УДК 004.02

Є.В. НІКІТЕНКО[*] , Є.В. РИНДИЧ[*]

ІНСТРУМЕНТАЛЬНИЙ ЗАСІБ ВІДДАЛЕНОГО СПОСТЕРЕЖЕННЯ ЗА ПОКАЗНИКАМИ ДАТЧИКІВ

[*] Чернігівський національний технологічний університет, м. Чернігів, Україна

Анотація. *Розроблено архітектуру інструментального засобу, який дозволяє стежити за станом показників датчиків, збирати статистичну інформацію з використанням сучасних мобільних і бездротових технологій та рішень. Засіб складається з програмного й апаратного забезпечення, яке у сукупності взаємодіє між собою за принципом клієнт-сервер, використовуючи різні протоколи обміну інформацією. Створено випробувальний зразок, який доводить коректність роботи інструментального засобу.*
Ключові слова: *USART, Android, C++, ThingSpeak, веб-сервер.*

Аннотация. *Разработана архитектура автоматизированного средства, которая позволяет следить за состоянием показаний датчиков, собирать статистическую информацию с использованием современных мобильных и беспроводных технологий и решений. Средство состоит из программного и аппаратного обеспечения, которое в совокупности взаимодействует между собой по принципу клиент-сервер, используя различные протоколы обмена информацией. Создан испытательный образец, доказывающий корректность работы инструментального средства.*
Ключевые слова*: USART, Android, C++, ThingSpeak, веб-сервер.*

Abstract. *It was developed the architecture of the automated tool that allows monitoring the status of the sensors indicators and gathering statistic information using modern wireless and mobile solutions and techniques. The tool consists of hardware and software which interact together with each other on the principle of the client-server model using a variety of communication protocols (HTTP, USART etc.). The test sample proving the correctness of the instrumental tool has been created.*
Keywords: *USART, Android, C++, ThingSpeak, web-server.*

1. Постановка проблеми

Комунікаційні мережі міцно увійшли в життя сучасного суспільства. Їх широке поширення дозволяє використовувати доступні канали зв'язку не тільки для спілкування та обміну повідомленнями, але і для управління різноманітними пристроями, реалізуючи таким чином концепцію моніторингу або діагностики приладів. Існує безліч технологій, технічних рішень, транспортних протоколів, що дозволяють проектувати на своїй основі системи, які здійснюють обмін керуючою інформацією і орієнтовані на бездротове з'єднання. Більшість таких рішень завжди стикається з проблемами енергоефективності, мобільності, відкритості та вартості. Як правило, задовольнити всі чотири вимоги – досить складне завдання. У цій статті розглядається можливе рішення перерахованих вище проблем для подібних систем. Метою статті є розробка прототипу пристрою, що дозволяє віддалено стежити за показниками датчиків, використовуючи відкриті сучасні технології, комунікації та бездротові мережі.

2. Аналіз досліджень і публікацій

Велике поширення як комунікації між пристроями отримали Wi-Fi-мережі [1]. Ефективним засобом комунікації між декількома пристроями, використовуючи всього лише одну лінію зв'язку, є протокол 1-Wire. Для взаємодії між двома пристроями дуже інтенсивно використовуються різні послідовні інтерфейси, такі як RS-232 і UART [2, 3]. У складних системах зазвичай не є можливим застосування якоїсь однієї технології. Найкращим компромісом дуже часто виявляється поєднання різних протоколів. Так, для мереж датчиків фір-

ми Dallas зручно використовувати 1-Wire [4], в той же час для інтенсивного обміну між двома пристроями краще використовувати, наприклад, UART або USB. Дуже часто також використовуються Wi-Fi-мережі датчиків. Як і в будь-якій іншій мережі, тут розглядаються такі показники:

1) надійність − операції з передачі інформації повинні здійснюватися з деякою заздалегідь заданою ймовірністю успіху;

2) енергоефективність − часта заміна батарей живлення в мобільних пристроях − одна з найважливіших проблем у даній області;

3) масштабованість − дуже важливо мати можливість легко розширювати і доповнювати систему новими властивостями та компонентами. Особливо це актуально для великих систем;

4) адаптивність − бажано, щоб пристрій або систему, що розроблюються, можна було застосувати в різних апаратних і програмних конфігураціях;

5) швидкодія − більша швидкодія забезпечує кращу інтерактивність і взаємодію з користувачем, дозволяє включити більше число функцій [5].

Дані твердження справедливі для більшості пристроїв.

3. Основні елементи систем «Інтернету речей»

Будь-яка система − це набір деяких елементів і зв'язків між ними. Для системи, яка реалізується, такими елементами можуть бути, наприклад:

1. Датчики (температури, вологості). Не можна говорити про те, що система без датчиків є повноцінною. Вона у такому випадку не може взаємодіяти з навколишнім середовищем і реагувати на її зміни.

2. Керуючі пристрої (наприклад, мікроконтролери різних виробників). Система завжди працює за закладеними у ній програмами, програмами, які виконуються на якомунебудь контролері, і, можливо, під управлінням певної операційної системи (найчастіше − це ОС, базовані на ядрі Linux).

3. Кінцеві пристрої (реле, транзистори і т.п.). Система створюється заради якоїсь мети. Вона повинна сформувати керуючі сигнали для виконання роботи. Причому, це не обов'язково може бути управління моторами, верстатами. Як результат роботи також можуть бути і сформовані бази даних.

4. Пристрої вводу/виводу (клавіатура, екран). Повністю автоматичних систем поки що не існує. Будь-яка система передбачає деяку автоматизацію. Проте на даний момент все ще немає можливості читати думки людини і працювати без втручання користувача або фахівця в певні моменти часу. Тому взаємодія з людиною повинна бути організована якимось чином за допомогою систем вводу/виводу.

4. Архітектура системи

Будь-яка система обробки і передачі даних повинна забезпечувати масштабованість, високу швидкість роботи і надійність. Система, що розробляється, складається з пристрою зняття показників датчиків (включаючи передавальний пристрій, що має вихід у мережу Інтернет по бездротовій Wi-Fi-мережі), сервера ThingSpeak і кінцевих клієнтських пристроїв (рис. 1). Пристрій зняття показників датчиків складається з трьох компонентів: мережі датчиків, мікроконтролера і передавального пристрою. Мережа датчиків являє собою набір пристроїв, що дозволяють знімати показники характеристик навколишнього середовища і взаємодіяти з центральним пристроєм по шині 1-Wire. Центральний пристрій − це мікроконтролер Atmega8L, який має низьку ціну, мале енергоспоживання і достатню продуктивність для поставлених завдань. Виконуючи одну повноцінну інструкцію за один такт, ATmega8L досягає продуктивності 1 MIPS/МГц, дозволяючи досягти оптимального

співвідношення продуктивності та споживаної енергії [6]. Для передачі даних по мережі використовується модуль ESP8266, який є відносно дешевим і підтримує весь потрібний спектр можливостей для передачі і прийому даних, взаємодії з сервером. Як сервер використовується ThingSpeak – відкритий сервер, вихідні тексти якого доступні на GitHub. ThingSpeak може обробляти GET- і POST-запити. Результат запиту сервер може повертати у форматі XML, JSON або ж простим текстом з відповіддю. Як клієнтський пристрій передбачається використання будь-якого пристрою з встановленою версією ОС Android не нижче 2.2 або будь-якого іншого пристрою, що має вихід у мережу Інтернет, дозволяє виконувати і відображати результати HTTP-запитів. Підтримка платформи Android API 10 забезпечує сумісність з 99% всіх доступних пристроїв на платформі Android в цілому.

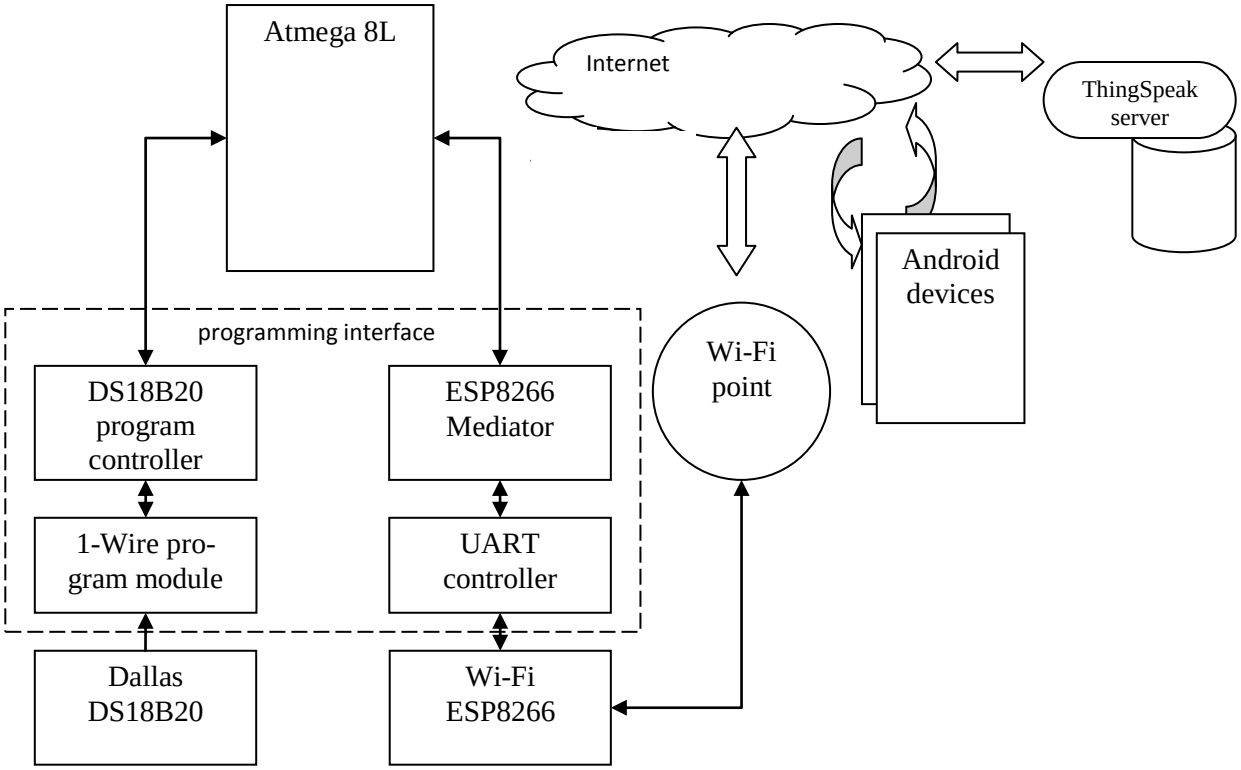

Рис. 1. Архітектура системи

5. ThingSpeak-сервер

Вихідні тексти сервера доступні на GitHub (https://github.com/iobridge/thingspeak). За допомогою даного сервісу і модуля ESP8266 можна створити програмно-апаратний комплекс, який дозволяє реалізувати віддалену взаємодію та управління тими чи іншими системами, навіть якщо джерело управління і виконавчі пристрої знаходяться на протилежних точках земної кулі.

Можна використовувати вже готовий сервіс, який доступний за адресою http://thingspeak.com, або ж його можна встановити локально на свій сервер. ThingSpeak надає зручний і простий інтерфейс REST API, спеціально розроблений для мініатюрних пристроїв.

Таким чином, використовуючи даний сервіс, можна забезпечити компонентам системи взаємодію між собою на будь-якій відстані, використовуючи вже готові рішення. Проект документований, відповідна документація доступна на офіційному сайті (https://thingspeak.com/docs).

6. Зв’язки між компонентами

Датчики – пристрої, взаємодіяти з якими відносно нескладно і дані з них можуть збирати прості 8-бітні контролери. Була написана програма по збору даних із датчиків температури по інтерфейсу 1-Wire та пересилання їх на сервер за допомогою Wi-Fi-модуля ESP8266 на 8-бітному контролері Atmega8 з 8 Кб пам'яті програм. Як результат – написана програма займає 64% доступної пам'яті (або всього лише 5.12 Кб). І це все з урахуванням того, що реалізація інтерфейсу 1-Wire на контролері була програмною. Крім того, модулі ESP8266 є самодостатніми і можуть працювати без зовнішніх контролерів. Існують також готові прошивки, які вже можуть працювати з датчиками DHT11/22, BMP085/180, BH1750, DS18B20, AM2321 і деякими іншими і відправляти дані на сервер.

Програмне забезпечення написано таким чином, щоб у випадку розриву з'єднання модуль ESP переходив у режим Wi-Fi-точки доступу, дозволяючи підключитися до нього і задати нову конфігурацію.

Блок-схему алгоритму представлено на рис. 2.

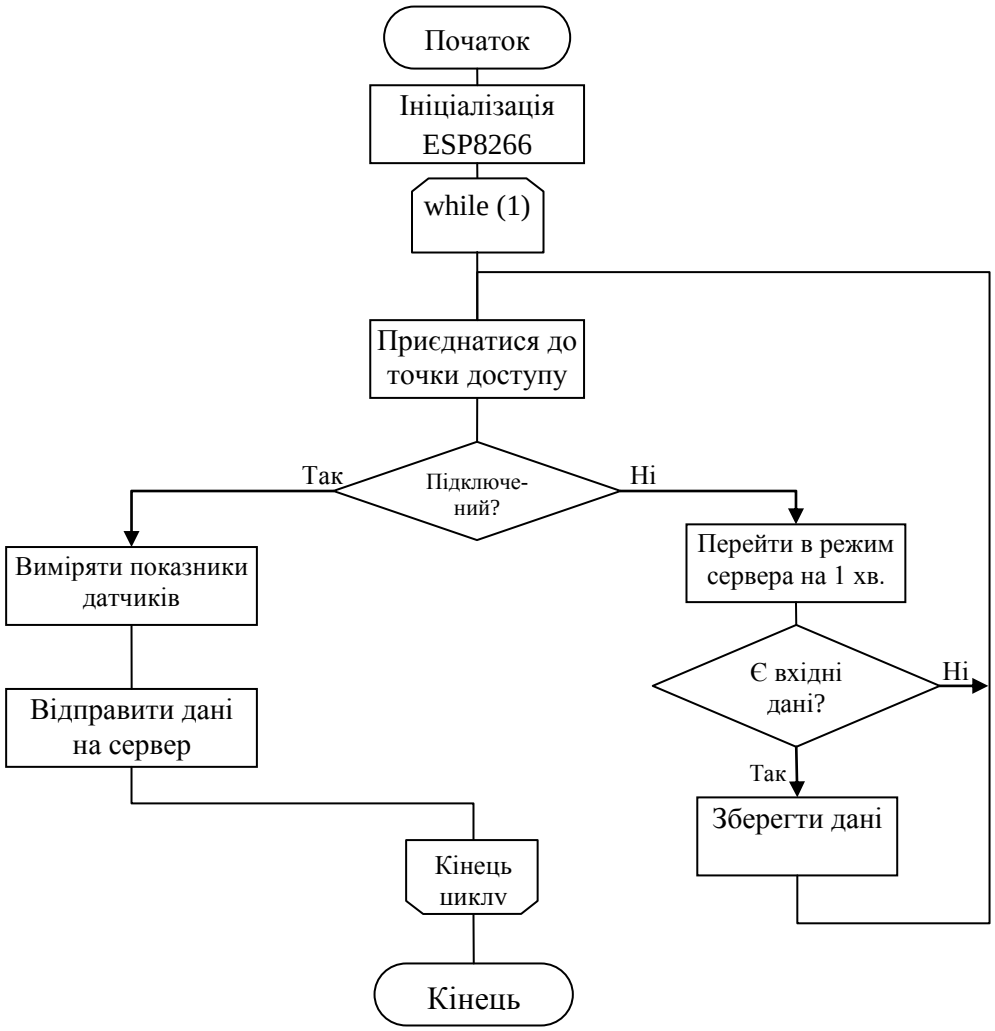

Рис. 2. Алгоритм однієї ітерації циклу програми

У мережі Інтернет можна зустріти інформацію про високе енергоспоживання Wi-Fi-модуля. Однак, оскільки система, що розробляється, не є системою реального часу, то постійна активність не потрібна. ESP8266 споживає менше 0.5 мА в режимі підтримки зв'язку з точкою доступу Wi-Fi і менше 60 мкА в режимі глибокого сну з працюючим годинником реального часу.

Взаємодія з Wi-Fi-модулем відбувається по UART. Для зручності та прискорення роботи з ним у пам'яті контролера був організований кільцевий буфер прийнятих даних.

```
char rxUartBuffer[UART_BUFFER_SIZE];
volatile unsigned char bufferPointer = 0;
```

Таким чином, відсилка управляючих команд зводиться до таких функцій:

```
const char AT[] = "AT\0";
const char AT_PREFIX[] = "AT+\0";
void sendCommand(const char *cmd, char *params){
      _delay_ms(10);
      if (strcmp(cmd, AT) == 0) sendStr(AT);
      else{
        sendStr(AT_PREFIX);
        sendStr(cmd);
        if (params != 0){
          sendStr("=\0");
          sendStr(params);
        }
      }
}
uchar push(char *patternOk){
  resetBuffer();
  sendStr("\r\n\0");
  uchar times = DEF_TIMES;
  while (times != 0){
    _delay_ms(DEF_PAUSE);
    if (patternOk != 0 && strstr(rxPuartBuffer, patternOk)){
      _delay_ms(50);
      return OK;
    }
    times--;
  }
  return FAIL;
}
```

Тоді функції управління модулем спрощуються. Наприклад:

```
uchar espIsConnected(){
  sendCommand("CWJAP?\0", 0);
  return push("OK");
}
uchar espGetMode(){
  sendCommand("CWMODE?\0", 0);
  return push(":1") == OK ? MODE_STATION : MODE_AP;
}
...
```

Запит на збереження даних у базі даних ThingSpeak буде мати такий вигляд:

```
uchar sendRequest(const char *ip, int port, const char *data, uchar
bPush){
  char cmd[] = "CIPSTART\0"; char pars[50];
  sprintf(pars, "\"TCP\",\"%s\",%d\0", ip, port);
  sendCommand(cmd, pars);
  if (push("Linked") == FAIL){
    sendCommand(cmd, pars);
    if (push("CONNECT") == FAIL) return FAIL;
```

```
  }
  sprintf(pars, "%d\0", strlen(data)-1);
  sendCommand("CIPSEND\0", pars);
  if (push(">") == FAIL) return FAIL;
  sendStr(data);
  if (bPush == PUSH) return push("OK");
  else return OK;
}
```

Приклад запиту/відповіді:

```
> AT+CWJAP="<AP Name>","<Password>"
OK
> AT+CWJAP?
+CWJAP:"<AP Name>"
OK
> AT+CIPSTART="TCP","<IP address>",<PORT>
OK
Linked
> AT+CIPSEND=<QUERY LENGTH>
> GET /update?api_key=<API_KEY>&field1=<VALUE>
SEND OK
+IPD,1:1
OK
Unlink
```

Мікроконтролер Atmega8L зі зниженим енергоспоживанням може працювати з максимальною частотою, рівною 8 МГц. Звернення до виконуючого пристрою по шині 1-Wire складається з операцій скидання, відповіді-присутності, ідентифікації, команди і читання/запису даних. Кожна операція I_i складається з N_i бітів, кожен біт передається за один фіксований часовий слот, рівний $T = 60$ мкс. Таким чином, загальний час виконання будь-якої операції обчислюється за такою формулою:

$$t = T\sum_{i=1}^{N} N_i = T \cdot (N_{СК} + N_{ВІД} + N_{ІД} + N_{КОМ} + N_{Д}), \qquad (1)$$

де $N_{СК} = 8$ − кількість слотів команди скидання, $N_{ВІД} = 1$ − кількість слотів команди присутності, $N_{ІД} = 64$ − кількість слотів команди ідентифікації пристрою, $N_{КОМ} = 16$ − кількість слотів опису команди (читання даних), $N_Д = 60$ − кількість слотів передачі даних (для значення температури DS18B20).

Таким чином, загальний час дорівнює

$$t = (8 + 1 + 64 + 16 + 10) \cdot 60\text{мкс} = 99 \cdot 60 = 5{,}9 \text{ мс} \approx 6 \text{ мс}.$$

Перевірка CRC-коду не відіграє суттєвої ролі, так як виконується за частки мікросекунд. Дані від Wi-Fi-модуля по UART обробляються апаратно і зберігаються в кільцевому буфері по перериванню, тому затримки в ньому теж несуттєві. Деяку невизначеність може вносити передача даних у мережу Інтернет, але її необов'язково проводити після кожного вимірювання. Кращим способом тут буде накопичення певної кількості виміряних значень від датчиків у буфері пам'яті і відправка даного буфера за один раз. Це знімає навантаження з мережі і ліній UART між контролером і Wi-Fi-модулем і забезпечить ще більшу швидкодію. Крім того, такий метод ще й знизить енергоспоживання Wi-Fi-модуля.

7. Клієнтське програмне забезпечення

У разі віддаленого управління найбільш прийнятним варіантом буде використання протоколу HTTP. Керуючі команди – це GET- і POST-запити до сервера. ThingSpeak вже забезпечує можливість прийому команд та їх обробки. Однак бажано написати відповідне програмне забезпечення (ПЗ). Враховуючи стан ринку IT в даний час, можна зробити висновок, що практична більшість користувачів використовує мобільні пристрої з наперед встановленою ОС Android. Тому цілком логічним виглядає рішення написати клієнтське програмне забезпечення саме для цієї ОС. До складу Android SDK входить популярна бібліотека Apache, яка значно спрощує розробку додатків, що працюють з мережею. Таким чином, на пристрої, які завжди знаходяться поруч з користувачем, можна виводити всю необхідну інформацію про стан системи. Користувач може в будь-який час перевірити її і прийняти необхідні рішення.

Більше того, безумовною перевагою прийнятого підходу є те, що одне і те ж ПЗ можна використовувати як для локального, так і для віддаленого управління. Для локального управління потрібно всього лише під'єднатись до Wi-Fi-точки доступу, в якій працює пристрій.

Можливості подальшого розвитку та поліпшення проекту:

– можливість додавання шифрування, використовуючи протокол HTTPS;

– у майбутньому можна буде відмовитися від бібліотеки Apache в Android-клієнті на користь нових засобів роботи з мережею, що надаються Android 6.0 Marshmallow;

– виключити використання контролерів Atmega і перенести всі їх функції в модулі ESP8266;

– для великих проектів або для застосування в умовах виробництва можна використовувати, наприклад, Raspberry Pi, який дозволяє підняти локальний ThingSpeak сервер, забезпечуючи, таким чином, повну автономність і працездатність системи навіть при відключенні від мережі Інтернет. При наступному підключенні всю інформацію можна буде легко синхронізувати;

– забезпечити інтеграцію ThingSpeak з відкритими хмарними платформами, наприклад, OpenShift (якщо потрібно проводити складні обчислення) і Dropbox (якщо потрібно зберігати великі обсяги даних).

– додати зручний Web-інтерфейс.

8. Висновки

Як було показано в даній статті, на сучасному етапі розвитку обчислювальної техніки і загального наукового прогресу цілком реально створювати мініатюрні, швидкодіючі, мобільні, енергоефективні пристрої та системи з мінімальною вартістю. Обсяг бінарного коду ПЗ для управління датчиками, передачі даних у мережу Інтернет та прийняття керуючих команд від віддаленого користувача склав менше 70% всієї пам'яті програм мікроконтролера Atmega8L. Було використано 8 ніжок мікроконтролера з 20. Це означає, що спроектовану систему можна вдосконалювати і додавати до неї додатковий функціонал. Також було розроблено клієнтський додаток для ОС Android. Запуск програми можливий на 99% всіх можливих Android-пристроїв. Розроблений пристрій був перевірений на практиці і показав стабільну роботу. Таким чином, можна зробити висновок, що розроблена система може використовуватися як самостійний пристрій, а також як складова частини іншої, складнішої і більшої системи.

СПИСОК ДЖЕРЕЛ

1. The best choice for enterprise IoT networking is Wi-Fi [Електронний ресурс]. – Режим доступу: http://searchmobilecomputing.techtarget.com/tip/The-best-choice-for-enterprise-IoT-networking-is-Wi-Fi.
2. 1-Wire ® (Protocol) Dallas Semiconductor/Maxim [Електронний ресурс]. – Режим доступу: coecsl.ece.illinois.edu/ge423/sensorprojects/1-wire_full.doc.
3. Serial Communication [Електронний ресурс]. – Режим доступу: https://learn.sparkfun.com/tutorials/serial-communication.
4. An Introduction to 1-Wire Technology [Електронний ресурс]. – Режим доступу: hivetool.org/w/images/4/40/Intro_to_1-Wire.doc.
5. Design Principles of Wireless Sensor Networks Protocols for Control Applications [Електронний ресурс]. – Режим доступу: https://people.kth.se/~kallej/papers/wsn_design_springer11.pdf.
6. Datasheets (ESP8266, Atmega8, DS18B20 etc.).

Стаття надійшла до редакції 30.10.2017