного матеріалу та результатів дослідження.

Сутність запропонованих узагальнених ММ_П з одночасною спектральною та бітово-зрізовою декомпозиціями (ММ_П_СБЗД) полягає у формуванні з явного кольорового зображення, а саме з його трьох R,G,B складових (BZ1, BZ2, BZ3) шляхом матричних аналого-цифрових перетворень (АЦП) наборів М-розрядів-зрізів та додаткового зображення CON_S, наприклад, згорнутого специфічним h – функціональним перетворенням, (у експерименті це по-елементне додавання за модулем 2 всіх 24=3*8 зрізів), їх конкатенації та застосуванні КП до спільного масиву A_SC, шляхом множення його зліва і справа на матриці перестановок (МК чи його степені), що еквівалентно перемішуванню рядків та стовбців масиву. На рис. 1-5 зображені результати моделювання прямого та оберненого криптографічних перетворень кольорового зображення на основі ММ_П_СБЗД. Програмні модулі та формули для моделювання процесів створення ключів, зашифрування та розшифрування кольорових зображень на основі ММ_П_СБЗД з формуванням кольорової криптограми та матриці ознаки її цілісності показані на рис.1,2. Кожне спектральне складове спочатку перетворювалось у 8 бітових зрізів за допомогою формул на рис.1b, а потім всі 24 зрізи (матриці кодів Грея) конкатенувались з утвореним з них контрольним зрізом CON_S у масив A_SC за допомогою формул на рис.1c. З урахуванням обмежень на обсяг роботи та висвітлень в роботах [4-8] аспектів генерування ключів (матриць перестановок) та протоколів їх обміну [10] тут ми відмічаємо лише той позитивний факт, що в ММ_П_СБЗД достатньо одного матричного ключа (KPS), розмірність якого для експериментів була 640*640 відповідно до розмірів (128*128) взятого для перетворень зображення. За допомогою матричної процедури чи подібних їй при зміні степенів МК у відповідності до скалярних ключів η^{sl} та η^{sp}, (рис.2), формували масив C_ASC з об'єднаного масиву A_SC, а з нього результуючу кольорову криптограму (Ar, Ag, Ab) та її цифровий підпис C_CON (дивись рис.2, 3) шляхом обернених цифро-аналогових перетворень (ЦАП), формувань відповідних R,G,B спектральних складових кольорової криптограми та контрольного зрізу, що є по суті відповідним цифровим підписом (ЦП) криптограми.

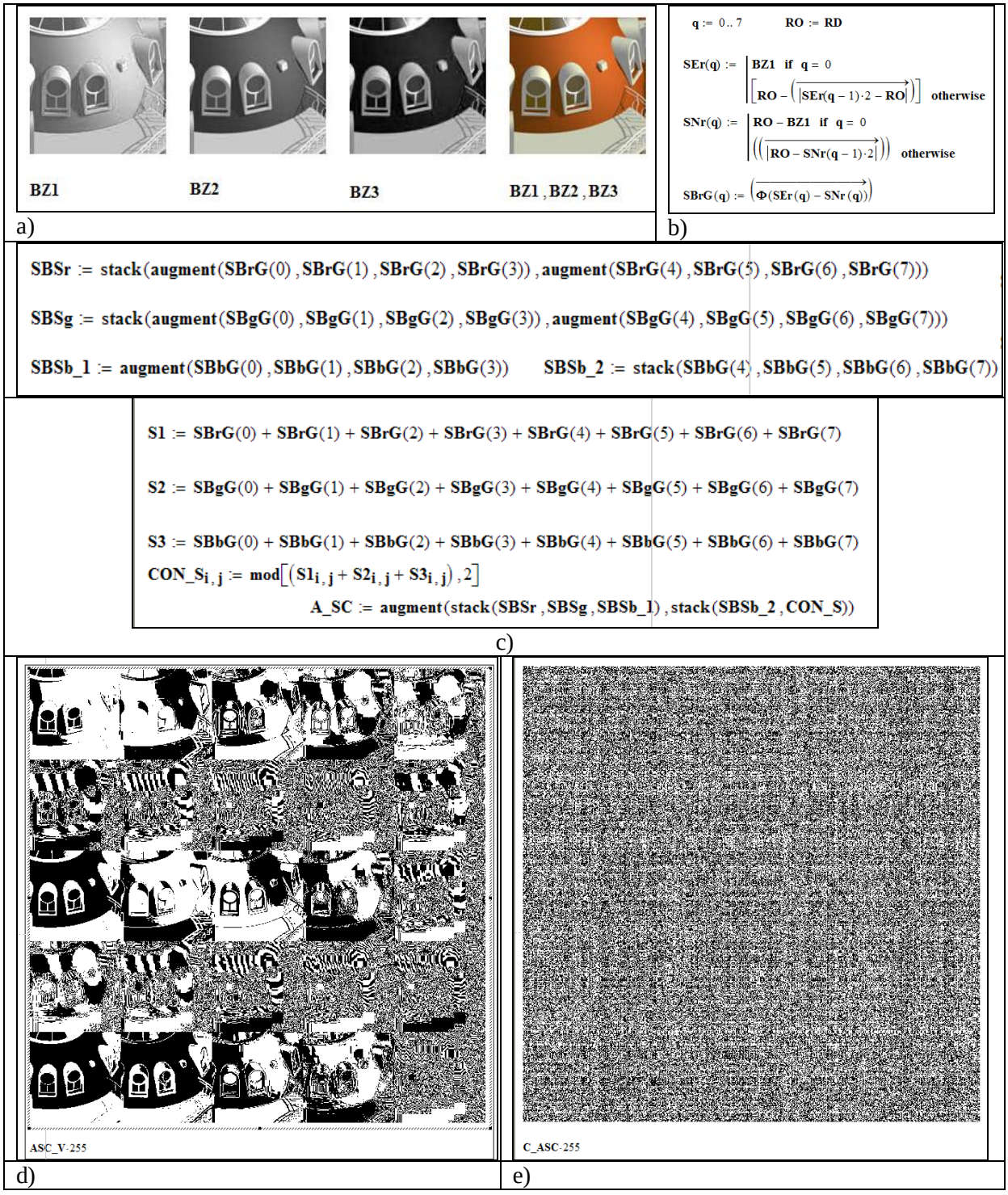


Рис. 1. Зображення та фрагменти вікон Mathcad, формули, що використовувались для моделювання:a) – R, G, B складові кольорового зображення (праворуч); b) - Формули, що використовувались для прямого АЦ (у бітові зрізи) та оберненого ЦА перетворень; c) – Формули для конкатенації всіх 24 зрізів та 25-го контрольного CON_S у масив A_SC; d) – Відновлений бітовий масив ASC_V, що є і масивом A_SC; e) – Криптограма з цифровим підписом C_CON у бітово-зрізовому представленні.

Процедури виділення бітових зрізів, їх перекодування з кодів Грея у двійкові зріз-коди для зворотної процедури формування трьох R, G, B складових результуючої кольорової криптограми (Ar, Ag, Ab) та її цифрового підпису C_CON (бітовий масив!) з масиву C_ASC (бітово-зрізове представлення криптограми та її ЦП !) здійснювались за допомогою формул на рис.2.

$\eta sl := 5 \quad \eta sln := 7 \quad KPSO := KPS^{T} \quad \eta sp := 9 \quad \eta spn := 3 \quad 128 \cdot 4 = 512$

$C\_ASC := KPS^{\eta sl} \cdot A\_SC \cdot KPS^{\eta sp} \quad 128 \cdot 5 = 640$

$ASC\_V := KPSO^{\eta sl} \cdot C\_ASC \cdot KPSO^{\eta sp}$

$Cr0 := submatrix(C\_ASC, 0, 127, 0, 127)$
$Cr1 := submatrix(C\_ASC, 0, 127, 128, 255)$
$Cr2 := submatrix(C\_ASC, 0, 127, 256, 383)$
$Cr3 := submatrix(C\_ASC, 0, 127, 384, 511)$
$Cb0 := submatrix(C\_ASC, 512, 639, 0, 127)$
$Cb1 := submatrix(C\_ASC, 512, 639, 128, 255)$
$Cb2 := submatrix(C\_ASC, 512, 639, 256, 383)$
$Cb3 := submatrix(C\_ASC, 512, 639, 384, 511)$

$Cr4 := submatrix(C\_ASC, 128, 255, 0, 127)$
$Cr5 := submatrix(C\_ASC, 128, 255, 128, 255)$
$Cr6 := submatrix(C\_ASC, 128, 255, 256, 383)$
$Cr7 := submatrix(C\_ASC, 128, 255, 384, 511)$
$Cb4 := submatrix(C\_ASC, 0, 127, 512, 639)$
$Cb5 := submatrix(C\_ASC, 128, 255, 512, 639)$
$Cb6 := submatrix(C\_ASC, 256, 383, 512, 639)$
$Cb7 := submatrix(C\_ASC, 384, 511, 512, 639)$

$Br7 := Cr0 \quad Bg7 := Cg0 \quad Bb7 := Cb0$

$Br6 := (Br7 \oplus Cr1) \xrightarrow{} \quad Bg6 := (Bg7 \oplus Cg1) \xrightarrow{} \quad Bb6 := (Bb7 \oplus Cb1) \xrightarrow{}$

$Br5 := (Br6 \oplus Cr2) \xrightarrow{} \quad Bg5 := (Bg6 \oplus Cg2) \xrightarrow{} \quad Bb5 := (Bb6 \oplus Cb2) \xrightarrow{}$

$Br4 := (Br5 \oplus Cr3) \xrightarrow{} \quad Bg4 := (Bg5 \oplus Cg3) \xrightarrow{} \quad Bb4 := (Bb5 \oplus Cb3) \xrightarrow{}$

$Br3 := (Br4 \oplus Cr4) \xrightarrow{} \quad Bg3 := (Bg4 \oplus Cg4) \xrightarrow{} \quad Bb3 := (Bb4 \oplus Cb4) \xrightarrow{}$

$Br2 := (Br3 \oplus Cr5) \xrightarrow{} \quad Bg2 := (Bg3 \oplus Cg5) \xrightarrow{} \quad Bb2 := (Bb3 \oplus Cb5) \xrightarrow{}$

$Br1 := (Br2 \oplus Cr6) \xrightarrow{} \quad Bg1 := (Bg2 \oplus Cg6) \xrightarrow{} \quad Bb1 := (Bb2 \oplus Cb6) \xrightarrow{}$

$Br0 := (Br1 \oplus Cr7) \xrightarrow{} \quad Bg0 := (Bg1 \oplus Cg7) \xrightarrow{} \quad Bb0 := (Bb1 \oplus Cb7) \xrightarrow{}$

Рис.2. Формули, що використовувались для створення матриці перестановок KPS та оберненої до неї KPSO, скалярних ключів ηsl та ηsp, матричних процедур для реалізації моделей зашифрування (утворення криптограми C_ASC з об'єднаного масиву A_SC) та розшифрування (відновлення масиву ASC_V) та процедур виділення бітових зрізів, їх перекодуванні з кодів Грея у двійкові зріз-коди для зворотної процедури формування трьох R, G, B складових результуючої кольорової криптограми (Ar, Ag, Ab) та її цифрового підпису C_CON (бітовий масив!).

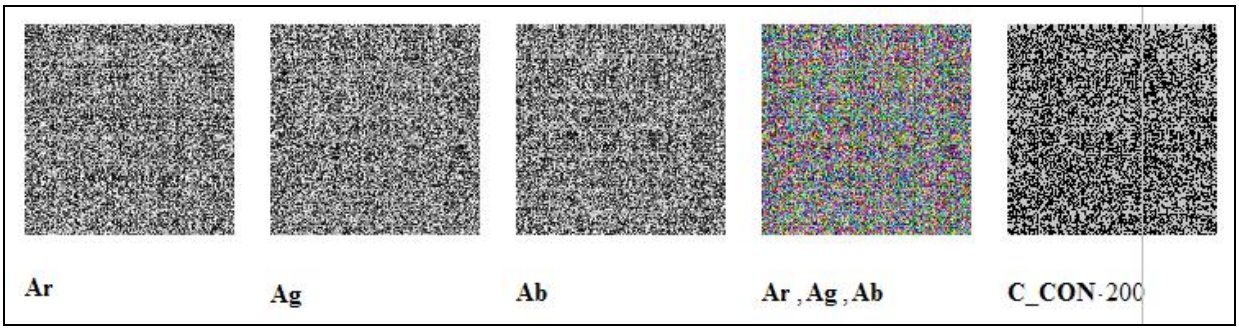


Рис.3. Утворені зашифруванням три R (Ar), G (Ag), B (Ab) складові результуючої кольорової криптограми (Ar, Ag, Ab) та її цифровий підпис C_CON (бітовий масив!).

Аналогічними процедурами, але вже за допомогою обернених перестановок з використанням ключа KPSO та аналогічних скалярних ключів відновлюється масив ASC_V, а з нього формуються відновлені (розшифроване) зображення та його контрольний підпис для верифікації. Отримані

результати моделювання процесу розшифрування при використанні правильних ключів, що на рис.5, свідчать про коректність моделей та правильне відтворення кольорового та всіх проміжних зображень на різних кроках перетворень. Результати моделювання процесу розшифрування при використанні неправильних ключів чи наявних втручаннях, що будуть наводитись у доповіді також засвідчили адекватність та стійкість моделей. Гістограми, що на рис.4, та їх аналіз також підтверджують якісну роботу моделей.

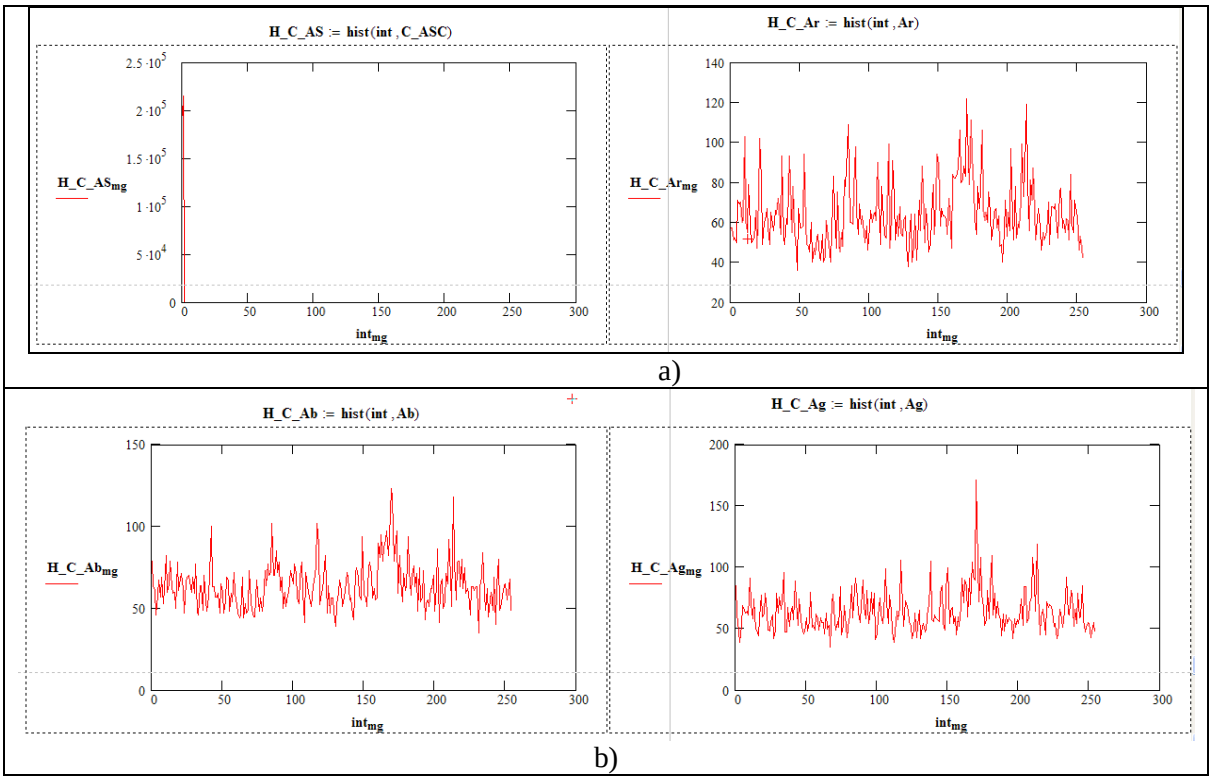


Рис.4. Гістограми криптограми C_ASC (вона ж - масиву A_SC) у бітово-зрізовому вигляді та утворених трьох R (Ar), G (Ag), B (Ab) спектральних складових результуючої кольорової криптограми (Ar, Ag, Ab).

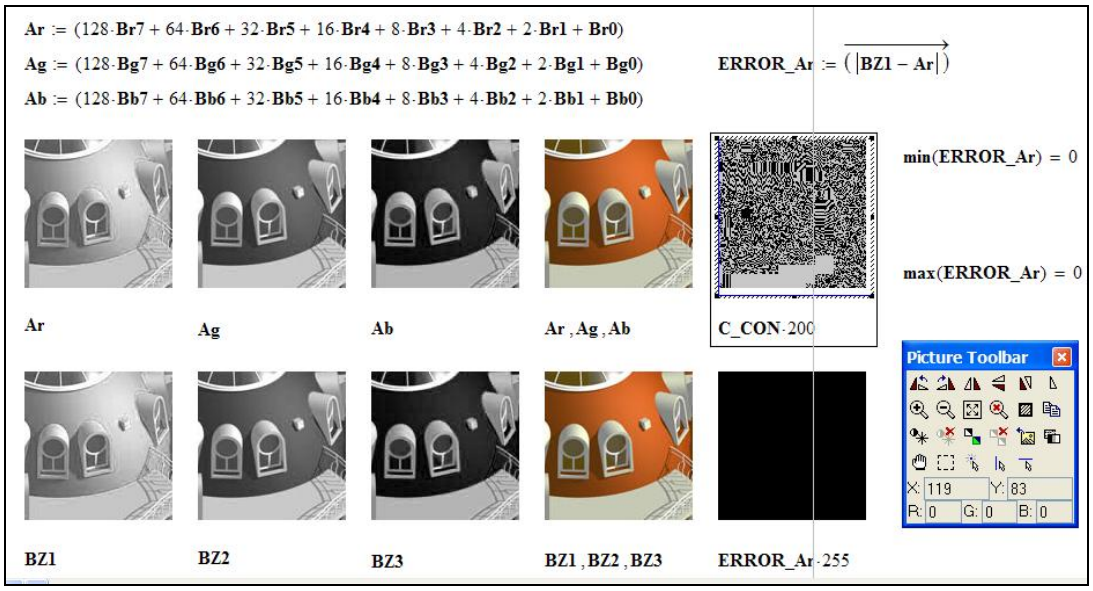


Рис.5. Відновлені розшифруванням та ЦА перетвореннями (формули вгорі!) зображення (верхній ряд) та початкові (спектральні складові та кольорове у нижньому ряду) для порівняння та верифікації точності моделей криптографічних перетворень та їх цілісності.

У роботах [3-6] було показано можливість збільшення ентропії криптограм на основі подібних моделей практично аж до 7,5-7,8 біт на точку чорно-білого зображення. Результати моделювання свідчать про неможливість без знання ключів відтворити початкове зображення. У випадку внесення спотворень при знанні особою ключів можна визначити наявність втручань, а порівнянням апріорного та розшифрованого зображень навіть усунути їх. Стосовно крипостійкості розглянутих моделей, то зазначимо, що, як було показано в [3], з урахуванням сьогоднішніх методів та засобів пошуку ключа гарантована стійкість аналогічних по суті базових моделей на основі матриць перестановок, яка залежить від потужності множини ключів (вона пропорційна n!, де n – розмірність матричних ключів), забезпечується уже при розмірності, рівній 32*32, а в нас матричні ключі розмірністю щонайменше 640*640, тобто відповідна потужність (640!) ще на багато порядків створює запас стійкості.

Висновки

Результати моделювання прямого та оберненого КП на основі ММ_П_СБЗД свідчать про коректну роботу цих моделей, їх адаптованість до різних форматів, зручність (всього один МК та його степені!), функціональність (можливість визначати факти втручань та порушень цілісності ІО, знаходити їх та усувати) та ефективність (орієнтація на матричні процесори). Розглянуті процедури створення необхідних МК, матриць перестановок та їх обміну, оцінена крипостійкість. Моделі ускладнюють здійснення прихованих атак та внесення перекручувань.

1. Ємець В. Сучасна криптографія. Основні поняття / В. Ємець, А. Мельник, Р. Попович. – Львів: БаК, 2003. – 144 с.
2. Красиленко В.Г. Моделювання матричних алгоритмів криптографічного захисту / В.Г. Красиленко, Ю.А. Флавицька // Вісник НУ «Львівська політехніка» «Комп'ютерні системи та мережі». - № 658. – С. 59-63.
3. Красиленко В.Г., Матричні афінні шифри для створення цифрових сліпих підписів на текстографічні документи / В.Г. Красиленко, С.К. Грабовляк // Системи обробки інформації. – Х.: ХУПС, 2011. – Вип. 7(97**). –** С. 60 – 63.
4. Красиленко В.Г. Матричні афінно-перестановочні шифри для шифрування та дешифрування зображень / В.Г. Красиленко, С.К. Грабовляк // Системи обробки інформації. - Х.: ХУПС, 2012. – Вип. 3 (101).-т. 2. **–** С. 53-62.
5. Красиленко В.Г. Матричні моделі криптографічних перетворень зображень з матрично-бітово-зрізовою декомпозицією і перемішуванням та їх моделювання / В. Г. Красиленко, Д.В. Нікітович //Матеріали 68 НТК «Сучасні інформаційні системи і технології. Інформаційна безпека», ч. 3, секції 3-4. – Одеса, ОНАЗ ім.. О.С.Попова, 2013. – С.139-143.
6. Красиленко В.Г. Криптографічні перетворення зображень на основі матричних моделей перестановок з матрично-бітовозрізовою декомпозицією та їх моделювання / В. Г. Красиленко, В. М. Дубчак // Вісник Хмельницького національного університету. Технічні науки. - 2014. - № 1. - С. 74-79.
7. Красиленко В.Г. Моделювання модифікованих матричних моделей криптографічних перетворень зображень з верифікацією цілісності криптограм / В.Г. Красиленко, Д.В. Нікітович // Матеріали II міжнародної науково-практичної Інтернет-конференції «Інформаційні технології: теорія, інновації, практика». – Полтава: ПолтНТУ, 2015. – С. 86-89.
8. Красиленко В.Г. Моделювання криптографічних перетворень зображень на основі їх матрично-бітовозрізової декомпозиції та матричних моделей перестановок з верифікацією цілісності / В.Г. Красиленко, Д.В. Нікітович // Всеукраїнська науково-практична конференція молодих вчених та студентів «Перспективні напрямки сучасної електроніки, інформаційних і комп'ютерних систем» (MEICS-2015). – Дніпропетровськ: Дніпропетровський національний університет ім. Олеся Гончара, 2015. - С. 32-34.
9. Красиленко, В.Г. Моделювання матричних афінних алгоритмів для шифрування кольорових зображень / В. Г. Красиленко, К. В. Огородник, Ю.А.Флавицька // Комп'ютерні технології: наука і освіта: тези доповідей V Всеукр. наук.-пр. конф. – К., 2010. – С.120-124.
10. Красиленко В. Г. Алгоритми формування двовимірних ключів для матричних алгоритмів криптографічних перетворень зображень та їх моделювання / В. Г. Красиленко, В. І. Яцковський, Р. О. Яцковська // Системи обробки інформації. - 2012. - Вип. 8. - С. 107-110.

УДК 681.6 : 004.9
Сальніков О.В.[1], Мартинюк О.С.[2], Шолом П.С.[3]
[1]Комунальний заклад «Луцький навчально-виховний комплекс «Загальноосвітня школа І-ІІІ ступенів № 22 – ліцей» Луцької міської ради»
[2]Східноєвропейський національний університет імені Лесі Українки
[3]Луцький національний технічний університет

ТЕХНОЛОГІЇ ВИГОТОВЛЕННЯ ТА ВИКОРИСТАННЯ 3D-ПРИНТЕРА

Сальніков О.В., Мартинюк О.С., Шолом П.С. Технології виготовлення та використання 3D-принтера. Обґрунтовано перспективи впровадження засобів 3D-моделювання та проведено апробацію технології, доступної для самостійного виготовлення та використання базового зразка 3D-принтера. Експериментально підтверджено можливості технології 3D-моделювання, доцільності та можливості самостійного проектування та виготовлення 3D-конструкцій.
Ключові слова: 3D-принтер, калібрування, сопло екструдера, Prusa Mendel i2, Arduino IDE, OpenScad, Repetier Host, Slic3r

Сальников А.В., Мартынюк А.С., Шолом П.С. Технологии изготовления и использования 3D-принтера. Обосновано перспективы внедрения средств 3D-моделирования и проведена апробация технологии, доступной для самостоятельного изготовления и использования базового образца 3D-принтера. Экспериментально подтверждено возможности технологии 3D-моделирования, целесообразности и возможности самостоятельного проектирования и изготовления 3D-конструкций.
Ключевые слова: 3D-принтер, кали бровка, сопло екструдера, Prusa Mendel i2, Arduino IDE, OpenScad, Repetier Host, Slic3r

Salnikov O., Martyniuk O., Sholom P. Technology of manufacturing and use of 3D-printers. The prospects of 3D-modeling tools introduction are justified and the technology approbation, available for self-manufacture and use of the base sample of 3D-printers, is conducted. The possibility of 3D-modeling technology, the feasibility and possibility of independent design and manufacture of 3D-structures is experimentally confirmed.
Keywords: 3D-printer, calibration, an extruder nozzle, Prusa Mendel i2, Arduino IDE, OpenScad, Repetier Host, Slic3r

Постановка проблеми. Історія розвитку засобів для друку об'ємних зразків налічує вже майже три десятиліття. В останні роки інтерес до них став особливо зростати. Багато промислових компаній почали активно використовувати 3D-принтери, що забезпечує скорочення витрат на виробництво складної технічної продукції. Продукти 3D-друку користуються великим попитом серед дизайнерів, архітекторів, конструкторів. Ця технологія дає можливість їм у найкоротші терміни отримувати високоякісні прототипи виробів, макети та заготовки. У недалекому майбутньому виникне потреба у фахівцях цієї галузі. Тому **актуальною** є проблема вивчення технологій 3D-моделювання та можливостей самостійного проектування та виготовлення 3D-принтерів.

Аналіз можливостей технологій 3D-моделювання у різноманітних сферах людської діяльності засвідчив їх тотальне застосування, зокрема, за даними експертів DHL, кожна сім'я, що проживає в країні розвиненого світу, вже до 2050 р. використовуватиме тривимірний друк в домашніх умовах. Це означає, що надрукувати велосипед, меблі, посуд і аксесуар не буде проблемою для непрофесійних користувачів.

Мета досліджень полягає у теоретичному обґрунтуванні можливостей технологій 3D-моделювання, проектуванні, виготовленні та апробації 3D-принтера. Задля досягнення мети наукової роботи було поставлено і реалізовано низку технологічних завдань: аналіз можливостей технологій 3D-моделювання у різноманітних сферах людської діяльності; проектування та виготовлення 3D-принтера; проведення апробації конструкції в умовах практичного застосування.

Серед **методів дослідження**, використаних в рамках даної роботи, варто виокремити наступні: методи узагальнення і порівняння – на етапі виявлення основних технологій 3D-друку, метод опису – на етапі представлення наявних типів принтерів, і експериментальної моделі на основі конструкції 3D-принтера Prusa Mendel i2; а також програмне середовище Arduino IDE, програма для створення 3D моделей OpenScad, програма Repetier Host для створення G-кодів моделей, необхідних для друку та завантаження цих кодів на комп'ютер, програма Slic3r для задання шляхів руху сопла екструдера – на етапі налаштування 3D-принтера.

З урахуванням **прикладного** характеру роботи **наукова новизна дослідження** обґрунтована перспективами впровадження засобів 3D-моделювання та апробації технології, доступної для самостійного виготовлення та використання базового зразка 3D-принтера.

Практична значимість полягає у експериментальному підтвердженні можливостей технологій 3D-моделювання, доцільності проектування та виготовлення 3D-принтера. Перевагою роботи є представлення усіх етапів створення принтера, використаного програмного забезпечення щодо його налаштування, також узагальнена послідовність налаштування принтера. Роботу виконано в лабораторії основ автоматики та електронно-обчислювальної техніки Східноєвропейського національного університету імені Лесі Українки, м. Луцьк.

Виклад основного матеріалу роботи.

На вітчизняному ринку формується попит на 3D-технології. І хоча такі принтери вже є в наявності в українських Інтернет-магазинах, та брак офіційних дилерів для більшості виробників 3D-принтерів та фахівців, які можуть їх обслуговувати [5] ще тривалий час відчуватиметься. Звідси – актуальність самостійного проектування та виготовлення 3D-принтера.

Особливості технології виготовлення 3D-принтера.

Для експериментальної моделі обрано конструкцію 3D-принтера Prusa Mendel i2. Оскільки основна її частина досить легка у конструюванні, адже складається з декількох шпильок різної довжини, декількох лінійних валів, набору друкованих деталей, гайок, шайб, декількох підшипників тощо, а тому зібрати можна за короткий термін [2, 3]. Усі деталі наявні для продажу в Україні, що забезпечує доступність виготовлення.

Механіка. Основний каркас Prusa Mendel i2 зібраний з 13 шпильок, 17 пластикових деталей та дрібних гайок і представляє собою трикутну призму з правильним трикутником в основі, направлену бічною стороною вниз, як показано на рисунку 1. Рух по осі X здійснюється по двох лінійних валах, скріплених між собою двома пластиковими деталями. На одній з цих деталей кріпиться мотор з шестернею, що має 16 зубців, а на іншій – підшипник. Між мотором та підшипником натягується ремінь, обидва кінці якого кріпляться до каретки, яка на 3-ох лінійних підшипниках рухається по даній конструкції. Ця конструкція на 4-ох лінійних підшипниках прикріплених до двох описаних вище деталей рухається по двох вертикально закріплених до корпусу лінійних валах. Вал з двома підшипниками та правою деталлю зображено на рисунку 2.

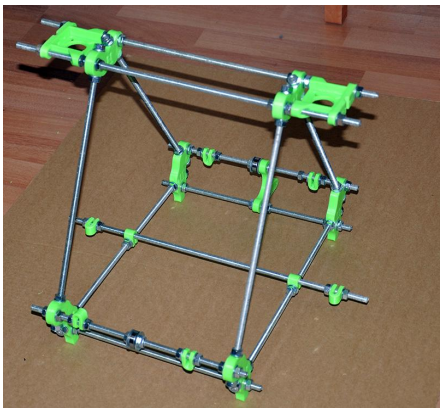


Рис. 1. **Геометрична форма конструкції каркасу Prusa Mendel i2**

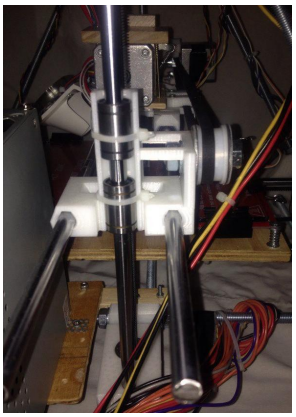


Рис. 2. **Конструкція платформи руху по осі X**

Рух по осі Z, здійснюють два мотори розташовані у верхній частині конструкції. До цих моторів через дві алюмінієві муфти кріпляться дві шпильки, нижні кінці яких закріплені у підшипниках основного каркасу. При обертанні цих шпильок, закріплені у двох деталях осі X гайки підіймають усю конструкцію осі X, за чим слідує і підняття екструдера, закріпленого на каретці.

Рух по осі Y забезпечено переміщенням платформи на якій розташовується виріб, що друкується. До нижньої частини каркасу кріпляться два лінійні вали, по яких на чотирьох лінійних підшипниках і рухається платформа. До спеціальної деталі, закріпленої у тій же нижній частині каркасу, кріпиться мотор з такою ж, як і для осі X, шестернею. Через цей мотор і, розташований на протилежній стороні корпусу, підшипник протягується ремінь, обидва кінці якого кріплять невеличкими пластиковими деталями до платформи.

Останній 5-й мотор використовується для подачі пластикового прутка. На ньому кріпиться спеціальна шестерня для подачі прутка. Мотор кріпиться до конструкції, що складається з невеличкого

корпусу з пружиною та опорним колесом. А ця конструкція кріпиться до каретки, на якій закріплений хотенд. Також до каретки прикріплюється вентилятор з невеличким корпусом для обдування хотенда. Оскільки наша модель тестова, то в цій конструкції замість шестерні із зубцями для подачі пластику використовується шестерня з гумовим покриттям. Метою такого експерименту є визначення якості друку із таким технічним оформленням екструдера, який зображено на рисунку 3.

Можливе також зменшення довжини багатьох валів; оскільки запропонована модель принтера, як зазначалось вище є тестовою, використано 6 однакових лінійних валів (по 0,5 м діаметром 8 мм кожен), задля можливості їх наступного використання у інших моделях. Загальна довжина використаних шпильок (М8) складає більше 6 м. Набір пластикових деталей виготовлений з АВС пластику. Зовнішній вигляд принтера зображено на рисунку 4.

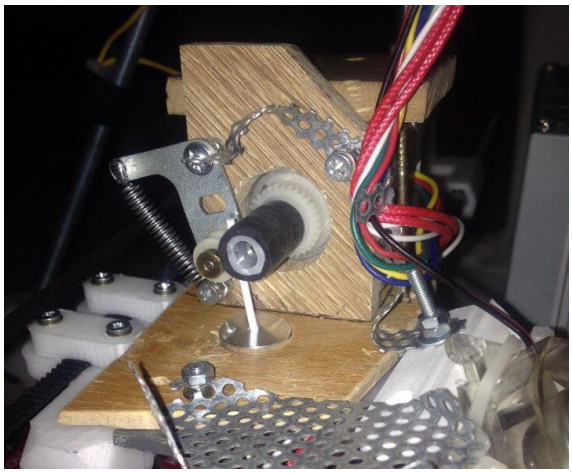


Рис. 3. **Конструкція екструдера**

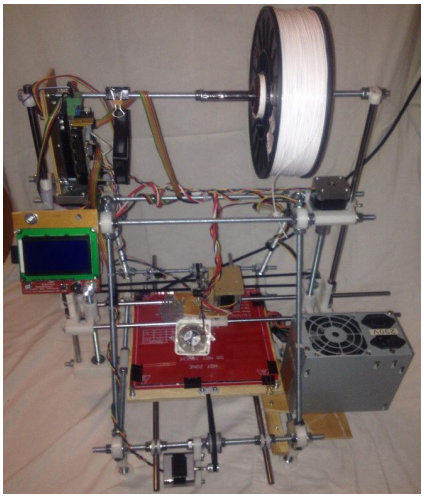


Рис. 4. **Зовнішній вигляд принтера**

Електроніка. В основі електронної складової приладу – Arduino Mega 2560, а також зв'язуючий шилд Ramps 1.4. Мотори контролюються чотирма однаковими драйверами A4988: вісь X, Y та екструдер по 1-му драйверу на кожен мотор, вісь Z – 1 драйвер на обидва мотори. Роз'єми під дані драйвери знаходяться у Ramps 1.4. Живиться електроніка через 16-ти амперний комп'ютерний блок живлення потужністю 300 Вт.

На платформі на чотирьох пружинах закріплюється нагрівальне дно моделі МК, необхідне для друку АВС пластиком і не тільки. Пружини слугують для можливості регулювання правильного положення нагрівального дна відносно принтера та хотенду. У дно також вмонтовується терморезистор (100 кОм) для вимірювання температури дна і прикріплюється канцелярськими затискачами скло, на якому розташовуватиметься виріб. Хотенд складається з радіатора, сопла, нагрівного елемента і такого ж, як і у нагрівальному дні терморезистора.

В нашому випадку використано сопло 0,3 мм, хотенд моделі J-Head (прямий) розрахований на пластиковий пруток діаметром 1,75 мм, радіатор та нагрівник стандартні.

На кожній з осей закріплені кінцеві вимикачі, представлені звичайними важільними кнопками. Ці вимикачі слугують у принтері датчиками початкового положення, координати якого по кожній з осей дорівнюють 0. Також у даній моделі наявні два вентилятори різного розміру: більший для охолодження електроніки, менший для обдування хотенда. Всі електронні складові окрім одного з вентиляторів (більшого) підключаються до Ramps 1.4, як показано на рисунку 5. На схемі не вказано підключення дисплея, оскільки вона аналогічна для більшості принтерів RepRap, зокрема на Ramps 1.4.

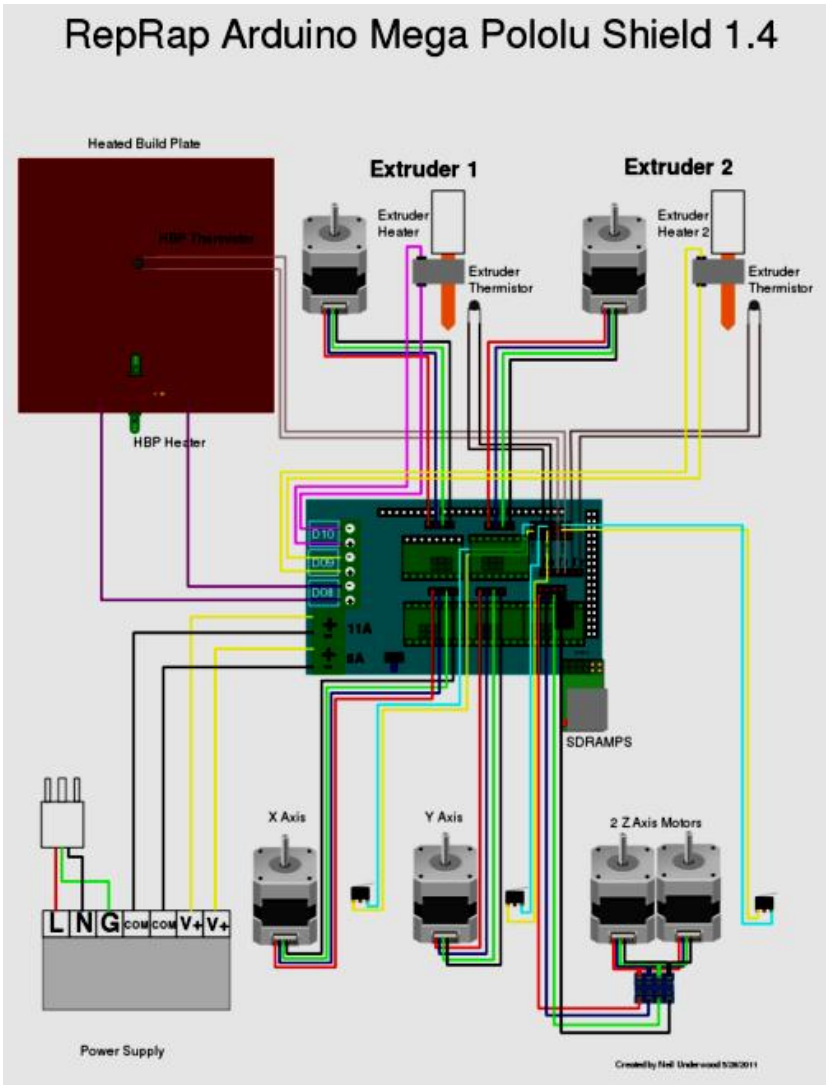


Рис. 5. **Схема комутації Ramps 1.4 [6]**

Дисплей (рис. 6) надає такі переваги:

- перевірка працездатності принтера;
- попередній підігрів панелі та хотенда, що є швидшим ніж автоматичний підігрів при запуску процесу друкування;
- можливість автономної роботи принтера (без комп'ютера), з умовою наявності карти пам'яті з завантаженими файлами для друку;
- моніторинг процесу друкування виробу;
- наявність клавіші екстреної зупинки;
- можливість калібрування принтера в автономному режимі чи під інший вид пластику;
- наявність у платі дисплея роз'єму під карту пам'яті (до 16 Гб), вмонтованого бузера (мікродинаміка) та зазначеної вище кнопки екстреної зупинки.

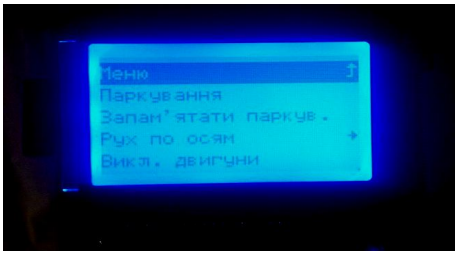


Рис. 6. **Зовнішній вигляд дисплея**

Робота з Prusa Mendel i2: особливості калібрування принтерів

Особливості калібрування механіки полягають у забезпеченні паралельності між валами осі X та двома шпильками у верхній частині принтера, тобто площиною у якій закріплені обидва мотори, що рухають вали по осі Z, а також у регулюванні нагріваючого дна так, щоб відстань від сопла екструдера до кожного з кутів дна була однаковою. Наступним кроком калібрування є встановлення кінцевих вимикачів у свої положення, зокрема закріпити вимикач по осі Z так, щоб коли механізм опустить екструдер до нагріваючого дна на відстань, що складає товщину аркушу А4, даний вимикач спрацював і повідомив, що екструдер розміщений у крайньому нижньому положенні, як показано на рисунку 7.

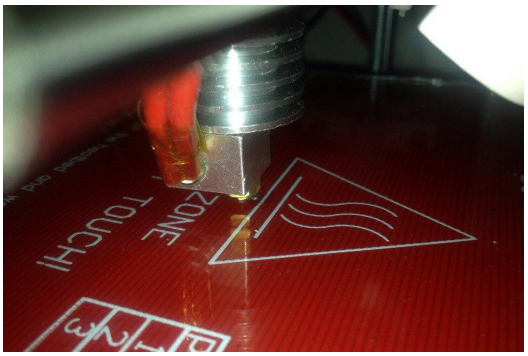


Рис. 7. **Калібрування механіки принтера**

Далі, аналогічно для інших осей, потрібно закріпити решту вимикачів. Для роботи з принтером, а також для створення G-кодів моделей, необхідних для друку, та завантаження цих кодів на комп'ютер ми використовували програму Repetier Host, вигляд вікна якої зображено на рисунку 8.

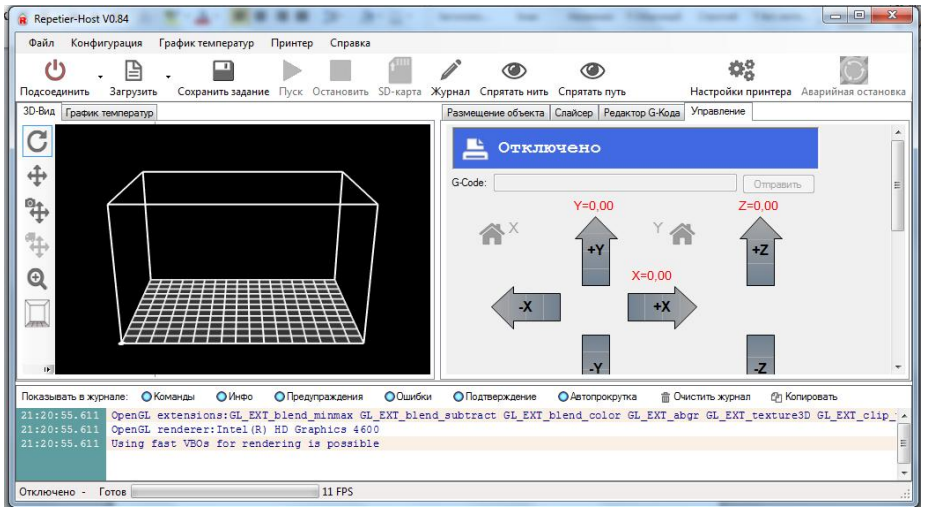


Рис. 8. **Вікно програми Repetier Host**

Використовувалось також середовище для програмування Arduino IDE останньої версії та програма для створення 3D моделей OpenScad. Програмне налаштування полягає у внесенні у створену в середовищі Arduino IDE програму всіх особливостей вашого принтера. В нашому випадку проводилась програмна інверсія всіх двигунів, кінцевих вимикачів, також редагування швидкості передачі даних, внесення розмірів області в якій може друкуватись певний виріб, внесення всіх швидкостей, інформація про керуючу плату, типи терморезисторів та температурні межі. Програма завантажується у Mega 2560 через програмне середовище Arduino. Також була проведена українізація меню принтера, оскільки у наявній програмі не існувало коду з українською мовою. Для нашої програми використано прошивку Marlin 1.1.0-RC3.

Після завантаження в плату відредагованої прошивки можна підключити принтер по USB кабелю до комп'ютера і починати працювати в Repetier Host-і. Програма має вмонтований слайсер – програму, що ділить задану 3D модель, завантажену в STL форматі, на шари, кожен з яких складається з

певних шляхів, по яких буде рухатись кінчик сопла екструдера, щоб надрукувати нам задану деталь. Усі ці шляхи записуються автоматично в G-код. Для завантаження G-коду моделі на карту пам'яті задля роботи принтера в автоматичному режимі потрібно користуватись окремим слайсером. Ми обрали Slic3r (рис. 9), оскільки він зручний у використанні, а також завантажується разом з попередньою програмою, як її підпрограма.

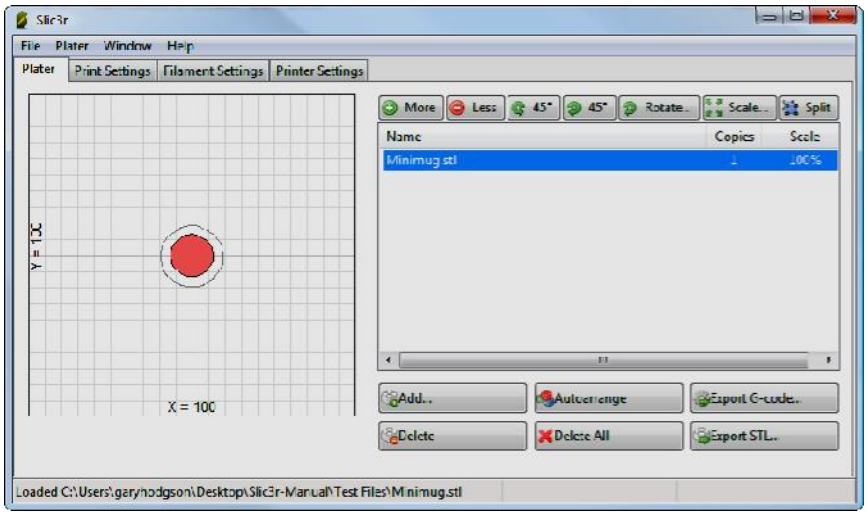


Рис. 9. **Вікно програми Slic3r**

При роботі з принтером в автоматичному режимі можна відредагувати: максимальну та мінімальну межі температури для екструдера та нагрівального дна, початкові положення відносно кінцевих вимикачів, швидкості та кроки моторів, діаметр прутка. Також можна налаштувати режими автоматичного підігріву для певних видів пластику, перевірити управління всіма системами в режимі реального часу та робочу здатність принтера.

Під час роботи з принтером на його дисплеї відображаються температури екструдера та нагрівального дна на даний момент часу і їх цільові значення, а також чи працює і з якою потужністю кулер для обдування хотенда, швидкість друку у відсотках, яку в режимі реального часу можна змінити (рис. 10).

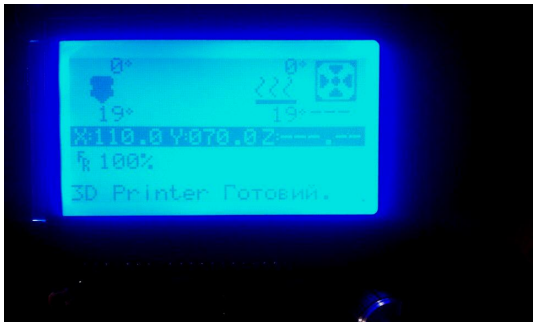


Рис. 10. **Інтерфейс дисплея принтера, головне меню**

Послідовність налаштування принтера:

1) увімкнути принтер у мережу та за допомогою меню запустити попередній обігрів для обраного виду пластику;

2) завантажити 3D-модель у STL форматі в Repetier Host чи Slic3r;

3) підготувати і запустити друк:

- якщо другий крок пов'язаний з Repetier Host-ом, то вміючи користуватись програмою, під'єднуємось до принтера, формуємо файл і запускаємо на виконання;

- якщо зі Slic3r-ом, то аналогічно формуємо файл, після чого завантажуємо його на карту пам'яті і поміщаємо її у слот під дисплеєм; вибираємо в меню файл з карти і активізуємо його;

4) прослідкувати, аби після запуску принтер виконав три важливих дії: 1) підігрів дно та хотенд; 2) виконав паркування; 3) перед друком «обвів» ділянку, на якій друкуватиметься виріб, тонким шаром пластику (цей процес забезпечує так зване «прочищення» екструдера).

Під час друку можуть виникнути певні проблеми:

- деталь відкріпилась від дна – потрібно скористатись одним з відомих способів: покрити поверхню перед друком або лаком для волосся, або розчином ацетону з пластиком, яким виконується друк;
- деталь «розпливлась» – потрібно зменшити максимальну температуру чи збільшити обдув хотенду;
- деталь вийшла не суцільною – збільшити температуру чи зменшити обдування.

На рисунку 11 представлено зразки 3D-конструкцій самостійно спроектованих в середовищі Repetier Host, та виготовлених на 3D-принтері.

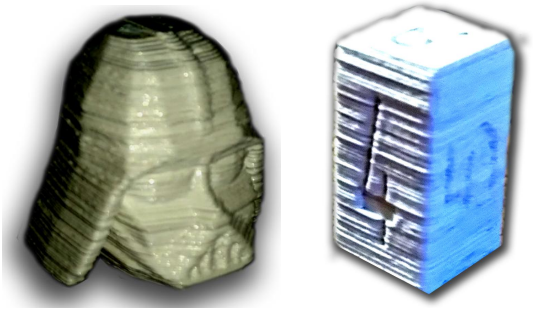


Рис. 11. **Зразки конструкцій, виготовлених методом 3D-моделювання**

Висновки. Спроектовано та виготовлено експериментальну модель 3D-принтера на основі конструкції Prusa Mendel i2. Представлено особливості технології виготовлення 3D-принтера, які доповнені візуальними зображеннями. Особливості калібрування принтера деталізовано до пропонованого програмного забезпечення – середовища Arduino IDE, програми для створення 3D моделей OpenScad, програми Repetier Host для створення G-кодів моделей, необхідних для друку та завантаження цих кодів на комп'ютер, програми Slic3r для задання шляхів руху сопла екструдера.

Проведено апробацію виготовленої власноруч конструкції, що дало можливість не лише послідовно узагальнити основні етапи створення 3D-принтера, але й виявити проблеми, які можуть з'явитися під час друку, та причини їх виникнення і шляхи подолання.

Загалом, у роботі обґрунтовано перспективи впровадження засобів 3D-моделювання та проведено апробацію технології, доступної для самостійного виготовлення та використання базового зразка 3D-принтера. Експериментально підтверджено можливості технології 3D-моделювання, доцільності та можливості самостійного проектування та виготовлення 3D-принтера.

1. 3D printing [Electronic resource]. – Mode of access: https://uk.wikipedia.org/wiki/3D-друк
2. 3D-печать на пороге новой промишленной революции [Electronic resource]. – Mode of access: http://innotech.kiev.ua/ru/news/3d--pechat-na-poroge-novoy-promishlennoyrevolyutsii#sthash.5BrAlFfc.JdQFhmtp.dpbs
3. 3D-принтер − что это такое и как он работает [Electronic resource]. – Mode of access: http://prostocomp.com/articles/43-apparatnoye-obespecheniye/117-3d-printer.html
4. Prusa Mendel iteration 2/ru [Electronic resource]. – Mode of access: http://reprap.org/wiki/Prusa_ Mendel_iteration_2/ru
5. PRINT CONFERENCE. KIEV. Выставка-конференция передовых технологий 3D-печати и сканирования. [Electronic resource]. – Mode of access: http://3dprintconf.com.ua
6. RAMPS 1.4 [Electronic resource]. – Mode of access: http://reprap.org/wiki/RAMPS_1.4

УДК 658.562
Філіппова М.В. к.т.н, доцент, Данилюк О.А, магістрант, Демченко М.О., асистент
Національний технічний університет України «Київський політехнічний інститут»

КЕРУВАННЯ ТРУДОВИМИ РЕСУРСАМИ НА ВИРОБНИЦТВІ ПРИ ВПРОВАДЖЕННІ СИСТЕМИ МЕДЖМЕНТУ ЯКОСТІ

Філіппова М.В., Данилюк О.А, Демченко М.О. Керування трудовими ресурсами на виробництві при впровадженні системи менеджменту якості. Розглянуто існуючі підходи до керування трудовими ресурсами на підприємствах. Запропоновано використання нової системи керування трудовими ресурсами, яка основана на впровадженні системи менеджменту якості відповідно до міжнародних стандартів. Розглянуто показники оцінки стану та ефективності процесу керування трудовими ресурсами, які дозволяють побудувати більш продуктивні форми взаємодії між трудовими ресурсами та керівництвом. Розроблено програмне забезпечення керування трудовими ресурсами на підприємстві.
Ключові слова: система менеджменту якості, трудові ресурси, персонал, інформаційні технології, виробництво.

Филиппова М.В., Данилюк А.А, Демченко М.А. Управление трудовыми ресурсами на производстве при внедрении системы менеджмента качества. В статье рассмотрены существующие подходы к управлению трудовыми ресурсами на предприятиях. Предложено использование новой системы управления трудовыми ресурсами, которая основана на внедрении системы менеджмента качества в соответствии с международными стандартами. Рассмотрено показатели оценки состояния и эффективности процесса управления трудовыми ресурсами, которые позволяют построить более продуктивные формы взаимодействия между трудовыми ресурсами и руководством. Разработано программное обеспечение управления трудовыми ресурсами на предприятии.
Ключевые слова: система менеджмента качества, трудовые ресурсы, персонал, информационные технологии, производство.

Filippova M.V., Данилюк A.A, Demchenko M.O. Management of human resources in the production of the introduction of quality management system. The article deals with existing approaches to the management of human resources in enterprises. The use of a new system of human resources management that is based on the implementation of quality management system in accordance with international standards. It was considered assessment of the performance and efficiency of the management of human resources that allow you to build a more productive forms of cooperation between labor and management. Human resources management software was developed in the enterprise.
Keywords: quality management system, human resources, personnel, information technology, production.

Вступ. Сучасна ситуація в Україні, яка склалася з керуванням трудовими ресурсами є результатом існування проблем та суперечностей, економічної діяльності, яка орієнтується на короткий термін, незбалансованістю трудових ресурсів та робочих місць, що тривалий час не вирішувались та продовжують загострюватися.

Для розв'язання даної ситуації на підприємствах необхідно впроваджувати систему менеджменту якості в процес керування трудовими ресурсами (КТР), яка вимагає ретельного дослідження, оскільки в ньому беруть участь як внутрішні суб'єкти (засновники, керівники, профспілка), так і зовнішні (органи державної влади, кадрові агентства, споживачі).

При впроваджені інформаційних систем управління проектами на підприємствах, слід пам'ятати, що окрім використання інформаційних систем потребує певних змін в процесах управляння. Основним базисом підприємства є його трудові ресурси та їх професійний рівень.

Використання процесного підходу до управління ефективністю діяльності підприємства та його трудовими ресурсами дозволяє побудувати найбільш продуктивні форми організації взаємодії персоналу та керівництва, а також якісно перетворити динамічні та структурні характеристики трудового потенціалу підприємства й основі цього підвищити інноваційну сприйнятливість продукції. Тобто на підприємстві повинна функціонувати система управління якістю продукції, що представляє собою організаційну структуру, яка чітко розподіляє відповідальність, процедури, процеси і ресурси, необхідні для керування якістю.

Постановка наукової проблеми. Метою даної роботи є аналіз існуючих підходів до КТР з метою організації системи КТР на виробництві з використанням системи менеджменту якості, як один зі шляхів підвищення ефективності управління проектами підприємства. Розробка програмного забезпечення керування персоналом на виробництві.

Аналіз існуючих підходів до КТР. На шляху до підвищення ефективності КТР є багато перешкод як об'єктивних, дія зовнішніх чинників, так і суб'єктивних. Суб'єктивні перешкоди полягають це помилки керівників при роботі з трудовими ресурсами. Також ефективність КТР залежить від стану

ринку праці. Як відомо, ринок праці — це система соціально-економічних відносин, пов'язаних з попитом і пропозицією робочої сили; сфера взаємодії покупців і продавців праці; механізм, який в придатних економічних умовах саморегулюється завдяки узгодженню ціни та якісних і кількісних характеристик праці, а в умовах кризової ситуації та трансформації економіки певні його сегменти регулюються владою. Запропоноване визначення вказує не тільки на вплив соціальних та економічних факторів на ринок праці, а й на зміну в його регулюванні відповідно напрямку їх динаміки. Це дозволяє охопити більш широкий спектр модифікаційних структур аналізованої категорії.

Ринок праці надає руху всім іншим ринкам, всім іншим ресурсам, тому що тут формується і розподіляється за професіями, підприємствами, регіонами і галузями та включається в дію найбільш важливий національний ресурс – трудовий. По-перше, через нього забезпечуються зайнятість економічно активного населення, його включення у сферу виробництва та послуг, створюється можливість одержання необхідного заробітку [1,2]. По-друге, через ринок йде комплектування підприємств потрібною робочою силою, у необхідних кількостях і за належною якістю. Ринок показує, які професії потрібні, а які є в надмірності, на що мають звертати увагу безробітні у своїй підготовці, перепідготовці, розширенні наявних знань і умінь для одержання роботи [4]. Тобто, ринок праці є важливим джерелом інформації. По-третє, ринок через конкуренцію найманих працівників стимулює їх до розширення професійної майстерності, сприяє підвищенню їх кваліфікації й універсалізації [3,5]. По-четверте, ринок праці регулює потоки трудових ресурсів, що складаються на ньому, забезпечує розподіл і перерозподіл трудових ресурсів в зв'язку зі структурними змінами в економіці [3].

Виклад основного матеріалу й обґрунтування отриманий результатів дослідження.

Ефективне використання трудових ресурсів є однією із найважливіших завдань керівника, навіть якщо за них відповідають спеціальні служби або відділи, КТР, а також кадрова політика, прийняття важливих рішень, усе це покладається на функції керівника.

Головною метою управління персоналом є створення умов для нормального функціонування, розвитку та ефективного використання кадрового потенціалу організації. Процес управління персоналом включає такі підпроцеси, як кадрове планування, залучення та відбір персоналу, набір і звільнення, навчання і розвиток, мотивацію і винагороду, організацію діяльності, оцінку та атестацію кадрів.

Центральним підпроцесом керування персоналом є кадрове планування, що взаємодіє з усіма підпроцесами і забезпечує:

- організацію необхідним і достатнім кадровим складом;
- підбір працівників, які відповідають потребам бізнесу;
- необхідний рівень кваліфікації працівників і розвиток ТР;
- активна участь працівників в діяльності організації.

Планування трудових ресурсів (ТР) повинно бути об'єднано з основними планами організації і скоординовано з виконанням таких функцій, як переміщення кадрів, навчання, аналіз роботи і розвиток (рис. 1).

Одним з основних завдань кадрового планування є визначення потреби в персоналі, що розуміється як необхідність кількісного і якісного підбору склад персоналу, який визначається відповідно до обраної стратегії розвитку підприємства. В якості ресурсу виступають працівники підприємства з досягнутими рівнями компетенції, бажаннями, мотиваціями, устремліннями.

Слід особливо відмітити, що одним з основних напрямків управління персоналом є залучення і відбір персоналу, який визначає якість персоналу організації для успішного бізнесу і ефективності функціонування організації після реалізації проекту реструктуризації. Під залученням персоналу маються на увазі різноманітні способи використання джерел персоналу (зовнішні і внутрішні).

Зауважимо, що залучення і відбір персоналу повинен супроводжуються оцінкою кандидатів, включаючи як зовнішніх кандидатів, так і працівників організації, що плануються до переміщень.

Завдання служби по КТР, яка здійснює оцінку кандидатів при прийомі на роботу, має полягати в тому, щоб відібрати такого працівника, який в змозі досягти очікуваного організацією результату. Оцінка при прийомі - це одна з форм попереднього контролю якості людських ресурсів організації. Центральним показником при оцінці працівника є рівень його компетентності, який розуміється ширше, ніж традиційно.

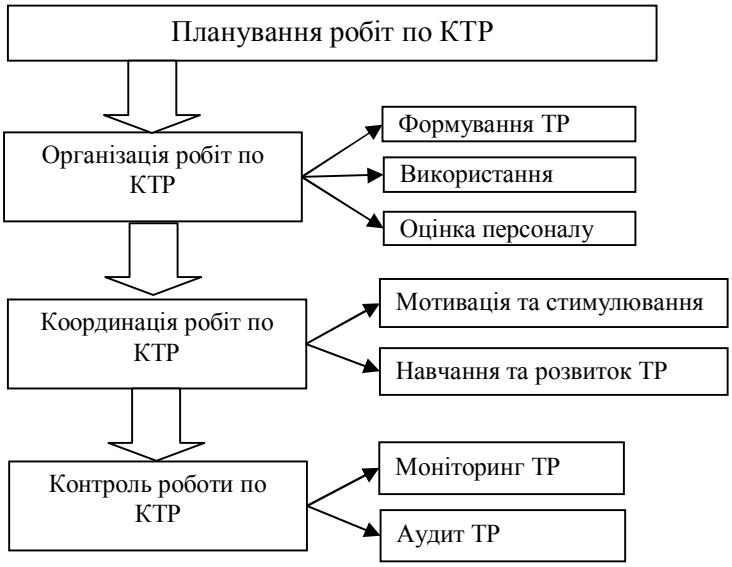


Рис. 1 Процес КТР

Ефективність діяльності персоналу істотно залежить від його компетентності, тому одним з основних завдань управління персоналом є управління компетенцією ТР - показник, що аналізується і керований в рамках практично всіх підпроцесів управління персоналом[1]:

• кадрового планування - визначення якісної і кількісної потреби організації в персоналі;

• залучення та відбору персоналу - визначення вимог до компетенції кандидатів і відбір відповідних працівників;

• навчання і розвитку персоналу - визначення шляхів і способів підвищення компетентності персоналу;

• мотивації персоналу - вироблення шляхів мотивації і стимулювання потреб працівників щодо підвищення компетентності;

• оцінка і атестація персоналу - проведення ефективного аналізу компетентності персоналу відповідно до вимог бізнесу.

Можна представити КТР як про процеси оцінки, контролю, організації підвищення компетенції шляхом навчання, підвищення кваліфікації, перепідготовки персоналу, прийому на роботу висококомпетентних працівників.

В якості базису для формування системи КТР використано ДСТУ ISO 9001:2009 «Системи управління якістю. Вимоги», де сформульовані основні вимоги до людських ресурсів підприємства. Серед основних факторів впливу на умови КТР них можна виділити потребу в персоналі в потрібній кількості і необхідної якості (тобто певну чисельність працівників, які мають необхідні професійні компетенції), постійне вдосконалення професійних знань співробітників, високий рівень трудової дисципліни, роботу згуртованої команди, ставлення до праці як до першої життєвої необхідності та ін. Таким чином, видно, що варіанти входу в процес управління персоналом можуть істотно відрізнятися. Вибір пріоритетів, як правило, залежить від рівня розвитку організації, обраної нею концепції управління бізнесом і концепції управління персоналом [7].

Нормативний рівень трудового потенціалу показує, якими кількісними і якісними характеристиками, а також інноваційними, мотиваційними та іншими можливостями повинен володіти персонал організації для вирішення планових виробничих завдань і досягнення цілей компанії [6]. Отже, виходом з процесу можна вважати реалізований рівень трудового потенціалу, що відповідає вимогам підприємства та виробництва.

Для управління персоналом керівнику організації необхідні такі ресурси:

• фахівці з КТР;

• інформація про об'єкт управління;

• матеріально-технічні ресурси для оснащення служби КТР;

• фінансові ресурси для приведення трудового потенціалу організації у відповідність до вимог менеджменту якості.

Розробка регламенту процесу управління кадрами передбачає приведення у відповідність певним умовам або створення необхідної документації організаційного, організаційно-методичного, організаційно-розпорядчого, технічного, нормативного, техніко-економічного та економічного характеру, а також формування нормативно-довідкових матеріалів, що встановлюють норми, правила, методи управління персоналом.

Залежно від масштабів організації, обсягу і структури діяльності з управління кадрами в рамках процесу менеджменту персоналу вибудовується розгалужена система показників.

Таким чином, впровадження системи менеджменту якості СМЯ в процес управління персоналом підприємства вимагає ретельного дослідження, оскільки в ньому беруть участь як внутрішні суб'єкти (засновники, керівники, профспілка, комісії по трудових спорах, HR-менеджери), так і зовнішні (органи державної влади, контрольно-ревізійні служби, кадрові агентства, споживачі), та обов'язки між ними розділити непросто в силу відмінності інтересів (рис. 2).

Рис. 2 – Покращення процесу КТР

На основі процесного підходу КРТ розроблено програмне забезпечення (ПЗ) керування трудовими ресурсами на виробництві при впровадженні системи менеджменту якості. Інтерфейс «Системи керування персоналом» зображено на рис. 3.

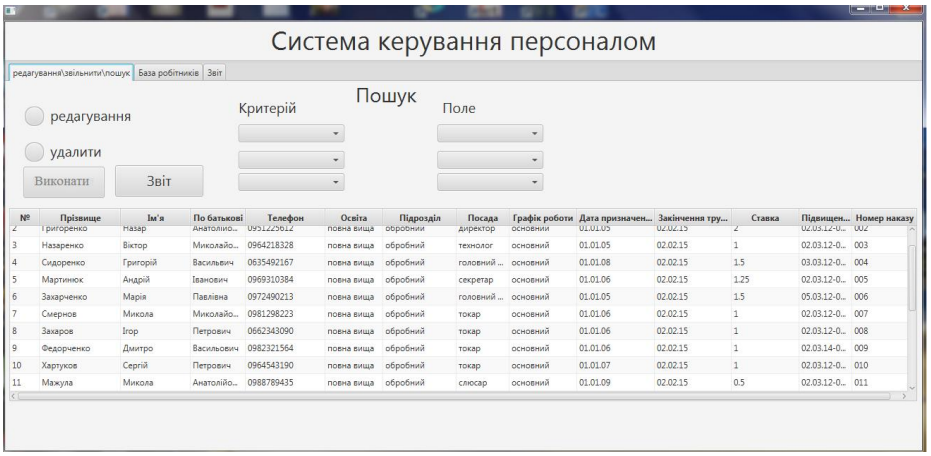


Рис. 3 Інтерфейс «Системи керування персоналом»

Дане ПЗ дає можливість слідкувати за роботою персоналу на виробництві, рівнем освіченості, кваліфікації, терміном роботи та терміном проходження підвищення кваліфікації. Включає в себе створення бази робітників підприємства, подальшого редагування цієї бази та видалення з її непотрібної інформації. На рис. 3 представлено інтерфейс ПЗ « Система керування персоналом».

Висновки

Таким чином, використання процесного підходу до управління ефективністю діяльності підприємства та його трудовими ресурсами дозволяє вибудувати найбільш продуктивні форми організаційної взаємодії персоналу, керівництва, власників компанії інвесторів, а також якісно перетворити динамічні і структурні характеристики трудового потенціалу підприємства та на основі цього підвищити його інноваційну сприйнятливість.

На додаток до агітаційних заходів необхідно організувати цільове навчання і підвищення кваліфікації співробітників організації. Для цього слід сформувати перелік цільових напрямків, за якими потрібно найближчим часом навчити персонал, що дозволить до моменту початку впровадження СМЯ мати якісний кадровий потенціал, тобто фахівців, здатних зробити введення і налагодження механізмів підвищення якості на місцях. При цьому необхідно орієнтуватися на подальшу побудову нової СМК, відповідно до якої нагляд за її реалізацією здійснюється найбільш кваліфікованими менеджерами різних рівнів управління.

Важливим елементом підготовки процесу впровадження СМЯ є актуалізація нормативної бази по основних виробничих і бізнес-процесів підприємства та приведення нормативної і технічної документації у відповідність сучасним вимогам, стандартам і умовам функціонування структурних підрозділів організації. Введення процедур підвищення якості праці часто вимагає зміни корпоративної культури підприємства, в першу чергу культури і кваліфікації керуючого персоналу середньої ланки. Саме від цієї групи фахівців залежить, чи вдасться залучити рядових співробітників в процес тотального вдосконалення технологій і контролю якості, змусити їх усвідомити, що підвищення якості веде до зниження витрат. Навчання вдосконалення якості та оцінка ефективності управління підприємством припускають дотримання таких принципів, як безперервність і залучення членів суспільства на всіх етапах навчання, а також виробничої діяльності, що веде до появи такого поняття, як самонавчальна компанія, тобто організація, в якій процес навчання є постійним і поступово стає своєрідною потребою персоналу.

На необхідність проведення заходів по підвищенню кваліфікаційного рівня персоналу може вплинути реалізація різних функцій інформаційної системи управління проектами. Процес впровадження розробленого ПЗ потребує системного підходу, що передбачає планування комплексу робіт і контроль за їх здійсненням. Використання інформаційних технологій в керуванні персоналом на підприємстві потребує проведення попередніх робіт по впровадженню даного ПЗ з метою автоматизації процесів ефективного керування трудовими ресурсами.

1. Бабушкина, Е.А. Управление эффективностью компании, / Е.А. Бабушкина http://www.cfin.ru/management/strategy/competit/efficiency_factors.shtml)
2. Шипунов, В.Г.. Основы управленческой деятельности./ В.Г. Шипунов, Е.Н. Кишкель– М.: Высшая школа, 1999.

3. Хучек, М. Стратегия управления трудовым потенциалом предприятия. / М .Хучек – М.: Прогресс-Универс, 2000. – 315 с.
4. Шмелева, А.Н. Оценка эффективности управления предприятиями./ А.Н. Шмелева— Пенза: Информационно-издательский центр Пензенского государственного университета, 2006. — 160 с.20
5. Ховард, К. Принципы менеджмента: управление в системе цивилизованного предпринимательства: учеб. пособие./ К/ Ховард, Э. Коротков– М.: ИНФРА-М, 1996.
6. Кибанов А.Я. Мотивация и стимулирование персонала: взаимосвязь понятий / Кадровик. Кадровый менеджмент, № 6, 2008.
7. ДСТУ ISO 9001: 2009 (ISO 9001: 2008, IDT) Национальный стандарт Украины. Системы управления качеством. Требования
8. Иллюстрованный самоучитель по Java [Електронний ресурс]. – Режим доступу: http://computers.plib.ru/programming/Java/
9. Ситник В.Ф**.,** Писаревська Т.А**.,** Єрьоміна Н.В**.,** Краєва О.С**.** Основи інформаційних систем: Навч. посібник / За ред. В.Ф. Ситника. — К.: КНЕУ, 1997. — 252 с.

УДК 004.421.2
Гришанович Т.О. к.ф.-м.н., доцент., Єрмейчук С. Ю.студент
СНУ імені Лесі Українки

АНАЛІЗ АЛГОРИТМІВ ПОШУКУ

Гришанович Т.О., Єрмейчук С.Ю. Аналіз алгоритмів пошуку. Важливе місце у сучасні теорії алгоритмів відводиться алгоритмам пошуку, оскільки задача відшукання даних (пошук даних за заданим ключем, пошук підпослідовності у послідовності) є однією із тих задач, що має прикладне застосування. Застосування того чи іншого алгоритму пошуку для вирішення конкретної задачі є досить складною проблемою, вирішення якої потребує не лише досконалого володіння саме цим алгоритмом, але й всебічного розглядання того чи іншого алгоритму, тобто визначення усіх його переваг і недоліків.
Ключові слова: алгоритм; складність алгоритмів; часова характеристика; машина Тьюрінга; асимптотична часова складність; продуктивність алгоритму.

Гришанович Т.О., Ермейчук С.Ю. Анализ алгоритмов поиска. Важное место в современной теории алгоритмов отводится алгоритмам поиска. Задача поиска связана с нахождением заданного значения, называемого ключом поиска (*search key*), среди заданного множества. Существует огромное количество алгоритмов поиска, так что есть из чего выбирать. Их сложность варьируется от самых простых алгоритмов поиска методом последовательного сравнения, до чрезвычайно эффективных, но ограниченных алгоритмов бинарного поиска, а также алгоритмов, основанных на представлении базового множества в иной, более подходящей для выполнения поиска форме. Последние из упомянутых здесь алгоритмов имеют особое практическое значение, поскольку применяются в реально действующих приложениях, выполняющих выборку и хранение массивов информации в огромных базах данных.
Ключевые слова: алгоритм; сложность алгоритмов; временная характеристика; машина Тьюринга; асимптотическая временная сложность; производительность алгоритма.

Hryshanovych T.O.,Yermeychuk S. Y. Analysis of the search of algorithms. Important role in the modern theory of algorithms given to algorithms search because the task of finding data (search data for a given key, search subsequence of the sequence) is one of those tasks that has practical applications. The use of a search algorithm to solve a specific problem is quite complex problem whose solution requires not only fluency by this algorithm, but also comprehensive consideration of an algorithm that determine all its advantages and disadvantages. In computer science, a search algorithm is an algorithm for finding an item with specified properties among a collection of items which are coded into a computer program, that look for clues to return what is wanted. The items may be stored individually as records in a database; or may be elements of a search space defined by a mathematical formula or procedure, such as the roots of an equation with integer variables; or a combination of the two, such as the Hamiltonian circuits of a graph. Algorithms for searching virtual spaces are used in constraint satisfaction problem, where the goal is to find a set of value assignments to certain variables that will satisfy specific mathematical equations and inequations. They are also used when the goal is to find a variable assignment that will maximize or minimize a certain function of those variables.
Keywords: algorithm; complexity of algorithms; temporal characteristics; Turing machine; asymptotic time complexity; the productivity of the algorithm.

Постановка наукової проблеми та її значення. Важливою проблемою сучасного програмування є його математизація, розробка формалізованих мов проектування алгоритмів і програм, а також їх абстрактних моделей. Засоби проектування, аналізу алгоритмів є особливо актуальними у зв'язку з важливими процесами комп'ютеризації й автоматизації діяльності суспільства. До таких засобів, зокрема, відносяться алгебри алгоритмів, орієнтовані на вирішення проблем формалізації, обґрунтування правильності, покращення алгоритмів (за обраними критеріями).

Мета статті є аналіз алгоритмів відшукання даних за заданим ключем. У відповідності до поставленої мети визначено такі **завдання дослідження**: розглянути алгоритми пошуку даних за заданим ключем; дослідити основні поняття теорії складності алгоритмів; вивчити методику оцінки часової складності алгоритмів; зробити порівняльний аналіз часової складності алгоритмів пошуку за заданим ключем.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження. Пошук – одна з найпоширеніших у програмуванні дій. Він же являє собою завдання, на якому можна випробовувати різні СТРУКТУРИ даних у міру їх появи. Існує кілька основних варіацій цієї задачі, і для них створено багато різних алгоритмів, наприклад (у статичних структурах даних): лінійний пошук, пошук діленням навпіл, пошук у таблиці, прямий пошук рядка, пошук у рядку (алгоритми Кнута-Моріса-Пратта і Боуера-Мура).

Послідовний (лінійний) пошук. Найпростішим методом пошуку елемента, який знаходиться в неврегульованому наборі даних, за значенням його ключа є послідовний перегляд кожного елемента набору, який продовжується до тих пір, поки не буде знайдений потрібний елемент. Якщо переглянуто весь набір, і елемент не знайдений – значить, шуканий ключ відсутній в наборі. Цей метод ще називають методом повного перебору. Для послідовного пошуку в середньому потрібно *N/2* порівнянь. Таким

чином, порядок алгоритму – лінійний – *O(N)*. Програмна ілюстрація лінійного пошуку в неврегульованому масиві приведена в наступному прикладі, де *a* – початковий масив, key – ключ, який шукається; функція повертає індекс знайденого елемента.

```
int LinSearch(int *a, int key)
{
    int i = 0;
    while ( (i<N) && (a[i] != key) )
        i++;
    return i;
}
```

Якщо елемент знайдено, то він знайдений разом з мінімально можливим індексом, тобто це перший з таких елементів. Рівність *i=N* засвідчує, що елемент відсутній.

Єдина модифікація цього алгоритму, яку можна зробити, – позбавитися перевірки номера елементу масиву в заголовку циклу (*i<N*) за рахунок збільшення масиву на один елемент у кінці, значення якого перед пошуком встановлюють рівним шуканому ключу – key – так званий „бар'єр" [1].

```
int LinSearch(int *a, int key)
{
    a[N] = key;
    i = 0;
    while (a[i] != key)
        i++;
    return i;  }     // i<N – повернення номера елемента
```

Поняття алгоритмічно нерозв'язуваних проблем звичайно важливе і практично значуще. Розв'язування багатьох задач, як це не парадоксально, пов'язане саме з алгоритмічною нерозв'язаністю. Але з розвитком обчислювальної техніки все більше уваги стали приділяти не просто наявності алгоритму рішення деякого класу задач, а й ефективності цих алгоритмів. При використанні алгоритмічних моделей (фінітний комбінаторний процес машини Поста, абстрактну машину Тюрінга чи машини з довільним доступом) не виникає проблем із обмеженням ресурсів (стрічка є нескінченною, а час необмеженим). Але в реальних ЕОМ і пам'ять, і час обмежені. Ось чому замало знати про існування того чи іншого алгоритму – необхідно мати уявлення про необхідні ресурси, а саме: чи може певна програма (алгоритм) розміститися в пам'яті ЕОМ; чи дасть вона результат за належний час.

Дослідженням цих питань і займається розділ теорії алгоритмів – аналіз складності алгоритмів. *Складність алгоритмів* – кількісна характеристика, яка визначає час, що необхідний для виконання алгоритму (часова складність), і об'єм пам'яті, необхідний для його розміщення (ємнісна складність). Часова та ємнісна складність тісно пов'язані між собою. Обидві є функціями від розміру вхідних даних. Складність розглядається, за звичай, для машинних алгоритмічних моделей (ЕОМ), оскільки в них час і пам'ять присутні у явному вигляді. Часова характеристика (фізичний час виконання) складності алгоритму –це величина $\tau^* t$, де t – кількість дій алгоритму (елементарних команд), а τ – середній час виконання однієї операції (команди). Кількість команд t визначається описом алгоритму в певній алгоритмічній моделі і не залежить від фізичної реалізації цієї моделі. Середній час τ – величина фізична і залежить від швидкості обробки сигналів в елементах і вузлах ЕОМ. Ось чому об'єктивною математичною характеристикою складності алгоритму в певній моделі є кількість команд t.

Ємкісна характеристика складності алгоритмів визначається кількістю комірок пам'яті, що використовуються в процесі його обчислення. Ця величина не може перевищувати кількість дій t, що перемножена на певну константу (кількість комірок, які використовуються при виконанні однієї команди). В свою чергу, кількість дій t може перевищувати об'єм пам'яті (за рахунок використання повторень в одних і тих же комірках). До того ж проблема пам'яті технічно долається легше, аніж проблема швидкодії, яка має фізичне обмеження – швидкість розповсюдження фізичних сигналів (*300 км/с*). Ось чому часова складність вважається більш суттєвою характеристикою алгоритму. Слід зазначити, що часова складність алгоритму не є постійною величиною і залежить від розмірності задачі (об'єм пам'яті для зберігання даних) – кількість комірок для різних даних. Отже, *складність алгоритму* – функція, значення якої залежить від розмірності n даних задачі.

Зазвичай говорять, що час виконання алгоритму або його часова складність має порядок $T(n)$ від вихідних даних розмірністю n. Одиниця виміру $T(n)$ точно не визначена, її розуміють як кількість кроків, що виконані на ідеалізованому комп'ютері [3,4]. У більшості випадків, при визначенні часової

складності алгоритму *T(n),* розуміють максимальний час виконання алгоритму по всім вхідним даним, так звану часову складність алгоритму *Tmax(n)* у найгіршому випадку. Також розглядають і *Tavg(n)* як середній (в статистичному сенсі) час виконання алгоритму на всіх вихідних даних розмірністю n. Хоча *Tavg(n)* є досить об'єктивною мірою визначення часової складності, проте часто неможливо стверджувати рівнозначність всіх вхідних даних. Таким чином, на практиці середній час виконання алгоритму знайти складніше, ніж час у найгіршому випадку. Ось чому, зазвичай, використовують час у найгіршому випадку як міру часової складності алгоритму *T(n)*. Обчислення часової складності алгоритму *Tmin(n)* у найсприятливішому випадку хоча і буде предметом нашої уваги, проте ми повинні розуміти, що значення цієї функції не повинно суттєво впливати на вибір того чи іншого алгоритму. Це пояснюється декількома причинами: по-перше, знаючи час роботи в найгіршому випадку, можна гарантувати, що виконання алгоритму закінчиться за якийсь час, навіть не знаючи, якого вигляду буде вихідна послідовність; по-друге, на практиці «погані» входи (для яких час роботи близький до максимуму) можуть часто траплятися [2,5].

Формальний опис машини Тьюрінга. Машина Тьюрінга (МТ) складається з: нескінченної стрічки, що поділена на комірки. В кожній комірці може бути записаний один із символів кінцевого алфавіту $A=\{a_0, a_1, a_2, \ldots, a_m\}$, що називається зовнішнім алфавітом (ЗА). Для кожної машини Тьюрінга можна задати власний ЗА; керуючого пристрою, що може знаходитися в одному із внутрішнього станів $Q=\{q_1, q_2, \ldots, q_n\}$. Кількість елементів Q визначає об'єм "внутрішньої пам'яті" машини Тьюрінга. У множині Q вирізняють два спеціальні стани: початковий q_1 та кінцевий q_z (або ! – знак оклику), де z – не числовий індекс, а мнемонічна ознака кінця. Таким чином, машина Тьюрінга починає роботу в стані q_1 та, потрапивши в q_z зупиняється; каретки, що переміщуючись вздовж стрічки, може: записувати в комірку символ зовнішнього алфавіту; зміщуватися на комірку праворуч/ліворуч чи залишатися на місці. Функціонування машини Тьюрінга можна описати так: в залежності від внутрішнього стану машини Тьюрінга (q_i) та символу зовнішнього алфавіту на стрічці (a_j) відбувається запис нового символу зовнішнього алфавіту (a'_j), зміщення каретки (d) та перехід до нового внутрішнього стану (q'_i). Таким чином, робота машини Тьюрінга визначається системою команд вигляду: $q_i\, a_j \rightarrow q'_i\, a'_j\, d$ [2,3,6].

Машина з довільним доступом до пам'яті (RAM) Аналіз алгоритму полягає в тому, щоб передбачити потрібні для його виконання ресурси. Іноді оцінюється потреба в таких ресурсах, як пам'ять, пропускна здатність мережі або необхідне апаратне забезпечення, але найчастіше визначається час обчислення. Шляхом аналізу деяких алгоритмів, призначених для розв'язку однієї та тієї самої задачі, можна легко обрати найбільш ефективний з них. В процесі такого аналізу може також виявитись, що декілька алгоритмів приблизно рівноцінні, а всі інші варто відкинути.

З урахуванням того, що алгоритми реалізуються у вигляді комп'ютерних програм, в більшості випадків в якості технології аналізу алгоритмів буде використовуватись модель узагальненої однопроцесорної машини з довільним доступом до пам'яті (Random-Access Machine – RAM). В цій моделі команди процесора виконуються послідовно; операції, які виконуються одночасно, відсутні. У цю модель входять ті команди, які зазвичай можна знайти в реальних комп'ютерах: арифметичні (додавання, віднімання, добуток, ділення, обчислення остачі від ділення, наближення дійсного числа найближчим більшим чи найближчим меншим цілим числом), операції переміщення даних (завантаження значення в пам'ять, копіювання) та керуючі операції (умовне та безумовне галуження, виклик підпрограми і повернення з неї). Для виконання кожної такої інструкції потрібно певний фіксований проміжок часу. В моделі RAM є цілочисловий тип даних та тип чисел з плаваючою комою. Також передбачається, що є верхня межа розміру одного слова даних. У моделі RAM, яка розглядається, не моделюється ієрархія пристроїв пам'яті, які сьогодні розповсюджені у звичайних комп'ютерах. Таким чином, кеш та віртуальна пам'ять відсутні у RAM. Моделі, які включають в себе таку ієрархію, набагато складніше моделі RAM, тому вони можуть ускладнити роботу. Окрім цього, аналіз, який заснований на моделі RAM, зазвичай чудово прогнозує продуктивність алгоритмів, які виконуються на реальних машинах.

Аналіз навіть простого алгоритму в моделі RAM може вимагати значних зусиль. В число необхідних математичних інструментів може увійти комбінаторика, теорія ймовірностей, алгебраїчні перетворення та здатність ідентифікувати найбільш важливі доданки в формулі. Оскільки поведінка алгоритму може відрізнятись для різних наборів вхідних значень, буде потрібна методика обліку, яка описує поведінку алгоритмів за допомогою простих та зрозумілих формул [1].

Методика оцінки часової складності алгоритмів. Один із способів визначення часової ефективності алгоритмів полягає в наступному: на основі даного алгоритму потрібно написати програму

і виміряти час її виконання на певному комп'ютері для вибраної множини вхідних даних. Хоча такий спосіб популярний і, безумовно, корисний, він породжує певні проблеми. Визначений час виконання програми залежить не тільки від використаного алгоритму, але й від архітектури і набору внутрішніх команд даного комп'ютера, від якості компілятора, і від рівня програміста, який реалізував даний алгоритм. Час виконання також може суттєво залежати від вибраної множини тестових вхідних даних. Ця залежність стає очевидною при реалізації одного й того ж алгоритму з використанням різних комп'ютерів, різних компіляторів, при залучені програмістів різного рівня і при використанні різних тестових даних. Щоб підвищити об'єктивність оцінки алгоритмів, учені, які працюють в галузі комп'ютерних наук, прийняли *асимптотичну часову складність* як основну міру ефективності виконання алгоритму [5]. У більшості випадків алгоритми реалізуються програмами, які записані операторами мов програмування високого рівня. Отже, виникає необхідність запису цих алгоритмів в термінах „кроків", кожен з яких повинен виконуватися за скінчений фіксований час після трансляції їх у машинну мову будь-якої ЕОМ. Проте, точно визначити час виконання будь-якого кроку алгоритму дуже важко, адже цей час залежить не тільки від „природи" самого кроку, а й від процесу трансляції і машинних інструкцій певного комп'ютера. Ось чому замість точної оцінки ефективності алгоритму, що придатна (актуальна, валідна) для певної обчислювальної системи, прийнято використовувати менш точну, але більш загальну асимптотичну часову складність як основну міру ефективності виконання алгоритму.

Для опису швидкості росту часової складності алгоритмів використовується Θ-символіка („тета-символіка"). Наприклад, коли говорять що час виконання $T(n)$ деякого алгоритму має порядок $\Theta(n^2)$, тобто $T(n)=\Theta(n^2)$, то вважають, що існують такі константи c_1, $c_2>0$ і таке число n_0, що $c_1n^2\le T(n)\le c_2n^2$ для всіх $n\ge n_0$. Оскільки, як було зазначено вище, складність алгоритму – це функція, значення якої залежить від розмірності n даних задачі, то запис $T(n)=\Theta(n^2)$ можна подати у вигляді $f(n)=\Theta(n^2)$. Враховуючи різні значення пропорційності для n різних алгоритмів, точнішим буде використання запису $f(n)=\Theta(g(n))$, де $g(n)$ – деяка функція від n. Зміст цього запису полягає у тому , що існують такі константи $c_1,c_2>0$ і таке число n0, що $0\le c_1g(n)\le f(n)\le c_2g(n)$ для всіх $n\ge n_0$. Якщо для деякого алгоритму $f(n)=\Theta(g(n))$, то функцію g(n) вважають асимптотично точною оцінкою його складності.

Для пояснення скористаємося прикладом. Нехай часова складність деякого алгоритму $T(n)=(1/2)n^2-3n$. Перевіримо, що цей алгоритм має швидкість росту $\Theta(n^2)$, тобто $(1/2)n^2-3n=\Theta(n^2)$. Для цього зазначимо константи c_1,c_2 і число n_0, так щоб нерівність $c_1n^2\le(1/2)n^2-3n\le c_2n^2$ виконувалася для всіх $n\ge n_0$. Розділимо нерівність на n^2 і отримаємо $c_1\le 1/2-3/n\le c_2$. Таким чином, для виконання другої нерівності достатньо прийняти $c_2=1/2$. Перша ж є нерівність виконується, наприклад, при $n_0=7$ і $c_1=1/14$.

Зазначимо, що запис $f(n)=\Theta(g(n))$ включає в себе дві оцінки росту швидкості: верхню $O(g(n))$ („о від же від ен") і нижню $\Omega(g(n))$ („омега від же від ен").

Вважають, що: $f(n)=O(g(n))$, якщо знайдеться така константа $c>0$ і таке число n_0, що $0\le f(n)\le cg(n)$ для всіх $n\ge n_0$; $f(n)=\Omega(g(n))$, якщо знайдеться така константа $c>0$ і таке число n_0, що $0\le cg(n)\le f(n)$ для всіх $n\ge n_0$. Із наведеного випливає, що для функції $f(n)$ властивість $f(n)=\Theta(g(n))$ виконується тоді і тільки тоді, коли $f(n)=O(g(n))$ і $f(n)=\Omega(g(n))$. Спрощення, яких ми припускалися при оцінці складності алгоритму $T(n)$, базуються на правилах виконання операцій додавання та множення в Θ-символіці.

Правило сум. Нехай $T_1(n)$ і $T_2(n)$ – час виконання двох програмних фрагментів P_1 і P_2. $T_1(n)$ має швидкість росту $O(f(n))$, $T2(n)$ - $O(g(n))$. Тоді $T_1+T_2(n)$, тобто час послідовного виконання фрагментів P_1 і P_2, має швидкість росту $O(max(f(n),g(n))$. Для доказу цього нагадаємо, що існують константи c_1,c_2, n_1 і n_2 такі, що за умови $n\ge n_1$ виконується нерівність $T_1(n)\le c_1f(n)$, та аналогічно $T_2(n)\le c_2g(n)$, якщо $n\ge n_2$. Нехай $n_0=max(n_1,n_2)$. Якщо $n\ge n_0$, то, очевидно, що і $T_1(n)+T_2(n)\le c_1f(n)+c_2g(n)$. Звідси випливає, що при $n\ge n_0$ справедлива нерівність $T_1(n)+T_2(n)\le(c_1+c_2)max(f(n),g(n))$. Остання нерівність і означає, що $T1(n)+T2(n)$ має порядок росту $O(max(f(n), g(n)))$.

Правило добутку. Якщо $T1(n)$ і $T2(n)$ мають швидкість росту $O(f(n))$ і $O(g(n))$ відповідно, то добуток $T1(n)*T2(n)$ має швидкість росту $O(f(n)*g(n))$. З правила добутку випливає, що $O(n^2/2)$ еквівалентно $O(n^2)$.

Часто говорять, що час виконання алгоритму має порядок $T(N)$ від вхідних даних розміру N. Одиниця вимірювання $T(N)$ точно не визначена, але в більшості випадків розуміють під нею кількість інструкцій, які виконуються на ідеалізованому комп'ютері. Для багатьох програм час виконання дійсно є функцією вхідних даних, а не їх розміру. У цьому випадку визначають $T(N)$ як час виконання в найгіршому випадку, тобто, як максимум часів виконання за всіма вхідними даними розміру N. Поряд з тим розглядають $T_{ср}(N)$ як середній (в статистичному розумінні) час виконання за всіма вхідними

даними розміру *N*. Хоча $T_{ср}(N)$ є достатньо об'єктивною мірою виконання, але часто неможливо передбачити, або обгрунтувати, рівнозначність усіх вхідних даних. На практиці середній час виконання знайти складніше, ніж найгірший час виконання, так як математично це зробити важко і, крім цього, часто не буває простого визначення поняття „середніх" вхідних даних. Тому, в основному, користуються найгіршим часом виконання як міра часової складності алгоритмів [1,2].

Продуктивність алгоритму оцінюють за порядком величини. Говорять, що алгоритм має складність порядку $O\left(f\left(N\right)\right)$, якщо час виконання алгоритму росте пропорційно функції $f\left(N\right)$ із збільшенням розмірності початкових даних N. O – позначає „величина порядку".

Приведемо деякі функції, які часто зустрічаються при оцінці складності алгоритмів. Функції приведемо в порядку зростання обчислювальної складності зверху вниз. Ефективність степеневих алгоритмів звичайно вважається поганою, лінійних – задовільній, логарифмічних – хорошою.

Таблиця 1.1 - Функції, що використовуються при обчислення складності алгоритмів

Функція	Примітка
f(N)=C	C – константа
f(N)=log(log(N))	
f(N)=log(N)	
f(N)=NC	C – константа від нуля до одиниці
f(N)=N	
f(N)=N*log(N)	
$f(N)=N^C$	C – константа більша одиниці
$f(N)=C^N$	C – константа більша одиниці
f(N)=N!	тобто 1*2* … N

Оцінка з точністю до порядку дає верхню межу складності алгоритму. Те, що програма має певний порядок складності, не означає, що алгоритм буде дійсно виконуватися так довго. При певних вхідних даних, багато алгоритмів виконується набагато швидше, ніж можна припустити на підставі їхнього порядку складності. У числових алгоритмах точність і стійкість алгоритмів не менш важлива, ніж їх часова ефективність.

Аналіз складності алгоритму корисний для розуміння особливостей алгоритму і звичайно знаходить частини програми, що витрачають велику частину комп'ютерного часу. Надавши увагу оптимізації коду в цих частинах, можна внести максимальний ефект в збільшення продуктивності програми в цілому. Іноді тестування алгоритмів є найбільш відповідним способом визначити якнайкращого алгоритму. При такому тестуванні важливо, щоб тестові дані були максимально наближені до реальних даних. Якщо тестові дані сильно відрізняються від реальних, результати тестування можуть сильно відрізнятися від реальних .

Опишемо базові правила аналізу складності алгоритмів. У загальному випадку час виконання оператора або групи операторів можна розглядати як функцію з параметрами – розміром вхідних даних і/або одної чи декількох змінних. Але для часу виконання програми в цілому допустимим параметром може бути лише розмір вхідних даних.

Час виконання операторів присвоєння, читання і запису звичайно має порядок $O\left(1\right)$. Час виконання послідовності операторів визначається за правилом сум. Тому міра росту часу виконання послідовності операторів без визначення констант пропорційності співпадає з найбільшим часом виконання оператора в даній послідовності. Час виконання умовних операторів складається з часу виконання умовно виконуваних операторів і часу обчислення самого логічного виразу. Час обчислення логічного виразу часто має порядок $O\left(1\right)$. Час для всієї конструкції if-then-else складається з часу обчислення логічного виразу і найбільшого з часів, який необхідний для виконання операторів, що виконуються при різних значеннях логічного виразу. Час виконання циклу є сумою часів усіх часів виконуваних конструкцій циклу, які в свою чергу складаються з часів виконання операторів тіла циклу і часу обчислення умови завершення циклу (часто має порядок $O\left(1\right)$). Часто час виконання циклу обчислюється, нехтуючи визначенням констант пропорційності, як добуток кількості виконуваних операцій циклу на найбільший можливий час виконання тіла циклу. Час виконання кожного циклу, якщо в програмі їх декілька, повинен визначатися окремо.

Для пояснення методики оцінки часової складності алгоритмів T(n) скористаємося так званими „простими" алгоритмами. До цієї групи відносять алгоритми впорядкування обміном, впорядкування вибором та впорядкування вставками.

Впорядкування обміном. Алгоритм впорядкування обміном базується на принципі порівняння пари сусідніх елементів до тих пір, доки не будуть впорядковані всі елементи. Щоб описати основну ідею цього методу, який іноді називають методом „бульбашки", уявимо, що елементи зберігаються в послідовності (масиві), розташованому вертикально. Елементи, що мають малі значення, є більш „легкішими" і „спливають" нагору подібно бульбашкам. При першому проході уздовж масиву (перегляд починається знизу), береться перший елемент послідовності і його значення по черзі порівнюється зі значеннями наступних елементів. Якщо зустрічається елемент з більш „важким" значенням, то ці елементи міняються місцями. При зустрічі з елементом, що має більш „легше" значення, цей елемент стає „еталоном" для порівняння, і всі наступні елементи порівнюються з цим новим, більш „легшим" елементом. У результаті елемент з найменшим значенням опиниться вгорі послідовності.

Під час другого проходу вздовж масиву знаходиться елемент із другим по величині значенням, що міститься під елементом, який було знайдено при першому перегляді послідовності. Процес повторюється до тих пір, доки не будуть впорядковані всі елементи послідовності. Відзначимо, що під час другого і наступних переглядів послідовності немає необхідності переглядати елементи, знайдені за попередні перегляди, адже вони мають значення менші за значення елементів, що залишилися. Іншими словами, під час і-го перегляду не перевіряються елементи, що знаходяться на позиціях вище і.

Проілюструємо роботу алгоритму впорядкування обміном наступним прикладом, що записаний на псевдокоді.

```
(1) for i←n-1 step -1 until 1 do
(2) for j←1 step 1 until I do
(3)         if a[j] > a[j+1] then
begin
(4)                 swap(a[j], a[j+1])
end
```

Звернемо увагу на те, що підпрограма-процедура *swap* рядку (4) використовується в багатьох алгоритмах впорядкування для перестановки елементів місцями. Код цієї процедури наведено нижче:

```
procedure swap (x,y)
begin
temp ← x
x ← y
y ←temp
end
```

Для визначення часової складності алгоритму виконаємо наступні дії. Біля кожного рядку алгоритму зазначимо його вартість (число операцій) і кількість разів, за яку виконується цей рядок. Зауважимо, що рядки усередині циклу виконуються на один раз менше, ніж перевірка, оскільки остання перевірка виводить з циклу. Для кожного *j* від 1 до 2 і підрахуємо, скільки разів буде виконаний рядок (3), і позначимо це число через *kj*. Аналогічну дію виконаємо і для рядку (4).

Таблиця 1.2 - Визначення часової складності алгоритму впорядкування обміном

Команда	Вартість	Кількість виконань
for i←n-1 *step* -1 *until* 1 *do*	c_1	n
for j←1 *step* 1 *until* i *do*	c_2	n-1

if a[j] > a[j+1] *then*	c_3	$\sum_{j=1}^{i} k_j$
swap (a[j], a[j+1])	c_4	$\sum_{j=1}^{i} t_j$

Рядок вартістю *c*, що повторений *m* разів, дає внесок *cm* в загальну кількість операцій. Склавши внески всіх рядків, одержимо вираз, що позначає часову складність алгоритму впорядкування обміном:

$$T(n) = c_1 n + c_2(n-1) + c_3 \sum_{j=2}^{i} k_j + c_4 \sum_{j=2}^{i} t_j$$

Обчислимо суму кількості виконань для рядку (3).

$$\sum_{j=1}^{i} k_j = \frac{1+n-1}{2}(n-1) = \frac{n(n-1)}{2}$$

Таким чином, часова складність алгоритму T(n) дорівнює:

$$T(n) = c_1 n + c_2(n-1) + c_3\left(\frac{n(n-1)}{2}\right) + c_4 \sum_{j=2}^{i} t_j =$$

$$= c_1 n + c_2 n - c_2 + \frac{c_3 n^2}{2} - \frac{c_3 n}{2} + c_4 \sum_{j=2}^{i} t_j =$$

Як бачимо, функція T(n) – квадратична, тобто має вигляд $T(n)=an^2+bn+c$, де константи *a*, *b* і *c* визначаються значеннями $c_1, \ldots c_4$.

Слід зауважити, що час роботи алгоритму залежить не тільки від *n*, але і від того, який саме масив розмірністю *n* поданий їй на вхід. Для алгоритму впорядкування обміном найбільш сприятливий випадок, коли послідовність уже впорядкована. Процедура обміну *swap* у такому разі не буде виконана жодного разу, а часова складність алгоритму $T_{min}(n)$ дорівнюватиме:

$$T_{min}(n) = \left(\frac{c_3}{2}\right) n^2 + \left(c_1 + c_2 + \frac{c_3}{2}\right) n - c_2$$

Таким чином, у найбільш сприятливому випадку час $T_{min}(n)$, необхідний для обробки масиву розміру *n*, залишається бути квадратичною функцією від *n*.

Якщо ж масив розташований у зворотному (спадному) порядку, час роботи алгоритму буде максимальним: кожен елемент *a[j]* прийдеться міняти місцями з елементом *a[j+1]* . При цьому $t_j = j$.

Обчислимо суму кількості виконань для рядку (4).

$$T_{max}(n) = \left(\frac{c_3}{2}\right) n^2 + \left(c_1 + c_2 + \frac{c_3}{2}\right) n - c_2 + c_4 \sum_{j=2}^{i} t_{j=}$$

$$= \left(\frac{c_3}{2}\right) n^2 + \left(c_1 + c_2 + \frac{c_3}{2}\right) n - c_2 + c_4\left(\frac{n^2 - n}{2}\right) =$$

$$= \left(\frac{c_3}{2} + \frac{c_4}{2}\right)n^2 + \left(c_1 + c_2 - \frac{c_3}{2} - \frac{c_4}{2}\right)n - c_2$$

Як бачимо, у гіршому випадку час роботи алгоритму $T_{max}(n)$ також є квадратичною функцією від n.

Аналіз середнього значення часової складності алгоритму впорядкування обміном $T_{avg}(n)$ залежить від обраного розподілу ймовірностей, і на практиці реальний розподіл може відрізнятися від передбачуваного, який, зазвичай, вважають рівномірним. Іноді рівномірний розподіл моделюють, використовуючи генератори випадкових чисел. Проте, можна припустити, що в середньому обмін відбувається приблизно у $i/2$ випадках і його загальна кількість приблизно дорівнює значенню $(1+2+..+n)/2 \approx n^2/4$. У такому випадку середнє значення часової складності алгоритму впорядкування обміном $T_{avg}(n)$ можна показати формулою:

$$T_{avg}(n) = \left(\frac{c_3}{2}\right)n^2 + \left(c_1 + c_2 - \frac{c_3}{2}\right)n - c_2 + c_4\left(\frac{n^2 - n}{4}\right) =$$

$$= \left(\frac{c_3}{2} + \frac{c_4}{2}\right)n^2 + \left(c_1 + c_2 - \frac{c_3}{2} - \frac{c_4}{2}\right)n - c_2$$

Звичайно, алгоритм впорядкування обміном можна легко модифікувати. Один із шляхів полягає у тому, що можна позбавитися великої кількості непотрібних порівнянь, запам'ятавши чи відбувався обмін на попередньому кроці. Якщо обміну не було, то послідовність вважається впорядкованою, і алгоритм закінчує роботу. Цей процес удосконалення алгоритму можна продовжити, якщо запам'ятовувати не тільки сам факт обміну, але і місце (індекс) останнього обміну. Адже зрозуміло, що всі пари елементів з індексом, меншим від j, вже упорядковані, і наступні перегляди можна закінчувати на цьому індексі. Проте, всі вдосконалення жодним чином не впливають на кількість обмінів – вони лише зменшують кількість надлишкових повторних перевірок. Нажаль, обмін двох елементів – більш ресурсоємна операція, ніж порівняння, тому всі наші вдосконалення дають лише незначний ефект [2].

Таким чином, аналіз алгоритму впорядкування обміном показав, що ніяких переваг, окрім легко запам'ятовуючої назви, цей алгоритм не має. Проте, саме на прикладі цього алгоритму ми розглянули методику визначення часової складності алгоритмів, яку надалі застосуємо при аналізі інших алгоритмів.

Висновки й перспективи подальших досліджень. Застосування того чи іншого алгоритму пошуку для вирішення конкретної задачі є досить складною проблемою, вирішення якої потребує не лише досконалого володіння саме цим алгоритмом, але й всебічного розглядання того чи іншого алгоритму, тобто визначення усіх його переваг і недоліків. Звичайно, необхідність застосування саме простих алгоритмів пошуку очевидна. Адже складні алгоритми пошуку не дають бажаної ефективності в роботі програми. Але завжди треба пам'ятати й про те, що кожний простий алгоритм пошуку поряд із своїми перевагами може містити і деякі недоліки.

В нашій статті ми розглянули деякі алгоритми пошуку та їх реалізацію мовою С++, реалізували програмне використання методів пошуку, а також дослідили переваги алгоритмів пошуку, ефективність їх використання, визначили деякі недоліки окремих алгоритмів.

Програма містить реалізацію деяких методів пошуку даних. Дане дослідження не є завершеним. Надалі буде проводитись модернізація та реструктуризація створеного програмного продукту, задля розширення області його практичного застосування. Оскільки, теорія алгоритмів – молода та не докінця досліджена наука, наповнена великим багажем знань, вдосконалення та розширення предметної області програмного продукту, і надалі буде актуальним питанням для досліджень.

1. Cormen, Thomas H. Introduction to Algorithms (2nd ed.) / [Cormen, Thomas H.; Leiserson, Charles E., Rivest, Ronald L., Stein, Clifford]. – MIT Press and McGraw-Hill. ISBN 0-262-03293-7., 2001. – Глава 12. – 345 с.
2. Ахо А. Структуры данных и алгоритмы / Ахо А., Хопкрофт Дж., Ульман Дж. : [Пер. с англ.] // [Уч. пос.] – М. : Издательский дом «Вильямс», 2000. –384 с. : ил.
3. Ахо А. Построение и анализ вычислительных алгоритмов / Ахо А., Хопкрофт Дж., Ульман Дж. – М. : Мир, 1979. – 536 с.
4. Кормен Т. Алгоритмы : построение и анализ / Кормен Т., Лейзерсон Ч., Ривест Р. – М. : МЦНМО, 2001. – 960 с.
5. Лісовик Л. П. Теорія алгоритмів : [Навч. Посібник] / Л. П. Лісовик, С. С. Шкільняк. – К. : Видавничий поліграфічний центр : Київський університет, 2003. –163 с.
6. Прийма С. М. Теорія алгоритмів: [Навч. Посібник] / Прийма С. М. – Мелітополь : МДПУ, 2004. – 48 с.:іл.

УДК 004.6
Клятченко Я.М., Тарасенко Г.О., Тарасенко-Клятченко О.В.
Національний технічний університет України «Київський політехнічний інститут»

РЕАЛІЗАЦІЯ ПОРІВНЯННЯ ЧИСЕЛ В НЕГАПОЗИЦІЙНИХ СИСТЕМАХ ЧИСЛЕННЯ

Клятченко Я.М., Тарасенко Г.О., Тараскенко-Клятченко О.В. Реалізація порівняння чисел в негапозиційних системах числення. В статті розглядаються можливості покращити архітектурно-структурні характеристики комп'ютерних засобів за рахунок подання чисел в негапозиційних системах числення. Запропоновано структури операційного пристрою для порівняння двох операндів.
Ключові слова: негапозиційна система числення, позиційна арифметика, порівняння двійкових операндів.

Клятченко Я.М., Тарасенко Г.О., Тарасенко-Клятченко О.В. Реализация сравнения чисел в негапозиционных системах исчисления. В статье рассматриваются возможности улучшить архитектурно-структурные характеристики компьютерных средств за счёт представления чисел в негапозиционных системах исчисления. Предложены структуры операционного устройства для сравнения двух операндов.
Ключевые слова: негапозиционная система исчисления, позиционная арифметика, сравнение двоичных операндов.

Klyatchenko Y.M., Tarasenko G.O., Tarasenko-Klyatchenko O.V. Implementation of numbers comparison in negative-base numeral systems. The options to improve architectural-structural characteristics are examined in the article through the numbers presentation in negative-base numeral systems. The structures of operational arrangement for two operands comparison are proposed.
Key words: negative-base numeral system, positional arithmetic, comparison of binary operands.

Подання чисел в негапозиційних системах числення (тобто, позиційних з від'ємною основою) є заманливою можливістю поліпшити структурно-функціональні та техніко-економічні характеристики комп'ютерних операційних засобів [1-3]. Перш за все, ця заманливість обумовлена значним прискоренням найбільш масових арифметичних операцій додавання-віднімання, оскільки сигнали переносу (для віднімання сигналу боргу) у випадку використання такого числення, не розповсюджується далі, ніж на один розряд (на одну позицію) [1,2]. Це, в свою чергу, дозволяє отримати час операції додавання-віднімання, який не залежить від довжини операндів, оскільки проблема «довгих ланцюжків» (так звана ''вічна проблема'' позиційної арифметики взагалі [4]) в такому разі не виникає. Іншою заманливою особливістю подання чисел в позиційних численнях з від'ємною основою є відсутність спеціальної знакової позиції (розряду) [1]. Зважаючи на вартість сучасних апаратних цифрових засобів для подання чисел, перевага в скороченні довжини розрядної сітки в один розряд (від відкидання знакового розряду) для операційних пристроїв широкого призначення може вважатися несуттєвою. Однак, для спеціалізованих комп'ютерних операційних засобів ця обставина може бути вирішальною. При цьому слід мати на увазі, що переваги подання чисел в негапозиційних численнях, що стимулюють дослідницький інтерес до них, до певної міри ''врівноважуються'' складністю виконання інших операцій з таким поданням чисел. Особливо це стосується операцій порівняння негапозиційних операндів. Складнощі навіть наштовхнули деяких комп'ютерних аналітиків-арифметистів до думки про неможливість взагалі порівняння операндів в негапозиційних численнях [2]. Відомі також рекомендації щодо попереднього переводу чисел до зміщених систем числення [1,3] заради необхідності виконання такого порівняння.

Далі з метою спростування сумнівів щодо можливості технічного відтворення операції порівняння чисел в негапозиційному численні пропонуються деякі алгоритмічні та структурні рішення, які є цілком реальними з позицій досягнень сучасної комп'ютерної схемотехніки.

Нехай A і B – двійкові операнди в позиційній системі числення з основою *k=-2*, причому:

$$A = \sum_{i=0}^{n} a_i(-2)^i; B = \sum_{i=0}^{n} b_i(2^{-i});\ a_i, b_i \in \{0,1\};\ i = \overline{0,n};$$

$$i = \overline{0,n}; n = 2l$$

Тут n – число всіх розрядів для подання операндів, яке без втрати загальності викладу будемо вважати парним; *l* – число розрядів в поданні операндів *A* і *B*, які мають один і той же знак їх ваги (інакше – число розрядів з додатною і від'ємною вагою однакове і дорівнює *l*). Очевидно, що за прийнятих позначень максимальний операнд в такому негапозиційному численні може бути обчислений як:

$$A_{max} = 2^0 + 2^2 + 2^4 + \cdots + 2^{n-2} = \sum_{i=0}^{n-1} 2^{2i},$$

а найменший операнд як:

$$A_{min} = -(2^1 + 2^3 + 2^5 + \ldots + 2^{n-1}) = -\sum_{i=1}^{n-1} 2^{2i-1}$$

Наприклад, якщо *n=8, l=4*, то ваговий ряд для такої негапозиційної системи має вигляд *-128, 64, -32, 16, - 8, 4,- 2, 1*, а діапазон можливих цілих чисел знаходиться між $A_{max} = 85$ і $A_{min} = -170$. Реалізація порівняння операндів *A* і *B* означає встановлення факту виконання одного із співвідношень: *A=B, A>B, A<B.*

Основними властивостями подання операндів в такій негапозиційній системі, що забезпечують досягнення поставленої мети, є наступні:

1. Знаки ваги окремих розрядів чергуються оскільки $(-2)^i > 0$, якщо *i* – парне (включаючи випадок *i*=0) та $(-2)^i < 0$, якщо *i*-непарне. Це дозволяє розділити кожен окремий операнд на додатну і від'ємну частину та виконувати операції з кожною частиною окремо.
2. Будь-яка ненульова цифра, що знаходиться на *i*-му місці (рахуючи справа наліво) в запису операнда за абсолютною величиною її кількісного еквівалента перевищує кількісний еквівалент будь-якої комбінації цифр, що знаходиться справа від *i*-ї позиції. Ця властивість дозволяє в багатьох випадках проводити порівняння чисел орієнтуючись на позиції крайніх зліва одиниць в поданні операнда.
3. Знак операнда в цілому або деякої його частини визначається знаком ваги найстаршого ненульового розряду. Отже якщо група із *m* старших розрядів операнда *A* тотожна (збігається) групі із *m* старших розрядів операнда *B*, то їх можна виключити із розгляду, а остаточний висновок про співвідношення *A* і *B*, робити по *n-m* – розрядних фрагментах *A* і *B*, які залишаються.

Таким чином, алгоритм реалізації багатомісної операції порівняння операндів *A* і *B*, поданих в негапозиційному численні, полягає в наступному:

1. На основі двійкових наборів (векторів), що відповідають операндам *A* і *B*, утворити вектор *C* шляхом їх покомпонентного додавання за модулем 2. Якщо всі компоненти вектора *C* дорівнюють нулю, то *A=B* і операція порівняння закінчена. В іншому разі перейти до п.2.
2. Шляхом пріоритетного порівняння компонент c_i вектора *C* утворити маркерний вектор *D*, який містить лише одну одиницю в позиції d_M, що відповідає найстаршій одиниці вектора *C*. Це означає, що в векторах *A* і *B* всі цифри лівіше позиції d_M не впливають на виконання співвідношення порівняння.
3. Шляхом покомпонентного порівняння в парах (a_i, d_i) та (b_i, d_i) визначити цифри a'_M і b'_M в векторах *A* і *B*, які знаходяться на позиції маркерного розряду d_M в векторі *D*.
4. Визначити парність (чи непарність) позиції маркерного розряду d_M у векторі *D*. Позначимо $P(d_M) = 1$ – факт парності d_m, $N(d_M) = 1$ – факт непарності d_m. Очевидно, що $P(d_M)$ та $N(d_M)$ можуть бути тільки взаємно інверсними.
5. На основі значень a'_M, b'_M, $P(d_M)$, $N(d_M)$ відповідно до таблиці сформувати результат порівняння чисел *A* і *B*. Алгоритм закінчено.

a'_M	b'_M	$P(d_M)$	$N(d_M)$	Співвідношення між операндами
1	0	1	0	$A>B$
0	1	0	1	$A>B$
1	0	0	1	$A<B$
0	1	1	0	$A<B$
0	0	0	0	$A=B$

Зауважимо, що у вхідній частині таблиці наведені тільки правильні можливі значення a'_M, b'_M, $P(d_M)$ та $N(d_M)$. Інші їх комбінації за правильного виконання попереднього алгоритма неможливі, або ж вказують на неправильне виконання якихось кроків алгоритма.

Приклад 1.

Нехай A=01110010 (це 46_{10}) і B=10001010 (це -138_{10}). Тоді значення векторів C, D і окремих змінних після кожного кроку виконання алгоритму можна показати наступною діаграмою станів.

A=01110010, B=10001010

1-й крок C=11111000

2-й крок D=10000000

3-й крок $\begin{cases} a_M = 0 \\ b_M = 1 \end{cases}$

4-й крок $\begin{cases} P(d_M) = 0 \\ N(d_M) = 1 \end{cases}$

5-й крок $A>B$

Приклад 2.

A=11101100 (це -102_{10}), B=11100111 (це -77_{10}). Для цього випадку діаграма станів має вигляд:

A=11101100, B=11100111

1-й крок C=00001011

2-й крок D=00001000

3-й крок $\begin{cases} a_M = 1 \\ b_M = 0 \end{cases}$

4-й крок $\begin{cases} P(d_M) = 0 \\ N(d_M) = 1 \end{cases}$

5-й крок $A<B$

На рис 1. показана структурна схема апаратної реалізації вищеприведеного алгоритма.

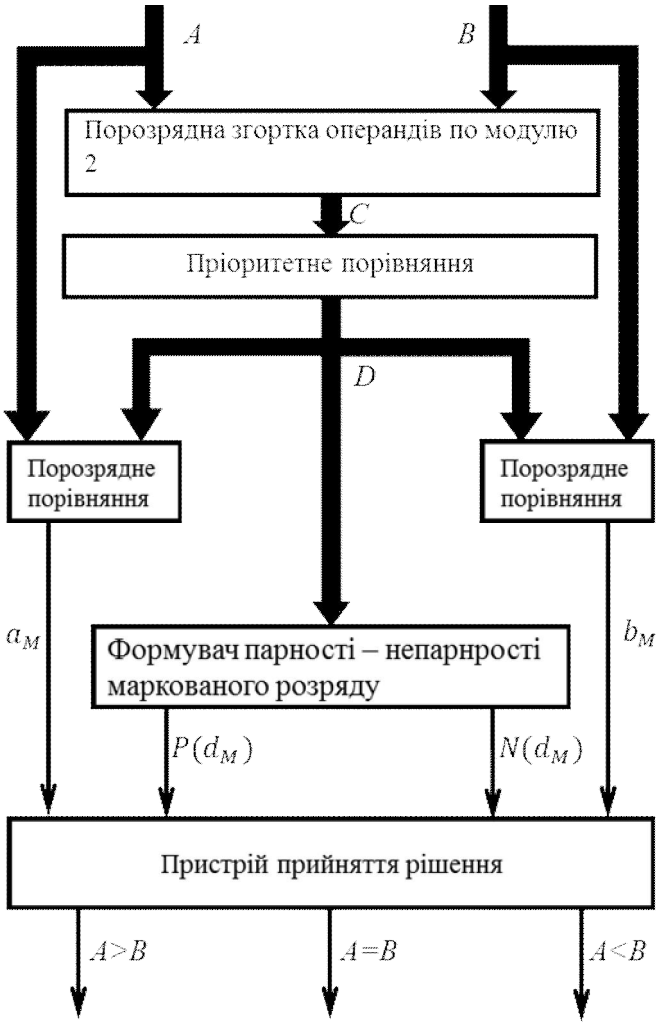


Рис.1. Пристрій для порівняння двох операндів.

Аналіз структури на рис.1 показує, що всі структурні елементи, які використані для встановлення співвідношень порівняння операндів у негапозиційному численні, можуть бути реалізовані комбінаційними схемами, що функціонують у двозначному структурному алфавіті. Очевидно також, що всі вони можуть бути відтворені на базі стандартних логічних ресурсів сучасних програмовних логічних інтегральних середовищ (ПЛІС). Отже, реалізація порівняння операндів в негапозиційному численні є принципово можливою і здійсненною без переводу операндів до позиційної системи числення. Стосовно структури на рис.1 слід зауважити, що оптимізація апаратних засобів, використовуваних на деяких кроках алгоритма, наприклад, заміна комбінаційного формувача парності на лічильники, може породити нові ефективніші реалізації.

1. Корнійчук В.І., Тарасенко В.П., Тарасенко-Клятченко О.В. Основи комп'ютерної арифметики/ В.І. Корнійчук, В.П.Тарасенко, О.В.Тарасенко-Клятченко, -К.: «Корнійчук», -2014. -с.44-49.
2. Король І.Ю., Тарасенко В.П. До питання про виконання арифметичних операцій в мінус-двійковій системі числення, «Проблеми автоматизації управління»/ І.Ю. Король, В.П.Тарасенко, -К.: вид-во НАУ, -2008, -с.195-206.
3. Петришин М.Л. Переваги перетворення форми інформації в негапозиційних системах числення. Матеріали 5-ї МНТК ''Інформаційні технології'' та комп'ютерна інженерія в Прикарпатському національному університеті імені Василя Стефаника, 2008, с.110-111.
4. Карцев М.А. Арифметика цифровых машин. М:, «Наука», 1969, 576 с.

УДК 004.002
Мельник В.М., Багнюк Н.В., Мельник К.В., Жигаревич О.К.
Луцький національний технічний університет

РЕАЛІЗАЦІЯ С- ТА JAVA-ІНТЕРФЕЙСІВ ДЛЯ АСИНХРОННОГО РЕЖИМУ ПЕРЕДАЧІ ДАНИХ

В.М. Мельник, Н.В. Багнюк, К.В. Мельник, О.К. Жигаревич. Реалізація С- та JAVA-інтерфейсів для асинхронного режиму передачі даних. Розглянута технологія передачі даних, що реалізована з застосуванням асинхронного режиму передачі (АРП), яка пропонує власні можливості для користувачів. Для використання переваги АРП з одночасним поєднанням його роботи в прикладних програмах необхідно використовувати потужний програмний інтерфейс для прикладних додатків (ПІПД). В роботі реалізовано три різні варіанти програмного інтерфейсу для прикладних додатків з використанням вільнодоступних власних АРП-функцій. Представлені також дизайн та реалізація ПІПД для АРП на мові С для робочих станцій Digital Alpha, що працюють з системою Digital UNIX. Описано і ПІПД для АРП мовою Java, який пропонує програмування Java-сокетів в середовищі, звичному для Java-програмістів. З метою виявлення пропускних характеристик були співставлені різні концепції та архітектурні підходи
Ключові слова: Технологія передачі, асинхронний режим передачі, програмний інтерфейс для прикладних додатків, сокети, архітектура.

V.M. Melnyk, N.V. Bahnyuk, K.V. Melnyk, O.K. Zhyharevych. C- and java- interface implementation in asynchronous data transfer mode. Flexible transmission technology is represented through an asynchronous transferring mode (ATM), which proposals new services set for users. In aim to satisfy ATM's strength advantage and to integrate access with ATM services into particular applications, it need to use a powerful application programming interface (API). Three different variants of such API was implemented here with native ATM functions accessing. It is also presented ATM API design and implementation in C for Digital Alpha workstations running under Digital UNIX system. There also Java ATM API is described with offering the socket programming environment, suitable for Java-programmers. The different attitudes and architectural approaches are compared in order to give throughput information
Keywords: transmission technology, asynchronous transferring mode, application programming interface, sockets, architectural approaches

В.М. Мельник, Н.В. Багнюк, К.В. Мельник, О.К. Жигаревич. Реализация С- и JAVA-интерфейсов для асинхронного режима передачи данных. Рассмотрена технология передачи данных, реализована с применением асинхронного режима передачи (АРП), которая предлагает свои возможности для пользователей. Для использования преимущества АРП с одновременным сочетанием его работы в программах необходимо использовать мощный программный интерфейс для прикладных приложений (ПИПД). В работе реализовано три различных варианта программного интерфейса для приложений с использованием доступных собственных АРП-функций. Представлены также дизайн и реализация ПИПД для АРП на языке С для рабочих станций Digital Alpha, работающих с системой Digital UNIX. Описаны и ПИПД для АРП на языке Java, который предлагает программирование Java-сокетов в среде, привычной для Java-программистов. С целью определения пропускных характеристик были сопоставлены различные концепции и архитектурные подходы
Ключевые слова: Технология передачи, асинхронный режим передачи, программный интерфейс для прикладных приложений, сокеты, архитектура

Вступ

В останні роки комунікаційні технології різко повернули до впровадження опто-волоконних мереж і вдосконалення технологій передачі даних, особливо таких як SONET/SDH. На базі подібних розробок високошвидкісні мережі стали реальністю, зокрема, на основі технології *асинхронного режиму передачі* (АРП) даних для Broadband-ISDN. Однак, зовсім нові можливості, які запропоновані АРП, поглинаються традиційними протоколами, що функціонують між додатками і мережевими службами. Наприклад, при передачі даних з використанням IP через АРП [1] або LAN-емуляцію [2] спостерігається спадання асинхронного режиму передачі до рівня традиційного протоколу канального зв'язку і приховування нової функціональності. Єдина можливість для додатка скористатися послугами, наданими АРП, – це використання власного АРП-інтерфейсу, який надає додаткам прямий доступ до функціональності рівнів адаптації при асинхронному режиму передачі.

Форум АРП визначив семантичний опис такого АРП-інтерфейсу для прикладного програмування (АРП ПІПД) [3,4], який було реалізовано на декількох архітектурах, таких як UNIX-похідних машинах, Linux [5] і для комп'ютерів під управлінням MS-DOS або операційної системи Plan 9 та передуючих адаптерів [6]. Останній підхід був портований на FreeBSD і [7] також має відношення до розробки, реалізації операційної системи та сигнальної підтримки споріднених додатків [7], що є прикладом транспортного обслуговування, запропонованого на вершині рівня адаптації асинхронного режиму передачі AAL-5.

В роботі описуються способи проектування та дослідження власної ППІД АРП для робочих станцій, що працюють під управлінням Digital Alpha Digital UNIX. Подається Java-реалізація для ППІД АРП по лінії рекомендацій та матеріалах АРП-формуму на її вершині для даного асинхронного режиму.

Мова С для ППІД АРП на Digital UNIX

Перший програмний інтерфейс для прикладних додатків АРП, представлений в цій роботі, був розроблений на Digital UNIX (раніше OSF/I) на робочих станціях Digital Alpha мовою С [8]. Для Digital UNIX версій 3.2 і 4.0 підсистема ППІД фіксується навколо модуля управління з'єднанням (МУЗ) в якості основного модуля (рис. 1). Стосовно трьох різних класів модулів ця архітектура встановлює драйвери пристроїв, модулі сигналізації і модулі конвергенції. В залежності від потреб додатків, модулі кожного класу можуть бути додані до МУЗ, забезпечуючи додаткову функціональність.

Як правило, драйвери пристроїв походять з методів передачі поданих даних, сформованих середовищем. Для підтримки кожного сигнального протоколу модуль сигналізації зареєстрований в МУЗ. Сигнальні канали обробляють МУЗ як регулярні (що переключаються або постійні) віртуальні канали. Конвергенція модулів, нарешті, служить для того, щоб подолати додатками можливий розрив між МУЗ та іншими модулями в просторі користувача. Реалізація IP-через-АРП, який поставляється з МУЗ, є прикладом такого модуля збіжності.

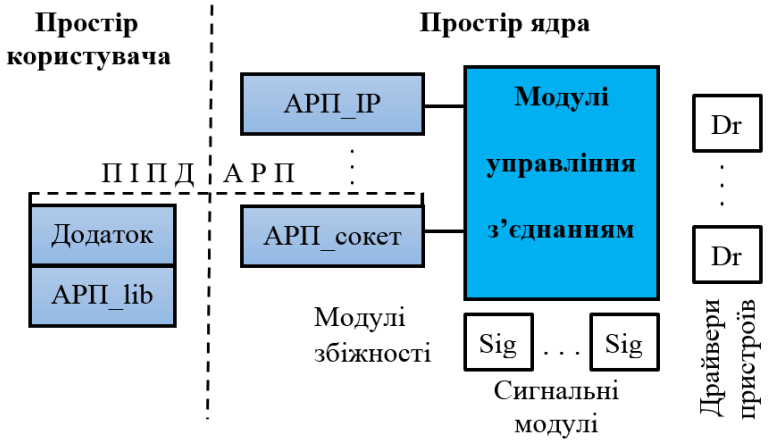


Рис. 1 – Структура підсистеми АРП digital UNIX

ППІД АРП було реалізовано також у вигляді модуля збіжності. Якщо говорити в деталях, то ППІД АРП складається з двох частин: нового модуля конвергенції atmsocket, розташованого в просторі ядра, і бібліотеки libatm в просторі користувача. Atmsocket взаємодіє з центральним МУЗ в ядрі і виконує такі завдання, як обробка контекстів зв'язку і переадресації даних користувача. Додатки отримують доступ до функцій, що забезпечують модуль atmsocket за допомогою бібліотеки libatm. Декілька програм можна пов'язати з цією бібліотекою для одночасної роботи і всі вони будуть напрямлені на один модуль atmsocket в ядрі. Для того, щоб здійснювалась взаємодія між libatm і atmsocket, ППІД АРП повинен перейти межі між ядром і простором користувача Digital UNIX.

У даному відтворенні не вводилось нових системних викликів для взаємодії з ядром, тому що це призвело б до проблем появи нових вільних кроків роботи операційної системи. У такому випадку ППІД АРП повинні бути повторно інтегровані в кожній новій версії системи. Натомість, ППІД використовує концепцію стандартизованого UNIX-сокета як інтерфейс до ядра. Однак ППІД АРП не реалізує сокетний інтерфейс АРП. Сокет тільки з'єднує виклики функцій з бібліотеки libatm в модуль atmsocket і навпаки. То ж позитивним є те, що функціональність наведеного ППІД можна легко розширювати.

Цей підхід аналогічний до Fore API [9] який також використовує інтерфейс сокетів BSD або потоковий інтерфейс System V, як канал зв'язку між API і драйвером пристрою. Більшість інтерфейсних функцій отримані безпосередньо з відповідних примітивів семантики опису форуму АРП [3,4]. Більш детально, ППІД АРП пропонує виклики функцій з програми (libatm) до модуля atmsocket, які називатимемо викликами на нижчий рівень "downcalls" і зворотного виклику функцій для індикації, наданої від модуля atmsocket до додатка, які теж назвемо викликами на вищий рівень "upcalls". Передбачені downcall-функції, можуть бути згруповані в певні категорії, де кожна категорія міститиме функції, що пов'язані з локальними завданнями, такими як створення або видалення локального

ідентифікатора зв'язку з кінцевою точкою з'єднання (КТЗ) (наприклад, такою як *ATM_associate_endpoint()*).

Іншими категоріями є, відповідно, встановлення, звільнення та переривання вихідних або вхідних з'єднань (наприклад, таких як *ATM_connect_outgoing_call()*). Крім того, дані надсилаються функцією *ATM_send_data()*, а інші функції взаємодіють з індикаторами та підтвердженнями, які обробляються інтерфейсом (наприклад, *ATM_process()*, як було вже описано). Аналогічним чином, upcalls-виклики, які додаток може зареєструвати в ПІПД, можуть також бути згруповані в класи. Є спеціальні функції для повідомлення додатка про отримання виклику або надходження даних. Крім того, є зворотні або callback-виклики, які підтверджують завершення або відмову від операції як додавання частки в груповий зв'язок (яка не підтримується до версії Digital UNIX 4,0e). Також повідомляється, коли з'єднання готове для обробки даних, що будуть передаватися. І нарешті, також є функція, яка вказує на завершення виклику.

Реалізація ПІПД базується на одній нитці виконання. Щоб уникнути блокування додатка, коли, наприклад, він очікує вхідний виклик, всі downcall-виклики ПІПД були реалізовані як неблокуючі функції. Для того, щоб дозволити можливість ПІПД виконувати upcall-виклики до додатка, має бути виділена нитка виконання. Це робиться шляхом виклику downcall-функції ATM_process() з відповідним значенням для кінцевої точки з'єднання (КТЗ). В залежності від показників очікування API далі викликає відповідну upcall-функцію і, таким чином, виступає в якості диспетчера. Додатку, який спонукає асинхронний виклик upcall-функцій, потрібно створити ще один потік, який блокується до тих пір, поки сокет, який пов'язаний з КТЗ, згенерує подію.

Спроектований ПІПД АРП був використаний для реалізації різних рівнів протоколів над АРП [8]. Спеціальний механізм моніторингу QoS був інтегрований в ПІПД [10], а контроль якості обслуговування QoS підтримувався новими послугами спеціального підрівня конвергенції ПСПК (або *Service Specific Convergence Sublayer*) протоколом для шару сервісу рівнів адаптації асинхронного режиму передачі загального використання. Такий QoS-моніторинг ПСПК об'єднує додаткову інформацію управління, таку як часові мітки і порядкові номери, в регулярний потік даних. За допомогою цієї інформації та спеціального підходу вирівнювання в протоколі трейлера, QoS-моніторинг ПСПК здатний підтримувати моніторинг шару рівнів адаптації асинхронного режиму передачі з кінця в кінець, а також моніторинг стрибків шару рівня асинхронної передачі. Крім того, такий механізі потужного підвищення контролю вже інтегрований в останню версію ПІПД [11] і може бути легко активований шляхом комунікації з типом ПСПК-сервісу нашого QoS-моніторингу ПСПК при встановленні з'єднання. Оцінка з кінця в кінець QoS-моніторингу інформації для ПСПК може бути виконана за допомогою спеціалізованого QoS-монітора [12], який в даний час інтегровано в модуль armsocket [11] в просторі ядра. Через це, детальні результати моніторингу на базі нового QoS-моніторингу ПСПК все ж не можуть бути отримані.

ПІПД для АРП Java в Digital UNIX

Насамперед було втілено ряд пропозицій для того, щоб надати доступ до функціональності АРП в Java [13]. Однією із таких пропозицій для розширення Java є пропозиція операцій з сокетами стилю BSD [14-16]. Іншою – є пропозиція розширення пакета java.net з метою імітувати функції Java-сокетів якомога ближче [17-20], для того щоб реалізувати функціональність, описану в роботі [4], яка пізніше була розширена в наступних роботах. Цей підхід був узятий на нашу Java-бібліотеку ПІПД, що реалізує основні функції специфікації для ядра і був успішно протестований в Java Development Kit (JDK) 1.1.7 і 1.2/2.0.

Для доступу до функцій "іншими" мовами в Java функціонує доступний власний Java-інтерфейс (BI Java). Інструмент Java-h генерує заголовки С-функцій із оголошень в Java-класі. Виклик методу функцією, оголошеного як власного в результатах Java, здійснюється у відповідній С-функції, що викликається. Однак, параметри передаються дещо інакше, ніж із С-функцій. Замість передачі параметрів безпосередньо, деякі вказівники на, наприклад, середовище Java, передаються на функцію, яка, в свою чергу, повинна знайти відповідні фактичні змінні. Це означає, що не можливо прямим способом викликати функції регулярної бібліотеки С з Java в UNIX, але замість цього повинен бути введений проміжний шар, який, як мінімум, визначає параметри і відслідковує синтаксичні відмінності. Рисунок 2 зліва демонструє архітектуру ПІПД Java, у тому числі проміжний шар та ПІПД мови С. Потрібно бути обережним і не слід плутати "власні послуги АРП " і "власні функції інтерфейсу" в цьому контексті.

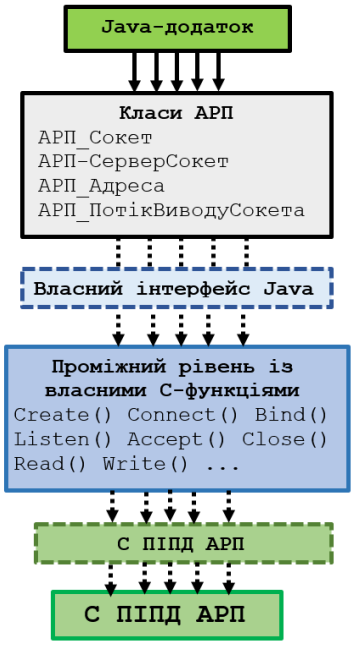


Рис. 2 – Архітектура ПІПД Java на Digital UNIX

AtmSocket створює клієнтський сокет, аналогічний сокету мови С, AtmServerSocket – це серверний сокет. Обоє вони використовують абстрактний клас AtmSocketImpl, який пропонує методи відповідно до "заводського" шаблону проектування, разом з класом PlainAtmSocketImpl. AtmSocketInputStream і AtmSocketOutputStream пропонує відповідно декілька функцій читання і запису, які відомі зі звичайних Java сокетів. Для вказівки адреси АРП використовується новий клас AtmAddress. Класи AtmBLLI і AtmBHLI представляють інформацію високого і низького рівня розширення АРП, відповідно. Параметри з'єднання визначаються за допомогою AtmConnAttr, але в даний час – ігноруються.

Оцінка продуктивності

Адаптери АРП, що використовувалися, пропонують лінійну швидкість ~155 Мбіт/с. Віднімаючи інформацію протоколу і управління, яка передається, шар АРП пропонує підтримку швидкості 135,63 Мбіт/с. Пропускна здатність, втрати блоків (або комірок) і отримані графіки системного навантаження залежать від багатьох факторів, а найбільше від розміру посланих блоків.

Використовуючи рівень потужності Digital UNIX в 8,98 кбайт, застосування IP через АРП на робочих станціях Digital UNIX дає продуктивність близько 60 Мбіт/с. ПІПД АРП, який ще не був оптимізований за швидкістю, досягає пропускної здатності від 85 до більш ніж 120 Мбіт/с при утворенні зв'язку між двома різними машинами. Низька продуктивність IP через АРП пояснюється його більш високими затратами на обробку протоколу. Якщо використовувати для двохстороннього з'єднання свіч АРП до однієї і тієї ж машини, то пропускна здатність через власний інтерфейс досягає ~40 Мбіт/с. У цьому випадку частина ядра АРП займає ~90% циклів процесора. Java ПІПД на таких же машинах досягає пропускної здатності відсилання до ~106 Мбіт/с (але з дуже високим рівнем втрат приймача, які, ймовірно, обумовлені неоптимальністю буфера обробки на приймальній стороні), при використанні з однієї машини на іншу і до 85 Мбіт/с (при значно нижчому рівні втрат) при відправці до свіча і назад до тієї же машини. Ці цифри нижчі, ніж для ПІПД С через додаткові затрати, пов'язані з проміжним шаром, а високий рівень втрат, пов'язаний з рівнем швидкості надсилання до 90 Мбіт/с спричиняє значно менший рівень отримування (rate2). З використанням удосконаленої віртуальної машини Java-fast можна збільшити вихідну досягнуту пропускну здатність за допомогою збільшеного зображення пам'яті (44 Мб замість 11 Мб) і вищого рівня втрат (імовірно приріст операції надсилання зростає швидше, ніж здійснення отримання). Використання повільнішого PrintStream замість DataOutputStream стимулює зростання пропускної здатності від 5 до 8 Мбіт/с при втратах близько 0,01%. Отже пропускна здатність сильно залежить від розміру блоку і значення вихідного рівня.

Висновок

В роботі були представлені три різні підходи для реалізації власного інтерфейсу АРП. У той час як реалізація ПІПД АРП мови С повністю інтегрована в архітектуру цифрового модуля управління з'єднанням (Connection Management Module), ПІПД Java був реалізований на вершині ПІПД мови С і якому необхідний проміжний рівень. Із-за різних архітектур (і складнощів), а також через різницю характеристик продуктивності основних машин, операційних систем і драйверів досягнуті в роботі значення пропускних здатностнй також відрізняються. Отримані цифрові значення збільшують впевненість, що є можливість налаштування коду, так щоб використовуючи сучасні машини, можна було б скористатися перевагами АРП за ради збільшення пропускної здатності при умові забезпечення відповідної підтримки драйверів для цього функціоналу.

1. M. Laubach: Classical IP and ARP over ATM, Request for Comments 1577, Internet Engineering Task Force (IETF), January 1994.
2. The ATM Forum: LAN Emulation over ATM, Version 2 - LUNI Specification; af-lane-0084.000, July 1997.
3. The ATM Forum: Native ATM Services: Semantic Description Version 1.0; af-saa-0048.000, February, 1996.
4. The ATM Forum: API Semantics for Native ATM Services using UN1 4.0; af-saa-0108.000, December 1998.
5. W. Almesberger: Linux ATM API, Draft, version 0.4, EPFL, LRC, Switzerland, http://lrcwww.epfl.ch/linux-atm/, July 19, 1996.
6. S. Keshav, H. Saran: Semantics and Implementation of a Native-Mode ATM Protocol Stack; AT&T Bell Laboratories Technical Memorandum, 1994, http://www.cs.att.com/csrc/keshav/papers.html
7. R. Ahuja, ***S.*** Keshav, H. Saran: Design, Implementation and Performance of a Native Mode ATM Transport Layer; Proc. IEEE INFOCOM, March 1996.
8. Stefan Dresler, Markus Hofmann, Claudia Schmidt, Hajo R. Wiltfang: A Native ATM API Suited for Multimedia Communication; Fourth International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS P7), Darmstadt, Germany, September 1997.
9. Edoardo Biagioni, Eric Cooper, Robert Sansom (Fore Systems Inc.): Designing a Practical ATM LAN; IEEE Network, March 1993 http://www.cs.att.com/csrc/keshav/papers.html.
10. H. R. Wiltfang, C. Schmidt: QoS Monitoring for ATM-based Networks; in Proceedings of the International Conference on Management of Multimedia Networks and Services; Montreal, Canada; July 8- 10, 1997.
11. S. Dresler, R. Lisak, H. Ritter, H.R. Wiltfang. Native ATM Interfaces in C and Java: Implementation and Experiences. Lecture notes in computer science 1309. – 1997. – Springer. – p. 352-471.
12. C. Schmidt, R. Bless: QoS Monitoring in High Peflorniance Environments, in Proceedings of the Fourth IFIP International Workshop on Quality of Service; Paris, France; March 6-8, 1996.
13. The ATM Forum/T. Jepsen, J. Shaffer: Java ATM API Description-Proposed Outline, Revision 1; contribution atm97-1044r1; February 1997.
14. The ATM Forum/T. Jepsen, S.A. Wright: A Java Syntax Instantiation of the Native ATM Services; contribution atm97-0772, September 1997.
15. The ATM Forum/T. Jepsen, S.A. Wright: A Java Syntax Mapping to the Native ATM Services; contribution atm97-0773, September 1997.
16. The ATM Forum/T. Jepsen, S.A. Wright: A Java Syntax Implementation of the Native ATM *Services;* contribution atm97-0774, September 1997.
17. The ATM Forum/J. Harford: Java ATMAPI from 50,000 Feet; contribution atm97-1110, December 1997.
18. The ATM Forum/J. Sung: Extension of the java.net package to support Native ATM Services, Revision 1; contribution atm98-0332r1, July 1998.
19. Microsoft: *Windows Sockets* 2 *API, Revision 2.2.0;* May 10, 1996.
20. The ATM Forum/Tom Jepsen, John Shaffer: *Java ATM API Working Document-Baseline Text;* contribution btd-saa-api-Java-00.03, April 1999.

УДК 004.002
Мельник В.М., Поліщук М.М., Здолбіцький А.П., Желобицький Я.К.
Луцький національний технічний університет

САЙТ ДЛЯ ТЕЛЕРАДІОКОМПАНІЇ З АВТОМАТИЧНИМ ЗАПИСОМ ЕФІРІВ, І АВТОНАПОВНЕННЯМ ІЗ ВЛАСНОГО ФАЙЛОБМІННИКА

Мельник В.М., Поліщук М.М., Здолбіцький А.П., Желобицький Я.К. Сайт для телерадіокомпанії з автоматичним записом ефірів, і автонаповненням із власного файлобмінника. Створено сайт для телерадіокомпанії та напрактично реалізовано на роботоздатність. З реалізовуваними перевагами сайт дає можливість проводити автозапис без втручання оператора зі збереженням запису зйомки на сайті чи в файлообміннику. На базі реалізації сайту досягнуто повної автоматизації праці операторів та економії часових затрат під час відтворення. Від оператора вимагається тільки початкова підготовка апаратури і старт, а отримання відео-матеріалу здійснюється автоматично на власний файлобмінник, не думаючи про оренду серверів або накопичувачі інформації. Доведено, що програмне і апаратне забезпечення є надійним і захищеним, включаючи власний архів.
Ключові слова: сайт, автоматичнй запис, прямий ефір, автонаповнення, файлобмінник, RSS.

Мельник В.М., Полищук Н.Н., Здолбицкий А.П., Желобицкий Я.К. Сайт для телерадиокомпании с автоматической записью эфиров, и автонаполнением из собственного файлообменника. Создан сайт для телерадиокомпании и напрактически реализован на работоспособность. С реализуемыми преимуществами сайт дает возможность проводить автозапись без вмешательства оператора с сохранением съёмки на сайте или в файлообменнике. На базе реализации сайта достигнуто полной автоматизации труда операторов и экономии временных затрат при воспроизведении. От оператора требуется только начальная подготовка аппаратуры и старт, а получение видео-материала осуществляется автоматически на собственный файлообменник, не думая об аренде серверов или накопителях информации. Доказано, что программное и аппаратное обеспечение является надежным и защищенным, включая собственный архив.
Ключевые слова: сайт, автоматическая запись, прямой эфир, автонаполнение, файлообменник, RSS.

V.M. Melnyk, Polishchuk M., Zdolbitskyy A.P., Zhelobytskyy Y.K. Site for TV & radio company with automatic recording broadcasts and auto-filling from own file share. A site for a broadcasting company is created and practically implemented for a workability. In addition with the involved advantages the site enables an automatic writing without operator invention during the interview, preserving it on the site or on the file storage. On the background of the site implementation there is achieved full automation of an operator to save the time and costs during playback. The service requires only initial equipment preparations and start, but getting the video material is automatic and placing it on the own file storage without thinking about renting servers or other information storages. It is proved that the software and hardware is reliable and secure, including its own archives.
Keywords: site, automatic recording, direct broadcasting, autofilling, file sharing, RSS.

Постановка роблеми. В сучасному світі окрім морально-етичних цінностей та грошової валюти не менш важливим ресурсом для людини є її особистий час. Як кажуть, «час – це гроші» і він в наш час може вигравати за рахунок автоматизації тієї чи іншої діяльності людини, сфери впливу чи області дії. В наш час рівень розвитку технологій досяг того рівня, що передача інформації в будь-яку точку світу займає лічені секунди [1].

Сьогодні важко не зауважити стрімкий розвиток технологій автоматизації та роботизації видів діяльності – це створення різноманітних дронів, роботів, роботизоваих систем, конвеєрів тощо. Безумовно, штучний інтелект, нейронні мережі та інші подібні досягнення є доволі актуальними питаннями, що інтегрують свою діяльність з багатьма сферами діяльності людини. Все це направлено на те, щоб скоротити людські ресурси, які людина затрачає, виконуючи той чи інший вид роботи, заощаджуючи людський час і капітал [2]. Здобутки та успіхи розробників в даній області діяльності, наприклад, таких як в роботах [4,5,6] полягали в тому, що перш за все їх продукт розвивається і набуває нових версій свого представлення. Інтерфейс та функціонал більш широкий та багатий різноманітними можливостями. Що стосується роботи, пов'язаної із записом ефірів, як наприклад робота [6], то тут автори досягли широкого користувальницького загалу для свого продукту, який також радує оновленнями та версіями. Відчутні оновлення спостерігаються і в функціоналі та інтерфейсі користувача-оператора з їх досить широкими можливостями.

Однак, на сьогодні актуальною залишається проблема автонаповнення сайтів ефірів, їх збереження на окремих носіях інформації та автовідтворення. Крім того, всі вищеперелічені дії потребують всесторонньої автоматизації. Важливе місце відводиться також економії часу та інших видів ресурсів, особливо при необхідності повторного автовідтворення інформації, що досить часто викликає навіть і труднощі в операторів.

Метою даної *роботи* послужило одне із особистих замовлень телерадіокомпанії «Крок назустріч» відносно телерадіокомпанії «Нова Волинь» в інтеграції з поставленим завданням під час проходження практики в розробницькій компанії «InternetDevels».

Постановка завдання. В роботі ставилося завдання створення сайту, який би інтегрував в собі відео- та аудіо-публікації, різного роду статті, інформацію про компанію і т.і. Великим внеском стала наша ідея впровадження в поставлені завдання також і автоматизації роботи операторів. Місце зберігання матеріалів також пропонується вибрати оператору за власним бажанням, а саме – створити власний файлобмінник. Згідно реконструкції деякого базового скрипта повинен відбуватися автоматичний запис ефірів без втручання оператора – що відображатиме перше інтегроване завдання. В якості другого завдання транслювання на сайт повинно здійснюватися за допомогою RSS. Основу скрипта файлобмінника було також узято із іншого доступного джерела Інтернет, який потрібно також було досить перебудувати і налаштувати для якісної узгодженої організації роботи всієї системи.

Результати та їх обговорення. Перед тим, як будемо обговорювати інструментальні підходи розробки та удосконалення процесу автонаповнення сайту, опишемо коротко його послідовність дій. Як відомо, щонеділі о 21.25 вмикається запис радіо-ефіру на сайті телерадіокомпанії «Нова Волинь» (рис. 1), який триває впродовж двадцяти хвилин. Результати прямої трансляції автоматично записуються в файл, який, в свою чергу, зберігається на власному файлобміннику. Після завершення запису трансляції, він передається на сайт, що і зумовлює суть його автонаповнення.

На рис. 1 зображена головна сторінка сайту, URL на який посилається сайт - htrk-krok.com.ua.

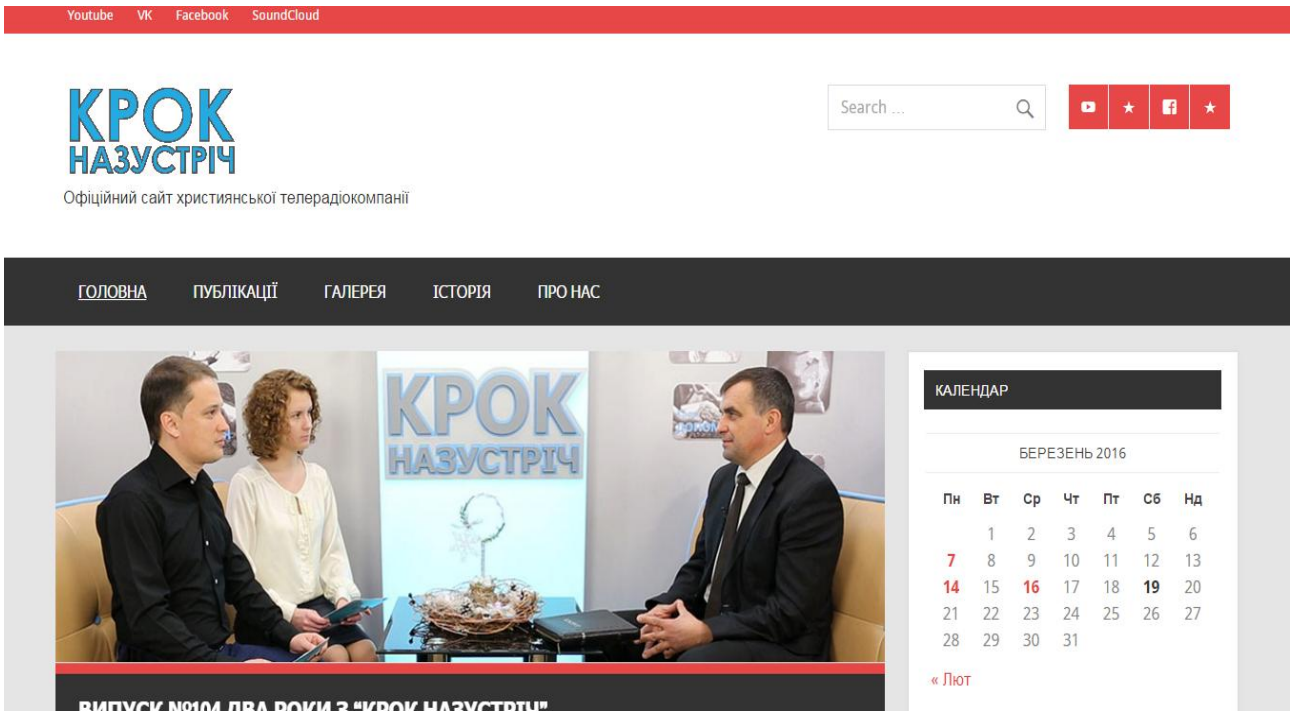


Рис. 1 – Головна сторінка сайту

Реалізація поставлених завдань здійснюється за наступними кроками. Розробка сайта була здійснена засобами PHP-5. За допомогою розробленого скрипта, основу якого було взято із одного доступного Інтернет-джерела [3], вдосконалено і спрямовано під реалізацію власної задачі. Це скрипт RadioCMS, приклад адмін-панелі якого представлений на рис. 2. Його функція полягає в тому, щоб проводити запис радіоефірів онлайн з Інтернет, чи будь-якого іншого ресурсу. Зміни в ньому були здійснені наступні: опрацьований інтерфейс та редагований і оптимізований код налаштовувалися для автоматичного запису в директорію з доступом, власне, оператора. Без змін залишилася основна функція скрипта – це засіб для запису ефірів, і головна функція. Для реалізації програмного продукту були використані інструменти середовищ розробки PHP 5 та C++.

Згідно реконструкції базового скрипта відбуватиметься автоматичний запис ефірів без втручання користувача [5]. Записаний файл має налаштування за замовчуванням, згідно яких він повинен бути розміщений в процесі роботи на власний файлобмінник. Це відбувається наступним шляхом: як

зазначалося, директорія для збереження може бути довільною з доступом для користувача-оператора, що добивається за допомогою елементарних налаштувань. Транслювання на сайт відбувається за допомогою RSS і власного «граббера» [8], який розроблявся нами самостійно. RSS (Realy Simple Syndication) – це сервіс, що представляє інформацію, і дозволяє її вилучати та відображати в потрібному місці інтернет-джерела в потрібній формі та представлені згідно з побажанями користувача. RSS-граббер (або RSS-агрегатор) – це, свого роду, додаток, який дозволяє в автоматичному режимі проводити збір інформації, яка експортується, і транслювати її у формат RSS чи Atom. В нашій роботі він виконує вищеописану роль. Налаштування його для роботи, місць проведення сканування та захоплення інформації відбувається саме з панелі адміністратора плагіну (рис. 2). Він дозволяє здійснювати взяття обраної інформації з Інтернет-джерела, яке підтримується для проведення даної операції, та відображати її в обраному користувачем місці Інтернет-ресурсу, який також обирається за замовчуванням. Для цього було обрано саме RSS-агрегатор [8], тому що, на нашу думку, він – є єдиним інструментом, що може вирішувати дану операцію найефективніше. За допомогою «крона» відбувається автоматичний запуск модифікованого PHP-скрипта, який без відома і наявності користувача сканує ділянку з RSS, де знаходяться відповідні потрібні теги, тобто в коді прописано «дати запит на ту сторінку». Після сканування території, він додає відповідний RSS собі в базу (CRON мовою PERL).

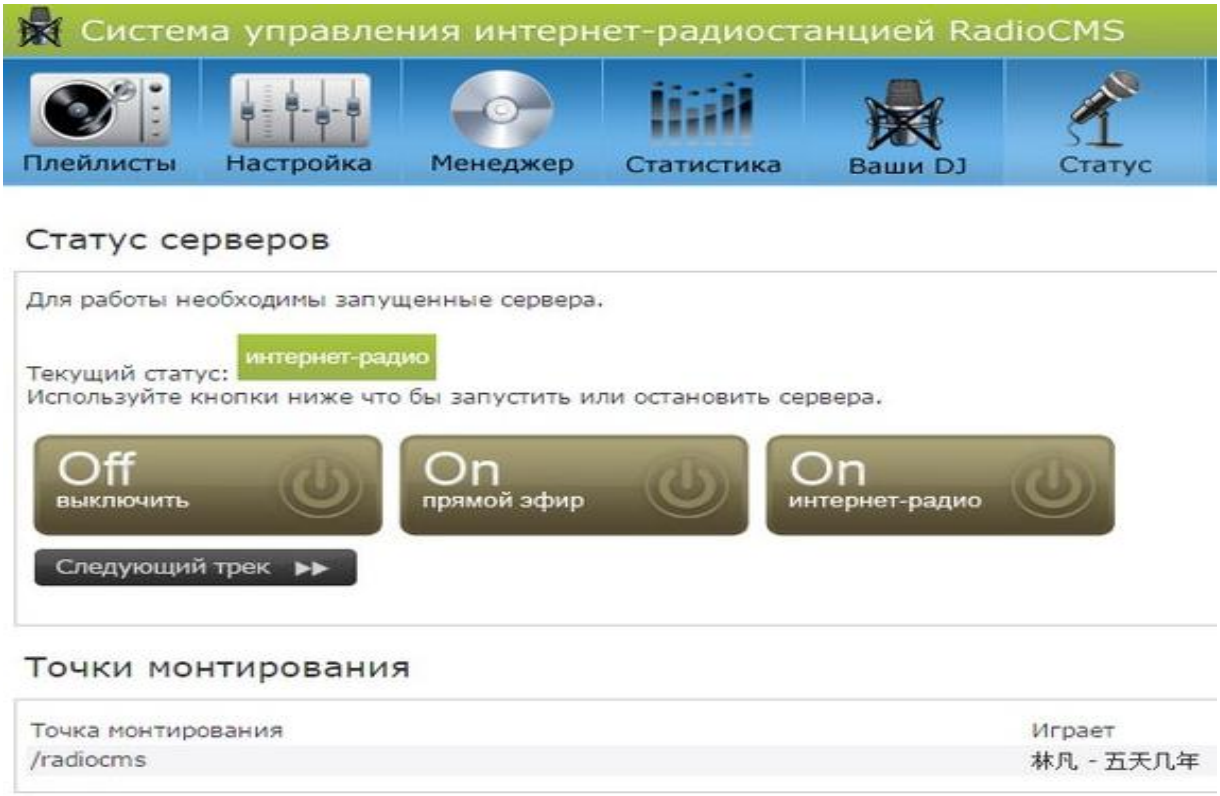


Рис. 2 – Адміністраторська панель для запису ефірів

Основу скрипта файлобмінника, приклад якого представлений на рис. 3, було також узято із іншого доступного джерела Інтернет [4], що знову ж був дещо перебудований і налаштований суто для реалізації поставленої задачі від телерадіокомпанії. Його було встановлено на веб-ресурс. Наразі, доступ до файлів можливий за посиланням www.soundcloud.com. Наш скрипт являє собою повноцінний файлообмінник з "дружнім" до користувача інтерфейсом. У вихідному скрипті були здійснені деякі виправлення в коді як збоку його оптимізації згідно поставленого завдання, так і збоку оптимізації роботи самого сервісу. Було також реалізовано можливість задавання шляху для транслювання збережених матеріалів і шляху отримання їх на файлобміннику. Інтерфейс виправлявся також з урахуванням зручності його використання користувачем. Скрипт розміщено за власниими побажаннями замовників та їх поглядами. Він представляється на подобі хостера, в якому всі файли зберігаються на хостінгу і являє собою візуальну оболонку для файлового провідника.

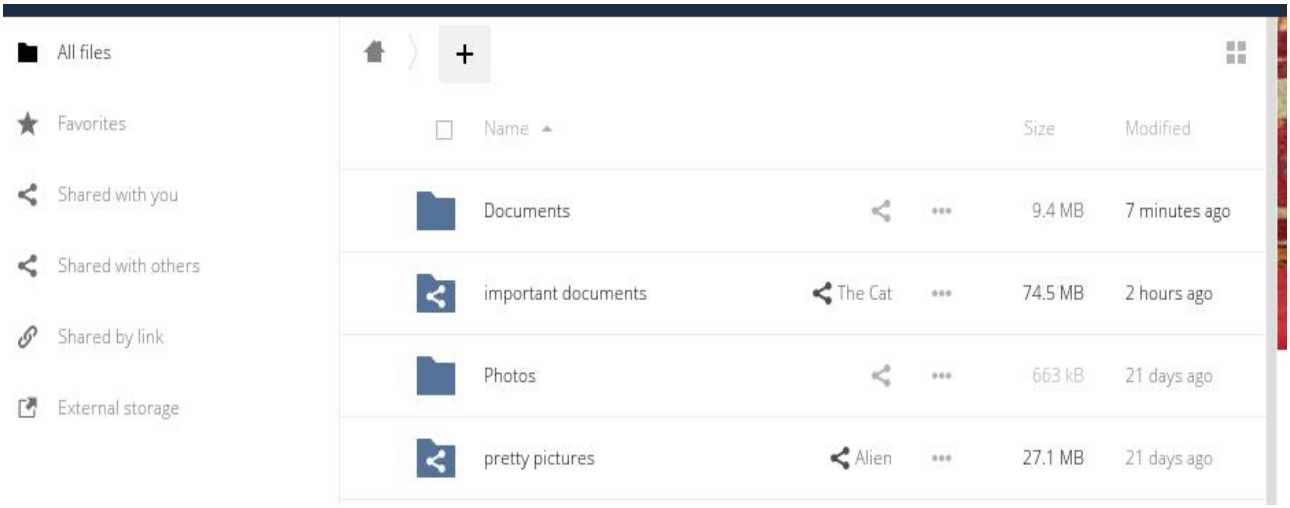
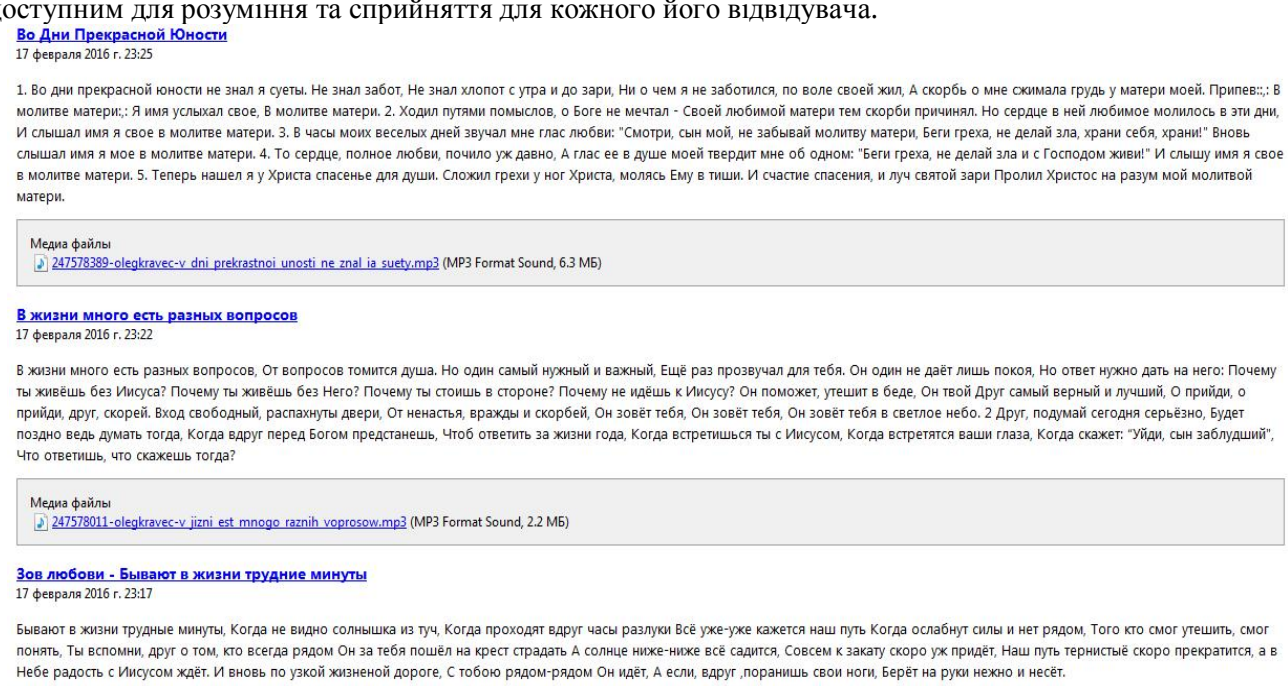


Рис. 3 – Вигляд файлобмінника для запису трансляцій

На сайт також додано декілька власних скриптів та модулів, наприклад, таких як панелі, socsharing, загальна добудова теми і т.д. Це зроблено для того, щоб покращити її зовнішній вигляд, а також зробити інтерфейс представлення зручним та гнучким для користувачів з метою відобразити сайт доступним для розуміння та сприйняття для кожного його відвідувача.


Во Дни Прекрасной Юности
17 февраля 2016 г. 23:25

1. Во дни прекрасной юности не знал я суеты. Не знал забот, Не знал хлопот с утра и до зари, Ни о чем я не заботился, по воле своей жил, А скорбь о мне сжимала грудь у матери моей. Припев:;,: В молитве матери;,: Я имя услыхал свое, В молитве матери. 2. Ходил путями помыслов, о Боге не мечтал - Своей любимой матери тем скорби причинял. Но сердце в ней любимое молилось в эти дни, И слышал имя я свое в молитве матери. 3. В часы моих веселых дней звучал мне глас любви: "Смотри, сын мой, не забывай молитву матери, Беги греха, не делай зла, храни себя, храни!" Вновь слышал имя я мое в молитве матери. 4. То сердце, полное любви, почило уж давно, А глас ее в душе моей твердит мне об одном: "Беги греха, не делай зла и с Господом живи!" И слышу имя я свое в молитве матери. 5. Теперь нашел я у Христа спасенье для души. Сложил грехи у ног Христа, молясь Ему в тиши. И счастие спасения, и луч святой зари Пролил Христос на разум мой молитвой матери.

Медиа файлы
247578389-olegkravec-v_dni_prekrastnoi_unosti_ne_znal_ia_suety.mp3 (MP3 Format Sound, 6.3 МБ)

В жизни много есть разных вопросов
17 февраля 2016 г. 23:22

В жизни много есть разных вопросов, От вопросов томится душа. Но один самый нужный и важный, Ещё раз прозвучал для тебя. Он один не даёт лишь покоя, Но ответ нужно дать на него: Почему ты живёшь без Иисуса? Почему ты живёшь без Него? Почему ты стоишь в стороне? Почему не идёшь к Иисусу? Он поможет, утешит в беде, Он твой Друг самый верный и лучший, О прийди, о прийди, друг, скорей. Вход свободный, распахнуты двери, От ненастья, вражды и скорбей, Он зовёт тебя, Он зовёт тебя, Он зовёт тебя в светлое небо. 2 Друг, подумай сегодня серьёзно, Будет поздно ведь думать тогда, Когда вдруг перед Богом предстанешь, Чтоб ответить за жизни года, Когда встретишься ты с Иисусом, Когда встретятся ваши глаза, Когда скажет: "Уйди, сын заблудший", Что ответишь, что скажешь тогда?

Медиа файлы
247578011-olegkravec-v_jizni_est_mnogo_raznih_voprosow.mp3 (MP3 Format Sound, 2.2 МБ)

Зов любови - Бывают в жизни трудние минуты
17 февраля 2016 г. 23:17

Бывают в жизни трудные минуты, Когда не видно солнышка из туч, Когда проходят вдруг часы разлуки Всё уже-уже кажется наш путь Когда ослабнут силы и нет рядом, Того кто смог утешить, смог понять, Ты вспомни, друг о том, кто всегда рядом Он за тебя пошёл на крест страдать А солнце ниже-ниже всё садится, Совсем к закату скоро уж придёт, Наш путь тернистыё скоро прекратится, а в Небе радость с Иисусом ждёт. И вновь по узкой жизненой дороге, С тобою рядом-рядом Он идёт, А если, вдруг ,поранишь свои ноги, Берёт на руки нежно и несёт.

Рис. 4 – Приклад транслювання матеріалів на сторінку

На рис. 4 наведено приклад транслювання на сторінку. Нижче представлений приклад однорядкового фрагменту коду для транслювання на сторінку одного елемента, в якому описується шлях, звідки здійснюється RSS-сервіс.

```
<rss xmlns:atom="http://www.w3.org/2005/Atom" xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd" version="2.0">
<channel>
```

Нижче, у представленому коді визначаються області обраного контенту для його обрання.

```
<atom:link href="http://feeds.soundcloud.com/users/soundcloud:users:206470194/sounds.rss" rel="self" type="application/rss+xml"/>
```

<atom:link href="http://feeds.soundcloud.com/users/soundcloud:users:206470194/sounds.rss?before=247576838" rel="next" type="application/rss+xml"/>
…

Тут представлені інші контент-записи, які сервіс транслює згідно з попередньо обраним випадком, який представлений нижче.

Код транслювання контенту здійснюється наступним чином. Обирається веб-посилання, та області «граббінгу». Зберігається необхідна інформація, а саме, час публікації, шлях, автор, назва публікації.

```
<item>
    <guid isPermaLink="false">tag:soundcloud,2010:tracks/247577185</guid>
    <title>Зов любови - Бывают в жизни трудние минуты</title>
    <pubDate>Wed, 17 Feb 2016 21:17:37 +0000</pubDate>
<link>
      https://soundcloud.com/olegkravec/zov-lyubvi-byvayut-v-zhizni-trudnye-minuty
</link>
<itunes:duration>00:04:40</itunes:duration>
<itunes:author>Oleg Kravec</itunes:author>
<itunes:explicit>no</itunes:explicit>
<itunes:summary>
```

Для прикладу наведемо текстову частину публікації.

```
ТЕКСТОВА ЧАСТИНА ПУБЛІКАЦІЇ

</itunes:summary>
<itunes:subtitle>Бывают в жизни трудные минуты, Когда не видно сол…</itunes:subtitle>
<description>

ТЕКСТОВА ЧАСТИНА ПУБЛІКАЦІЇ
```

В наступній частині коду представлені заключні теги, повторення інформації про опублікований запис, а також відомомсті про автора з посиланням на інтернет-джерело, звідки береться аватар.

```
</description>
    <enclosure type="audio/mpeg" url="http://feeds.soundcloud.com/stream/247577185-olegkravec-zov-lyubvi-byvayut-v-zhizni-trudnye-minuty.mp3" length="0"/>
    <itunes:image href="http://i1.sndcdn.com/avatars-000205731250-o4mtcw-original.jpg"/>
</item>
</channel>
</rss>
```

Споріднені роботи та обговорення. Використання трансляції у вигляді «U to U» [7], як варіант, є недоцільним, в той час як за допомогою RSS є стимуляція оптимізувати код, скоротити навантаження на процесор в порівнянні з використанням PHP-стрімів [7]. До нашої уваги під час розробки брались деякі схожі роботи з RSS-транслюванням, однак недоліком їх залишалося те, що «граббери» і сама трансляція не підтримується значним числом сайтів [4]. Наш продукт наразі є сумісним і це стало його другою цінністю розробки. Деталі сумісності полягають в тому, що плагін був розроблений «за останнім писком» «веб-моди», тобто структура і сам код виконані за новітніми методами, підходами, алгоритмами і т.і., що враховують тенденцію нинішніх веб-ресурсів. Це дозволяє йому буди сумісним майже для всіх загальновживаних ресурсів. Не менш важливим є те, що в мережі немає безкоштовних «грабберів», а якщо і знайуться такі, то програма буде не працездатною по багатьом причинам. Перша і найголовніша причина – це версії, які з кожним разом з'являються і потребують доопрацювання. В деяких випадках потрібно повністю уникнути в те, що мав на увазі програміст, і зазвичай «дописати» сервіс.

Що стосується встановлення подібних сторонніх програм, то це доволі трудомістке завдання, тому що буває дуже складно знайти на теренах Інтернет зрозумілу інструкцію встановлення. Це ж може стосуватися ситуації людей, які висвітлили власні рішення на форумах і т.п.

Наш продукт має зрозумілу інструкцію встановлення і, звичайно ж, не виникає проблем з комерційністю. Широкі можливості сучаних «грабберів» [8] не завжди доречні, але ж від цього страждає інтерфейс, що зумовлює його незручність та неприйнятність. Це можна побачити на рис. 5, де наведений приклад налаштування трансляції RSS-«граббера», взятого з одного доступного інтернет-джерела [4]. Мінімалізм, що пропонує наш продукт – це налаштування суто під потреби користувача для сумісної роботи.

Дослідженню RSS-агрегаторів посвячено і роботу [4,5], в якій здобутки та успіхи розробників в даній області діяльності полягали в тому, що перш за все їх продукт розвивається і набуває нових версій свого представлення. Інтерфейс та функціонал стає більш широкий та багатий різноманітними можливостями. Що стосується роботи, пов'язаної із записом ефірів [6], то тут було досягнуто широкого користувальницького загалу для розробленого продукту, який також радує оновленнями та версіями. Відчутні оновлення спостерігаються і в функціоналі та інтерфейсі користувача-оператора з їх досить широкими можливостями.

Недостатком із наведених робіт є те, що в них не реалізовано автозапису без втручання оператора. Потрібно шукати широкозагальний доступний канал, керувати його налаштуваннями, тримати ефір на записі, а оператора – постійно на своєму робочому місці. Далі приходиться опрацювати записаний файл для досягнення бажаного результату, завантажити його в потрбіну директорію, або ж за допомогою нашого скрипта RadioCMS [5] є можливість без втручання користувача зберігати записані файли в обрану директорію за допомогою API. Зауважимо, все це потребує часу, матеріальних, людських та інших ресурсових затрат. Втрати також спостерігаються при становленні стороннього подібного програмного забезпечення. Однак, наша розробка має чітку інструкцію та гнучкі можливості як в процесі її встановлення, так і в процесі використання.

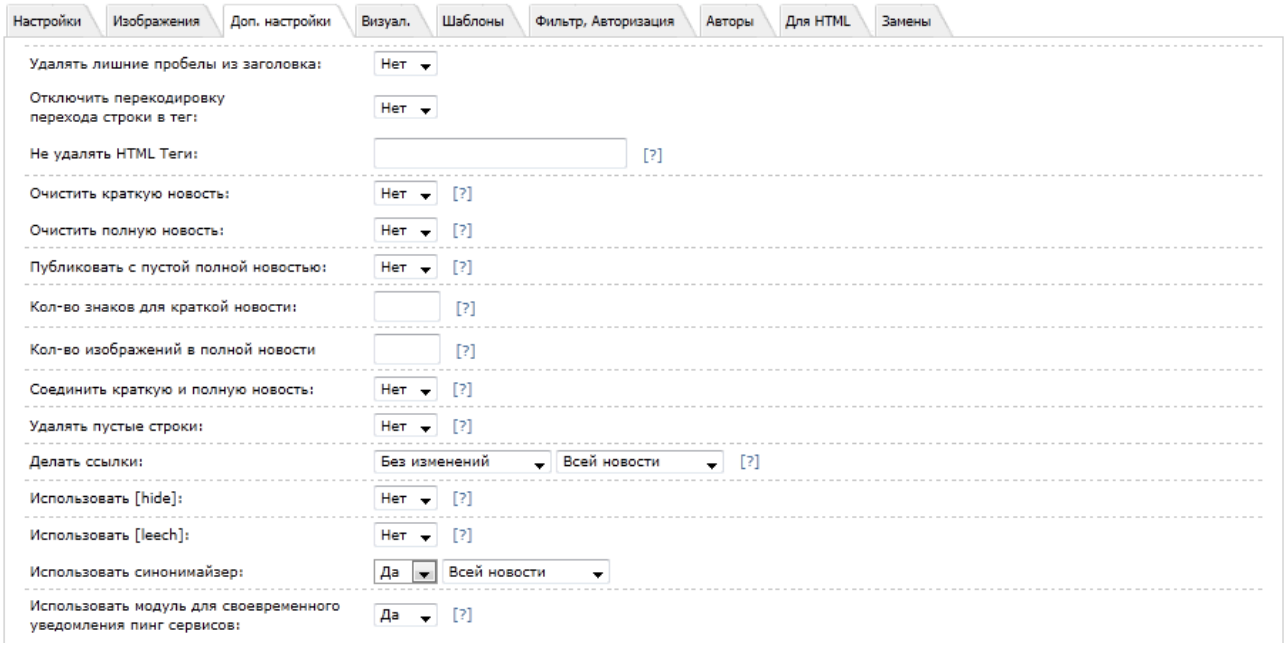


Рис. 5 – Приклад налаштування «граббера»

Порівняно з роботами [4-6] проект нашої розробки пропонує реалізацію запису ефірів ТБ\відео як безпосередньо з сайту трансляції, так і з youtube, наприклад. Транслювання може відбувтися не тільки для сайту на «чистому» PHP. Також дається можливість збереження файлу безпосередньо на сайт, або ж в будь-яку іншу обрану (чи віддалену) директорію (комп'ютер, накопичувач і т.д.). Вдосконалений нами «граббер» налаштовується стикує з CMS WordPress через використання відповідного самостійно розробленого плагіна, який є безкоштовним. А тому, запропоновану в роботі ідею також можна з легкістю реалізовувати і на CMS.

Порівнюючи наші попередні роботи, хочеться відмітити і такі недоліки та переваги нинішньої роботи. За час розробки було досягнуто використання сервера на основі «пінгвінчика» Debian для коректної роботи програми з метою запису ефірів. Зсилаючись на роботу [8], нами був прийнятий раціональний підхід при виборі рішення та використанні «граббера» та плагінів для оптимізації та покращення роботи транслятора і, відповідно, трансляції. Також нами було виявлено численні атаки на сайт. Захист сайту успішно запобігали плагіни, такі як Advance Login Style, Limit Login Attempts і т.д.

Висновок. Створений та випробуваний на роботоздатність сайт для телерадіокомпанії з реалізацією ряду налаштувань та переваг, які дають можливість проводити автозапис без втручання оператора з автозбереженням з'йомки на сайті чи в файлообміннику.

На базі реалізації сайту досягнуто повної автоматизації праці операторів, яке полягає в збереженні матеріалу та економії часових затрат під час відтворення. Оператору потрібно тільки стартувати і отримати відео-матеріал на власний файлобмінник, не думаючи про оренду серверів та накопичувачі інформації. Доведено, що програмне і апаратне забезпечення є надійним і захищеним, включаючи власний архів.

1. Мельник В.М., Желобицький Я.К. Сайт для телерадіокомпанії з автоматичним записом ефірів, і автонаповненням з власного файлобмінника // Збірник тез доповідей міжнародної науково-практичної конференції студентів і молодих учених «Інноваційні напрямки розвитку освіти, сфери послуг і технологій» / Укладачі: Л. І. Стешенко, А. В. Голуб, С. О. Кізим. - 2016. – 372с.
2. Каганюк А.К., Желобицкий Я.К Внедрение инновационных технологий струнного транспорта Юницкого «Skyway» в существующую транспортную систему // Міжвузівський збірник "Комп'ютерно-інтегровані технології: освіта, наука, виробництво – Луцьк: Видавництво ЛНТУ. Вип. 21. – 2015. – С. 75-81.
3. Багнюк Н.В., Желобицький Я.К., Кравець О.Р. URL-Shortener // Міжвузівський збірник тез доповідей "Комп'ютерно-інтегровані технології: освіта, наука, виробництво – Луцьк: Видавництво ЛНТУ. Вип. 22. – 2016. – С.
4. Шолом П.С., Желобицький Я.К., Супронюк В. Створення терморегулятора фірми «OPAL» // Міжвузівський збірник тез доповідей "Комп'ютерно-інтегровані технології: освіта, наука, виробництво – Луцьк: Видавництво ЛНТУ. Вип. 22. – 2016. – С.
5. Каганюк О.К., Желобицький Я.К., Впровадження іноваційних технологій струнного транспорту Юницького «Skyway» в існуючу транспортну систему // Збірник наукових робіт / виробництво – Х. Українська державна академія залізничного транспорту. – 2016. – С.
6. Електронний ресурс «Ламеркомп»: http://www.lamerkomp.ru/load/ internet/rss_agregatory/53
7. Електронний ресурс «ІнстантСіЕмЕс»: http://www.instantcms.ru/forum /thread13200.html
8. Електронний ресурс «МайДів»: http://soft.mydiv.net/win/collections/ show-Zapis-radio.html
9. Електронний ресурс «ПіЕйчПі» : http://php.net/manual/ru/book. stream.php
10. Електронний ресурс «Вікіпедія»: https://ru.wikipedia.org/wiki/RSS-агрегатор
11. Електронний ресурс «Вікіпедія»: https://ru.wikipedia.org/wiki/RSS\

УДК 681.515.8
Мельник В.М., Шклярський Б.М.
Луцький національний технічний університет

ПРОГРАМА УПРАВЛІННЯ МАГАЗИНОМ

Мельник В.М., Шклярський Б.М. Програма управління магазином. Прооблено дослідження та розробка програмного продукту обслуговування магазину було реалізовано для потреби малих торгових компаній. Недорогій та доступний у користуванні програмний продукт управління магазином був спроектований в програмному середовищі розробки C++ Builder 2009, який здійснює взаємодію з двохтабличною базою даних, розробка якої основана на технології ADO. В програмі реалізовано найнеобхідніші операції купівлі-продажу – це додавання, видалення товару, переміщення товару в кошик при покупці, очищення клієнтського кошика, а також операції пошуку товару в базі товарів та видалення непотрібних товарних позицій. Для зручності користування розроблено спрощену і доступну форму для адміністрування загальної бази даних магазину.
Ключові слова: програма управління, торгівля, автоматизація, управління торгівлею.

Мельник В.М., Шклярский Б.М. Программа управления магазином. Проделана исследования и разработка программного продукта обслуживания магазина было реализовано для нужд малых торговых компаний. Недорогой и доступный в пользовании программный продукт управления магазином был спроектирован в программной среде разработки C ++ Builder 2009, который осуществляет взаимодействие с двохтабличною базой данных, разработка которой основана на технологии ADO. В программе реализовано необходимые операции купли-продажи - это добавление, удаление товара, перемещение товара в корзину при покупке, очистки клиентского корзины, а также операции поиска товара в базе товаров и удаления ненужных товарных позиций. Для удобства пользования разработана упрощенная и доступную форму для администрирования общей базы данных магазина.
Ключевые слова: программа управления, торговля, автоматизация, управление торговлей.

Melnyk V.M., Shklyarskyy B.M. Store management program. Synchronization of research and development of software maintenance shop was implemented for the needs of small trading companies. Cheap and accessible to use management software store was designed in the software development environment C ++ Builder 2009, which interacts with dvohtablychnoyu database, the development of which is based on technology ADO. The program implemented the most necessary purchases and sales - is the addition, removal of goods, movement of goods in the basket the purchase, cleaning trash client and search operation in a product catalog and remove unwanted headings. For convenience, developed a simplified and accessible form for administration of a common database store.
Keywords: program management, trading, automation, control trade.

Постановка наукової проблеми: Для автоматизації купівлі та продажу в будь-якому магазині завжди потрібне спеціальне програмне забезпечення, яке зазвичай знаходиться в декількох видах відображення, представляючи різні інтереси покупців чи просто відвідувачів.

Здійснювати контроль в магазині Ви не зможете належним чином, не маючи програми для магазину.

В зв'язку з розвитком комп'ютерних технологій, все більше підприємців почали цікавитись автоматизацією усіх процесів купівлі-продажу. Враховуючи це, на програмному ринку постала проблема розробок недорогої, якісної і, найголовніше, зручної та доступної в користуванні програмої продукції для управління магазином.

На сучасному етапі ведення торгівлі немислиме без засобів автоматизації. Система автоматизації магазину дозволяє спростити щоденну працю персоналу, забезпечити високий рівень управлінського і фінансового обліку і контролю роботи з постачальниками, знизити ресурсові витрати, оптимізувати торговий процес, підвищити ефективність бізнесу, а також лояльність покупців завдяки більш швидкому і якісному обслуговуванню. Тому кожен підприємець прагне забезпечити свій магазин недорогою, якісною і простою в користуванні програмою обслуговування.

Створення програм такого типу, пов'язане із використанням великих об'ємів інформації та зв'язку з об'ємними базами даних. Поряд з тим необхідно забезпечити захист інформації в базах даних та знизити до мінімуму ймовірність її витоку та несанкціонованого доступу до неї.

Більшість додатків такого типу створюються за допомогою технології ADO (*ActiveX Data Objects*), розробленої компанією Microsoft. Дана технологія дозволяє представляти дані з різних джерел в об'єктно-орієнтованому вигляді, що є дуже зручним в роботі з базами даних.

На даний час існує багато програм для управління магазином. Проте, більшість з них є дорогими і орієнтовані на великі торгові точки. Розроблена ж програма орієнтована на дрібних торговців і є дешевою.

Аналіз останніх досліджень і публікацій. Аналізуючи останні дослідження і розробки подібних програм було виявлено, що статті і публікації подібних продуктів не є вільно доступними. Здебільшого це пов'язано з тим, що подібні програми є комерційними і ,в основному, розробляються з орієнтацією на великі підприємства. Проте деякі компанії все ж надають вичерпну інформацію стосовно своїх продуктів на сайтах.

Наприклад, розробник програми «Мой склад» на своєму сайті описує можливості даної програми і навіть дає змогу встановити пробну версію[1]. Але дана програма є платною і має багато зайвих і незрозумілих функцій, які недоречні для малого бізнесу. Доволі цікаве рішення пропонує компанія abmcloud на своєму сайті [2]. Суть ідеї полягає у встановленні програми управління на віддаленому сервері. Але ,як зазначалось вище, більше інформації про дану програму розробники не дають у вільний доступ. У цій сфері свою програму також пропонує компанія RS-BALANCE, але, як зазначено на їх сайті для роботи з програмою потрібне навчання, що свідчить про її складність [5]. Найбільш простими і зрозумілими є програмні продукти TRUE SHOP [3] та програма команди розробників USU [4]. Дані програми мають найпростіший інтерфейс з усіх наведених вище, проте також потребують деяких спеціальних навиків для роботи з ними.

Виділення невирішених раніше частин загальної проблеми. Як вже зазначалось вище, більшість програм подібного характеру є платними і складні в користуванні. Дана розробка покликана спростити роботу дрібних торгових точок. Вона має зрозумілий і легкий в користуванні інтерфейс. Тобто для роботи з програмою не потрібно ніяких спеціальних знань чи навичок. Всі функції компактно зібрані в одному вікні, що значно полегшує роботу і є основою з відмінностей від інших наявних продуктів. Застосування технології ADO дозволяє збільшити швидкість виконання завдань [6].

Мета та завдання роботи (статті)

Основною метою даної роботи є створення додатку для магазину, який би міг полегшити роботу закладу. Також проведений порівняльний аналіз розробленої програми з існуючими, в ході якого виявлені основні переваги і недоліки першого.

Виклад основного матеріалу й обґрунтування отриманих результатів розробки.

Магазин є кінцевим пунктом збуту товару. Кожний магазин має свій асортимент товару для роздрібної торгівлі. В магазині присутні товари загалом різного виду, в залежності від виду торгівлі, що представляється самим магазином. В даній роботі розглянемо магазин, який торгує опалювальною технікою. До основних наявних товарів належать: бойлери, котли, колонки, радіатори, каміни, стабілізатори напруги до котлів, запасні частини до опалювальної техніки та інші комплектуючі.

Продажа товару покупцеві здійснюється наступним чином:

1. Прийняття товару в магазин – товари поступають на склад за накладними його поставки;
2. Безпосередня продажа з магазину покупцеві – відбувається за видатковими чеками.

Товари зберігаються до їх продажу в складських приміщеннях або на вітринах магазину. Облік магазину передбачає наступні етапи: додавання даних про товар в магазині при його приході, видалення даних про товар з товарообліку магазину, зберігання даних про товар, можливості їх сортування, перегляду та редагування тощо.

Кожен з цих етапів є досить необхідним і відповідальним для ведення загального обліку в магазині. Для реалізації намічених вище пунктів ставилося в роботі завдання створення програми, яка могла б в повному обсязі забезпечувати виконання всіх описаних операцій купівлі та продажу з можливістю відображення такої інформації, як вид продукції, назва товару, ціна (за одиницю товару), Загальний інтерфейс програми зображено на рисунку 1. Як видно з рисунка, він є простим для сприйняття користувачем і легким у використанні, що мало б сприяти швидкому вивченню інформації покупцем та прийнятті рішень у здійсненні покупок, аналізу даних та товарних характеристик, що виводяться програмним продуктом для аналізу.

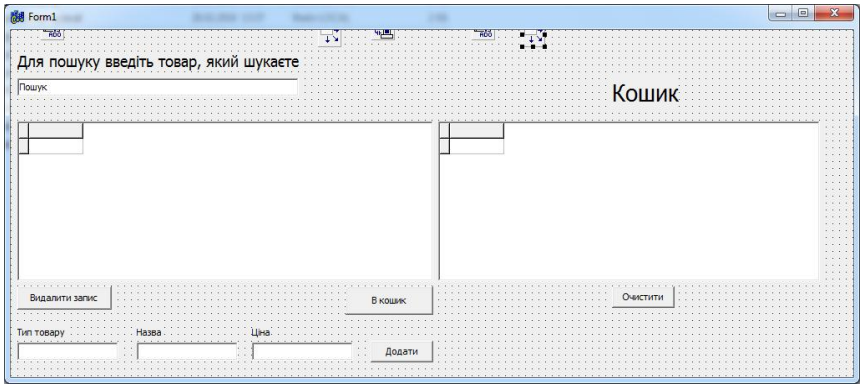


Рис.1. Вигляд вікна програми.

Як видно, в процесі розробки було додано зручну форму для додавання товару у базу даних, яка відіграє важливу роль в адмініструванні самої бази товарів. Програма та розроблена форма дозволяють користувачу-адміністратору без значних зусиль та спеціальних навичок додати новий товар до існуючої бази даних. На сьогодні ця реалізація відіграє одну із важливих особливостей подібних програмних продуктів, створених для автоматизації процесів покупки-продажу в будь-якому магазині.

Архітектура бази даних товарів подана на рис. 2. Основа її створення – програмне середовище Microsoft Access 2007. Вона інтегрує в собі дві таблиці, а саме, таблицю зі списком товарів та таблицю-кошик покупця.

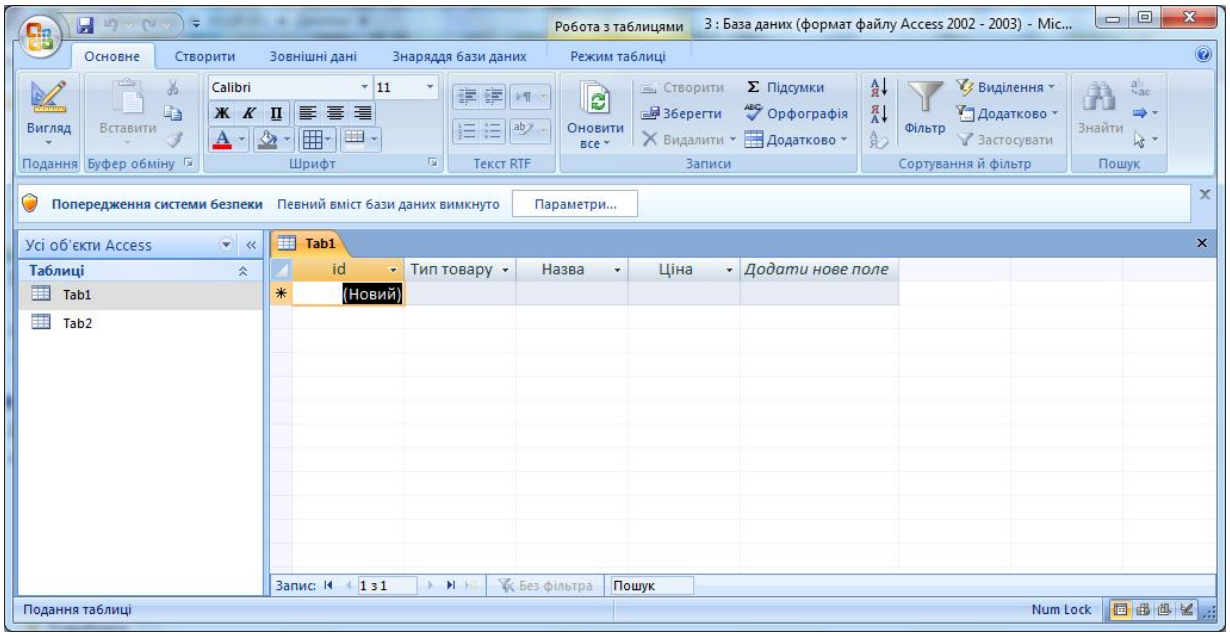


Рис.2. – Архітектура бази даних товарів.

В ході перегляду товару та в процесі аналізу його характеристик під час здійснення продажу, користувач може виділити товар зі списку, проаналізувати детальніше наведену інформацію про нього та натиснути кнопку «в кошик», що знаходиться під списком товару. Після цього вибрана позиція товару опиниться в таблиці справа. Це зручно, коли покупець купляє кілька найменувань товару. Одразу стає видно, який товар потрібно завантажити зі складу і продати в даний момент і, саме, цьому покупцю. Після завершення продажу, користувач може очистити кошик, натиснувши відповідну кнопку під таблицею. Після цього всі дані з кошика (другої таблиці) видаляться.

Також легко і доступно можна видалити будь-яку позицію товару і з основної бази даних. Наприклад, якщо товар вже не продається, то його просто потрібно виділити у списку, натиснувши кнопку «видалити запис». Після цієї дії найменування товару буде видалено із загальної бази даних.

Адміністратору магазину зовсім не потрібно витрачати на це багато часу і застосовувати якісь спеціальні знання. Для цього вистачить всього кілька кліків мишею.

Зв'язок програми з базою даних та команди операцій додавання, виділення та інші – реалізовані мовою SQL, що забезпечує швидкодію та надійність взаємодії пргограми з базою товарів. Наприклад, додавання товару у базу даних здійснюється за допомогою наступного SQL-запиту:

```
void __fastcall TForm1::Button2Click(TObject *Sender)
{
  ADOQuery1->Close();
  ADOQuery1->SQL->Clear();
  ADOQuery1->SQL->Add("INSERT INTO Tab1 ([Тип товару], [Назва],[Ціна]) VALUES
                      ('"+Edit2->Text+"', '"+Edit3->Text+"', '"+Edit4->Text+"')");
  ADOQuery1->ExecSQL();
  ADOQuery1->Close();
  ADOQuery1->SQL->Clear();
  ADOQuery1->SQL->Add("SELECT *FROM Tab1");
  ADOQuery1->Open();
}
```

Як видно з вищеописаного коду, всі операції здійснюються SQL-запитами. В середовищі розробки C++ Builder 2009 виконання таких запитів забезпечує компонент ADOQuery, що ще раз підтверджує багатогранність та широкий функціонал даного середовища розробки.

Роботи подібного характеру

На сьогодні існує уже багато програм управління магазином. Багато з них дозволяють підключати зовнішні пристрої, а саме: сканери штрих-кодів, чекові принтери, тощо. Більшість таких програм є платними, розробленими у вигляді веб-сайту, але не можна не врахувати їх ефективність та розповсюдженість.

Однією з таких програм є True Shop (рис.3) – це програма для автоматизації роздрібних магазинів[3]. Вона підходить як для автоматизації операцій купівлі-продажу малого магазину, так і для автоматизації торговельної мережі.

Основні можливості програми:

- Занесення товарів, операція продажу і повернення. Дисконтні карти.
- Статистика операцій продажу.
- Інвентаризація.
- Можливість паралельної роботи декількох користувачів в мережі.
- Підтримка сканерів штрих-кодів, фіскальних реєстраторів.
- Можливість автоматизації мережі магазинів-філіалів. Обмін даними тут здійснюється за допомогою flash-карти або через мережу Інтернет.
- Гнучка система обмежень прав користувачів, авторизація при вході в програму.
- Обмеження на перегляд підрозділів магазину продавцями.
- Заборона на перегляд собівартості продавцями.
- Можливість здійснення покупок через Інтернет, спеціальний режим для повільних каналів зв'язку.
- Можливість занесення фотокопій товарів.
- Вивантаження товарів на сайт магазину.
- Друк цінників на звичайних принтерах і на принтерах етикеток.
- Можливість завести необмежену кількість додаткових полів товару (колір, розмір і т.д.) для конкретного магазину з реалізацією їх пошуку.

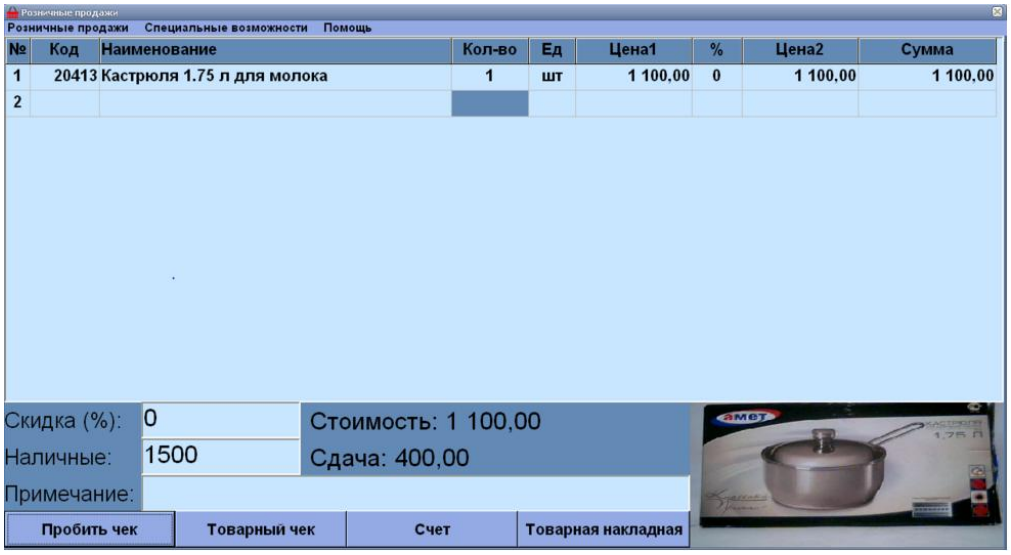


Рис.3. Вікно програми True Shop

Недоліком програми є те, що її інтерфейс є складнішим і додавати товар безпосередньо через програму неможливо. Додавання товару треба заносити у саму базу, що є незручністю для адміністратора магазину.[3]

Ще однією подібною програмою є «Мій склад» [1] (рис.4). Дана програма є платною і досить дорогою, особливо для малих та середніх магазинів. Вона також дозволяє підключити сканер штрих-коду та чековий принтер, а також дозволяє автоматизувати робоче місце касира магазину. Робота програмного додатка може активізуватися з будь-якого комп'ютера, ноутбука, нетбука і планшетника, підключеного до інтернету[1]. Зручний простий інтерфейс, який на перший погляд користувача не містить нічого зайвого. Реалізована також реєстрація операцій продажу, розрахунок здачі, робота з поверненнями, закриття зміни. Надана підтримка сканера штрих-кодів, друк товарних чеків, оф-лайн режим торгівлі і інтеграція з фіскальним реєстратором, – що є дуже зручним і значно полегшує роботу продавця. Але такі зручності є досить дорогими, потребують значних затрат і є нераціональними для невеликих підприємств.

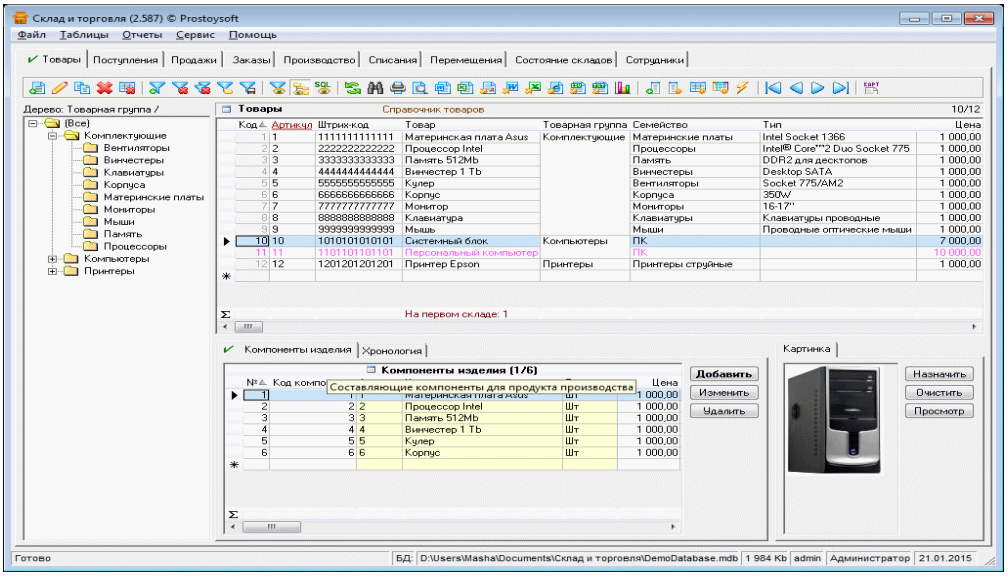


Рис.4. Вікно програми «Мій склад»

Висновки та перспективи подальшого дослідження. В результаті проробленого дослідження та розробки програмного продукту було реалізовано потребу, особливо малих торгових підприємств, у недорогій та доступно простій у користуванні програмі управління магазином, яка

була спроектована в програмному середовищі розробки C++ Builder 2009. Розроблений програмний продукт взаємодіє з двохтабличною базою даних, розробка якої основана на технології ADO. В програмі реалізовано найнеобхідніші операції купівлі-продажу – це додавання, видалення товару, переміщення товару в кошик при покупці, очищення клієнтського кошика. Поряд з цим також реалізовано пошук товару в базі даних усіх наявних товарів та видалення непотрібних товарних позицій. Для зручності користування розроблено спрощену і доступну форму для додавання найменувань товарів у загальну базу даних магазину.

На майбутнє, як і кожен продукт, дану програму можна вдосконалити або переробити під потреби конкретного магазину-споживача чи міні-торгової компанії. Середовище розробки і C++ Builder 9 та стартова структура розробленого програмного продукту дає широкі можливості для її модифікації та перспективи вдосконалення. Враховуючи потреби кінцевого споживача, можна і здійснити реалізацію більш досконалго захисту інформації за допомогою оптимізації вихідного коду та розмежування доступу обслуговуючого персоналу магазину за допомогою створення форми з іменем користувача та паролем. Також можна вдосконалювати опис товару, додавши йому додаткових властивостей, а, відповідно, і полів в таблицях бази товарів, якщо цього потребуватиме замовник.

1. http://www.moysklad.ru/avtomatizacija_magazina
2. http://abmcloud.com/управление-розницей/
3. http://osinavi.ru/index.php
4. http://usu.kz/app_programma_dlya_magazina.php
5. http://www.rs-balance.ru/wholesale/
6. Гайджарги В.І. Основи проектування та використання баз даних. К.: Політехніка, ТОВ фірма Періодика, 2004.-256 ст.

УДК 53:004.924
Муляр В. П., к. пед. н., доцент, Яцюк С. М., к. пед. н., доцент
Східноєвропейський національний університет імені Лесі Українки

ЕЛЕМЕНТИ КОМП'ЮТЕРНОЇ ГРАФІКИ У ВІЗУАЛІЗАЦІЇ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ ФІЗИЧНИХ ЯВИЩ І ПРОЦЕСІВ

Муляр В. П., Яцюк С. М. Елементи комп'ютерної графіки у візуалізації результатів моделювання фізичних явищ і процесів. У статті досліджено можливості моделювання в науці й у навчальному процесі. Розглянуто основні прийоми комп'ютерної графіки та їх використання для візуалізації результатів моделювання фізичних процесів і явищ.
Ключові слова: моделювання, комп'ютерне моделювання, комп'ютерний експеримент, комп'ютерна графіка, навчальний процес.

Муляр В. П., Яцюк С. М. Элементы компьютерной графики в визуализации результатов моделирования физических явлений и процессов. В статье исследованы возможности моделирования в науке и в учебном процессе. Рассмотрены основные приемы компьютерной графики и их использование для визуализации результатов моделирования физических процессов и явлений.
Ключевые слова: моделирование, компьютерное моделирование, компьютерный эксперимент, компьютерная графика, учебный процесс.

Mulyar V. P., Yatsyuk S. M. Elements of computer graphics in visualization of results of modeling of physical phenomena and processes. The capabilities of modeling in science and in the learning process are analyzed in the article. The basic techniques of computer graphics and their use for visualization of modeling of physical processes and phenomena are described.
Keywords: modeling, computer simulation, computer experiment, computer graphics, learning process.

Постановка проблеми в загальному вигляді. У процесі пізнання і практичної діяльності людина широко застосовує різноманітні моделі. Під моделюванням розуміють заміну вивчення явища в натурі вивченням аналогічного явища на моделі. Основний зміст моделювання полягає в тому, щоб за результатами дослідів із моделями можна отримати потрібну інформацію про досліджуваний об'єкт, безпосереднє вивчення якого становить значні труднощі або зовсім неможливе. Значно розширилися можливості цього методу з появою сучасних комп'ютерів. Вони поєднали в собі матеріальні та мислені математичні моделі, що дало змогу на цій основі одержати новий вид моделювання – комп'ютерне.

Моделювання стало невід'ємним складником навчального процесу в вищій школі. І це закономірно, адже сам процес формування знань пов'язаний із перетворенням у свідомості студента одних моделей в інші, які є похідними від перших, але точнішими, з більшим наближенням до абсолютної істини. Ознайомлення студентів із методами науки, зокрема із методом моделювання, дає їм можливість зрозуміти логіку наукового пізнання, осмислити його методологію. Моделювання дає можливість викладачеві глибше розкрити зміст фізичних понять, ознайомити студентів із сучасною експериментальною базою фізики, розкрити важливе значення методів дослідження фізичних явищ і процесів, озброїти студентів системою фізичних знань у тісному зв'язку з методами наукових досліджень.

Аналіз досліджень і публікацій. Фундаментальні дослідження в галузі моделювання зробили такі вчені як М. Алексєєв, М. Амосов, В. Вєніков, Б. Глинський, В. Глушков, Ю. Жданов, Б. Кедров, І. Новік, І. Ревзін, В. Штофф, А. Уємов та багато інших. Вагомий внесок у розвиток комп'ютерного моделювання зробив академік О. А. Самарський. Саме ним була запропонована знаменита тріада «модель – алгоритм – програма» і розроблена технологія комп'ютерного моделювання, яка використовується для вивчення фізичних явищ. Дослідженню дидактичних функцій методу моделювання присвячені праці Л. Калапуші, В. Попковича, М. Солодухіна, Л. Зоріної та інші. Вчені вважають, що моделювання в навчальному процесі з фізики має ту специфічну особливість, що воно одночасно виступає методом наукового пізнання, є частиною змісту навчального матеріалу з фізики та ефективним засобом її вивчення [1, с. 4].

Формулювання цілей статті. Мета статті – розглянути основні прийоми комп'ютерної графіки та їх використання для візуалізації результатів моделювання фізичних явищ і процесів.

Виклад основного матеріалу. Комп'ютер є ефективним засобом моделювання реальних предметів або матеріалів. Широкі можливості мають комп'ютери для розв'язання математичних задач. Як відомо, не всі задачі можна розв'язати аналітично, тобто отримати розв'язок у вигляді формул. У такому випадку використовують числові методи, за допомогою яких можна отримати лише приблизний результат. Наближені розрахунки на комп'ютерах дозволяють підвищити їхню

точність і швидкість. Як показує практика, лише з використанням комп'ютерної техніки можна розв'язати задачі: в яких за однією і тією ж формулою необхідно провести обчислення декілька разів з наступною побудовою графіків; у процесі розв'язання яких виникають рівняння високих степенів або трансцендентні рівняння, які легко розв'язуються тільки числовими методами; в яких потрібно знайти екстремуми функцій, якщо ці екстремуми неможливо знайти аналітично; у яких потрібно знайти визначений інтеграл, обчислення якого потребує використання числових методів; з оптимізації простих конструкцій і процесів; у яких необхідно застосувати числові методи обробки експериментальної залежності $y(x)$ (підбір функціонального масштабу, апроксимація методом найменших квадратів, лінійна екстраполяція, обчислення похибок тощо); у яких потрібно розв'язувати диференціальні рівняння; у яких виникає необхідність розв'язання системи лінійних рівнянь; на спектральний аналіз (розкладання в ряд Фур'є) і синтез функції за відомим спектром [2, с. 4].

Окрім виконання числових розрахунків, комп'ютери надають широкі можливості для здійснення комп'ютерного експерименту, який використовується для виконання досліджень у таких напрямах, як-от [3, с. 21–22]: розрахунок ядерних реакцій; розв'язування задач небесної механіки, астрономії та космонавтики; вивчення глобальних явищ на Землі, моделювання клімату, дослідження екологічних проблем, глобального потепління, наслідків ядерного конфлікту; розв'язування задач механіки суцільних середовищ; комп'ютерне моделювання різних технологічних процесів; розрахунок хімічних реакцій та біологічних процесів, розвиток хімічної та біологічної технологій; соціологічні дослідження, зокрема, моделювання виборів, голосування, поширення відомостей, зміна громадської думки, військових дій; розрахунок і прогнозування демографічної ситуації в країні та світі; імітаційне моделювання роботи різних технічних пристроїв; економічні дослідження розвитку підприємств.

Комп'ютерне моделювання є одним з найважливіших інструментів, що полегшує проникнення людини в таємниці науки. Моделювання дозволяє створювати вражаючі наочні образи, які сприяють розумінню досліджуваного явища і запам'ятовуванню важливих деталей у порівнянні з математичними рівняннями. Моделювання дозволяє унаочнити абстрактні закони і принципи, привернути увагу дослідника до тонких деталей досліджуваного явища, які вони не помічають під час безпосереднього спостереження. Графічне відображення результатів моделювання на екрані комп'ютера одночасно з анімацією досліджуваного явища чи процесу дозволяє досліднику легко сприймати великі обсяги змістовної інформації [4, с. 46].

Універсальних систем комп'ютерної наукової графіки, мабуть, немає через величезну різноманітність задач. Часто програми, які реалізовують наочне зображення розв'язку наукової задачі (майже завжди за результатами математичного моделювання), вбудовуються всередину основної програми, записуються на тій самій мові програмування.

Загальну мету наукової графіки можна сформулювати так: зробити невидиме й абстрактне «видимим». За допомогою машинної графіки можна побачити розподіл температури всередині неоднорідно нагрітого тіла складної форми без введення в нього сотень мікродатчиків, побачити розподіл металевих руд під землею без розкопок, розкрити будову невідомої планети за результатами радіолокації тощо. При цьому зрозуміло, що застосуванню машинної графіки повинна передувати математична обробка результатів експерименту, зокрема, створення відповідної математичної моделі і домовленість про сприйняття певних умовностей на рисунку. Більше того, зображення на екрані комп'ютера для тих, хто розуміє всю міру його умовності, приносить більшу користь, ніж тисячі чисел, що є результатом математичних обчислень.

Для розв'язання відносно нескладних задач не обов'язково застосовувати різноманітні стандартні пакети машинної графіки, оскільки при цьому часто виникає проблема поєднання різних систем програмування, яку не завжди легко розв'язати. Доцільніше, мабуть, орієнтуватися на ту мову програмування, на якій реалізується математична модель.

Наведемо декілька прикладів, які ілюструють можливості комп'ютерної графіки у візуалізації результатів моделювання фізичних процесів і явищ [5, с. 686–690].

Траєкторії руху тіл, графіки. У багатьох задачах доречно ілюструвати процес моделювання зображеннями рухомих об'єктів та їхніми траєкторіями. Обмежимося випадками плоских (двомірних) рухів, які легко відобразити на плоскому екрані комп'ютера.

Опишемо загальні моменти побудови графіків і траєкторій. Нехай у результаті обчислень ми отримали межі значень координат $[x_{\min}, x_{\max}]$ і $[y_{\min}, y_{\max}]$, та таблицю значень x і y в деякі моменти

часу $0, \tau, 2\tau, \ldots, n\tau$. Потрібно побудувати графіки залежності $x(\tau)$, $y(\tau)$ та траєкторію руху тіла. Проілюструємо це, використовуючи процедури *Delphi*.

Незвичайна орієнтація «екранної системи» координат створює певні проблеми під час побудови графіків і траєкторій. Ми хочемо виводити їх і задавати координати точок за «природною системою» координат *x, y*, зображеної на рис. 1, а графічні процедури (*Ellipse, MoveTo, LineTo, Rectangle* та ін.) сприймають аргументи в «екранній системі» *x', y'*.

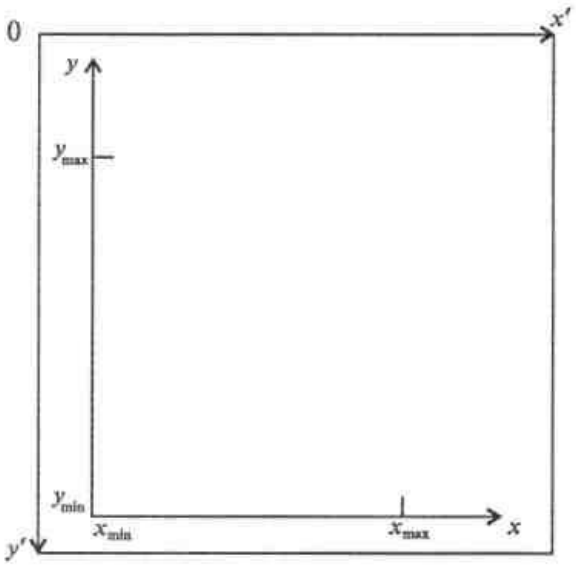


Рис. 1. Екранна і «природна» системи координат

Зробимо розмітку так, як показано на рисунку, і виконаємо лінійне перетворення координат:

$$x' = \alpha x + \beta;$$

$$y' = \chi y + \delta.$$

Якщо відомі роздільна здатність екрана – *M* точок по осі *x'* і *N* точок по осі *y'*, то для знаходження коефіцієнтів $\alpha, \beta, \chi, \delta$ достатньо зв'язати будь-які дві точки в різних системах координат, наприклад

$$\left(x = x_{\min}, y = y_{\min}\right) \Rightarrow \left(x' = 10, y' = N - 10\right);$$
$$\left(x = x_{\max}, y = y_{\max}\right) \Rightarrow \left(x' = M - 10, y' = 10\right)$$

(відступ на 10 позицій від межі екрана дасть можливість створювати підписи, розмітку осей та ін.). Маємо:

$$\begin{cases} 10 = \alpha \cdot x_{\min} + \beta \\ M - 10 = \alpha \cdot x_{\max} + \beta, \end{cases}$$

звідки

$$\alpha = \frac{M - 20}{x_{\max} - x_{\min}},$$

$$\beta = \frac{10x_{\max} - (M - 10)x_{\min}}{x_{\max} - x_{\min}}.$$

$$\begin{cases} N - 10 = \gamma \cdot y_{\min} + \delta \\ 10 = \alpha \cdot y_{\max} + \delta. \end{cases}$$

Маємо:

$$\gamma = \frac{20 - N}{y_{\max} - y_{\min}},$$

$$\delta = \frac{(N - 10)y_{\max} - 10y_{\min}}{y_{\max} - y_{\min}}.$$

Отже, переведення одних координат в інші здійснюється за формулами:

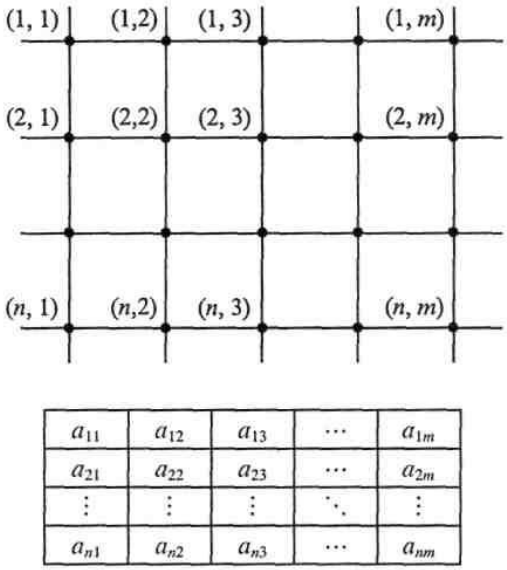
$$x' = \frac{(M-20)x + 10x_{\max} - (M-10)x_{\min}}{x_{\max} - x_{\min}},$$

$$y' = \frac{(20-N)y + (N-10)y_{\max} - 10y_{\min}}{y_{\max} - y_{\min}}.$$

Тепер достатньо поставити точку з потрібною координатою (x, y) за допомогою процедури *Pixels*, а ввівши її в цикл, зобразити графік або траєкторію. Якщо ж потрібно зобразити рух тіла, перед виведенням на екран зображення тіла достатньо стерти попереднє або скористатися іншими відомими прийомами програмування.

Ізолінії. У задачах моделювання часто виникає стандартна проблема побудови ліній (поверхонь), уздовж яких деяка функція має однакове значення, т. зв. ізоліній (ізоповерхонь). Це дуже поширена задача візуалізації характеристик деякого скалярного поля в наближенні суцільного середовища: ізотерми – лінії рівної температури; ізобари – лінії рівного тиску; ізолінії функції струму рідини або газу, за якими легко можна уявити собі їх потоки тощо.

Проаналізуємо типову процедуру побудови ізоліній на екрані комп'ютера. Нехай ми маємо двомірну таблицю значень деякої величини *A*, отриману в ході математичного моделювання; числа в цій таблиці відповідають значенням цієї величини у вузлах просторової сітки (рис. 2).



a_{11}	a_{12}	a_{13}	$\cdots$	a_{1m}
a_{21}	a_{22}	a_{23}	$\cdots$	a_{2m}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
a_{n1}	a_{n2}	a_{n3}	$\cdots$	a_{nm}

Рис. 2. Просторова сітка і відповідна їй таблиця значень величини *A*

Задамо деякий, абсолютно умовний, просторовий крок *h* між сусідніми вузлами по горизонталі й допоміжну систему координат, у якій вузол $(1,1)$ має координату $(0,0)$; вузол $(1,2)$ – координату $(h,0)$; вузол $(1,3)$ – координату $(2h,0)$ і т. д. Якщо крок по вертикалі h^*, то вузол (i,j) в цій системі має координату $((j-1)\cdot h, (i-1)\cdot h^*)$.

Заздалегідь знайдемо в таблиці найбільше й найменше значення величин a_{ij} – припустимо – це $a_{\min}$ і $a_{\max}$. Нехай b – деяке проміжне значення $a_{\min} < b < a_{\max}$. Розглянемо в загальних рисах, як побудувати ізолінію $A = b$. Будемо для цього (в циклі) проглядати спочатку всі пари найближчих чисел у першому рядку таблиці в пошуках такої пари, для якої b знаходиться «всередині». Припустимо, число *b* знаходиться між $a_{1,k}$ і $a_{1,k+1}$, тобто або $a_{1,k} < b < a_{1,k+1}$, або $a_{1,k} > b > a_{1,k+1}$.

За допомогою лінійної інтерполяції знайдемо відповідну горизонтальну координату точки, у якій $A = b$:

$$x = (k-1)\cdot h + \frac{b - a_{1,k}}{a_{1,k+1} - a_{1,k}} h$$

(координата *y* визначається номером горизонтальної лінії; у цьому разі $y = 0$).

Знайдені координати запам'ятаємо й переглянемо перший рядок у таблиці до кінця, потім проглянемо другий рядок і т. д. Закінчивши з переглядом рядків, ми отримаємо частину точок, які відповідають ізолінії $A = b$.

Після цього займемося переглядом стовпців. Припустимо, у другому стовпці знайшлася пара чисел, для якої число b знаходиться між $a_{p,2}$ і $a_{p+1,2}$. Вона дає наступну точку ізолінії. Закінчивши перегляд усіх стовпців, ми отримаємо максимально можливий набір координат точок, які належать цієї ізолінії. Вивівши їх на екран у потрібному масштабі, отримаємо точкове зображення ізолінії $A = b$, після чого можемо, узявши інше значення b, побудувати наступну ізолінію.

Умовні кольори, умовне контрастування. Ще один цікавий прийом сучасної наукової графіки – умовне розфарбовування. Він знаходить широке застосування в різних застосуваннях науки і є набором прийомів максимально зручної, хоча й дуже умовної, візуалізації результатів комп'ютерного моделювання.

Наведемо приклади. У різних дослідженнях температурних полів постає проблема наочного представлення результатів. Найпростіший – представити карту (креслення, план), у деяких точках якої позначені значення температури.

Інший спосіб – набір ізотерм – набагато ефективніший; до нього вдаються деякі газети, які подають стан і прогноз погоди. Але можна добитися ще більшої наочності, враховуючи, що більшості людей властиво, порівнюючи різні кольори, сприймати червоний як «гарячий», голубий як «холодний», а всі інші – між ними. Припустимо, що на деякій території температура в певний момент має в різних місцях значення від –25 °С до +15 °С. Розділимо цей діапазон на ділянки з кроком, який дорівнює, наприклад, 5° [–25, –20], [–20, –15]..., [+10, +15], і зафарбуємо перший із них у яскраво-голубий, останній – у яскраво-червоний, а всі інші – у проміжні відтінки голубого й червоного кольорів. У результаті отримаємо наочну картину температурного поля. Те саме можна робити, ілюструючи температурне поле і на поверхні оброблюваної на верстаті деталі, і на поверхні далекої планети. До подібного прийому візуалізації вдаються під час моделювання явища теплопровідності у стрижні, розподілу електричних полів.

Умовне розфарбовування може бути ще абстрактнішим, ніж в описаних вище випадках. При моделюванні складних органічних молекул комп'ютер може видавати результати у вигляді різнобарвної картини, на якій атоми водню зображено одним кольором, вуглецю – іншим і т. д., причому атом зображено кулькою (кружечком), у межах якої густина кольору змінюється відповідно до розподілу електронної густини.

Зображення в умовних кольорах і контрастах є потужним прийомом наукової графіки, який дає змогу зрозуміти будову не тільки плоских, а й об'ємних (тримірних) об'єктів. Зокрема, під час пошуку корисних копалин методами аерофотознімання з літаків або космічних супутників комп'ютери будують умовні кольорові зображення розподілу густини під поверхнею Землі. Подібних прикладів можна навести дуже багато.

Висновки. Комп'ютерне моделювання є потужним інструментом пізнання якісних і кількісних закономірностей природи. Важливим його етапом після завершення обчислень є усвідомлення результатів, представлення їх в максимально наочній і зручній для сприйняття формі. Візуалізація характеристик полів в наближенні суцільного середовища, зображення в умовних кольорах, представлення результатів у вигляді графіків, діаграм, траєкторій руху динамічних об'єктів дозволяє виділити й відобразити найважливіші для пізнання зв'язки в явищах, які часто бувають недоступними для безпосереднього спостереження, осмислити суть багатьох фізичних процесів.

1. Калапуша Л. Р. Комп'ютерне моделювання фізичних явищ і процесів / Л. Р. Калапуша, В. П. Муляр, А. А. Федонюк / Навч. посіб. для студ. вищих навч. закл. – Луцьк: РВВ «Вежа» Волин. нац. ун-ту ім. Лесі Українки, 2007. – 192 с.

2. Бурсиан Э. В. Задачи по физике для компьютера: Учеб. пособ. для студ. физ.-мат. пед. ин-тов / Э. В. Бурсиан.– М.: Просвещение, 1991.– 256 с.

3. Майер Р. В. Компьютерное моделирование: Учеб. пособие для студ. пед. вузов / Р. В. Майер. – Глазов, ГГПИ: 2014. – 531 с.

4. Булавин Л. А. Компьютерное моделирование физических систем / Л. А. Булавин, Н. В. Выгорницкий, Н. И. Лебовка. – Долгопрудный: Изд. дом «Интеллект», 2011. – 352 с.

5. Могилев А. В. Информатика: Учеб. пособие для студ. пед. вузов / А. В. Могилев, Н. И. Пак, Е. К. Хеннер; Под ред. Е. К. Хеннера. – 2-е изд., стер. – М.: Издательский центр «Академия», 2003. – 816 с.

УДК 004.43
Погорелов В.В., Марченко О.І.
Національний технічний університет України "Київський політехнічний інститут"

ОГЛЯД ВНУТРІШНІХ ФОРМ ПРЕДСТАВЛЕННЯ ПРОГРАМИ ДЛЯ ТРАНСЛЯЦІЇ З ПРОЦЕДУРНИХ МОВ ПРОГРАМУВАННЯ У ФУНКЦІОНАЛЬНІ МОВИ

В.В. Погорелов, О.І. Марченко. Огляд внутрішніх форм представлення програми для трансляції з процедурних мов програмування у функціональні мови. У статті, в контексті вирішення задачі трансляції з процедурних мов програмування у функціональні мови, розглянуто ряд внутрішніх форм представлення програми (ВФПП) у вигляді орієнтованих графів: граф потоку управління, форма статичного однократного присвоєння, розширена форма статичного однократного присвоєння та граф залежностей значень і станів. Запропоновано набір критеріїв для вибору ВФПП та проаналізовано відповідність вищезазначених ВФПП набору цих критеріїв.
Ключові слова. Граф потоку управління, форма статичного однократного присвоєння, розширена форма статичного однократного присвоєння, граф залежностей значень, граф залежностей значень і станів, внутрішнє представлення, трансляція програм.**Форм. 0, Таб. 1, Рис. 4, Літ. 21.**

В.В. Погорелов, А.И. Марченко. Обзор внутренних форм представления программы для трансляции из процедурных языков программирования в функциональные языки. В статье, в контексте решения задачи трансляции из процедурных языков программирования в функциональные языки, рассмотрен ряд внутренних форм представления программы (ВФПП) в виде ориентированных графов: граф потока управления, форма статического однократного присвоения, расширенная форма статического однократного присвоения и граф зависимостей значений и состояний. Предложен набор критериев для выбора ВФПП и проанализировано соответствие вышеупомянутых ВФПП набору этих критериев.
Ключевые слова. Граф потока управления, форма статического однократного присвоения, расширенная форма статического однократного присвоения, граф зависимостей значений, граф зависимостей значений и состояний, внутреннее представление, трансляция программ.

V.V Pogorelov, O.I. Marchenko. Survey of intermediate forms of program representation for translation from procedural programming languages into functional languages. The paper deals with the analysis of a range of intermediate representation forms (IRF) as oriented graphs in the context of translation of a program from procedural programming languages into the functional programming languages. In particular, the following graphs are examined: control flow graph, single static assignment, gated single assignment, value state dependence graph. A set of criteria for IRF selection is proposed and correspondence of the aforementioned IRFs to the set of criteria is analyzed.
Keywords. Control flow graph, single static assignment, gated single assignment, value dependence graph, value state dependence graph, internal representation form, translation of programs.

Вступ і постановка проблеми. На сьогоднішній день, в області задач сучасного системного програмування, знаходять застосування мови, що базуються на різних стилях програмування, таких, як об'єктно-орієнтований, аспектно-орієнтований, процедурний, логічний та функціональний. Серед перерахованих категорій мов програмування виділимо процедурні та функціональні мови. Процедурні мови виділяються серед інших за критерієм кількості програмного коду, вже написаного цими мовами, а функціональні мови – за критерієм перспективності їх використання у майбутньому.

Процедурний стиль програмування розглядає програму як опис процедури виконання дій (послідовність дій та умови їх виконання) для досягнення результату, де однією із базових структур є змінна, яка зберігає певне значення і дозволяє змінювати його протягом виконання алгоритму. На відміну від процедурних мов, процес виконання програми, що написана функціональною мовою програмування, ґрунтується на обчисленні результатів функцій від вихідних даних та від результатів інших функцій і не передбачає явного збереження стану програми. Відповідно, не передбачається й зміна цього стану. Під поняттям стану програми будемо розуміти множину значень усіх змінних, що доступні у конкретній точці програми у довільний момент її виконання.

Функціональний стиль передбачає, що процес програмування інтерпретується як обчислення значень функцій у математичному сенсі, причому порядок їх обчислення визначається наявністю даних для обчислення, а не порядком розташування функцій та їх викликів у тексті програми.

Зважаючи на той факт, що між процедурними та функціональними мовами програмування існує значний семантичний розрив, можна сказати, що проблема трансляції програм з процедурних мов у функціональні є нетривіальною. Коректність та ефективність трансляції великою мірою забезпечується внутрішньою формою представлення програми (ВФПП) у трансляторі. Тому вибір ВФПП, яка дозволить повною мірою відобразити взаємозв'язки вхідної програми на процедурній мові та ефективно

згенерувати суттєво відмінний код вихідної програми на функціональній мові, є актуальною науковою задачею.

Внутрішні форми представлення програм у контексті рішення задач із різних сфер проектування трансляторів в останні роки вивчали, наприклад, такі дослідники, як М. В. Зубов, А. Н. Пустигін, Э. В. Старцев [1, 2], А. А. Рибаков [3], В. А. Битнер, Н. В. Заборовский [4] та ін.

Проблемою побудови оптимізуючих компіляторів, використовуючи різні ВФПП, зокрема такі, як граф потоку управління (ГПУ), форма статичного однократного присвоєння (ФСОП), розширена форма статичного однократного присвоєння (РФСОП) та ін., займався А. Ю. Дроздов. Результати його роботи викладені на сторінках докторської дисертації науковця [5]. Вчений описує процес розробки алгоритмічних основ функціонування компонент оптимізуючих компіляторів, націлених на досягнення граничних рівнів продуктивності для платформ з явним паралелізмом, як на рівні команд, так і на рівні ядер процесора.

Серед зарубіжних вчених особливої уваги заслуговують Даниель Вейс [6], що запровадив поняття графу залежностей значень (ГЗЗ), Джеймс Стен'єр і Дес Ватсон, які запропонували розширений ГЗЗ – граф залежностей значень і станів (ГЗЗС) [7], а також Хельг Бахманн, який запропонував ефективний алгоритм трансформації ГЗЗС у ГПУ [8].

Після проведення детального аналізу літературних джерел не було виявлено комплексного огляду ВФПП, які є ядром ефективного оптимізуючого транслятора, у контексті рішення задачі трансляції з процедурних мов програмування у функціональні.

Метою дослідження є аналіз ефективності використання таких ВФПП, як ГПУ, ФСОП, ГЗП, ГЗЗ та ГЗЗС, у рамках рішення задачі трансляції з процедурних мов програмування у функціональні мови, та визначення критеріїв вибору доцільної ВФПП для цієї задачі.

Виклад основного матеріалу дослідження. У даній роботі важливу роль відіграють такі терміни як процедурна мова програмування (ПМП), функціональна мова програмування (ФМП) та чиста функціональна мова програмування (ЧФМП). У статтях різних авторів вищезгадані терміни часто інтерпретуються не однаково. У контексті даної статі терміни «процедурна (імперативна) мова програмування» та «функціональна мова програмування» будемо розуміти так, як визначено в [9, 10]. У свою чергу, ФМП можна поділити на два взаємовиключні підкласи: чисті (pure) та гібридні (impure) мови.

У літературних джерелах часто посилаються на неформальні визначення ЧФМП. Ці визначення базуються на поняттях прозорості посилань та незалежності порядку обчислення [11]. У даній статті, використовуючи термін ЧФМП, будемо використовувати формальне визначення, що було запропоноване вченим А. Сабри у [12].

Властивість чистоти ФМП тісно пов'язана з відсутністю побічних ефектів (ПЕ). У рамках вирішення задачі трансляції програми з ПМП у ФМП можна виокремити наступні типи ПЕ:

1. Присвоєння локальній змінній нового значення.
2. Читання глобальної змінної.
3. Модифікація глобальної змінної.
4. Виконання операторів вводу/виводу.
5. Виконання процедур/функцій, у яких параметр(и) передаються за адресою.

ПЕ із наведеного вище списку будемо позначати $ПЕ_i$ (i=1..5).

А. Сабри показав, що відсутність побічних ефектів виступає необхідною умовою чистоти ФМП [12]. Враховуючи дане твердження, можна запропонувати зазначені п'ять типів ПЕ як критерії якості ВФПП, що відображають властивість деякої ВФПП не містити відповідних $ПЕ_i$. Іншими словами, якщо початково код програми містить $ПЕ_i$, а після його перетворення у деяку ВФПП цей $ПЕ_i$ може бути усунений, тоді ця ВФПП відповідає критерію *i*.

Зазначені п'ять критеріїв якості ВФПП можна доповнити ще шостим критерієм, що показує практичну доцільність використання певної ВФПП. В даному контексті практична доцільність (критерій 6) означає можливість генерувати вихідний код одразу з ВФПП без додаткових перетворень.

На практиці, у якості ВФПП для процедурних мов програмування найчастіше виступає ГПУ [13]. ГПУ – це орієнтований граф, вершини якого є одиничними операторами чи базовими блоками. Базовий блок – це послідовність операторів з однією точкою входу та однією точкою виходу, що не містить операторів умовних та безумовних переходів. Дуги графа відображають можливі шляхи виконання програми. Дуга графа *(a, b)* є відображенням того, що потік управління досягне блоку *b* одразу після

виконання останнього оператора у блоці *a*. ГПУ, по суті, є відображенням структури потоку виконання операторів у програмі.

ГПУ будується лише для однієї підпрограми (процедури чи функції). Тобто, цей граф не має засобів для представлення міжпроцедурного потоку управління (МПУ). Ця інформація окремо описується графом викликів підпрограм (ГВП). ГВП – це орієнтований граф, вузли якого відповідають підпрограмам. Між вузлами графа *p* та *q* існує дуга *(p, q)* у випадку, якщо функція *p* викликає функцію *q*. Важливо відзначити, що ГВП не містить інформації про кількість та порядок виклику підпрограм. Цю інформацію можна отримати з ГПУ. Отож, на практиці ГПУ та ГВП найчастіше використовують сумісно.

При побудові ГПУ в обов'язковому порядку враховуються безумовні переходи, умовні переходи, оператори вибору, виклики функцій та процедур, цикли, оператори виходу з циклу (break) та продовження циклу (continue). Приклад побудови ГПУ наведено на рис. 1.

```
int n = READ();
int f = 1;
while(n > 0) {
  n--;
  if((n + 1) % 2 == 0){
        f = f * (n + 1);
  } else {
    f = f / (n + 1);
  }
}
int res = f;
PRINT(res);
```

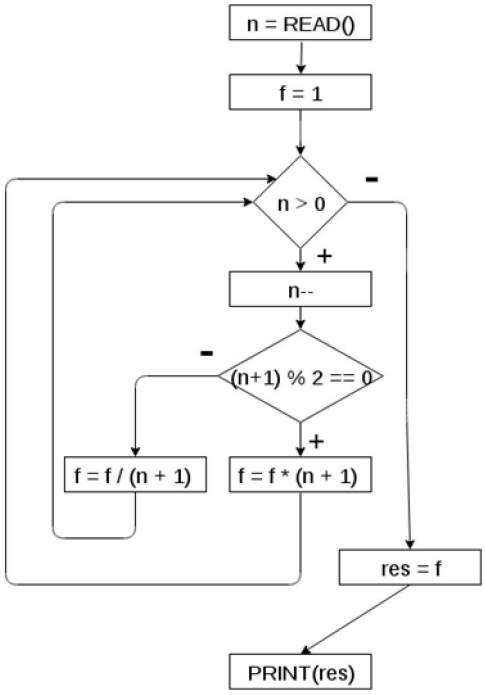


Рис.1. Приклад побудови ГПУ
авторська розробка

Рис. 1 наочно демонструє процедурну природу ГПУ: оператори виконуються покроково, а в результаті виконання операторів модифікуються значення змінних. ГПУ з самого початку створювався як ВФПП для процедурних мов програмування. Тобто цей граф не передбачалося використовувати для вирішення однієї з головних задач трансляції з процедурних мов програмування у функціональні – усунення зміни стану програми. Не важко побачити, що ГПУ не відповідає першим п'яти критеріям якості ВФПП із зазначеного вище переліку. Але, у той же час, ГПУ відповідає критерію 6 і, в принципі, може бути використаний як ВФПП для трансляції програм з ПМП у гібридні ФМП.

Недоліки ГПУ можна частково усунути, перетворивши цей граф у ФСОП. Базовий алгоритм перетворення ГПУ у ФСОП був запропонований у [14, 15], а модифікований алгоритм побудови ФСОП було запропоновано в [16]. У порівнянні з базовим модифікований алгоритм працює швидше.

Важливо зазначити, що у різних статтях значення поняття "форма статичного однократного присвоєння" трактується не однаково. У контексті даної статті будемо використовувати визначення ФСОП, що було запропоноване авторами статей [14, 15, 16].

Спрощене визначення ФСОП можна сформулювати наступним чином: ФСОП представляє собою такий ГПУ, у якому кожній змінній в програмі, що використовується у лівій частині

оператора присвоєння (ОП), відповідає рівно один такий оператор. Права частина ОП може включати довільну кількість змінних.

Для приведення ГПУ у ФСОП потрібно виконати наступні кроки:

1. Для кожної змінної програми *x*, для якої зустрічається більше одного ОП (позначимо їх a_i) та *x* використовується у лівій частині a_i, визначаються нові змінні x_i. Ліва частина усіх a_i замінюється відповідними змінними x_i.

2. Вводяться спеціальні функції φ_k, що використовуються у точках конвергенції x_i. Ці функції потрібні для зв'язування різних визначень змінної *x*, що досяжні у точках конвергенції.

На рис. 2 представлено приклад трансформації ГПУ, що заданий на рис. 1 у ФСОП.

```
int n1 = READ();
int f1 = 1;
int n3, f3;
while (true){
  int n2 = φ1(n1, n3);
  if(n2 <= 0){
    break;
  }
  int f2 = φ2(f1, f3);
  int n2_tmp = n2;
  n2_tmp--; // перезапис!
  n3 = n2_tmp;
  int f4, f5;
  if((n3 + 1) % 2 == 0){
    f4 = f2 * (n3 + 1);
  } else {
    f5 = f2 / (n3 + 1);
  }
  f3 = φ3(f4, f5);
}
int f6 = φ4(f1, f3);
int res = f6;
PRINT(res);
```

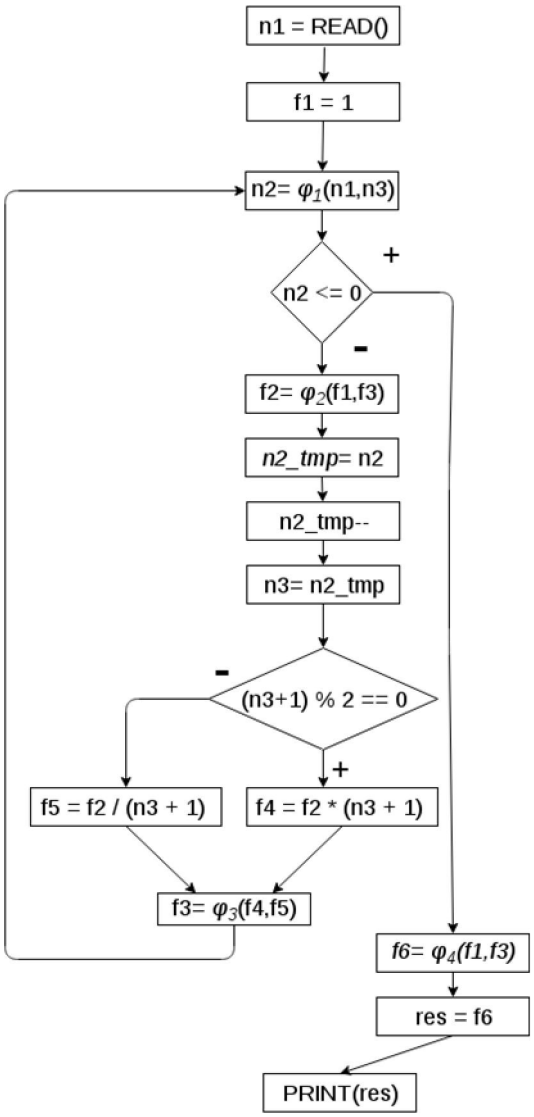


Рис. 2. Приклад трансформації коду програми у ФСОП
авторська розробка

Для того, щоб зберегти однократність присвоєння у програмі, представленій на рис. 2, для змінних *n* та *f*, вводяться нові змінні n_1, n_3 та f_1, f_3, f_4, f_5 відповідно. Для змінної *res* нові змінні не вводяться, оскільки вона використовується у лівій частині лише одного ОП. У точках конвергенції змінних n_1, n_3, f_1, f_3, f_4, f_5 вводяться спеціальні функції φ_1, φ_2, φ_3 та φ_4, що приймають два параметра. Визначення змінних n_2, f_2 та f_6 не є обов'язковим. Ці змінні можна замінити викликом функції φ_k у місцях їх використання.

ФСОП не містить визначення функцій φ_k, тому без додаткових перетворень згенерувати код з ФСОП у вихідну функціональну мову програмування неможливо. Отже критерій 6 для ФСОП не виконується. Базовий та модифікований алгоритми перетворення ГПУ у ФСОП не передбачають, що код вхідної програми містить глобальні змінні, виклики процедур і функцій. Тому ФСОП не задовольняє критеріям 2-5, а відповідає лише критерію 1.

Р.Балланс частково вирішив недоліки ФСОП, запропонувавши розширену форму статичного однократного присвоєння (РФСОП) [17]. У РФСОП вводяться кілька типів функцій φ_k:

1. γ-функція – функція, що відповідає заголовку умовного оператора *if-then-else*. Функція має вигляд $X3 = \gamma (P, X1, X2)$, де P – умова переходу конструкції *if-then-else*, $X1$ – змінна, що відповідає умові P, а $X2$ – змінна, що відповідає інверсній умові $\neg P$.

2. μ-функція визначає значення змінної в тілі циклу. Функція має вигляд $X2 = \mu (X0, X3)$, де $X0$ – змінна, що відповідає початковим установкам циклу, $X3$ – змінна, що використовується в тілі циклу.

3. η-функція визначає значення змінної на виході циклу. Функція має вигляд $X3 = \eta (P, Xfinal)$, де P – умова виконання (завершення) циклу, $Xfinal$ – змінна, що доступна за межами циклу.

На рис. 3 зображено результат перетворення ФСОП (рис. 2) у РФСОП.

```
int n1 = READ();
int f1 = 1;
int n3, f3;
while (true){
 //  μ - функція
 int n2 = is_bound(n3)?n3:n1;
 if(n2 <= 0){
   break;
 }
 //  μ - функція
 int f2 = is_bound(f3)?f3:f1;
 n3 = safe_dec(n2);
 int f4, f5;
 bool P1 = (n3 + 1) % 2 == 0;
 if(P1){
  f4 = f2 * (n3 + 1);
 } else {
  f5 = f2 / (n3 + 1);
 }
 //  γ - функція
 f3 = P1 ? f4 : f5;
}
bool P2 = n1 > 0;
//  η - функція
int f6 = P2 ? f3 : f1;
int res = f6;
PRINT(res);
```

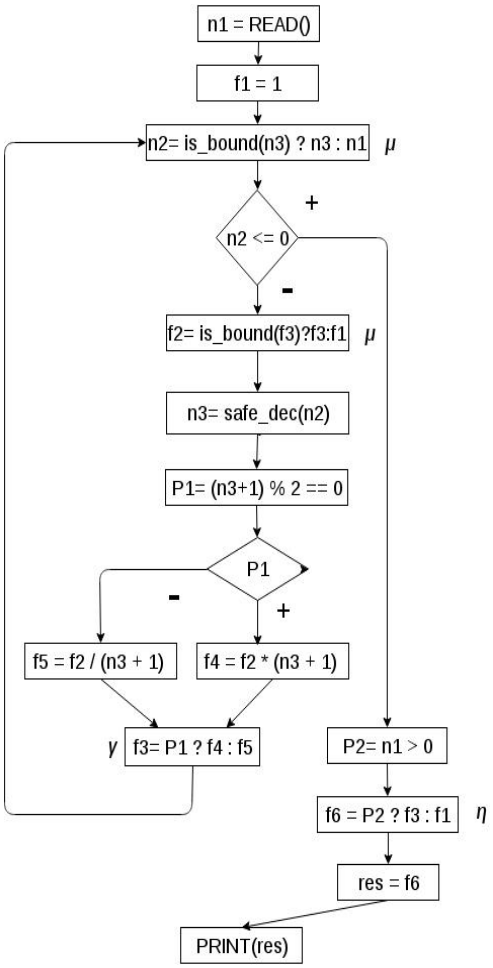


Рис. 3. Перетворення ФСОП у РФСОП
авторська розробка

РФСОП має ті ж властивості, що й ФСОП, але окрім того, у ній присутня уся необхідна інформація для генерації коду вихідної програми. Тому РФСОП відповідає не тільки критерію 1, а також критерію 6. РФСОП так само як і ФСОП не відповідає критеріям 2-5.

Хоча ФСОП та РФСОП надають деякі переваги у вирішенні поставленої проблеми у порівнянні з ГПУ, але вони є його модифікаціями. А ГПУ, як відомо, не відповідає жодному з п'яти критеріїв якості ВФПП. Тому наступним логічно обґрунтованим кроком є розгляд такої ВФПП, що не буде базуватися на ГПУ. Такою формою виступає ГЗЗ [18] та його модифікація ГЗЗС [19]. Властивості ГЗЗС не залежать від значень змінних, від порядку, в якому ці значення з'являються та від місця визначення змінних [6, 20]. ГЗЗС є паралельним представленням, у якому відношення порядку між операціями задається частково і базується лише на залежностях за даними та станами. Тобто, на відміну від ГПУ, ГЗЗС передбачає обчислення значень лише у разі потреби. Звідси слідує, що обчислення ГЗЗС може бути таким, що завершилось, навіть якщо виконання вхідної програми буде нескінченним. Ця особливість графу пояснюється тим, що значення, обчислення яких відбувається під час виконання програми але які не потрібні для побудови ГЗЗС, не виконуються.

ГЗЗС – це дводольний граф, що може бути визначений наступним чином [21]: $G = (N, E_V, E_S, L, N_0, N_\infty)$. Складовими частинами цього графа є:

1. N – множина вершин графа. Граф включає три типи вершин: операторні вершини, вершини-значення, вершини стану.
2. E_V – множина дуг графа ($E_V \subseteq N \times N$), що відповідають залежностям значень.
3. E_S – множина дуг графа ($E_S \subseteq N \times N$), що відповідають залежностям стану.
4. L – множина функцій, що асоціюють кожну вершину графа з відповідним оператором.
5. N_0 – початкова вершина графа.
6. N_∞ – кінцева вершина графа.

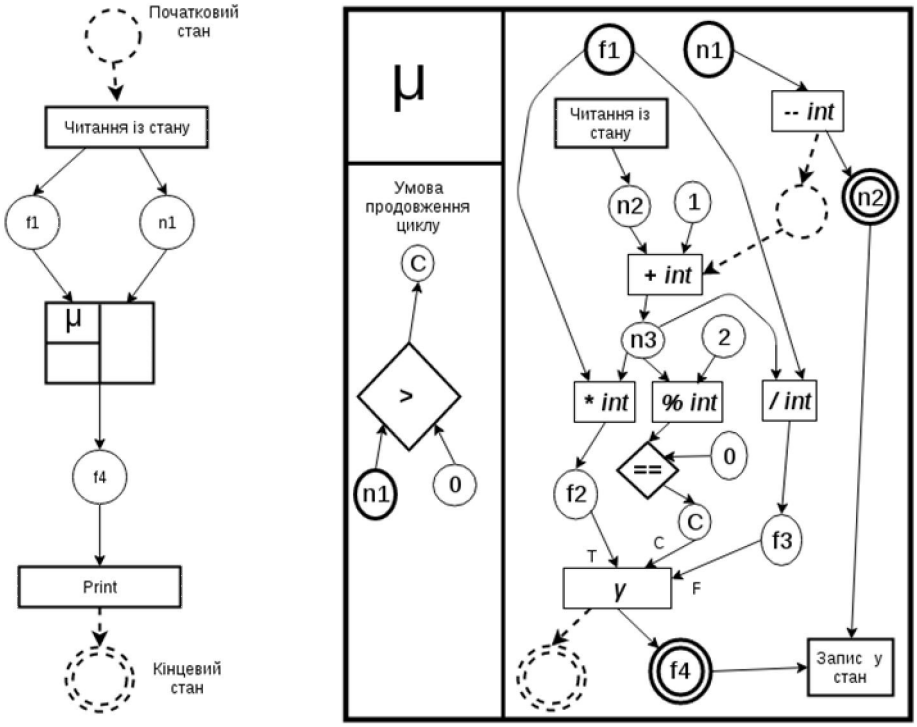


Рис. 4. Приклад побудови ГЗЗС
авторська розробка

Цикли та умовні оператори представляються у ГЗЗС за допомогою спеціальних вершин μ та γ.

На рис. 4 наведено приклад побудови ГЗЗС для програми, заданої на рис. 2.

У ГЗЗС непрямим чином забезпечується властивість однократного присвоєння, оскільки існує однозначна відповідність між операторними вершинами та вершинами-значеннями. Кожній вершині-значенню можна присвоїти ім'я, що буде унікальним і відповідатиме змінній програми у ФСОП.

Усі значення у ГЗЗС зчитуються із стану або продукуються у результаті виконання операторних вершин. Тому глобальні змінні при побудові ГЗЗС оброблюються аналогічно локальним. Таким чином ГЗЗС відповідає не тільки критерію 1, а й критеріям 2, 3, 5.

Кожна вершина, що відповідає оператору вводу/виводу продукує новий стан. Тому ГЗЗС відповідає критерію 4.

ГЗЗС містить усю необхідну інформацію для генерації вихідної програми, а отже відповідає критерію 6.

У рамках статті було проаналізовано відповідність ВФПП критеріям 1 – 6. Підсумок результатів аналізу наведено у таб. 1.

Таблиця 1. Підсумкова таблиця відповідності ВФПП критеріям 1 – 6
авторська розробка

ВФПП / Критерій	**1**	**2**	**3**	**4**	**5**	**6**
ГПУ	-	-	-	-	-	+
ФСОП	+	-	-	-	-	-
РФСОП	+	-	-	-	-	+
ГЗЗС	+	+	+	+	+	+

Висновки. Проведений аналіз ВФПП показав, що ГПУ не придатний для ефективного вирішення задачі трансляції програм із процедурних мов програмування у функціональні, оскільки ГПУ не відповідає жодному із п'яти критеріїв якості ВФПП. У той же час ГПУ відповідає критерію 6, тому ГПУ, теоретично, можна використовувати як ВФПП для трансляції програм у гібридні функціональні мови програмування.

На відміну від ГПУ ФСОП та РФСОП відповідають критерію 1. ВФПП, що відповідають критерію 1, можна використовувати для трансляції такої підмножини програм, які містять побічні ефекти лише типу 1. ФСОП не відповідає критерію 6, тому на практиці цю ВФПП використовувати недоцільно.

Найбільш придатним для вирішення вищезгаданої задачі є ГЗЗС, оскільки ця ВФПП відповідає усім запропонованим критеріям якості ВФПП. Крім цього, ГЗЗС відповідає критерію 6, тому його доцільно використовувати не тільки як теоретичний спосіб внутрішнього представлення програми у трансляторі, а і як основу для побудови реальної системи.

Серед актуальних проблем для проведення подальших досліджень можна виокремити задачу ефективної побудови ГЗЗС для програм, що містять оператори безумовного переходу (*goto*), завершення циклу (*break*) та продовження циклу (*continue*).

1. Зубов М. В. Применение универсальных промежуточных представлений для статического анализа исходного программного кода / М. В. Зубов, А. Н. Пустыгин, Е. В. Старцев. // Доклады Томского государственного университета систем управления и радиоэлектроники. – март 2013. – №1 (27). – С. 64–68.
2. Зубов М. В. Математическое моделирование универсальных многоуровневых промежуточных представлений для статического анализа исходного кода / М. В. Зубов, А. Н. Пустыгин, Е. В. Старцев. // Доклады Томского государственного университета систем управления и радиоэлектроники. – сентябрь 2014. – №3 (33). – С. 94–99.
3. Рыбаков А. А. Алгоритм создания случайных графов потока управления для анализа глобальных оптимизаций в компиляторе / Алексей Анатольевич Рыбаков. // Международная конференция по параллельным и распределенным компьютерным системам. – Харьков, 13-14 марта 2013. – С. 269–275.
4. Битнер В. А. Построение универсального линеаризованного графа потока управления для использования в статическом анализе кода алгоритмов / В. А. Битнер, Н. В. Заборовский. // Модел. и анализ информ. систем. – 2013. – Т. 20, №2. – С. 166–177.
5. Дроздов А. Ю. Компонентный подход к построению оптимизирующих компиляторов: дис. докт. техн. наук : 05.13.11 / Дроздов Александр Юльевич – Москва, 2010. – 307 с.
6. Value dependence graphs: representation without taxation / D.Weise, R. F. Crew, M. Ernst, B. Steensgaard. // POPL '94 Proceedings of the 21st ACM SIGPLAN-SIGACT symposium on Principles of programming languages. – 1994. – Pp. 297–310.

7. Stanier J. The value state dependence graph revisited / J. Stanier, L. Alan. // In Proceedings of the Workshop on Intermediate Representations. – 2011. – Pp. 53–60.
8. Bahmann H. Perfect Reconstructability of Control Flow from Demand Dependence Graphs / H.Bahmann, N. Reissmann, J. Magnus, J. C. Meyer. // ACM Transactions on Architecture and Code Optimization. – January 2015. – Volume 11 Issue 4. – Article No. 66.
9. Reinhard W. Compiler Design: Virtual Machines / W. Reinhard, S. Helmut., 2011. – Springer. – Pp. 7–55.
10. Reinhard W. Compiler Design: Virtual Machines / W. Reinhard, S. Helmut., 2011. – Springer. – Pp. 57–104.
11. Launchbury J. State in Haskell / J. Launchbury, J. Peyton, L. Simon. // Lisp and Symbolic Computation. – 1995. – №8. – Pp. 293–341.
12. Sabry A. What is a purely functional language? / Amr Sabry. // Journal of Functional Programming. – Volume 8 Issue 1. – January 1998. – Pp. 1 – 22.
13. Compilers: Principles, Techniques, and Tools / Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman. – 2nd Ed. – Boston: Addison-Wesley Longman Publishing Co., 2006.
14. Rosen B. K. Detecting equality of variables in programs / B. K. Rosen, M. N. Wegman, F. K. Zadeck. // Proceeding POPL '88 Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages. – 1988. – Pp. 1–11.
15. Rosen B. K. Global value numbers and redundant computations / B. K. Rosen, M. N. Wegman, F. K. Zadeck. // Proceeding POPL '88 Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages. – 1988. – Pp. 12–27.
16. An efficient method of computing static single assignment form / [R. Cytron, J. Ferrante, B. K. Rosen та ін.]. // Proceeding POPL '89 Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages. – 1989. – Pp. 25–35.
17. Ballance R. A. The program dependence Web: A representation supporting control-, data-, and demand-driven interpretation of imperative languages. / R. A. Ballance, A. B. Maccabe, K. J. Ottenstein. // In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'90). – 1990. – Pp. 257–271.
18. Byers D. Syntax-Directed Construction of Value Dependence Graphs / D. Byers, M. Kamkar, T. Palsson. // ICSM '01 Proceedings of the IEEE International Conference on Software Maintenance. – 2001. – Pp. 692 – 703.
19. Lawrence A. C. Optimizing compilation with the Value State Dependence Graph / Lawrence Alan C. – 2007. – 183 p.
20. Колчин А. В. Метод редукции анализируемого пространства поведения при верификации формальных моделей распределенных программных систем // Искусственный интеллект. –2013. – №4. –С. 113–126.
21. Johnson N. E. Code size optimization for embedded processors / Neil E. Johnson. – Technical Report UCAM-CL-TR-607. – November 2004. – 257 p.

УДК 519.876.5
Поморова О.В, д.т.н., проф., Тітова В.Ю., к.т.н, доцент, Медзатий Д.М., к.т.н, доцент
Хмельницький національний університет

ДОСВІД ВИКОРИСТАННЯ ОБЧИСЛЮВАЛЬНОГО ПРИСТРОЮ DE1-SOC У НАВЧАЛЬНОМУ ПРОЦЕСІ ТА НАУКОВИХ ДОСЛІДЖЕННЯХ КАФЕДРИ СИСТЕМНОГО ПРОГРАМУВАННЯ ХНУ

Поморова О.В, Тітова В.Ю., Медзатий Д.М. Досвід використання обчислювального пристрою DE1-SOC у навчальному процесі та наукових дослідженнях кафедри системного програмування ХНУ. У статті пропонується використання плати DE1-SOC, яка побудована за сучасною технологією ПЛІС, в якості учбового стенду для навчального процесу кафедри системного програмування Хмельницького національного університету. Плата являє собою автономний зовнішній пристрій, який підключається до ПК за допомогою розташованих на ній портів та інтерфейсів. Її використання в навчальному процесі дозволяє студентам краще зрозуміти принципи роботи сучасних обчислювальних пристроїв, проектувати власні пристрої та тестувати їх роботу за допомогою ресурсів плати.
Ключові слова: комп'ютерна інженерія, програмовані логічні інтегральні схеми, DE1-SOC, навчальний процес.

Поморова О.В, Титова В.Ю., Медзатый Д.М. Опыт использования вычислительного устройства DE1-SOC в учебном процессе и научных исследованиях кафедры системного программирования ХНУ.
В статье предлагается использование платы DE1-SOC, которая построена по современной технологии ПЛИС, в качестве учебного стенда для учебного процесса кафедры системного программирования Хмельницкого национального университета. Плата представляет собой автономное внешнее устройство, которое подключается к ПК с помощью расположенных на ней портов и интерфейсов. Ее использование в учебном процессе позволяет студентам лучше понять принципы работы современных вычислительных устройств, проектировать собственные устройства и тестировать их работу с помощью ресурсов платы.
Ключевые слова: компьютерная инженерия, программируемые логические интегральные схемы, DE1-SOC, учебный процесс.

Pomorova O.V., Titova V.Yu., Medzatyi D.M. Experience of using the computing device DE1-SOC for the educational process and scientific research at the System Programming department of KhNU. Using the board DE1-SOC, which is built on a modern CPLD technology, as a training bench for the educational process at the System Programming department of Khmelnitsky National University is proposed in this paper. The board is a standalone external device that connects to a PC by ports and interfaces located on it. Using the board provides to students better understanding the work principles of modern computing devices, designing own devices and testing their work by using the board resources.
Keywords: Computer Engineering, Complex Programmable Logic Device, DE1-SOC, educational process.

Вступ. На сьогоднішній день, комп'ютерна інженерія це - технічні (апаратні) засоби та системне програмне забезпечення комп'ютерних систем і мереж універсального і спеціального призначення, а також їх компоненти. Зміст діяльності фахівців з комп'ютерної інженерії полягає в розробці апаратно-програмних засобів сучасних і перспективних інформаційних технологій, розробці та застосуванні комп'ютерних систем і мереж загального та спеціального призначення, їх системного програмного забезпечення, спеціалізованих комп'ютерних систем і мереж з оптимізованими параметрами, інтегрованих комп'ютерних систем, технічних засобів захисту інформації в комп'ютерних системах і мережах. Оволодіння зазначеними навичками та компетенціями неможливе без використання у процесі навчання сучасних комп'ютерних засобів.

Одним з перспективних напрямків сучасних технологій розробки апаратно-програмних засобів є технологія програмованих логічних інтегральних схем (ПЛІС). Її сучасний розвиток дозволив не тільки замінити ПЛІС мікросхеми малого та середнього ступеня інтеграції у спеціалізованих цифрових та комп'ютерних пристроях, але й дозволив використовувати ПЛІС у навчальному процесі в якості учбових стендів.

Серед переваг такого використання можна виділити наступні [1-2]:

- підтримка реалізації складних цифрових електронних схем та пристроїв, які за своєю потужністю не поступаються сучасним процесорам;
- простота реалізації зазначених схем та пристроїв за рахунок використання систем автоматизованого проектування (САПР), які пропонують розробники ПЛІС.

Учбові стенди на ПЛІС можуть бути реалізовані в трьох варіантах:

- як автономний зовнішній пристрій, програмування якого здійснюється за допомогою персонального комп'ютера (ПК) через спеціальний інтерфейс JTAG;
- як плата розширення ПК з інтерфейсом шини PCI, PCI Express, AGP або інших;

- як автономний зовнішній пристрій, який підключається до ПК за допомогою послідовних або паралельних інтерфейсів (USB, LPT тощо).

Перевагою першого варіанту є те, що такий стенд може працювати автономно без ПК. Перевагою другого – можливість обміну даними з ПК в режимі реального часу. Третій варіант поєднує в собі переваги двох попередніх, а тому надає більш широкі можливості для навчального процесу.

Одним з прикладів ПЛІС, яка відноситься саме до третього класу, є обчислювальний пристрій DE1-SoC від Terasic Technologies. Він використовується в якості учбового стенду на кафедрі системного програмування Хмельницького національного університету.

Зазначений обчислювальний пристрій містить двоядерний процесор ARM Cortex-A9, інтегровану оперативну пам'ять та множину інтерфейсів шин та портів, які використовуються для підключення сучасної периферії. Його структура наведена на рис. 1.

Даний стенд використовується в рамках вивчення студентами предмету «Архітектура комп'ютерів». Предмет розрахований на два семестри, та складається з 15 лабораторних робіт.

Метою лабораторних робіт першого семестру є вивчення можливостей мови програмування VHDL, яка є базовою мовою при розробці апаратури сучасних обчислювальних цифрових систем [3].

Для цього студентам даються завдання з проектування комбінаційних схем, простих обчислювальних пристроїв, зокрема таймерів, лічильників адрес та інших. Наявні на платі засоби візуалізації: світлодіоди та цифрові дисплеї дозволяють наочно перевірити правильність функціонування спроектованих пристроїв та зрозуміти принципи їх роботи.

Метою лабораторних робіт другого семестру є набуття студентами навичок у розробці:

- апаратних засобів обміну даними між процесором і зовнішніми пристроями, обробки переривань програми та прямого доступу до пам'яті;
- мікропроцесорів та мікроконтролерів на базі мікропроцесорних комплектів ВІС.

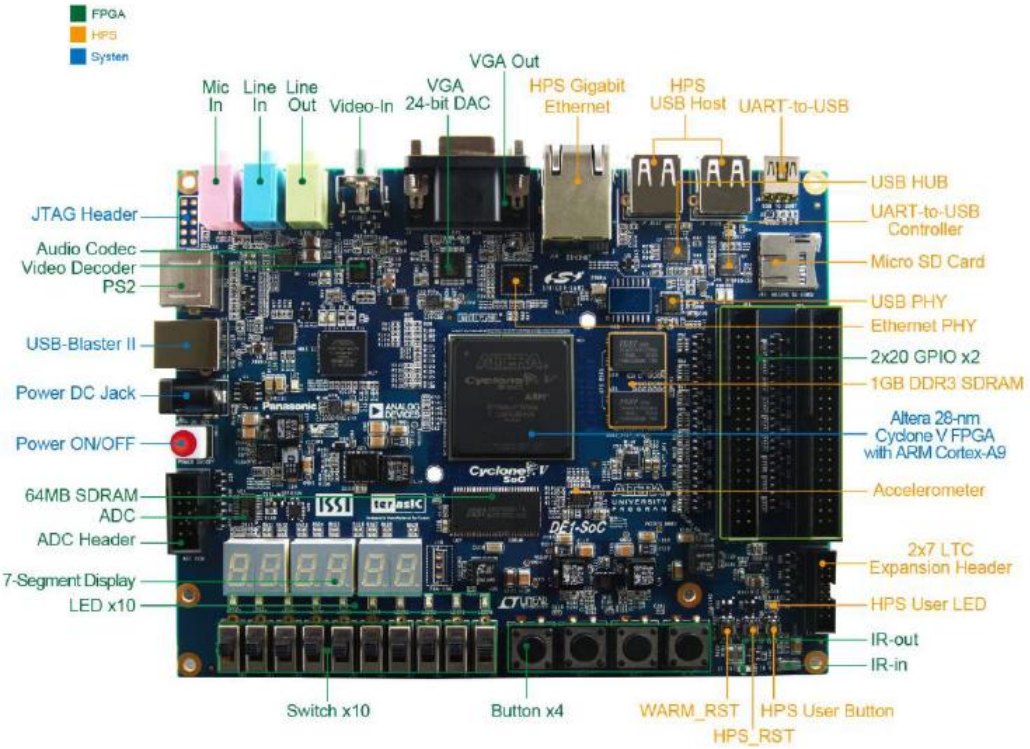


Рис. 1. Структура обчислювального пристрою DE1-SoC, який використовується в якості учбового стенду

Для цього студентам надаються різноманітні завдання, серед яких можна виділити:

- реалізацію арифметично-логічного пристрою (АЛП), який виконує арифметичні та логічні операції, задані в індивідуальному завданні, та встановлює задані у завданні прапорці. Ресурси плати дозволяють протестувати роботу реалізованого АЛП та краще зрозуміти принцип його функціонування;

- реалізацію основної та стекової пам'яті заданого обсягу, доступ до яких здійснюється за алгоритмами, заданими в індивідуальному завданні. В ході виконання лабораторної роботи студенти мають змогу використовувати інтегровану на плату пам'ять та тестувати різні способи доступу до неї;
- реалізацію послідовного та паралельного портів, які функціонують за заданими в індивідуальному завданні алгоритмами. Ресурси плати дозволяють використати реалізовані порти для підключення зовнішніх пристроїв та візуалізації передачі інформації з них;
- реалізацію керуючого автомату, який виконує дії або обчислення згідно індивідуального завдання, та мікропроцесора за архітектурою, заданою в індивідуальному завданні.

Набуття навичок в реалізації подібних автоматів та мікропроцесорів дозволить студентам проектувати та програмувати мікроконтролери та мікропроцесори в їх майбутній професійній діяльності.

Реалізація усіх зазначених пристроїв здійснюється за допомогою САПР Quartus II 15.0. Дана програма відрізняється зручним інтерфейсом, який значно спрощує процес проектування та розробки будь-яких пристроїв та дозволяє не тільки виконувати завдання в межах лабораторних робіт, але й створювати повноцінні апаратні реалізації пристроїв, які можуть бути використані в сучасному виробництві.

Приклад виконання лабораторних робіт за допомогою САПР Quartus II 15.0 наведено на рис. 2-5. За допомогою програмного коду студенти можуть створювати власні пристрої: прості, такі як мультиплексор (рис. 2) або декодер, та складні, такі як арифметично-логічний пристрій (рис. 3) або керуючий автомат (рис. 4). За допомогою графічних файлів існує можливість поєднувати пристрої між собою, створюючи цифрові схеми на їх основі (рис. 5).

```
entity mux is
        port (a, b, c, d, e1,e2: in std_logic;
              pout: out std_logic);
end mux;

architecture multiplexor of mux is
         begin
        process (e1, e2, a, b, c, d)
                begin
                if e1='0' and e2='0' then pout <= a;
                        elsif e1='0' and e2='1' then pout <= b;
                        elsif e1='1' and e2='0' then pout <= c;
                        else pout <= d;
                end if;
        end process;
end multiplexor;
```

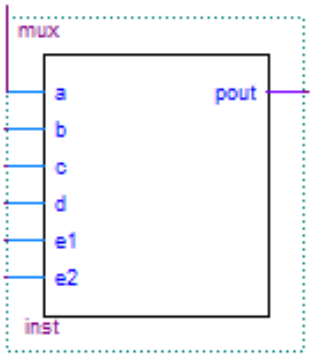


Рис. 2. Приклад створення пристрою, який працює за принципом мультиплексора

Окрім навчального процесу, плата використовується як стенд для наукових досліджень та студентських проектів.

На сьогоднішній день студентські колективи працюють над наступними проектами:
- реалізація на базі ПЛІС логічних відеоігор та грального автоматів;
- реалізація на базі ПЛІС системи оповіщення;
- реалізація на базі ПЛІС системи керування електричною підстанцією.

```
process ( reset, op_code, in1, in2)
   variable tmp_out: std_logic_vector (15 downto 0);
begin
 if reset = '1' then
  tmp_out:= "0000000000000000";
  case op_code is
     when op_add =>
     out1<=in1 + in2;
     flag_ov<= 'Z';
     when op_mul =>
     tmp_out:= in1 * in2;
          out1<=tmp_out(7 downto 0);
     if (tmp_out (15 downto 8) /= "00000000") then
       flag_ov<= '1';
    else
      flag_ov<= '0'; end if;
      when op  rol =>
      tmp_out (7 downto 0):= in1;
       out1(7 downto 1)<= tmp_out (6 downto 0);
       out1(0)<= tmp_out (7);
       flag_ov<= 'Z';
       when others =>
       out1<= "ZZZZZZZZ"; flag_ov<= 'Z';
    end case;
  else out1<="00000000"; flag_ov<= 'Z';  end if;
end process;
end behaviour;
```

Рис.3. Приклад створення арифметичного-логічного пристрою, який виконує операції додавання, множення та циклічного зсуву.

На основі обчислювального пристрою виконується магістерська робота, присвячена апаратній реалізації природничих алгоритмів, таких як: алгоритм рою бджіл, мурашиний алгоритм, генетичний алгоритм, алгоритм зграї вовків та інші, та дисертаційна робота, присвячена виявленню поліморфних вірусів в операційних системах та комп'ютерних мережах. У цій роботі обчислювальний пристрій виконує роль плати-прискорювача роботи персонального комп'ютера, яка містить спеціальне програмне забезпечення та здійснює моніторинг програмного коду на наявність у ньому тіл вірусів.

Висновки. Сучасні навчальні стенди на основі ПЛІС дозволяють значно розширити можливості навчального процесу при підготовці студентів технічних спеціальностей, зокрема спеціальності комп'ютерна інженерія.

Студенти використовують такі стенди при виконанні лабораторних та практичних робіт, оволодіваючи навичками розробки сучасних програмно-апаратних засобів.

На основі стендів проводяться наукові та дослідницькі роботи, що дозволяє перевірити теоретичні розробки на практиці та отримати більш якісні наукові результати.

Також сучасні ПЛІС використовуються у якості плат-прискорювачів, виконуючи емуляцію таких програмних засобів, як антивіруси, штучні імунні системи, системи моніторингу ресурсів та вивільняючи ресурси комп'ютера для опрацювання поточних задач користувача.

Напрямом подальших робіт є розроблення нових лабораторних робіт, спрямованих на розробку студентами різноманітних обчислювальних та керуючих пристроїв, та на розширення тематик наукових і магістерських робіт, які можуть бути виконані на основі обчислювального пристрою DE1-SoC.

```
process (start, stan_next) -- регістр поточного стану
 -- якщо сигнал запуску дорівнює 0, автомат знаходиться в 000 стані
 -- якщо сигнал запуску дорівнює один автомат переходить зі стану в стан
begin
if start='0' then
STAN<="000"; else stan<=stan_next; end if;
end process;
 process (stan) -- логіка переходів
 -- визначає яке значення отримає поточний стан наступної миті
 begin
 if stan="000" then stan_next<="001"; end if;
 if stan="001" then stan_next<="010"; end if;
 if stan="010" then stan_next<="011"; end if;
 if stan="011" then stan_next<="100"; end if;
 if stan="100" then stan_next<="101"; end if;
 if stan="101" then stan_next<="110"; end if;
 if stan="110" then stan_next<="111"; end if;
 end process;
 process (stan) -- логіка виходу
 -- кожному поточному стану автомату у відповідність ставить певна дія або множина дій
 variable m:std_logic_vector (0 to 4);
 begin
 if stan="000" then y1<='0'; y2<='0';y3<='0';y4<='0';y5<='0'; end if;
 if stan="001" then m(0):=a1; m(1):=a2; m(2):=a3; m(3):=a4; m(4):=a5; end if;
 if stan="010" then m(0):=not m(0); end if;
 if stan="011" then m(1):=not m(1); end if;
 if stan="100" then m(2):=not m(2); end if;
 if stan="101" then m(3):=not m(3); end if;
 if stan="110" then m(4):=not m(4); end if;
 if stan="111" then y1<=m(0); y2<=m(1);y3<=m(2);y4<=m(3);y5<=m(4); end if;
 end process;
```

Рис.4. Приклад створення керуючого автомату, який перетворює двійкове число в обернений код

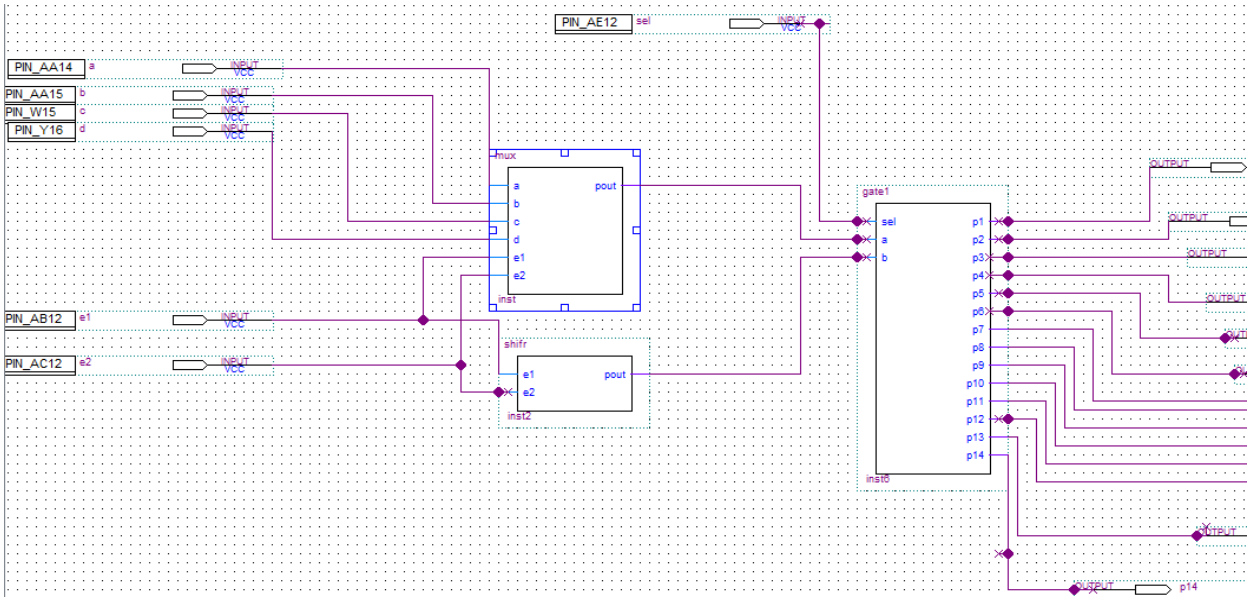


Рис. 5. Поєднання створених пристроїв на схемі та їх прив'язка до ніжок плати

1. Стешенко В.Б. ПЛИС фирмы ALTERA: проектирование устройств обработки сигналов Москва: Додэка, 2000. — 128 с.
2. Грушвицкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем на микросхемах программируемой логики СПб.: БХВ-Петербург, 2002. — 608 с.
3. Поляков А. К. Языки VHDL и Verilog в проектировании цифровой аппаратуры. — М.: СОЛОН-Пресс, 2003. — 320 с.
4. DeSoC User Manual / - http://courses.cs.washington.edu/courses/cse467/15wi/docs/DE1_SoC_User_Manual.pdf

УДК 681.513.2
[1]Рязанцев О.І., [1]Кардашук В.С.,[2] Бортник К.Я.
[1]Східноукраїнський національний університет імені Володимира Даля
[2]Луцький національний технічний університет

ЗАСТОСУВАННЯ ПРОГРАМНОЇ БІБЛІОТЕКИ АЛГОРИТМІЧНИХ ЕЛЕМЕНТІВ ДЛЯ ПРОЕКТУВАННЯ ТЕХНОЛОГІЧНИХ СХЕМ ПРОМИСЛОВОЇ АВТОМАТИЗАЦІЇ

Рязанцев О.І., Кардашук В.С., Бортник К.Я. Застосування програмної бібліотеки алгоритмічних елементів для проектування технологічних схем промислової автоматизації. Розглянуто застосування програмної бібліотеки алгоритмічних елементів для проектування технологічної схеми та програмного забезпечення автоматизації виробництва вінілхлориду. Проведена декомпозиція технологічного процесу на підсистеми та визначений перелік програмних модулів, необхідних для реалізації функцій керування. Запропоновані програмно-апаратні засоби реалізації системи керування. Наведені переваги використання модернізованої системи.
Ключові слова: програмна бібліотека алгоритмічних елементів, технологічна схема, система керування, програмно-логічний контролер, цифровий регулятор.

Рязанцев А.И., Кардашук В.С., Бортник К.Я. Применение программной библиотеки алгоритмических элементов для проектирования технологических схем промышленной автоматизации. Рассмотрено применение программной библиотеки алгоритмических элементов для проектирования технологической схемы автоматизации производства винилхлорида. Проведена декомпозиция технологического процесса на подсистемы и определен перечень программных модулей, необходимых для реализации системы управления. Предложены программно-аппаратные средства реализации системы управления. Приведены преимущества использования модернизированной системы.
Ключевые слова: программная библиотека алгоритмических элементов, технологическая схема, система управления, программно-логический контролер, цифровой регулятор.

Ryazantsev A.I., Kardashuk V.S., Bortnyk K.Y. The automation system of process control the production of gelatine. The application of the program library of algorithmic elements for design automation of technological production scheme of vinyl chloride. Spend the decomposition process in the sub-system and a list of software modules necessary for the implementation of the management system. Proposed software and hardware implementation of the management system. The advantages of using the upgraded system.
Keywords: software library of algorithmic elements, flow chart management system, software and logic controller, digital regulator.

Актуальність проблеми. Сучасні тенденції розвитку підприємств хлорорганічного синтезу характеризуються широким використанням потенційно небезпечних технологій та виробництв. Однією з пріоритетних задач промислової екології є кваліфікована переробка хлорорганічних відходів виробництва і споживання, а застосування в автоматизації технологічних процесів досягнень в галузі програмно-технічних засобів (ПТЗ) є одним із головних напрямків удосконалення діючих виробництв. Робота з модернізації системи автоматичного керування (САК) технологічного процесу (ТП) виробництва вінілхлориду проводилася в рамках науково-технічної програми співпраці між кафедрою комп'ютерної інженерії Східноукраїнського національного університету імені Володимира Даля та науково-виробничим підприємством «Уніконт» (м. Сєвєродонецьк), що займається розробкою і впровадженням програмно-логічних контролерів (ПЛК) і робочих станцій (РС) для інформаційних систем та САК ТП різноманітного призначення [1].

Аналіз досліджень об'єкта керування. Аналіз технологічного процесу виробництва вінілхлориду показав необхідність модернізації існуючого виробництва, застосування ПЛК з необхідним набором засобів контролю з метою керування параметрами температури, вібрації та концентрації хімічних продуктів CO, CO_2, HCl, Cl_2. За результатами дослідження виконана декомпозиція об'єкту керування, для розроблення програмного забезпечення системи визначений перелік вхідних та вихідних сигналів, визначені функції системи. Для прискорення процесу створення САК слід скористатись інструментальними засобами проектування технологічних схем ТП, що включає застосування програмної бібліотеки алгоритмічних елементів.

Рішення задачі. Дворівнева САК ТП виробництва вінілхлориду є складовою частиною системи утилізації відходів хлорорганічного синтезу [2] та призначена для підвищення екологічної безпеки хімічного підприємства, забезпечення технічного персоналу оперативною інформацією про стан процесу з метою подальшого прийняття рішень щодо зміни керуючих впливів при відхиленні параметрів від технологічного регламенту (рис. 1).

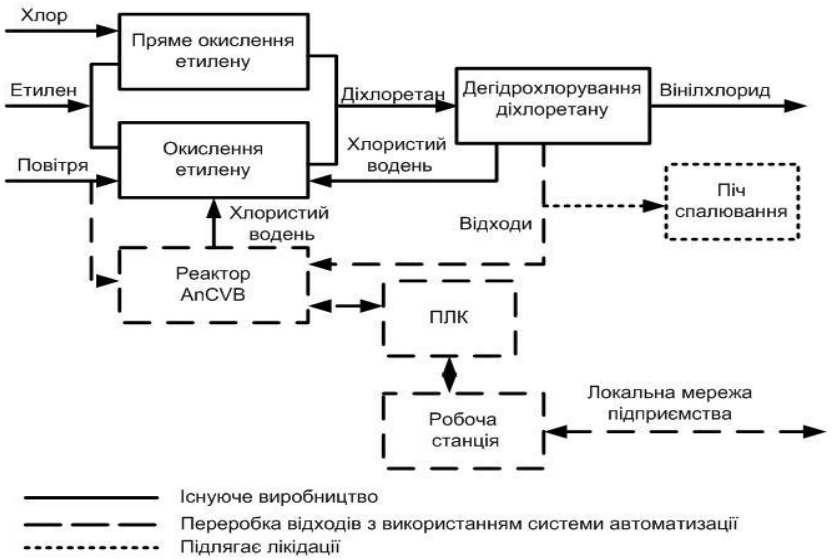


Рис. 1. Структурна схема модернізованої системи виробництва вінілхлориду

Переваги впровадження модернізованої системи виробництва вінілхлориду з використанням ПЛК полягає в тому, що повністю ліквідується піч спалювання, яка є джерелом шкідливих викидів незв'язаного хлору для здоров'я людини та навколишнього середовища продуктами згорання відходів.

Модернізована САК є програмно-технічним комплексом, який складається з технічних засобів та програмного забезпечення і забезпечує автоматизацію керування та контроль стану вихідних параметрів (концентрацією CO, CO_2, HCl, Cl_2). Керування процесом на нижньому рівні здійснюється за допомогою ПЛК з необхідним набором апаратних та програмних модулів, який забезпечує збір, первинну обробку інформації та видачу сигналів керування [3].

У відповідності з загальноприйнятими концепціями побудови САК при розробленні були враховані модульність, ієрархічність, інформаційна сумісність форматів даних, що дозволяє використовувати ПЛК та робочі станції різних виробників програмно-технічних засобів.

Одним із методів прискорення проектування схем промислової автоматизації є застосування інструментальних засобів на базі програмної бібліотеки алгоритмічних елементів.

Запропонована система проектування технологічних схем керування використовує графічний редактор PC-CAPS з пакету PCAD фірми Altium та програмні модулі, що описують роботу цих елементів.

Розроблена на базі графічного редактора програмна бібліотека основних алгоритмічних елементів містить 130 елементів та складається з наступних складових частин:

1. Комбінаційні логічні елементи. До цієї групи входять елементи, що виконують основні логічні операції (додавання, множення, інверсії і т.п.), а також шифратори, дешифратори, мультиплексори, демультиплексори, елементи порівняння та ін.

2. Елементи послідовного типу – тригери різноманітних типів, лічильники, регістри, елементи затримки.

3. Динамічні елементи – елементи пропорційності, інтегрування,диференціювання, фільтри з можливістю налаштування.

4. Статичні елементи – елементи виконання математичних операцій додавання, віднімання, ділення та ін.

5. Елементи нелінійності – порогові елементи та елементи релейної автоматики.

6. Перемикачі на 2, 4, 8 входів та різноманітні комутатори.

7. Елементи регулювання – елементи для реалізації основних законів керування ПІ, ПД, ПІД та ін.

Крім основної бібліотеки, додаткова бібліотека містить елементи роботи з модулями зв'язку з об'єктом, такі як, наприклад, модулі формування дискретних сигналів, аналогово-цифрові перетворювачі та ін. Всі розроблені елементи модулів зв'язку з об'єктом мають діагностичні виходи, що дозволяє системі контролювати стан їх працездатності.

Для побудови технологічних схем керування підсистеми температури, вібрації, кранами подачі компонентів визначені необхідні елементи програмної бібліотеки алгоритмічних елементів.

Покажемо застосування програмної бібліотеки алгоритмічних елементів на прикладі підсистеми керування температурою ТП [4]. В замкнутій одноконтурній підсистемі керування температурою керуючий вплив ліквідує відхилення вихідної величини від заданого значення (рис. 2).

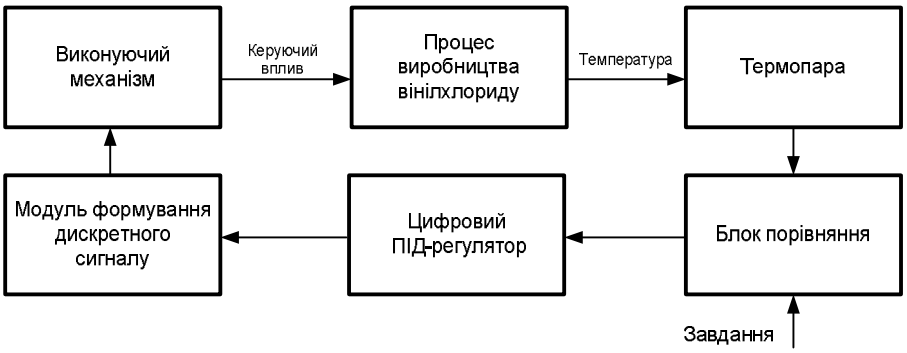


Рис. 2. Структурна схема керування підсистемою температури

Така система представляє собою замкнутий контур, який утворюється об'єктом керування та керуючим органом (система зі зворотнім зв'язком), забезпечує високу точність керування та є поширеним типом САК.

ПІД-регулятор при відхиленні величини температури від заданого параметру Тзд, видає сигнал відхилення ΔТ та впливає на регулюючий орган пропорційно відхиленню величини, що регулюється, інтегралу цього відхилення і швидкості зміни відхилення [5]:

$$y(t) = K_{\Pi}\left[x(t) + \frac{1}{T_I}\int x(t)dt + T_{\partial}\frac{dx(t)}{dt}\right]. \quad (1)$$

Передавальна функція ПІД-регулятора:

$$W_{ПІД}(p) = K_{\Pi}(1 + \frac{1}{T_I p} + T_{\partial} p) \quad . \quad (2)$$

Цифрові ПІД-регулятори, що реалізують дискретну форму рівняння (2), за можливостями налаштування є більш універсальними порівняно з іншими регуляторами.

В динамічному відношенні розроблений цифровий ПІД-регулятор (рис. 3) представляє собою паралельно з'єднані пропорційну, інтегральну та диференційну ланки, структурну схему якого наведено на рис. 4.

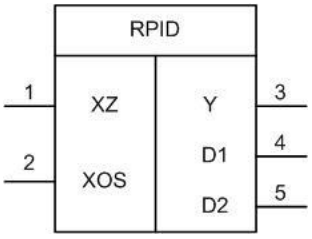


Рис. 3 – Умовне графічне зображення ПІД-регулятора

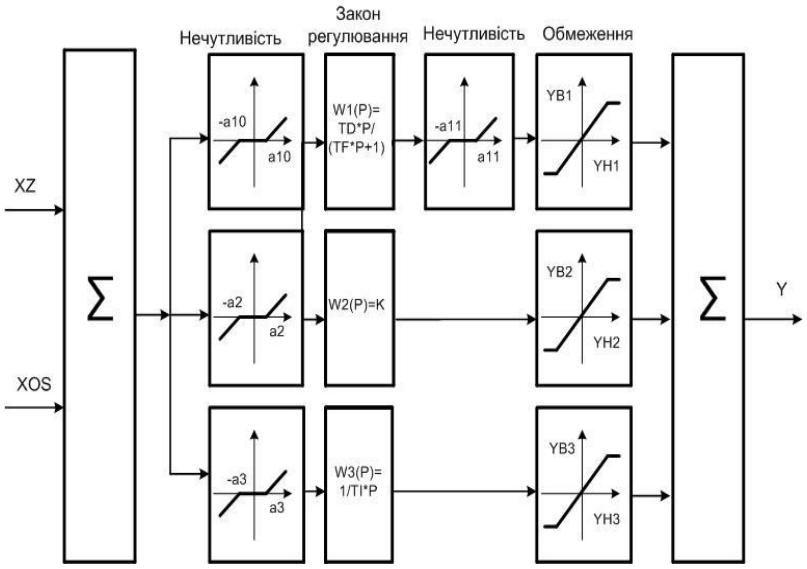


Рис. 4. Структура цифрового ПІД-регулятора

При розробленні програмного забезпечення цифрового ПІД-регулятора (модуль RPID) враховано, що величина керування обмежена певним діапазоном. Призначення елементів та параметри настроювання модуля RPID наведені в табл. 1та 2.

Таблиця 1 – Призначення елементів модуля RPID

Позначення	Вид	Призначення
XZ	Вхід	Аналоговий вхід завдання.
XOS	Вхід	Аналоговий вхід зворотного зв'язку.
Y	Вихід	Аналоговий вихід регулятора.
D1	Вихід	Дискретний вихід ознаки нижньої межі.
D2	Вихід	Дискретний вихід ознаки верхньої межі.

Таблиця 2– Параметри настроювання програмного модуля RPID

Позначення	Найменування
T	Крок (період) дискретизації (200 мс).
YH0,YB0	Нижнє та верхнє граничні значення вихідної величини.
a10,a11	Параметр "зони нечутливості" по каналу диференціювання.
Td	Постійна часу диференціювання.
Tf	Постійна часу фільтру.
NNUd, NNUf	Початкові значення диференційної складової та складової фільтру - NNUd=0, NNUf=0.
YH1,YB1	Граничні значення диференційної складової YH1=-32000, YB1=32000.
a2	Параметр "зони нечутливості" по каналу пропорціональності.
Kc2,Kz2	Коефіцієнт пропорційності закону регулювання = Kc2/Kz2.
YH2,YB2	Граничні значення пропорційної складової YH2=-32000, YB2=32000.
a3	Параметр "зони нечутливості" по каналу інтегрування.
Ti	Постійна часу інтегрування.

Nui	Початкове значення інтегральної складової NUi=0.
YH3,YB3	Граничні значення інтегральної складової YH3=-32000 YB3=32000.

При стрибкоподібній зміні відхилення величини температури цифровий ПІД-регулятор в початковий момент часу здійснює вплив на регулюючий орган (спрацьовує диференційна частина регулятора). Потім величина впливу падає до значення, яке визначається пропорційною частиною, після того, як в регуляторі поступово починає впливати астатична частина регулятора.

Параметрами настроювання цифрового ПІД-регулятора є коефіцієнт пропорційності Кп, постійна часу інтегрування ТІ і постійна часу диференціювання Тд (табл. 4).

Таблиця 3 – Параметри налаштування цифрового ПІД-регулятора температури

Найменування параметру	Значення
Коефіцієнт пропорційності, Кп	2,8
Постійна часу інтегрування, ТІ	2000
Постійна часу диференціювання, Тд	28
Зона нечутливості, °С	± 3
Завдання температури, °С	600

Для перевірки роботи підсистеми температури під керуванням цифрового ПІД-регулятора проведено розроблення програмного забезпечення імітатора температури. Для розроблення імітатора температури асимптотична температура розраховувалась по формулі:

$$T_0 = \frac{1500 \cdot P1 + 600}{P1 + 30}, \quad (3)$$

де – Р1 - % впливу виконавчого пристрою.

Обраний такт роботи системи складає 200 мс. Час опитування датчиків становить 1 с, тобто 5 тактів роботи системи.

Зміна температури фіксувалась через проміжки часу 2 с (10 тактів роботи системи) та розраховувалась по формулі:

$$DP1 = \frac{(|P1 - 30| + 30) \cdot (T_0 - T_2)}{3000}, \quad (4)$$

де – Т2 – перше значення з масиву температур (тобто 2 с назад).

При моделюванні температури час виходу на номінальне значення температури (600°С) від початкового (20°С) склав 20 хвилин.

Для керування кранами подачі вхідних компонентів розроблена технологічна схема, що містить програмні компоненти бібліотеки (рис. 6).

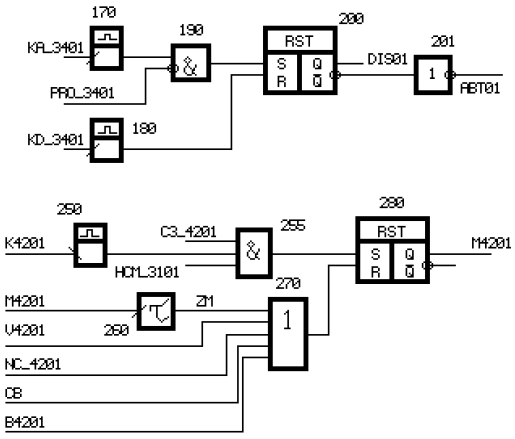


Рис. 6. Схема автоматів керування кранами подачі вхідних компонентів

Для кожного виконавчого пристрою розроблена мнемосхема керування, яка включає елементи та виконавчі органи такі як найменування та номер крану, можливість установки режимом керування та ін. (рис. 7).

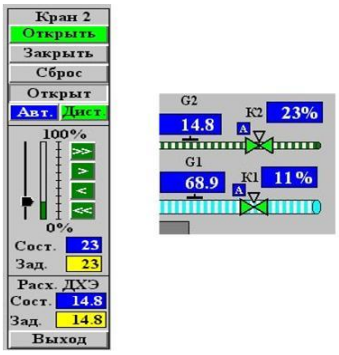


Рис. 7. Фрагменти мнемосхем керування виконавчими пристроями

При видачі дискретного сигналу для керування виконавчими пристроями (рис. 8) в програмний модуль MFDCZD заносить відповідний код каналу та проводиться діагностика модулю (вихід DDIA → логічна "1").

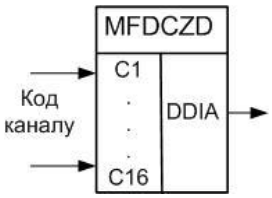


Рис. 4.6. Видача дискретного сигналу

Після генерації системи з підключеними необхідними програмними модулями програма функціонування завантажується у пам'ять ПЛК.

Висновки. Запропонована програмна бібліотека алгоритмічних елементів для проектування схем технологічного процесу виробництва вінілхлориду значно прискорює створення програмного забезпечення функціонування системи, завантаження програми функціонування в ПЛК та впровадження розробленої системи у виробництво. Наявність різноманітних елементів програмної бібліотеки дозволяє швидко та зручно змінювати алгоритми роботи системи. Основні теоретичні положення і результати розробленої АСК ТП виробництва вінілхлориду перевірялись з використанням програмних засобів на базі операційної системи реального часу QNX v.4.25 фірми QSSL (QNX Software System Ltd.), бібліотеки

алгоритмічних модулів «Уніконт», мови програмування та транслятора Watcom C++, графічної оболонки Photon microGUI, системи відображення Photon Application Builder фірми QSSL, SCADA-системи „Уніконт-М", SCADA-системи „Кварц", ПЛК Fastwell на базі процесора CPU686 фірми Octagon Systems.

1. Рязанцев О.І. Аналіз функцій системи автоматизації та програмно-технічних засобів керування процесом аерозольного нанокаталізу / О.І. Рязанцев, В.С. Кардашук // Вісник Східноукраїнського національного університету ім. Володимира Даля. – Луганськ: СНУ ім. В Даля, 2010. – № 7 (154), ч. 2. – С. 140–144.
2. Рязанцев О. І. Реалізація функцій керування на нижньому рівні в системі автоматизації керування процесом аерозольного нанокаталізу / О.І. Рязанцев, В.С. Кардашук // Вісник Східноукраїнського національного університету ім. Володимира Даля. – № 15 (186), ч. 2, 2012. – С. 170–175.
3. Рязанцев О.І. Методи та програмно-технічні засоби автоматизації керування процесом аерозольного нанокаталізу / О.І. Рязанцев, В.С. Кардашук // Радіоелектроніка, інформатика, управління. - Запоріжжя: ЗНТУ, 2011. - № 1(24). - С. 164-171.
4. Кардашук В.С. Математична модель підсистеми керування температурою для процесу знешкодження відходів хлорорганічного синтезу / Кардашук В.С., Пономарчук О.В., Маринич М.В. // 1-а міжнародна науково-практична конференція «Теоретичні і прикладні аспекти комп'ютерних наук та інформаційних технологій» TACSIT-2015, 15-16 травня, м. Сєвєродонецьк. – С. 21-24.
5. Ерофеев А.А. Теория автоматического управления: учебник [для вузов] / Ерофеев А.А. - [2-е изд., перераб. и доп.]. - СПб.: Политехника, 2002. - 320 с.: ил.

УДК 004.67
Савенко О.С., к.т.н, Лисенко С.М., к.т.н., Нічепорук А.О., асп.
Хмельницький національний університет

МЕТОД ВИЯВЛЕННЯ МЕТАМОРФНИХ ВІРУСІВ У КОРПОРАТИВНІЙ МЕРЕЖІ НА ОСНОВІ МОДИФІКОВАНИХ ЕМУЛЯТОРІВ

Савенко О.С., Лисенко С.М., Нічепорук А.О. Метод виявлення метаморфних вірусів у корпоративній мережі на основі модифікованих емуляторів. У статті представлено метод виявлення метаморфних вірусів з використанням мережних модифікованих емуляторів. Ідея методу полягає у формуванні оцінки схожості копій метаморфного вірусу, що досягається шляхом створення на кожному хості свого модифікованого мережного емулятора. Отримані копії метаморфних вірусів порівнюються з використанням метрики Дамерау-Левенштейна. Для формування висновку про інфікування системи метаморфним вірусом здійснюється класифікація отриманих копій з використанням моделей подібності метаморфних вірусів.
Ключові слова: метаморфний вірус, модифікований емулятор, відстань Дамерау-Левенштейна, опкоди.

Савенко О.С., Лысенко С.Н., Ничипорук А.А. Метод выявления метаморфных вирусов в корпоративной сети на основе модифицированных эмуляторов. В статье представлен метод обнаружения метаморфных вирусов с использованием модифицированных эмуляторов. Идея метода заключается в формировании оценки сходства копий метаморфный вируса, что достигается путем создания на каждом хосте своего модифицированного эмулятора. Полученные копии метаморфных вирусов сравниваются с использованием метрики Дамерау-Левенштейна. Для формирования вывода о инфицировании системы метаморфным вирусом осуществляется классификация полученных копий с использованием моделей сходства метаморфных вирусов.
Ключевые слова: метаморфный вирус, модифицированный эмулятор, расстояние Дамерау-Левенштейна, опкоды.

Savenko O.S., Lysenko S.M., Nicheporuk A.O. The method of identifying metamorphic virus in the corporate network based on modified emulators. The article presents a method of detection the metamorphic virus with using a modified network emulators. The idea of a method is to create the estimate of the similarity metamorphic copies of the virus, which is achieved by creating for each host his own modified emulator. Received copies of metamorphic viruses compare by using metrics Damerau-Levenshtein. To form the conclusion that the metamorphic virus infected systems, done the classification of obtained copies using models similarities of metamorfic viruses.
Keywords: metamorphic virus, modified emulator, distance Damerau-Levenshtein, opcodes.

Постановка проблеми. На сьогоднішній день виявлення комп'ютерних вірусів є одним з головних завдань інформаційної безпеки. Віруси завдають збитків як інформації так і самій робочій станції, що негативно позначається на робот всієї комп'ютерної системи вцілому. Серед всієї множини вірусних програм одне з провідних місць займають метаморфні віруси.

Виявлення метаморфних вірусів є складним завданням, у зв'язку з використанням ними техніки обфускації програмного коду. Використання обфускації дозволяє створювати різні копії одного і того ж вірусу. Сума збитків, що спричиняють віруси, зокрема метаморфні, сягають мільйонів доларів. За даними Symantec у 2011 метаморфний вірус Sality інфікував близько 3 мільйонів комп'ютерів у світі [1], а вже станом на кінець 2015 року, за даними ESET, увійшов в п'ятірку найбільш розповсюджених вірусів (1,43% від загальної кількості всіх виявлених загроз) [2], зберігаючи при цьому позитивну динаміку поширення.

Тому, актуальною постає задача розробки методу, що дозволить здійснювати виявлення нових метаморфних вірусів та копій вже існуючих, які здійснюють інфікування .PE .EXE файлів, шляхом приєднання власного коду до корисних програм (benign program).

Аналіз досліджень. Оскільки копії метаморфних вірусів змінюються від покоління до покоління, результати досліджень [3,6] показали неможливість виявлення метаморфних вірусів за допомогою сигнатурного аналізу.

Серед всіх відомих методів виявлення метаморфних вірусів можна виділити два основних підходи: статичні та динамічні методи виявлення. Статичні методи здійснюють аналіз підозрілого файлу без його безпосереднього виконання. Перевагою такої групи методів є трасування всіх можливих шляхів виконання програми, в той час як програма, що аналізується динамічними методами, може виконуватись тільки по одному шляху. Проте, у зв'язку з тим, що всі статичні методи передбачають попереднє дизасемблювання виконуваного файлу, вони не здатні здійснити аналіз виконуваного файлу, до якого було застосовано техніку шифрування або обфускації виконуваного коду [4].

Для відслідковування поведінки динамічні методи передбачають виконання підозрілого файлу. Одним з головних методів динамічного аналізу є емуляція виконання. Такі динамічні методи включають моніторинг API викликів, файловий моніторинг, монітор процесів, поведінковий аналіз та мережевий монітор [3-5]. Підозріла програма запускається у захищеному середовищі, що дозволяє здійснити аналіз її виконання. Проте, у випадку використання вірусними програмами антивідлашоджувальних та антиемуляційних технологій дані методи не здатні здійснити виявлення вірусу [6].

Обидва класи методів передбачають пошук сталих ознак. Цим ознаками можуть бути граф потоку управління програми, послідовність викликів API функцій, структурна інформація виконуваного файлу, опкоди підозрілої програми.

У роботі [7] запропоновано метод оцінки схожості метаморфних вірусів, шляхом побудови гістограми інструкцій для кожної підпрограми, з подальшим їх порівнянням за допомогою метрики city blocks. Проте, Описаний підхід є не ефективним для вірусів, що використовують техніку перемішування блоків (code transposition).

Іншим підходом виявлення метаморфних вірусів є формування метрики подібності метаморфних вірусів на основі графу потоку виконання програми [8]. Недоліком даного методу є відсутність оцінки послідовності API викликів.

Отже, аналіз предметної області показав необхідність у розробці та вдосконалені існуючих методів виявлення метаморфних вірусів.

Метод виявлення метаморфних вірусів у корпоративній мережі на основі модифікованих емуляторів. Запропонований метод передбачає порівняння зразка коду F_P метаморфного вірусу з його копію F_S ,що отримана з використанням модифікованих емуляторів, шляхом емуляції виконання на кожному хості у мережі підозрілої програми *P*. На рис.1 наведено узагальнену схему методу виявлення метаморфних вірусів у корпоративній мережі.

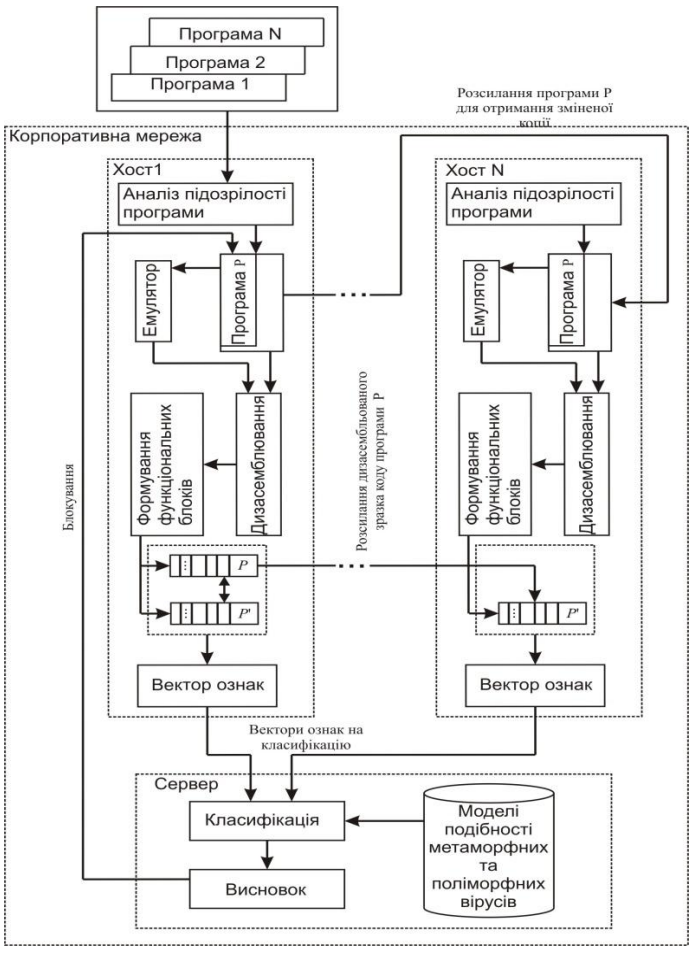


Рис. 1. Узагальнена схема методу виявлення метаморфних вірусів у корпоративній мережі

Для виявлення підозрілих дій на кожному хості корпоративної мережі використовується аналізатор підозрілості програми.

Кожна окрема дія, що виконується підозрілою програмою, не є небезпечною. Проте, виконання певної послідовності таких дій може свідчити про можливу небезпеку інфікування вірусом. Наприклад, якщо деяка програма записує себе в ключ системного реєстру автозавантаження, зчитує дані введені з клавіатури та з певним інтервалом надсилає повідомлення за певною адресою, то це свідчить про потенційну загрозу.

Кожна програма, що надходить в систему маркується як підозріла чи непідозріла, тобто:

$$P=\{suspicious, non\text{-}suspiciuous\}$$

Подамо вектор ознак, що визначає належність програми до одного з двох класів наступним чином:

$$\overline{U} = (M, Q, J, Y, L, N, H) \quad (1)$$

де, M – спроба програми отримати права адміністратора системи, Q – спроба відкриття або закриття системного порту, J – спроба видалення файлу, Y – створення файлу або процесу, L – перехоплення даних, що вводяться з клавіатури, N – розсилка повідомлень в мережу, H – створення або запис в системний реєстр.

Кожна ознака приймає значення 0 або 1, де 1 свідчить про активізацію відповідної ознаки, 0 – навпаки. Програму вважатимемо за підозрілу, якщо:

$$P = suspicious, if\ \forall u \in \overline{U}, (u_i = 1 \wedge u_j = 1)$$

Програма, для якої $P=suspicious$ надходить в систему виявлення метаморфного коду.

Для отримання зміненого зразку коду F_S, здійснюється емуляція виконання програми P. Процес емуляції виконання представляє собою розбір програмного коду на інструкції та імітацію їх виконання у віртуальному середовищі.

Використання однотипного емулятора на всіх хостах мережі не дозволить з високим ступенем достовірності здійснити виявлення метаморфних вірусів, оскільки і використання однакових емуляторів дозволить отримати лише однакові зразки коду. Для прояву метаморфних властивостей необхідним є забезпечення різних умов виконання шкідливого програмного коду. Тому, на кожному хості створюються модифіковані емулятори.

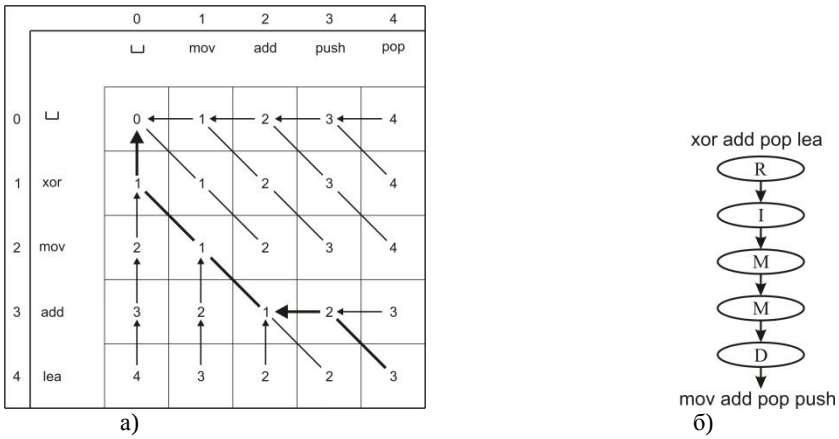
Структура емулятора наступна: віртуальний процесор, визначає набори інструкцій, що доступні для роботи (MMX, SSE, SSE2 та ін.) та включає в себе набори віртуальних регістрів; оперативна пам'ять та віртуальний стек; віртуальний мережний контролер; тип ОС (підтримка API-функцій, системного реєстру та портів) та модуль евристики.

Для протидії антиемуляційним технологіям, що використовуються у метаморфних вірусах, у емуляторі використовується модуль евристики. Для кожної операції, що виконується віртуальним CPU встановлюється фіксований час обробки та здійснюється перевірка повторів виконання певної операції (наприклад, якщо в тілі вірусу є цикл, що здійснює операцію інкременту змінної велику кількість разів).

З метою отримання початкового зразку коду F_P, здійснюється дизасемблювання програми P. Результатом процесу дизасемблювання є множина асемблерних інструкцій x86/x64. Для побудови вектора ознак, з метою спрощення реалізації, використовуються лише опкоди інструкцій, операнди відкидаються.

Отриманий лістинг дизасембльованих інструкцій розбивається на функціональні блоки (ФБ). Кожен ФБ складається з інструкцій, що розташовані між інструкціями переходів (jz, jnz, jmp та ін.).

З огляду на механізми створення метаморфними вірусами власних копій, що використовують техніки вставки, видалення та переміщення власних інструкцій, для пошуку схожості між ФБ двох зразків коду F_P та F_S використовується дистанція Дамерау – Левенштейна. Для пошуку дистанції Дамерау – Левенштейна використаємо алгоритм поліноміальної складності Вагнера-Фішера [9], який дає змогу сформувати найкоротший ланцюг перетворення, для приведення множини опкодів програми після емуляції у множину опкодів програми до емуляції.



а) б)

Рис. 2. Матриця Дамерау-Левенштейна для двох ФБ (а) та ланцюг перетворення ФБ1 в ФБ2 (б)

Розглянемо програму *P*, яка складається з множини асемблерних команд p_i, тобто $P=\{p_1, p_2,..., p_k\}$. Розіб'ємо програму *P* на функціональні блоки довільної довжини, що починаються із інструкцій умовного переходу jmp, jz та ін. і закінчуються ними, тобто $P=\{B_1, B_2,..., B_l\}$. Тоді можна записати: $P=\{B_1=\{p_1, p_2,..., p_{i-1}\},..., B_l=\{p_i, p_{i+1},..., p_k\}\}$.

Нехай функціональний блок *B*, що складається з множини опкодів, довжиною $|B|=m$ записується як $p_1, p_2,..., p_m$, де p_i представляє *i*-й опкод *p*. Підмножина опкодів $x_i, x_{i+1},..., x_j$, функціонального блоку *B* буде позначатись $B(i, j)$.

Вага перетворення опкода *a* в опкод *b* позначимо через $w(a,b)$. Таким чином, $w(a,b)$ – вага заміни одного опкоду на другий опкод, коли $a \neq b$, $w(b,a)$ – вага операції транспозиції, $w(a,\varepsilon)$ – вага видалення, а $w(\varepsilon,b)$ – вага вставки *b*.

В базовому алгоритмі вага всіх операцій дорівнює одиниці. В запропонованому алгоритмі вага операцій вставки та видалення опкоду становить 2, операція обміну – 1, а операції заміни одного опкоду на інший визначається наступними правилами:

$$w(a_i,b_j)=\left\{\begin{array}{c} 1, if\ (a_i=mov; b_j=push, b_{j+1}=pop)\ or \\ ((a_i=mov)\ or\ (a_i=xor)\ or\ (a_i=and)\ or\ (a_i=sub) \\ (b_j=mov)\ or\ (b_j=xor)\ or\ (b_j=and)\ or\ (b_j=sub))\ or \\ ((a_i=jnz)\ or\ (a_i=jz)\ or\ (a_i=jmp) \\ (b_j=jnz)\ or\ (b_j=jz)\ or\ (b_j=jmp)), a_i\in B^{F_P}, b_j\in B^{F_S}, a_i\neq b_j \\ 2, otherwise \end{array}\right\} \quad (2)$$

Нехай B_g та B_h – два ФБ, що складаються з послідовності опкодів (довжиною n та m відповідно) над скінченим алфавітом асемблерних інструкцій $A=(a_1, a_2,..., a_k)$, причому B_g ФБ програми F_P, позначимо $B_g^{F_P}$, а B_h – ФБ тієї ж програми після емуляції виконання F_S, позначимо $B_h^{F_S}$. Тоді відстань Дамерау – Левенштейна $dL(B_g^{F_P}, B_h^{F_S})$ обчислимо наступним чином:

$dL(B_g^{F_P}, B_h^{F_S})=OPT(N,M)$, де

$$OPT=\left\{\begin{array}{ll} 0, & i=0,\ j=0 \\ i, & j=0,\ i>0 \\ j & i=0,\ j>0 \\ \min\left\{\begin{array}{l} OPT(i, j-1)+w(a,\varepsilon) \\ OPT(i,-1j)+w(\varepsilon,b) \\ OPT(i,-1, j-1)+w(a,b) \\ OPT(i-2, j-2)+w(b,a) \end{array}\right. & j>0, i>0 \end{array}\right. \quad (3)$$

Після отримання відстані Дамерау-Левенштейна для двох блоків B_g та B_h, формується зважене усереднене значення відповідного параметру вектора ознак для всіх блоків коду. Для отримання зваженої усередненої оцінки параметрів використаємо показник центру розподілу зважене середнє арифметичне (4):

$$dL = \left\lfloor \frac{\sum_{i=1}^{n} dL_i * f_l}{\sum_{i=1}^{n} f_l} \right\rfloor \quad (4)$$

де,

dL_i – відстань Левенштейна для ФБ B_i, f_l – кількість ФБ з значенням dL_i

Оскільки відстань Дамерау-Левенштейна визначає мінімальне значення необхідних операції заміни, вставки, видалення та транспозиції, що відповідно є цілочисельним значенням, отримана ознака округлюється в меншу сторону, для знаходження найменшої різниці між копіями метаморфних вірусів.

Для решти ознак (кількість операцій співпадіння, вставки, видалення, заміни, транспозиції) унормування відбувається аналогічним чином.

Таким чином, вектор ознак схожості копій метаморфних вірусів на основі метрики Дамерау-Левенштейна подамо наступним чином:

$$\overline{S} = \langle dL, T, D, I, R, M \rangle \quad (5)$$

де, T – кількість необхідних операцій обміну опкодів для перетворення блоку програму F_p у F_s ($F_p = F_s$);

D – кількість необхідних операцій видалення опкоду;

I – кількість необхідних операцій вставки опкоду;

R – кількість необхідних операцій заміни відповідних опкодів;

M – кількість співпадінь між опкодами в функціональному блоці програми F_p та F_s.

Для формування нечіткого логічного висновку про інфікування системи метаморфним вірусом, отримані вектори ознак схожості надходять на серверну частину для їх класифікації. Результатом роботи системи є ступінь приналежності кожної копії до одного із сімейств метаморфних вірусів.

Експерименти. Для тестування та генрації копій метаморфних вірусів [12] було використано такі метаморфні генератори: Next Generation Virus Creation Kits (NGVCK), Second Generation Virus Generator (G2) та Virus Creation Lab for Win32 (VCL32). Тестування системи проводилось у корпоративній мережі, що складається з 80 комп'ютерів.

На кожному хості мережі були створені модифіковані мережні емулятори, параметри яких представлено в таблиці 1.

Таблиця 1. Параметри модифікованих мережних емуляторів для хостів мережі

Номер хоста	Початкові параметри мережних модифікованих емуляторів							
	Набір інструкцій	Тип ОС	Розмір віртуальної ОЗП	Системна дата	Віртуальний мережевий контролер	Час виконання емуляції інструкції (xor), од	Адреса початку емуляції	Архітектура ЦП
1	SSE2	W7	8	25.11	Так	1	0x001A4810	x86
2	Hyper-threading	W8.1	8	06.02	Ні	2	0x001A4620	x86
…	…	…	…	…	…	…	…	
N	MMX	W10	4	10.04	Ні	1	0x001A5214	x64

Для здійснення висновку про інфікування метаморфним вірусом використовується система нечіткого логічного висновку.

Система нечіткого логічного висновку побудована за допомогою пакету Fuzzy Logic Toolbox системи Mathlab. Для проведення досліджень в системі були задіяні наступні параметри: алгоритм – Мамдані, метод агрегування, метод акумуляції, метод дефазифікації. Система складається з 6 входів та

одного виходу. В якості функцій приналежності для входів було обрано трапецевидну, для виходу – трикутну. Кожна ознака вектора схожості копій метаморфних вірусів є вхідною лінгвістичною змінною для системи нечіткого логічного висновку. Для кожної лінгвістичної змінної задано терм множину Low, Medium та High.

Результатом роботи системи нечіткого логічного висновку є ступінь приналежності об'єкту до одного з чотирьох класів – трьох шкідливих та одного класу довірених додатків. Якщо ступінь приналежності невідомого об'єкту належить діапазону від 0 до 0,25, то невідомий об'єкт класифікується як довірений додаток; якщо ступінь приналежності лежить на проміжку від 0,26 до 1, то невідомий об'єкт відноситься до одного з класів метаморфних вірусів. Значення від 0,26 до 0,49 визначають 1 перший клас метаморфний вірусів, значення від 0,5 до 0,74 – 2 клас, значення від 0,75 до 1 – 3 клас. У таблиці 2 наведено результати нечіткого логічного висновку для підозрілого файлу. В результаті 15% копій не змінились, 5% віднесено до першого класу метаморфних вірусів, 11,25 – до третього та 68, 75% до другого.

Таблиця 2. Результати тестування системи

№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ
1	0,65	15	0,57	29	0,81	42	0,72	55	-	68	-
2	-	16	0,86	30	0,70	43	0,51	56	0,61	69	0,72
3	0,19	17	0,57	31	0,55	44	0,53	57	0,63	70	0,83
4	0,64	18	0,63	32	-	45	0,63	58	0,84	71	-
5	0,45	19	0,56	33	0,62	46	0,78	59	0,67	72	0,53
6	0,59	20	0,68	34	0,23	47	0,62	60	0,69	73	-
7	-	21	0,72	35	0,59	48	-	61	0,67	74	0,59
8	0,52	22	0,41	36	0,54	49	0,70	62	-	75	0,47
9	-	23	0,69	37	0,68	50	0,26	63	0,51	76	0,74
10	0,61	24	0,79	38	-	51	0,56	64	0,82	77	-
11	0,14	25	0,68	39	0,39	52	0,81	65	0,69	78	0,58
12	0,69	26	0,24	40	0,56	53	0,74	66	0,36	79	0,57
13	0,78	27	0,56	41	0,54	54	0,56	67	0,52	80	0,73
14	0,35	28	0,69								

Якщо підозрілий об'єкт віднесено до четвертого класу (значення від 0 до 0,25), то це свідчить, що даний зразок може бути довіреним додатком або метаморфним вірусом і потребує подальшого дослідження.

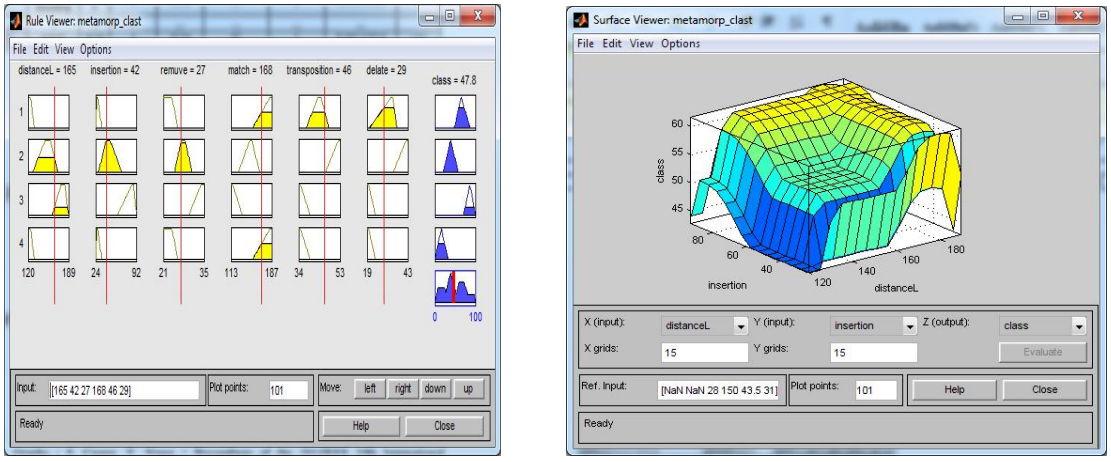


Рис. 3. Результат нечіткого логічного висновку для вектора схожості копій метаморфного вірусу.

Висновки. Аналіз предметної області показав необхідність у вдосконалені існуючих методів виявлення метаморфних вірусів. Авторами запропоновано метод виявлення метаморфних вірусі з використанням модифікованих мережних емуляторів на хостах локальної мережі. Класифікація вірусів здійснюється на основі системи нечіткого логічного висновку. Такий підхід дозволяє підвищити достовірність виявлення метаморфних вірусів. Віднесення досліджуваного програмного забезпечення до одного з класів дозволяє стверджувати, що таке програмне забезпечення є метаморфних вірусом, а не корисним додатком, до якого застосована техніка обфускації програмного коду. Було проведено ряд експериментів, і виявлено 85% копій, що були віднесені до одного з трьох класів метаморфних вірусів.

Список використаних джерел

1. Falliere, N.: Sality: Story of a Peer-to-Peer Viral Network. Technical report, Symantec Labs (2011)
2. Вирусная статистика. Обзор киберугроз ноября 2014 года [Електронний ресурс]. – режим доступу: https://www.esetnod32.ru/company/viruslab/statistics/?id= 896397
3. Alsagoff, S.N Malware self protection mechanism issues in conducting malware behavior analysis in a virtual environment as compared to a real environment. In: Proc. of International Symposium in Information Technology (ITSim), pp 1326-1331 (2010)
4. Bayer, U., Kirda, E., Kruegel C. Improving the Efficiency of Dynamic Malware Analysis. In: Proc. of 25th Symposium on Applied Computing (SAC), New York, pp. 1871-1878 (2010)
5. Park, Y., Reeves, D. S. Identification of Bot Commands by RunTime Execution Monitoring. In: Proc. of Security Applications Conference, ACSAC'09, Honolulu, pp. 321-330 (2009)
6. Peidai, X., Xicheng, L., Yongjun, W., Su, J. Eliminate Evading Analysis Tricks in Malware using Dynamic Slicing. In: International Journal of Security and Its Applications, vol.7, pp. 245-256 (2013)
7. Rad, B. B., Masrom, M.: Metamorphic Virus Variants Classification Using Opcode Frequency Histogram. In: 14th WSEAS international conference of computers, pp. 147-155, WSEAS (2010)
8. Cesare, S., Xiang, Y.: Malware variant detection using similarity search over sets of control flow graphs. In: 10th International Conference on Trust, Security and Privacy in Computing and Communications, Washington, DC, USA, pp. 181-189 (2011)
9. Wagner, R., Fisher, M.: The string to string correction problem. Journal of the ACM, 21, pp. 168-178 (1974)

УДК 004.738:004.9
Шолом П.С., Котвицька А.Ю., Самарчук В.Ф.
Луцький національний технічний університет

ЕЛЕКТРОННИЙ НАВЧАЛЬНИЙ ПІДРУЧНИК НА БАЗІ PHP FRAMEWORK YII2

Шолом П.С., Котвицька А.Ю., Самарчук В.Ф. Електронний навчальний підручник на базі PHP Framework Yii2. Розроблено електронний підручник зі зручним та інтуїтивно зрозумілим інтерфейсом для дисципліни «Інженерія програмного забезпечення». Окрім теоретичного наповнення підручник містить модуль тестування для перевірки знань по пройденому матеріалу.
Ключові слова: електронний підручник, MYSQL, СУБД, БД, YII2 FRAMEWORK, CSS, PHP, PHPSTORM, HTML

Шолом П.С., Котвицкая А.Ю., Самарчук В.Ф. Электронное учебное пособие на базе PHP Framework Yii2. Разработан электронный учебник с удобным и интуитивно понятным интерфейсом для дисциплины «Инженерия программного обеспечения». Кроме теоретического наполнения учебник содержит модуль тестирования для проверки знаний по пройденному материалу.
Ключевые слова: электронный учебник, MYSQL, СУБД, БД, YII2 FRAMEWORK, CSS, PHP, PHPSTORM, HTML

Sholom P., Kotvytska A., Samarchuk V. E-learning manual based on PHP Framework Yii2. The electronic manual with a convenient and intuitive interface for subject "Software Engineering" is developed. Apart from the theoretical content the textbook contains testing module to test knowledge on the passed material.
Keywords: electronic manual, MYSQL, database, database, YII2 FRAMEWORK, CSS, PHP, PHPSTORM, HTML

Постановка проблеми. На даний час усе більшої значимості набуває розробка та використання в навчальному процесі електронних навчальних систем, що розробляються із застосуванням гіпертекстових і мультимедійних технологій. Такі системи називають інтерактивними навчальними веб-матеріалами та можуть використовуватися не тільки для денної, заочної форм навчання, але й знайти широке застосування в дистанційній формі навчання для самостійного опрацювання навчального матеріалу. Тому розробка таких електронних засобів є досить **актуальною**, оскільки дасть можливість впровадити їх у навчальний процес.

Аналіз існуючих форм та видів електронних підручників показує, що вони є досить різноманітні – у вигляді простого тексту, текстові з оформленням (HTML, Electronic Publication, OpenDocument, SGML, XML, FictionBook, TeX, PDF, Microsoft HTMLHelp, PostScript, ExeBook, Mobipocket тощо), графічні растрові (TIFF, JPEG, DjVu і т.п.), мультимедіа книги (SWF, EXE, мультимедіа книга, аудіокниги і т.п.), книги у форматі Java-мідлетів для мобільних пристроїв тощо. Прийняття правильного рішення про вибір середовищ і технологій розробки та виду самого підручника залежить від обраної аудиторії користувачів.

Мета роботи полягає у розробці електронного підручника зі зручним та інтуїтивно зрозумілим інтерфейсом, який окрім теоретичного наповнення містив би модуль тестування для перевірки знань по пройденому матеріалу.

Виклад основного матеріалу роботи.

Перш ніж розробляти інтерактивний підручник, необхідно обрати платформу та визначити яким технічним новинкам віддати перевагу. При цьому потрібно враховувати можливу швидкість з'єднання у потенційних користувачів. Тому для створення електронного підручника було обрано PHP, PHP Framework Yii2, бази даних MySQL, HTML 5 та CSS. Причину вибору таких засобів обумовлено нижче.

PHP є мовою програмування, код якої можна вбудовувати безпосередньо в html-код сторінок, які, у свою чергу, будуть коректно оброблені PHP-інтерпретатором. Обробник PHP починає виконувати код після відкриваючого тегу (<?php) і продовжує виконання до того моменту, поки не зустріне закриваючий тег (?>).

Фреймворком є інфраструктура програмних рішень, що полегшує розробку складних систем. Спрощено дану інфраструктуру можна вважати своєрідною комплексною бібліотекою. Yii2 – це високопродуктивний компонентний PHP-фреймворк, призначений для швидкої розробки сучасних веб-додатків. Він є універсальним і може бути задіяний у всіх типах веб-додатків, що

використовують PHP. Завдяки його компонентній структурі і відмінній підтримці кешування, фреймворк особливо підходить для розробки таких великих проектів як портали, форуми, системи керування вмістом (CMS), інтернет-магазини або RESTful-додатки.

Основні переваги (позитивні сторони) Yii2:

- Подібно до багатьох інших PHP-фреймворків Yii2 втілює архітектурний шаблон MVC (Model-View-Controller) та сприяє організації коду відповідно до вимог шаблону.
- Yii2 дотримується філософії простого й елегантного коду.
- Yii2 є full-stack фреймворком і включає в себе перевірені можливості, які добре себе зарекомендували: конструктори запитів та ActiveRecord для реляційних та NoSQL баз даних, підтримка REST API, багаторівневе кешування та інше.
- Yii2 надзвичайно розширюваний − можна налаштувати або замінити практично будь-яку частину основного коду. Завдяки надійній архітектурі розширень Yii2 досить легко використовувати або розробляти поширюванні розширення.
- Дуже висока продуктивність щодо інших фреймворків і CMS-систем.
- Кешування сторінок / розділів, окремих фрагментів.
- Перехоплення і обробка помилок, автоматичне тестування.
- Зручна аутентифікація та управління доступами на основі ролей.
- Висока масштабованість (можливість розширювати можливості проекту з часом без зміни платформи або написання з нуля).
- Підтримує зворотню сумісність між версіями.

На даний момент існує дві основні версії Yii: 1.1 та 2.0. Версія 1.1 є попереднім поколінням і знаходиться у стані підтримки. Версія 2.0 − це повністю переписаний Yii, що використовує останні технології і протоколи, такі як Composer, PSR, простори імен, трейти і багато іншого. 2.0 – поточне покоління фреймворку. На цій версії будуть зосереджені основні зусилля кілька наступних років. Yii 2.0 потребує PHP 5.4.0 та вище, також необхідні знання ООП, оскільки фреймворк повністю слідує цій парадигмі. Composer – це відносно новий і вже досить популярний менеджер залежностей для PHP. Здійснивши опис бібліотек, від яких залежить проект, Composer встановить потрібні бібліотеки. Причому Composer – це не менеджер пакетів в класичному розумінні. Так, він оперує із сутностями, які називають «пакетами» або бібліотеками, але встановлюються вони всередину кожного проекту окремо, а не глобально (це одна з основних відмінностей від старого PEAR).

На рисунку 1 представлено загальну архітектурну схему Yii2 framework.

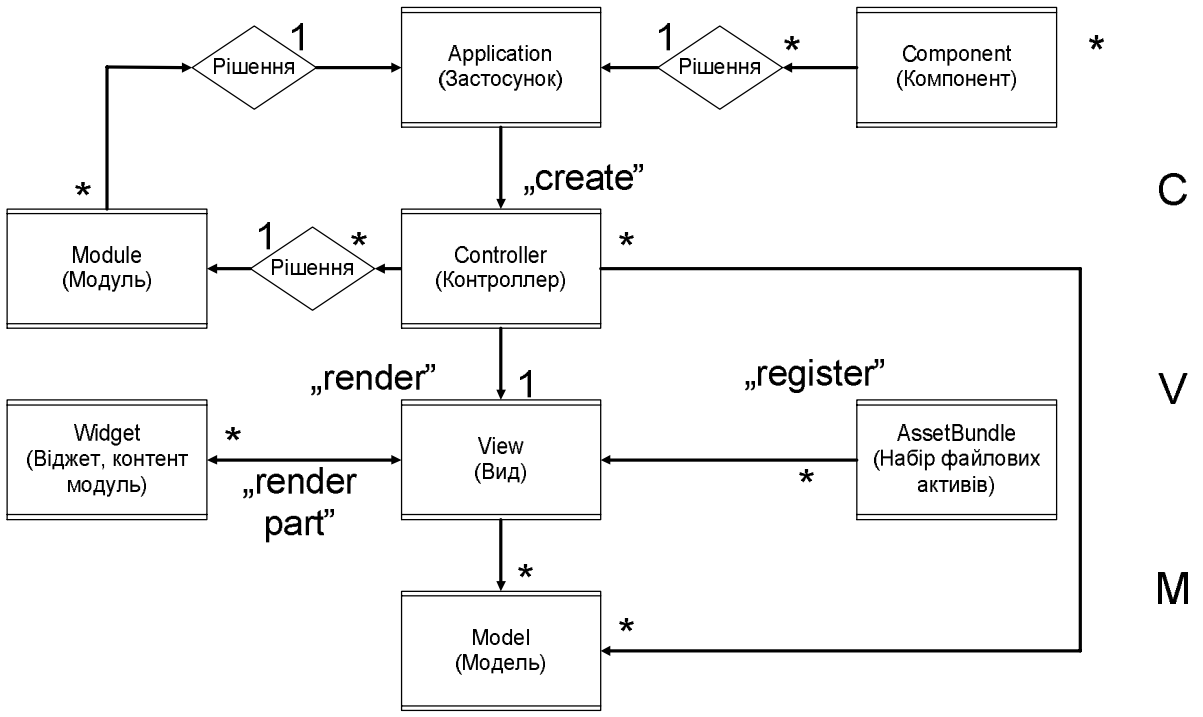


Рис. 1. **Загальна архітектура Yii2**

На даному рисунку можна побачити Aplication, що є точкою входу (для електронного підручника це файл index.php). Aplication підключає необхідні компоненти, працює з картинками і таблицями в залежності від потреб. По запиту Aplication визначає до якого контролера надійшов запит (так зване завдання) від користувача. В середині контролера є функція Ection яка безпосередньо обробляє запити. По обробці даних контролер може за необхідністю використовувати певні модулі, зокрема поширений модуль behavior (визначає модель поведінки, при створені виконує одні операції, при видалені − інші). Потім ідуть запити до Моделі. Після опрацювання ці всі дані за допомогою контролера надходять на View. View − це щось на кшталт шаблону файлів PHP, але з вставками HTML-коду. При необхідності в отриманні додаткових даних View може відправити запит, але це не бажано, оскільки така дія порушує архітектуру MVC. Для підключення JavaScript, CSS використовують AssetBundle. У ньому знаходиться перелік файлів, які необхідно підключити, у певній послідовність. Функція Register використовується для підключення AssetBundle. За допомогою Widget можна реалізовувати HMVC (ієрархічна модель-вид-контролер). Це одне із розширень архітектурного патерна (шаблону) MVC, що дозволяє вирішити проблеми масштабованості. Функція Render здійснює підстановку у змінну деякого вмісту, функція Render Part підставляє лише конкретну частину із цього вмісту, яка потрібна на даний момент.

Керуючись основоположними стратегіями архітектури Yii2 Framework, було розроблено архітектурну схему електронного підручника, що представлена на рисунку 2.

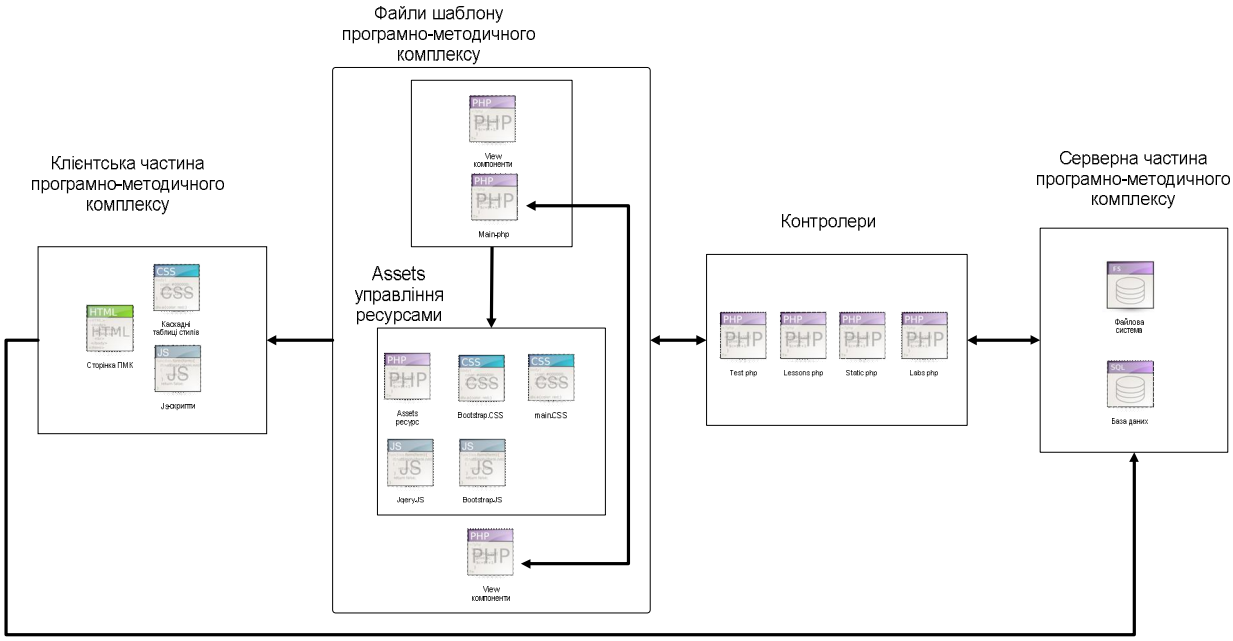


Рис. 2. **Архітектура електронного підручника**

При подачі користувачем запиту вступає в учать серверна частина електронного підручника з файлами системи та БД. Далі іде запит на обробку в контролер (при необхідності контролер може звертатись до БД). Потім оброблені дані передаються до View, де підключаються необхідні Assets. Після цього здійснюється рендерінг (замість змінних підставляється їх вміст) і на виході отримується HTML-файл із підключеними Java Scripts та CSS.

Одним із основних понять, з яким стикається розробник електронного підручника під час роботи із базами даних, є поняття міграцій. Міграція, в даному контексті, – це оновлення структури бази даних від однієї версії до іншої (зазвичай більш нової). У цьому сенсі термін «міграція» використовується в багатьох джерелах (особливо цьому посприяли міграції з gem'a Active Record, що входить до складу Ruby on Rails). Однак при використанні цього терміна виникає двозначність: людина, яка не знає контексту, швидше подумає, що мова йде про

перенесення бази даних з однієї СУБД на іншу (MySQL => Oracle), а то і зовсім про міграцію процесів / даних між нодами кластера. Тому пропонується у випадках, коли контекст неочевидний, використовувати більш точний термін: версійна міграція баз даних.

Як і вихідний код, структура бази даних змінюється в процесі розробки і підтримки програми. Приміром, під час розробки може знадобитися додати нову таблицю або вже після розміщення додатка на сервері додати індекс або стовпець. При цьому важливо відстежувати зміни в структурі бази даних (що звуться міграціями) точно так само, як це робиться для вихідного коду. Якщо вихідний код і база даних не відповідають один одному, то швидше за все програма не буде працювати. Саме тому в Yii2 є підтримка міграцій. Це ще одна із особливостей, через яку обрано даний фреймворк. Він дозволяє відстежувати зміни в базі даних, застосовувати міграції або відкочувати вже застосовані.

Нижче наведено покроковий процес використання міграцій при розробці:

- створення нової міграції (наприклад, створення нової таблиці);
- її завантаження у систему контролю версій (SVN, GIT або іншу);
- оновлення із системи контролю версій та отримання нової міграції;
- застосування міграції до своєї локальної бази даних.

В Yii2 управління міграціями проводиться через консольну команду yiic migrate, яка підтримує створення нових міграцій, їх використання, відкат і повторне застосування міграцій, перегляд історії міграцій та нових міграцій.

Розробку електронного підручника здійснено в інтегрованому середовищі розробки JetBrains PhpStorm. У підручнику передбачено дві ролі: адміністратор та користувач.

Для доступу до адмін-панелі адміністратору програмного комплексу потрібно спочатку пройзти авторизацію. Після входу адміністратору надається доступ для внесення змін в таки пункти головного меню як: Модулі, Тестування, Теми, Лабораторні роботи, Статичне наповнення. При натиснені на пункт меню «Модулі» з'являється випадаюче меню з підпунктами, з яких адміністратором може бути обраний будь який для його подальшої зміни. Обравши пункт випадаючого меню «Модулі», адміністратору стають доступні опції створення нового модуля та видалення / оновлення вже існуючого чи тільки що створеного модуля. Обравши пункт підменю «Запитання до модулів» стає доступним створення запитань, які будуть на модульному контролі, та відповідних налаштувань. Є можливість пошуку вже існуючих запитань. Це зроблено з метою полегшення роботи при створенні нових запитань. Можна видаляти, оновлювати та переглядати вже існуючі запитання. При виборі підпункту «Відповіді» відбувається перехід на сторінку, де адміністратором створюються можливі варіанти відповідей до тестового запитання. Адміністратор також має доступ до створення (наповнення) відповідною інформацією.

Користувач має доступ до лекційного матеріалу, лабораторних робіт та проходження тесту. Перед проходженням тесту користувачу необхідно зареєструватись. Приклад проходження тестування показано на рисунках 3 та 4.

Висновки. Розроблено електронний навчальний підручник для різних форм навчання з метою покращення продуктивності навчання та засвоєння знань студентів. Даний програмний продукт дає змогу розширити аудиторію слухачів дистанційного навчання та підняти рівень якісної самоосвіти студентів. Підручник містить модуль тестування, що має на меті скорочення витрат часу викладачів на проведення тестового контролю.

Розробку електронного підручника виконано в інтегрованому середовищі розробки JetBrains PhpStorm з використанням функцій PHP-framework Yii2. В якості локального веб-сервера для реалізації зв'язку «клієнт-сервер» використано Open Server.

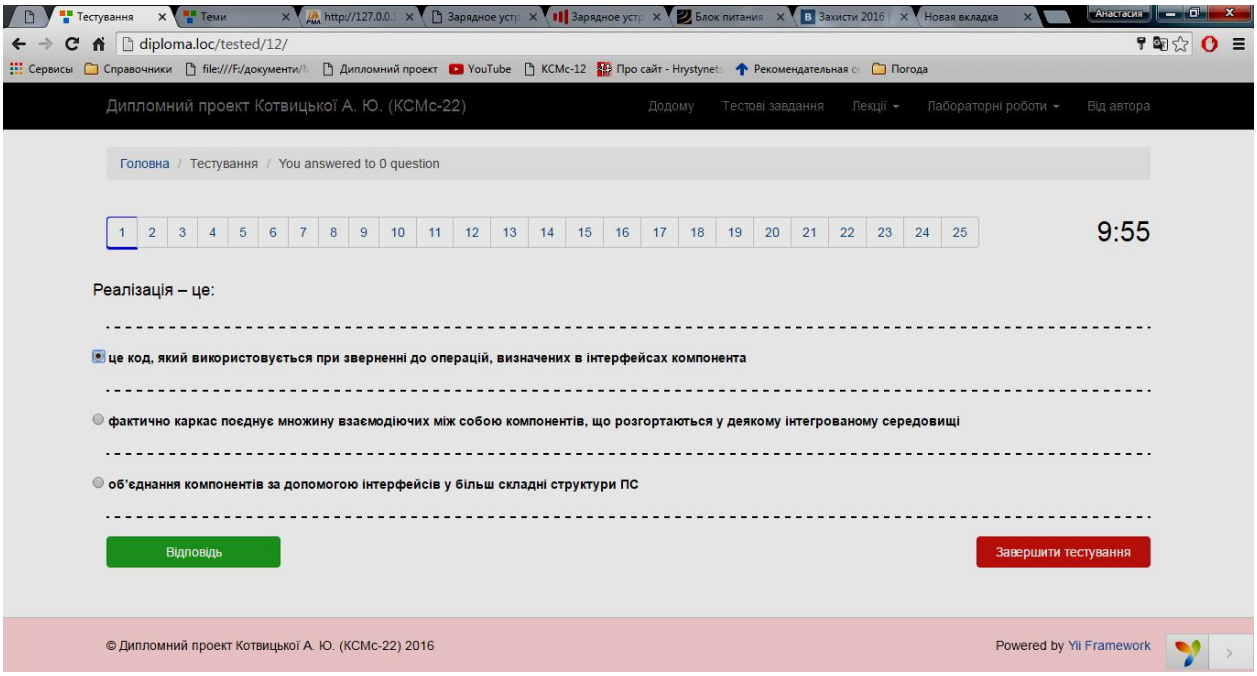


Рис. 3. **Відображення результатів тестування**

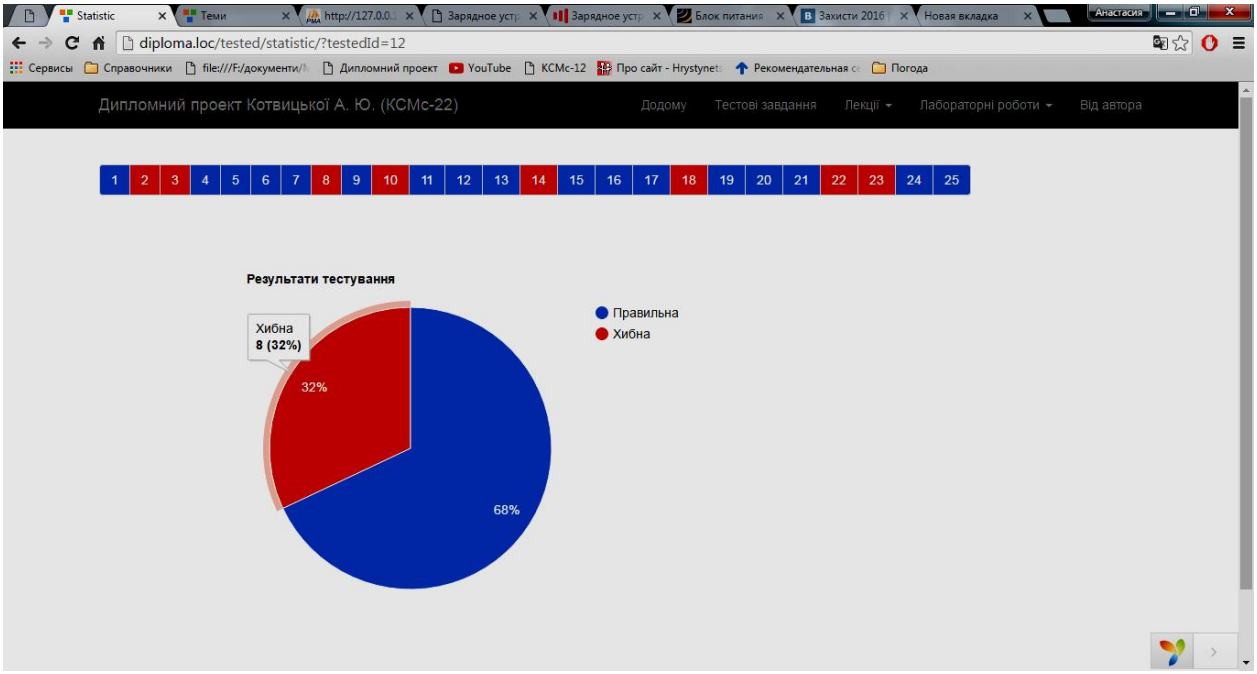


Рис. 4. **Відображення результатів тестування**

1. Yiifravework community: [Електронний ресурс] // Yii fravework community – Українська спільнота Yii fravework. – Режим доступу: https://yiiframework.com.ua/ru/doc/guide/2/
2. Safronov M., Winesett J. Web Application Development with Yii 2 and PHP. – Packt Publishing Ltd, 2014. [Електронний ресурс]. – Режим доступу: https://books.google.com.ua/books?hl=ru&lr=&id=XZefBAAAQBAJ&oi=fnd&pg=PT17&dq=framework+Yii2&ots=wYFbeq8sLA&sig=yAUGa7kfsTmK7LDqOrOew100R4U&redir_esc=y#v=onepage&q=framework Yii2&f=false
3. Марк Сафронов. Разработка веб-приложений в Yii 2. – М.: ДМК Пресс, 2015. − 392 с.
4. PHP.SU: [Електронний ресурс] // Основи PHP – Спільнота forum.php.su. – Режим доступу: http://www.php.su/php/?php/
5. Миграции баз данных. [Електронний ресурс]. – Режим доступу: https://yiiframework.com.ua/ru/doc/guide/2/db-migrations/

УДК 338.244:504.453
Филь Н.Ю
Харківський національний автомобільно-дорожній університет

МОДЕЛЬ ВИРТУАЛЬНОГО ОФИСА УПРАВЛЕНИЯ ПРОЕКТАМИ ЛИКВИДАЦИИ ПОСЛЕДСТВИЙ ЧРЕЗВЫЧАЙНЫХ ПРИРОДНЫХ СИТУАЦИЙ НА МАГИСТРАЛЬНЫХ АВТОДОРОГАХ

Филь Н.Ю. Модель виртуального офиса управления проектами ликвидации последствий чрезвычайных природных ситуаций на магистральных автодорогах. В статье разработана структурно-логическая модель организации виртуального офиса управления проектами ликвидации последствий чрезвычайных природных ситуаций магистральных автомобильных дорогах.
Ключевые слова: управление проектами, чрезвычайная природная ситуация, магистральная автомобильная дорога.

Філь Н.Ю. Модель віртуального офісу управління проектами ліквідації наслідків надзвичайних природних ситуацій на магістральних автодорогах. У статті розроблена структурна-логическая модель организации віртуального офісу управління проектами ліквідації наслідків надзвичайних природних ситуацій магістральних автомобільних дорогах.
Ключові слова: управління проектами, надзвичайна природна ситуація, магістральна автомобільна дорога.

Fil' N.U. Model of virtual project management office for the liquidation of natural emergency situation consequences on trunk roads. In the article the structural-logical model for organization of a virtual project management office for the liquidation of natural emergency situation consequences on trunk roads has been developed.
Keywords: project management, natural emergency situation, trunk road.

Анализ проблемы. Существующая тенденция к возрастанию масштабов чрезвычайных природных ситуаций (ЧПС) заставляет своевременно и обоснованно вырабатывать решения по ликвидации последствий ЧПС на магистральных автодорогах (МАД). С этой целью разрабатываются математические модели управления ЧПС на МАД, которые помогают формализовать и автоматизировать процессы принятия решений управления ликвидацией последствий ЧПС на МАД, помогают выбирать способы реализации принимаемых решений и оценивать их эффективность [1-2].

При возникновении ЧПС на МАД необходимо ликвидировать ее последствия в кратчайшие сроки, а выполнить эффективно все необходимые мероприятия можно только в том случае, если все эти действия осуществляются по единому алгоритму – технологии управления проектами ликвидацией последствий ЧПС на МАД [3].

Неотъемлемой составляющей современного офиса управления проектами является виртуальный офис – распределенная компьютерная система на базе телекоммуникационных сетей, которая позволяет пользоваться едиными программными средствами, едиными базами данных и знаний, осуществлять единый учет контроля, мониторинг работ по проекту, проводить видеоконференции, телекоммуникационные совещания в реальном режиме времени [4].

Анализ публикаций

В работе [4] рассмотрены основные принципы организации виртуального офиса. Подчеркнуто, что в настоящее время понятие и идеология виртуального офиса проекта приобретают все большее значение в связи с развитием сети Интернет и возрастанием значения программно-информационного и коммуникационного аспекта управления проектами.

В работе [5] проанализированы вопросы актуализации проектов городского благоустройства в системе качества городской жизни. Разработана структурно-логическая модель организации офиса управления проектами городского благоустройства. Предложенный подход позволил моделировать варианты наиболее оптимальных управленческих структур, способных снижать риски осуществления проекта, повышать гибкость, тем самым положительно влиять на качественные показатели проектов городского благоустройства.

Основы организации виртуального офиса проекта как самостоятельной инфраструктуры рассмотрены в работе [6]. В работе рассматриваются основные отличные особенности виртуальных и традиционных структур. Предложена схема организации виртуального проектного офиса и схема функций, которые являются основными модулями информационной системы виртуального офиса.

Опыт по реализации в Днепропетровской области в 2011-2013 гг. проектов по развитию системы предоставления электронных административных услуг анализируется в работе [7]. Исследуются проекты «Региональный виртуальный офис предоставления электронных административных услуг» и «Административные услуги: упрощенный доступ через почту».

Однако, остаются мало изученными вопросы разработки, внедрения и практического применения виртуальных офисов при ликвидации чрезвычайных ситуаций.

Цель исследования. Повышение эффективности управления проектами ликвидации последствий ЧПС на МАД за счет разработки и внедрения структурно-логической модели организации виртуального офиса управления проектами ликвидации последствий ЧПС на МАД.

Основные результаты исследования

Виртуальный офис − не привязанный к определенному, а представляющий собой программно-телекоммуникационную среду, обеспечивающую возможность работы и коммуникаций по единым стандартам [8].

Таким образом, при управлении проектами ликвидации ЧПС на МАД необходимо использовать специфическую инфраструктуру – виртуальный офис, который обеспечит эффективную реализацию проектов ликвидации последствий ЧПС на МАД в рамках системы компьютерных, коммуникационных и информационных технологий проектов ликвидации последствий ЧПС на МАД (рис. 1).

Рис. 1. Структурно-логическая модель организации виртуального офиса управления проектом ликвидации последствий ЧПС на МАД

Виртуальный офис − единая среда для принятия управленческих решений и коммуникаций.

В таблице 1 представлен анализ программных средств (ПС), которое используется для соответствующих групп процессов.

Таблица 1 – Элементы групп процессов офиса

Основные группы процессов	Назначение	Возможные типы программного обеспечения
1	2	3
Группа процессов инициации	Определяет и авторизует проект или фазу проекта	– Текстовые редакторы, электронные таблицы, СУБД, электронная почта – ПС финансового анализа программы и проектов – ПС документирования процессов, выбора проекта

Группа процессов планирования	Определяет и уточняет цели и планирует действия, необходимые для достижения целей и содержания, ради которых был начат проект	– ПС календарного планирования и управления проектами; – ПС контроля информации проекта
Группа процессов выполнения	Совмещает человеческие и другие ресурсы для выполнения плана управления проектом	– ПС интеграции и выполнения операций проекта; – ПС корректировки планов проекта
Группа процессов мониторинга и управления	Регулярно оценивает прогресс проекта и осуществляет мониторинг, чтобы обнаружить отклонение от плана реализации проекта	– ПС мониторинга соответствия текущих операций проекта, плана управления проектом с базовым планом выполнения проекта; – ПС контроля влияния на факторы, которые нарушают общее управление, для внедрения только одобренных изменений; – ПС наблюдения и управления рисками; – ПС администрирования контрактов
Группа завершающих процессов	Формализует принятие продукта, услуги или результата и подводит проект или фазу проекта к правильному завершению	– ПС завершения всех операций проекта – ПС ведения архива – ПС документооборота и формирования отчетов – СУБД.

В настоящее время существует несколько сотен ПС для автоматизации управления программами и проектами, так или иначе, реализующих перечисленные задачи. Однако реально, на отечественном рынке стабильно присутствует не более 10. Принципиальных функциональных отличий между ПС начального уровня на самом деле не так много.

Виртуальному офису по управлению проектами ликвидации последствий ЧПС на МАД необходимо обрабатывать динамичную информацию на каждом этапе отдельного проекта. Проведенный анализ литературы [9] показал, что на сегодняшний день не уделяется достаточного внимания научному обоснованию выбора ПС для автоматизации виртуального офиса по управлению проектами ликвидации последствий ЧПС на МАД.

Задача является достаточно сложной, так как предполагается наличие широкого спектра программного обеспечения для решений задач офиса, которое должно соответствовать организационным уровням управления программами и проектами [9].

Учитывая, что каждое ПС характеризуется рядом функциональных, интеграционных и затратных показателей, необходимо выбрать ПС, которое отвечало бы заданным критериям и ограничениям.

Для разработки модели выбора ПС виртуального офиса управления проектами ликвидации последствий проектами ЧПС на МАД введем следующие обозначения: переменная $x_i = \{0;1\}$, которая принимает два значения: $x_i = 1$, если выбрано i-е ПС; 0 – в противном случае.

Каждый вид ПС характеризуется рядом показателей:

F_i – масштабируемость *i*-го вида ПС, т.е. эффективное обслуживание различного числа клиентов одновременно $(i = \overline{1,i'})$, где i' - количество возможных ПС для автоматизации процессов виртуального офиса управления проектами ликвидации последствий ЧПС на МАД;

H_i – время непрерывной длительной работы i-го вида ПС, $(i = \overline{1,i'})$;

N_i – надежность i-го вида ПС, т.е. устойчивость не только к ошибкам пользователей, но и к сбоям в системе коммуникаций, $(i = \overline{1,i'})$;

Z_i – стоимость i-го вида ПС, $(i = \overline{1,i'})$;

S_i – сложность использования i-го вида ПС, $(i = \overline{1,i'})$;

K_i – высокий уровень безопасности i-го вида ПС, т.е. защиты и отслеживания, протоколирования информации на всех этапах функционирования, $(i = \overline{1,i'})$.

Некоторые из приведенных показателей определяются качественно, поэтому следует оказать им количественные значения по какой-то шкале, например в интервале от 0 до 1.

Заданы частные критерии эффективности, а также ограничения. Необходимо определить ПС с учетом заданных критериев и ограничений. Математическая модель имеет следующий вид.

Частные критерии:

− максимальная масштабируемость:

$$F = \max \sum_{i=1}^{i'} F_i x_i ; \quad (1)$$

− максимальное время непрерывной длительной работы:

$$H = \max \sum_{i=1}^{i'} H_i x_i ; \quad (2)$$

− максимальная надежность:

$$N = \max \sum_{i=1}^{i'} N_i x_i ; \quad (3)$$

− минимальная стоимость:

$$Z = \min \sum_{i=1}^{i'} Z_i x_i ; \quad (5)$$

− минимальная сложность использования:

$$S = \min \sum_{i=1}^{5} S_i x_i ; \quad (6)$$

− максимальная защищенность:

$$K = \max \sum_{i=1}^{i'} K_i x_i ; \quad (7)$$

Ограничения:

− должно быть выбрано только одно ПС:

$$\sum_{i=1}^{i'} x_i = 1; \quad (8)$$

− время непрерывной длительной работы ПС должна быть не меньше заданного $H_{зад}$:

$$\sum_{i=1}^{i'} H_i x_i \geq H_{зад} ; \quad (9)$$

− надежность ПС должна быть не меньше заданной $N_{зад}$:

$$\sum_{i=1}^{i'} N_i x_i \geq N_{зад} ; \quad (10)$$

− стоимость покупки ПС должна быть не больше заданной $Z_{зад}$:

$$\sum_{i=1}^{i'} Z_i x_i \leq Z_{зад} . \quad (11)$$

Приведенная модель (1)–(11) принадлежит к задачам многокритериального линейного дискретного программирования с булевыми переменными.

Для примера остановимся наиболее популярных ПС. Лидерами в данной области являются программы Microsoft Project, Primavera Project Planner и Spider Project.

Задача выбора ПС решается по обобщенному аддитивному критерию. Коэффициенты весомости критериев определяются экспертами [10].

Результатом решения стало программное средство Primavera Project Planner с максимальным обобщенным аддитивным критерием 0,76.

Выводы

Таким образом, в работе разработана структурно-логическая модель организации виртуального офиса ликвидации последствий ЧПС на МАД, основное назначение которого − обеспечение эффективной коммуникации различных специалистов в совместном выполнении работ по ликвидации последствий ЧПС на МАД.

Разработана математическая модель выбора ПО виртуального офиса ликвидации последствий ЧПС на МАД, что позволило научно обосновать выбор ПО для автоматизации виртуального офиса по управлению проектами ликвидации последствий ЧПС на МАД.

1. Нефёдов Л.И. Модели и методы управления чрезвычайными природными ситуациями на магистральных автомобильных дорогах / Л.И. Нефёдов, Н.Ю. Филь, Ю.Л. Губин, Е.М. Мельниченко. – Харьков : ХНАДУ, 2011. – 136 с.
2. Концепція захисту населення і території у разі загрози та виникнення надзвичайних ситуацій. Затверджено Указом Президента України від 26 березня 1999 р. – № 284–99.
3. Про основні напрямки державної політики України в галузі охорони довкілля, використання природних ресурсів і забезпечення екологічної безпеки. Постанова Верховної Ради України від 5 березня 1998 р. № 188-98-ВР.
4. Мазур И.И. Управление проектами: справочное пособие / И.И. Мазур, В.Д. Шапиро и др. – М.: Высшая школа, 2007. – 875 с.
5. Фесенко Т.Г. Формирование офиса управления проектами городского благоустройства. / Т.Г. Фесенко, П. А. Тесленко // Вісник НТУ «ХПІ»., 2015.– № 1 (1110) – С.72-76.
6. Шевченко О.В. Організація віртуального офісу проекту як самостійної інфраструктури // Вісник соціально-економічних досліджень, 2013 р.– 4 (51) – С. 371-375
7. Дрешпак В.М. Досвід реалізації проектів з розвитку системи надання електронних адміністративних послуг у Дніпропетровській області / В.М. Дрешпак., Вискуб О.А. // Публічне адміністрування: теорія та практика: Електронне видання., 2014.– 1 (11)
8. Гультяев А. К. Microsoft Project 2002. Управление проектами. Русифицированная версия / А.К. Гультяев. – М.: Корона-Принт, 2003. – 592 с.
9. Нефедов Л.И. Методологические основы синтеза офисов по управлению программами и проектами // Л.И.Нефедов, Ю.А. Петренко, М.В. Шевченко, А.Б. Биньковская – Х.: ХНАДУ. 2012.– 296 с.
10. Петров Е.Г. Методи і засоби прийняття рішень у соціально-економічних системах / Е.Г. Петров, М.В. Новожилова, І.В. Гребенник. – К.: Техніка, 2004. – 256 с.