УДК 004.415

ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОМИСЛОВИХ КОНТРОЛЕРІВ НА БАЗІ ГРАФІВ СТАНІВ

В.В. Ткачов, С.М. Проценко, О.О. Бойко, І.О. Погрібняк
(Україна, Дніпро, Національний ТУ «Дніпровська політехніка»)

В роботі запропоновано формальний підхід до проектування та розробки програмного забезпечення системи керування, що складається з трьох етапів: складання словесного опису алгоритму функціонування технологічного процесу, проектування програмного забезпечення системи керування у вигляді графу станів та розробки програмного забезпечення. Крім того наведено розроблений формальний підхід для переходу від графів станів до програмного забезпечення на мові Ladder Diagram. Отриманий підхід дозволяє виконувати перехід від графів станів до їх програмної реалізації використовуючи лише типові структурні елементи та потребує тільки їх налаштування відповідно до умов та дій дуг переходів.

Ключові слова: система керування, технологічний процес, об'єкт керування, промисловий контролер, проектування програмного забезпечення, розробка програмного забезпечення, автомат Мілі, граф стану, Ladder Diagram

The article proposes a formal approach to the design and development of control system software, which consists of three stages: compiling a verbal description of the process flow algorithm, designing the control system software as a state graph and software development. In addition, a formal approach to the transition from state graphs to Ladder Diagram software is presented. The resulting approach allows the transition from state graphs to their program implementation using conventional structural elements, and requires only adjusting them according to the conditions and actions of the transition arcs.

Keywords: control system, technological process, control object, industrial controller, software design, software development, Mealy machine, state graph, Ladder Diagram

В работе предложено формальный подход к проектированию и разработке программного обеспечения системы управления, состоящий из трех этапов: составление словесного описания алгоритма функционирования технологического процесса, проектирование программного обеспечения системы управления в виде графа состояний и разработки программного обеспечения. Кроме того приведен разработанный формальный подход перехода от графов состояний к программному обеспечению на языке Ladder Diagram. Полученный подход позволяет выполнять переход от графов состояний к их программной реализации, используя обычные структурные элементы, и требует только их настройки в соответствии с условиями и действиями дуг переходов.

Ключевые слова: система управления, технологический процесс, объект управления, промышленный контроллер, проектирования программного обеспечения, разработка программного обеспечения, автомат Мили, граф состояний, Ladder Diagram

Постановка проблеми. У сучасній промисловості всюди використовуються системи автоматизованого керування технологічними процесами та об'єктами, що забезпечує підвищення якості продукції, зменшення витрат ресурсів, виконання вимог охорони праці в зонах підвищеної небезпеки, за рахунок прибирання людини або значного зменшення її участі в процесах отримання, перетворення, передачі та використання енергії, матеріалів, виробів, інформації. На сьогоднішній день для керування в таких системах використовуються електронно-обчислювальні пристрої, які називаються промисловими контролерами, до яких відносяться: програмовані логічні контролери, розподілені системи керування та промислові комп'ютери [2]. При цьому основною проблемою на сьогоднішній час є відсутність базових та надійних принципів розробки та налагодження програмного забезпечення систем керування. Виробники промислових контролерів пропонують курси та допоміжні матеріали, що описують реалізацію типових операцій всіх рівнів складності, але в цих рішеннях відсутній системний підхід, кожне з них є індивідуальним та потребує суттєвого доопрацювання. В процесі навчання інженери по автоматизації вивчають середовище розробки, особливості реалізації мов програмування та функціональні можливості бібліотек, але питання представлення алгоритмів керування та переходу від них до програмного забезпечення залишається не розглянутими. Таким чином, присутній розрив між чітким та детермінованим підходом до проектування і обранням апаратного забезпечення систем керування на базі промислових контролерів та відсутністю такого при проектуванні та розробці їх програмного забезпечення. З цього випливає, що завдання створення формального підходу до проектування та розробки програмного забезпечення промислових контролерів є актуальним.

Аналіз існуючих матеріалів показав, що на ранніх етапах розвитку автоматизації технологічних процесів перед системами окремо ставилися завдання дискретного та безперервного керування. Завдання дискретного керування вирішувалися з використанням теорії цифрових автоматів на базі релейно контактної логіки [1]. Розвиток напівпровідникової техніки дозволив перейти до реалізації цифрових автоматів за допомогою цифрової схемотехніки, що істотно підвищило надійність функціонування систем за рахунок видалення механічних контактів. При цьому процес модернізації існуючих систем торкався тільки апаратної частини, логічна структура залишалася незмінною. Початок використання в промисловості електронно-обчислювальних машин призвів до переходу від апаратної реалізації цифрових автоматів до програмної. Для спрощення цього процесу була розроблена мова програмування релейних діаграм, яка дозволяє за рахунок простого повторення релейно контактних схем отримувати графічне програмне забезпечення. Подальший розвиток електронно-обчислювальних машин призвів до появи програмованих логічних контролерів, а в подальшому промислових

контролерів, однак це не призвело до розвитку проектування та розробки програмного забезпечення систем керування на базі цифрових автоматів.

При автоматизації використовуються два типи цифрових автоматів: комбінаційні та кінцеві. Комбінаційні автомати є логічними ланцюгами, що мають кілька входів та кілька виходів, в яких значення вихідних сигналів в кожен момент часу однозначно визначаються комбінаціями вхідних сигналів в той же момент часу. При проектуванні комбінаційних автоматів розробляється таблиці істинності, на підставі яких синтезуються логічні рівняння, що використовуються для обчислення функцій керування. Кінцеві автомати є математичними моделями, що мають кілька входів та кілька виходів, в яких значення вихідних сигналів, в кожен момент часу визначаються комбінаціями вхідних сигналів та поточним станом. Проектування кінцевих автоматів виконується на підставі таблиць переходів або графів станів.

Основна маса завдань які розв'язуються сучасними системами автоматизації вимагає реалізації алгоритмів керування на основі кінцевих автоматів, у свою чергу комбінаційні автомати здебільшого використовуються для обчислення допоміжних значень або окремих керуючих сигналів. Існують кінцеві автомати двох типів Мілі та Мура. Значення вихідних сигналів автомата Мілі залежать від стану автомата та вхідних сигналів, а автомата Мура лише від його стану. З урахуванням циклічності функціонування програмного забезпечення промислових контролерів, зміна вихідних сигналів в автоматі Мілі виконується одноразово при переході з одного стану в інший, а в автоматі Мура циклічно для поточного стану. Таким чином, в автоматі Мура при описі стану враховуються зміни вихідних сигналів для всіх варіантів переходів в даний стан, що робить реалізацію алгоритму керування непрозорою та істотно ускладнює налагодження та модифікацію програмного забезпечення в порівнянні з автоматом Мілі. Виходячи з цього, при вирішенні завдань керування використання автомата Мілі є переважним.

При розробці програмного забезпечення систем керування доцільним є графічне представлення алгоритмів керування, так як їх проектування, аналіз та експлуатація зрозумілі консультантам (інженерам-технологам), які не є фахівцями в обчислювальній техніці [6]. Крім того розробка програмного забезпечення систем керування не передбачає мінімізацію автоматів, так як це призводить до порушення сприйняття логіки функціонування системи, за рахунок усунення станів та переходів обумовлених технологічним процесом [4]. Виходячи з цього проектування програмного забезпечення на базі кінцевих автоматів необхідно виконувати на базі графів станів.

Дослідження показали, що процес проектування програмного забезпечення системи керування можливо розділити на три етапи: розробка словесного опису алгоритму функціонування технологічного процесу на підставі опису інженера-технолога, виділення станів системи керування та розробка графа станів.

В процесі розробки словесного опису алгоритму виділяються режими функціонування системи керування: ручний, автоматичний, пуск, стоп, аварійний [5]. Для кожного з режимів описуються допустимі технологічні операції та можливості переходів між ними. При описі технологічних операцій вказується їх послідовність, умови включення та виключення виконавчих пристроїв та контрольовані параметри. До умов відносяться: виконувана

технологічна операція, стан блокувань, необхідний стан датчиків та виконавчих пристроїв, необхідна послідовність дій, для включення або виключення пристрою. До контрольованих параметрів належать величини, які використовуються тільки для аналізу функціонування технологічного процесу або об'єкта керування. За результатами розробки алгоритму складаються таблиці вхідних, вихідних та контрольованих параметрів.

На підставі словесного алгоритму функціонування технологічного процесу або об'єкта керування виділяються стани, в яких може перебувати система керування. Спочатку визначаються загальні стани, що відповідають режимам роботи системи керування. Далі для кожного режиму виділяються стани які забезпечують виконання необхідної послідовності технологічних операцій. Стани зазвичай іменуються як SX, де S – скорочення від State (стан), а X – номер стану. Нумерація станів починається з нуля. Важливим є те, що стани та їх описи відносяться до системи керування, а не до технологічного процесу.

Відповідно до алгоритму функціонування технологічного процесу, виділеним станам та таблицям вхідних і вихідних параметрів розробляється граф станів. Спочатку на графі розміщують початковий стан та стани які відповідають режимам роботи системи керування, таким чином, що б між ними було якомога більше вільного простору. Стани позначаються колами, в яких зазначаються їх назви (рис. 1). Пари станів з'єднуються між собою дугами дій які виконуються при переході в новий стан, напрямок переходу між станами задається стрілкою, що розміщується на одному з кінців дуги. Над кожною дугою вказується умова переходу та виконувані дії, що розділяються між собою символом "/". Дії, що виконуються при знаходженні в стані, позначаються дугами дій сталого стану, що виходять та повертаються в цей же стан. Зазвичай такі дуги використовуються для паралельної перевірки значень параметрів та змінних, визначення моменту завершення роботи таймерів або зміни значення лічильників. Таке рішення значно спрощує структуру графа, за рахунок зменшення кількості станів та переходів між ними.

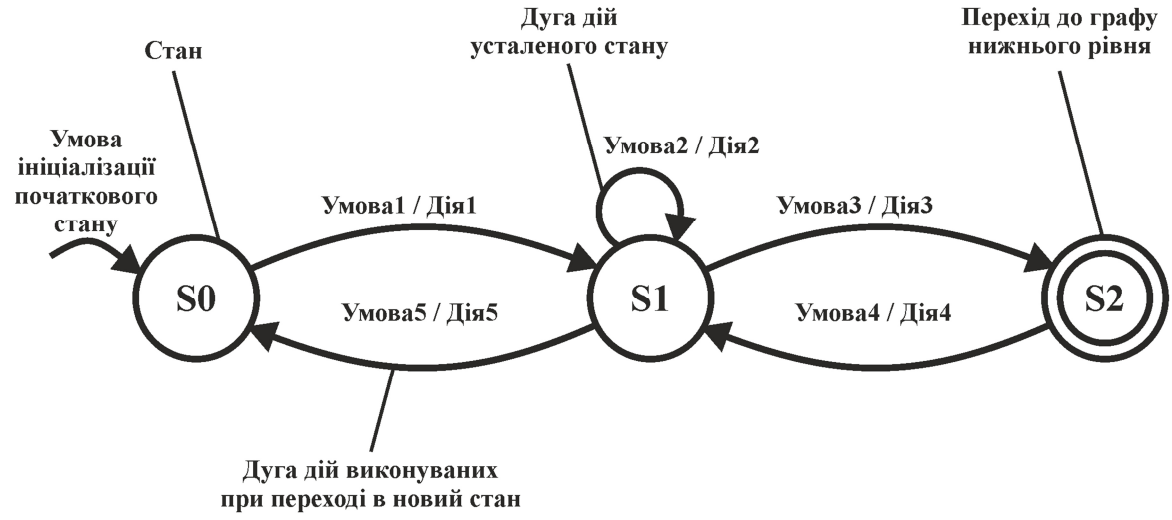

Рис. 1. Приклад графа станів

При проектуванні графів станів для розробки програмного забезпечення під умовами та діями мається на увазі більш широкі поняття, ніж використовувані в теорії цифрових автоматів. Це пов’язано з тим, що програмна реалізація не обмежена виконанням логічних операцій. Таким чином, умови включають перевірку бітових значень, обчислення логічних та математичних виразів, операції порівняння, інші типи операцій, що призводять до отримання логічного результату. У свою чергу дії включають в себе зміни бітових значень, запуск/зупинку таймерів, збільшення та зменшення значень лічильників, обчислення керуючих впливів, зміну налаштувань регуляторів, інші типи операцій.

Після розміщення станів режимів роботи системи керування, додаються стани необхідні для реалізації послідовностей технологічних операцій. З метою спрощення графа станів його незалежні гілки можуть бути винесені в окремі графи. При переході між такими графами функціонування графа верхнього рівня припиняється, до моменту повернення з графа нижнього рівня. Перехід позначається двома колами, в яких вказується назва графа або початковий стан при наявності наскрізної нумерації. Для позначення початкового стану використовується дуга з вільним вихідним кінцем.

Головною вимогою до проектування графів станів є максимально близьке відображення алгоритму функціонування системи керування, так як по ним розробляється програмне забезпечення яке є їх повною програмної копією.

На підставі запропонованого процесу проектування розроблено формальний підхід до у сього процесу розробки програмного забезпечення системи керування, який складається з трьох етапів:

– написання словесного опису алгоритму функціонування технологічного процесу;

– проектування програмного забезпечення, що включає складання таблиць вхідних, вихідних та контрольованих параметрів, виділення станів системи керування та розробку графів станів на основі кінцевого автомата Мілі;

– розробки програмного забезпечення на мові стандарту IEC 61131-3 Ladder Diagram.

Використання мови програмування Ladder Diagram обумовлено тим, що вона забезпечує легкість сприйняття логіки функціонування систем керування, простоту розробки та налагодження програмного забезпечення, швидкий пошук порушення функціонування датчиків та виконавчих пристроїв, що забезпечуються за рахунок її наочності та інтуїтивної зрозумілості для інженерів усіх електричних спеціальностей [3].

Розробка програмного забезпечення на базі графа станів який описується автоматом Мілі в першу чергу вимагає реалізації дуг дій виконуваних при переході в новий стан. Дані дуги описуються умовами переходу в нові стани та діями, виконуваними при переході. Дослідження показали, що найбільш повна реалізація дуги дій виконуваних при переході в новий стан передбачає три етапи: перевірку умови переходу, виконання дій відповідних переходу та завершення переходу в новий стан. З огляду на особливості розробки на мові програмування Ladder Diagram кожен етап зручно представити у вигляді

окремого ланцюга. У першому ланцюгу перевіряється знаходження системи керування в поточному стані та умова переходу в новий стан. Другий ланцюг на підставі поточного стану та нового стану виконує дії відповідні дузі переходу. Третій ланцюг завершує перехід в новий стан. Як видно з описаних функцій ланцюгів для їх реалізації необхідно зберігати поточний та новий стан системи керування.

Стан системи керування можна зберігати у вигляді цілого числа або бітових значень. Реалізація перевірок, установок та скидань бітових значень на мові програмування Ladder Diagram вимагає меншої кількості блоків в порівнянні з використанням цілих значень, крім того використовуючи бітові значення одночасно можна оперувати декількома станами системи керування (поточним та попереднім). Виходячи з цього, для зберігання станів системи керування доцільно використовувати при малій кількості станів окремі бітові змінні, а при великій масив бітових значень, логічного типу даних BOOL.

На підставі обраних рішень розроблена структура першого ланцюга, яка реалізує дугу дій виконуваних при переході в новий стан, в ланцюзі перевіряється знаходження системи керування в поточному стані та умова переходу в новий стан. При виконанні всіх умов встановлюється новий стан, в іншому випадку значення нового стану залишається незмінним (рис. 2). Під умовою переходу тут мається на увазі значення логічного результату виразу умови.

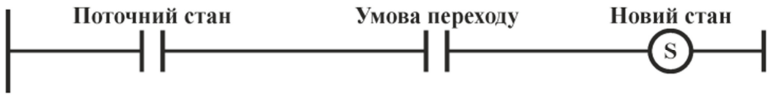

Рис. 2. Ланцюг установки нового стану

Другий ланцюг реалізує дії, що виконуються при зміні стану системи керування, в ньому перевіряється значення поточного та нового станів. При виконанні цієї умови вчиняються дії які відповідають переходу в новий стан (рис. 3).

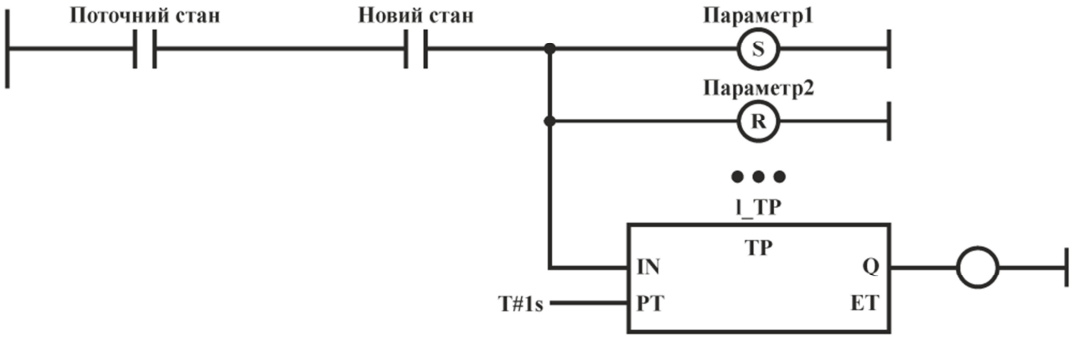

Рис. 3. Ланцюг дій

Третій ланцюг реалізує завершення переходу в новий стан, в ньому перевіряється значення поточного та нового станів. При виконанні цієї умови поточний стан скидається, а відпрацювання дуги дій виконуваних при переході в новий стан завершується (рис. 4).

Рис. 4. Ланцюг скидання поточного стану

При розробці програмного забезпечення яке реалізує граф станів на мові програмування Ladder Diagram основною вимогою є послідовне розміщення ланцюгів. Таким чином, усі три ланцюги, які реалізують одну дугу дій виконуваних при переході в новий стан повинні йти одна за одною зверху вниз. Таке жорстке обмеження пов’язане з тим, що на проміжку виконання програми між ланцюгом установки нового стану та ланцюгом скидання поточного стану активними є два стани. Дана вимога не накладає ніяких обмежень на реалізацію графа станів.

Відповідно до загальної структури реалізації графа станів на мові Ladder Diagram при задоволенні умов декількох дуг дій виконуваних при переході в новий стан, перехід буде виконаний по самій верхній. Таким чином, пріоритет дуги дій можна підвищити за рахунок переміщення її ланцюгів вгору або вниз.

Виходячи з опису функціонування промислових контролерів після перепрограмування ними виконується холодний перезапуску при цьому усі динамічні змінні стають рівними нулю або початковому значенню. Реалізація універсального завдання початкового стану системи керування вимагає створення окремого ланцюга на початку програмної реалізації графу стану. Даний ланцюг виконує перевірку значень всіх станів системи керування і в разі відсутності поточного стану встановлює початковий стан (рис. 5).

Рис. 5. Ланцюг встановлення початкового стану

Реалізація дуги дій сталого стану вимагає додавання окремо ланцюга, в якому виконується перевірка поточного стану та умов дій. Таких ланцюгів може бути декілька, при цьому дії для кожної дуги будуть виконуватися незалежно один від одного (рис. 6).

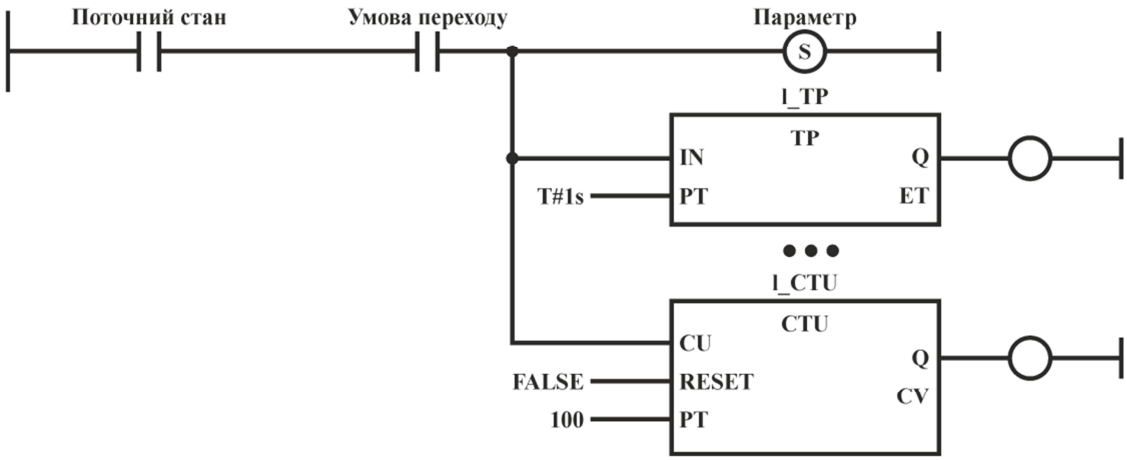

Рис. 6. Ланцюг дій сталого стану

Використовуючи отримані структури ланцюгів розроблено формалізацію переходу від графа станів до програмного забезпечення на мові програмування Ladder Diagram. В рамках даного підходу стан системи керування зберігаються в двійковому масиві, кожній дузі дій виконуваних при переході в новий стан відповідає ланцюг установки нового стану, ланцюг дій та ланцюг скидання поточного стану, кожній дузі дій сталого стану відповідає свій ланцюг який перевіряє знаходження у даному стані та умову переходу, для встановлення початкового стану системи керування використовується відповідний ланцюг який розміщується на початку програмної реалізації графу стану.

Висновки. В роботі проаналізовано сучасний стан питання проектування та розробки програмного забезпечення систем керування на базі промислових контролерів. На підставі чого встановлено, що на даний час відсутній системний підхід до цього питання, у технічній документації та літературі, а так само в навчальних курсах які надаються компаніями виробниками апаратного та програмного забезпечення промислових контролерів розглядаються тільки загальні питання пов’язані з використанням їх середовищ розробки та стандартних бібліотек. Виходячи з цього встановлено актуальність створення формального підходу до розробки програмного забезпечення промислових контролерів.

Аналіз питання розробки систем керування показав, що поява промислових контролерів розмила кордон між системами керування дискретними та безперервними об’єктами, що вимагає використання комплексного підходу до проектування та розробки програмного забезпечення таких систем. Основним способом вирішення даного типу питань є використання алгоритмів керування на базі кінцевих автоматів Мілі у вигляді графів стані.

Запропоновано формальний підхід до проектування та розробки програмного забезпечення системи керування, що складається з трьох етапів: складання словесного опису алгоритму функціонування технологічного процесу, проектування програмного забезпечення системи керування у вигляді графу станів та розробки програмного забезпечення. На підставі проведених

досліджень розроблено формальний підхід для переходу від графів станів до програмного забезпечення на мові Ladder Diagram. Даний підхід дозволяє виконувати перехід від графів станів до їх програмної реалізації використовуючи лише типові структурні елементи та потребує тільки їх налаштування відповідно до умов та дій дуг переходів.

Подальший розвиток дослідження передбачає дослідження складних питань проектування та розробки програмного забезпечення промислових контролерів на базі графів станів з метою їх формалізації. До таких питань відносяться:

– початкова ініціалізація системи керування після теплого перезапуску;

– реалізація ієрархічної структури програмного забезпечення;

– проектування та розробка програмного забезпечення при розпаралелюванні задач;

– проектування та розробка програмного забезпечення з урахуванням високопріоритетних завдань.

ПЕРЕЛІК ПОСИЛАНЬ:

1. Динесенко В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием / В.В. Динесенко. – М.: Горячая линия-Телеком, 2009. – 608 с.

2. Парр Э. Програмируемые контролеры: руководство для инженера / Э. Парр. – М.: БИНОМ. Лаборатория знаний, 2007. – 516 с.

3. Петров И.В. Программируемые контроллеры. Стандартные языки и инструменты / И.В. Петров. – М.: СОЛОН-Пресс, 2003. – 256 с.

4. Пушкарь М.С. Проектування систем автоматизації: навч. посібник / М.С. Пушкарь, С.М. Проценко. –Д.: Національний гірничий університет,2013.–268 с.

5. Ткачев В.В., Формальные методы разработки программного обеспечения для систем дискретного управления / В.В. Ткачев, С.Н. Проценко, Н.В. Козарь // Гірнича електро-механіка та автоматика: науково технічний збірник. – Дніпропетровськ, 2009. – С. 115-123.

6. Федоров Ю.Н. Справочник инженера по АСУТП: проектирование и разработка. Комплект в двух томах. Том 2 / Ю.Н. Федоров. – М.: Инфра-Инженерия, 2016. – 484 с.

УДК 004.93

РОЗРОБКА МОДУЛЮ ІНФОРМАЦІЙНОЇ СИСТЕМИ ІДЕНТИФІКАЦІЇ ОСОБИСТОСТІ

В.В. Гнатушенко, Д.О. Літвінов, Г.Ю. Станчиць
(Україна, Дніпро, Національна металургійна академія України)

Постановка проблеми. Розробка та теоретичне обґрунтування методів та алгоритмів, призначених для розпізнавання цифрових зображень у печатних документах, які отримані при скануванні, при ідентифікації особи.