УДК 004.04+ 004.4242

БОЛДАК А.О.,
ЄФРЕМОВ К.В.

ПРЕДМЕТНО-ОРІЄНТОВАНА МОВА АНАЛІТИЧНОЇ ОБРОБКИ ДАНИХ

В роботі розглядається предметно-орієнтована мова аналітичної обробки даних, використання якої для створення сценаріїв обробки даних не потребує від експертів знання інструментальних мов програмування, що суттєво знижує трудомісткість процесу розробки аналітичних звітів. Запропонована мова впроваджується в Світовому центрі даних з геоінформатики та сталого розвитку НТУУ «КПІ».

Work contains review of declarative domain-specific language for data mining, using of which doesn't demand knowledge of tool programming languages for development of data processing scenarios and that really reduces labor inputs in analytical reports development. Proposed language is implemented at ICSU World Data Center for Geoinformatics and Sustainable Development hosted by NTUU "KPI".

Вступ

Засоби аналітичної обробки даних [1], в основі якої лежать різноманітні методи статистичної обробки та аналізу даних, є потужним інструментом, що сьогодні входить до складу систем підтримки прийняття рішень. Нажаль, не дивлячись на те, що для більшості інструментальних мов програмування розроблено спеціалізовані бібліотеки для аналітичної обробки даних [2-4], ефективність використання такої обробки в межах універсальних засобів підтримки прийняття рішень залишається низькою. Одною з причин такого становища є те, що, з одного боку, розробка нових та(або) модифікація існуючих сценаріїв обробки даних здійснюється засобами інструментальної мови з використанням імперативної та об'єктно-орієнтованої парадигми програмування. З іншого боку, експерти, які повинні визначати сутність аналітичної обробки даних, а, відповідно, і розробляти сценарії, не мають належного досвіду програмування та вимушені звертатися за допомогою до програмістів, що, в свою чергу, збільшує трудомісткість створення та модифікації сценаріїв.

З огляду на зазначене вище, мета роботи полягає у зниженні трудомісткості створення та модифікації сценаріїв аналітичної обробки даних за рахунок розробки, реалізації та використання предметно-орієнтованої мови, яка не потребує від експерта спеціальних знань та навичок програмування та достатня для опису процесу перетворення даних.

Процес розробки нової предметно-орієнтованої мови складається з етапу створення абстрактного синтаксису, на якому визначається множина семантичних одиниць, етапу розробки синтаксису, на якому формально визначається подання семантичних одиниць синтаксичними, графічними або змішаними засобами, та етапу розробки правил трансформації абстрактного подання в таке, що виконується інструментальним оточенням.

Визначення семантичних одиниць

Сценарій аналітичної обробки даних є моделлю обчислювального процесу та може розглядатися як поглинаючий ланцюг Маркова з дискретними станами та автоматним часом [5]. Вважається, що в такій моделі перехід із одного стану до іншого здійснюється миттєво. Тому моменти часу, в які здійснюється перехід, пов'язують з подіями, а проміжки часу між подіями – з активністю (процесом). У відповідності до цього, сценарій може бути описаний як послідовністю подій, так і послідовністю процесів. Слід зауважити, що в моделі, орієнтованій на опис подій, припускається наявність формального апарату опису процесів, а в моделі, орієнтованій на опис процесів, навпаки, – певна формалізація подій. Використання обмеженого алфавіту подій з чітко визначеною в межах протоколу взаємодії процесів семантикою забезпечує підходу до опису сценаріїв аналітичної обробки даних, орієнтованих на процеси, переваги, пов'язані з тим, що модель подається як сукупність взаємодіючих процесів та більш адекватно (ніж сукупність подій) відображає структуру перетворень даних, які необхідно здійснити в сценарії.

Таким чином, сценарій аналітичної обробки даних є сукупність взаємодіючих процесів, кожен з яких може розглядатися як «чорна скринька»

$$P = \left\langle X, Y, X \xrightarrow{F} Y \right\rangle,$$

де X – множина вихідних даних, Y – множина результатів, $X \xrightarrow{F} Y$ – відображення, вихідних даних в результати, яке здійснює процес.

Процеси можуть використовувати дані та генерувати результати декількох типів, тому множини вхідних та вихідних даних можна подати як:

$$X = X_1 \times X_2 \times ... \times X_n$$
$$Y = Y_1 \times Y_2 \times ... \times Y_m \quad ,$$

де $X_i, i = \overline{1,n}$, $Y_j, j = \overline{1,m}$ визначають різні за семантикою та(або) типом вихідні дані і результати відповідно. Елементи вихідних даних $X_{k,i}, i = \overline{1,n}$ для процесу P_k можуть бути отримані від інших процесів P_l, які генерують $Y_{l,j}, j = \overline{1,m}$, при цьому повинна виконуватись умова суміщення типів даних

$$Y_{l,j} \subseteq X_{k,i}. \qquad (1)$$

Активація процесу P_k можлива лише у випадку, коли всі вхідні дані для процесу P_k готові, тобто отримані від процесів-попередників P_l, та при цьому не порушується умова суміщення типів даних (1). Отже, для організації взаємодії процесів достатньо однієї події готовності даних, яка ініціюється процесами після генерації вихідних даних.

Зрозуміло, що з моменту генерації результатів до моменту появи відповідної події готовності даних та активації наступного процесу існує певний проміжок часу, протягом якого проміжні дані повинні зберігатися. Слід також зауважити, що особливістю статистичної обробки даних є використання агрегатних перетворень (наприклад обчислення середнього значення, стандартного відхилення тощо), коли результат залежить від серії значень, що входять до статистичної вибірки. З огляду на це, процеси з'єднуються за допомогою черг (потоків) даних, до яких поступають результати та з яких вибираються вихідні дані у вигляді пакетів – елементів даних, з якими пов'язано тип даних. Відносно процесів черги даних можна розділити на вхідні, з яких процеси вибирають пакети вихідних даних, та вихідні, в які процеси поміщають пакети результатів обробки. Оскільки операції вибірки та поміщення в чергу ідентифікуються типом черги відносно процесів, черги можуть подаватися у неявній формі, за допомогою поняття вхідних та вихідних портів, які асоціюються з процесами. Таким чином, процеси з'єднуються в мережу (сукупність взаємодіючих процесів) за допомогою портів, а кожна пара «вихідний порт – вхідний порт» асоціюється з чергою (потоком) пакетів.

У відповідності до наявності вхідних і вихідних портів процеси можуть відноситись до трьох категорій: процеси-джерела даних, які не мають вхідних портів та призначені для вибірки даних з різних джерел способом, що виходить за рамки описуваної моделі; процеси-звіти, які не мають вихідних портів та асоціюються з виходами процесу аналітичної обробки даних; процеси-перетворення, які мають як вхідні, так і вихідні порти та призначені для перетворення даних. Оскільки процес не має іншого способу отримання даних, як вибірка пакетів з вхідних портів, виникає потреба у визначенні особливого типу процесів-джерел з постійним потоком пакетів – процесів-констант, які слугують для налаштування інших процесів.

Пакети, які є елементами потоків даних, що підлягають обробці, відносяться до певних типів даних, які є кількісними та (або) якісними оцінками властивостей деякої сукупності об'єктів $O = \{o_i\}, i = \overline{1,n}$, де o_i – ідентифікуюче значення номінальної шкали для $i-$ого об'єкту. Такі дані є відображенням виду:

$$O \xrightarrow{I_j} X^j, j = \overline{1,m}, \qquad (2)$$

де $I_j, j = \overline{1,m}$ – відображення, визначені на множині об'єктів ($D(I_j) = O$), з областями значень, що відповідають областям визначення показників X^j, тобто $E(I_j) = D(X^j)$.

Дані виду (2) можна подати в вигляді таблиці «об'єкт-властивість» [6]:

$$X = \left(x_{i,j}\right)_{i=1,j=1}^{n,m}, \qquad (3)$$

в якій рядок X_i відповідає набору значень, що характеризують властивості об'єкту o_i, а стовпчик X^j задає значення j-ого показника для всієї сукупності об'єктів. Така таблиця є однією з форм опису відношення, заданого в просторі показників

$$X \subseteq O \times E(X^1) \times ... \times E(X^m).$$

Даними для процесів можуть бути як таблиці виду (3), так і їхні перетини типу X_i та X^j, а також окремі значення $x_{i,j}$.

Таким чином, дані, що інкапсулюються в пакеті, можуть належати як до простих типів (значень), так і до структурних типів (масивів та таблиць).

Слід також зауважити, що дані, які містяться у вхідному пакеті, можуть бути структурно перетворені (бути декомпозовані, або, навпаки, об'єднані в структури) та пов'язані з іншими пакетами результуючого потоку. Також особливістю визначеного процесу обробки даних є неможливість втручання в нього, поки вхідні потоки даних не буде вичерпано. Тому, якщо необхідно виконувати аналіз результатів, то він здійснюється на окремій стадії обробки, на яку додаткову інформацію може бути передано завдяки використанню метаданих, що пов'язані з даними.

На рис.1 мовою UML [7] подано загальну схему відношень між семантичними одиницями предметно-орієнтованої мови аналітичної обробки даних, яка може бути використана для розробки відповідних інструментальних засобів.

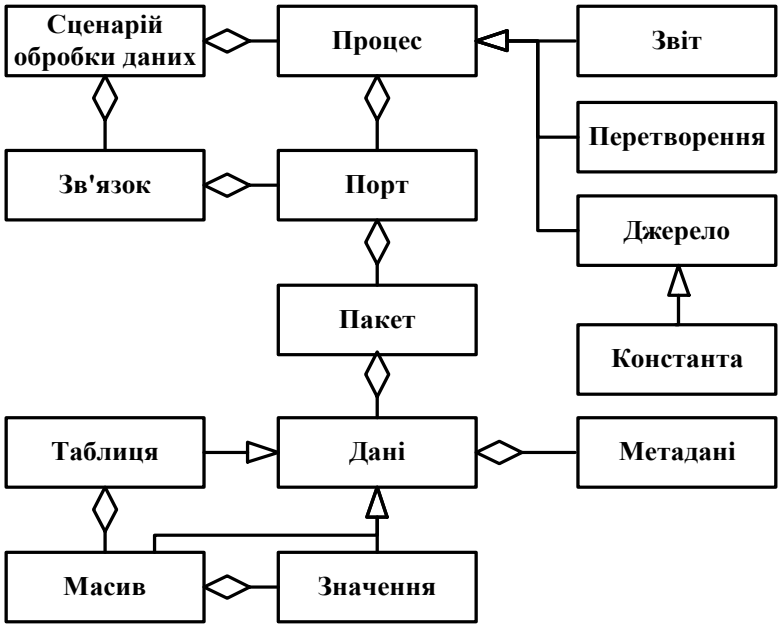

Рис.1. Схема відношень між семантичними одиницями предметно-орієнтованої мови аналітичної обробки даних.

Визначення синтаксису

Сценарій аналітичної обробки даних можна подати як граф, в якому множина вершин відповідає множені процесів, а дуги задають зв'язки між вихідними та вхідними портами. Незалежно від того, якими синтаксичними, графічними або змішаними засобами буде подано модель процесу аналітичної обробки даних, треба вирішити задачу ідентифікації класів процесів, екземплярів процесів, портів, тощо.

Кожен з процесів, які входять до складу сценарію, є екземпляром деякого класу, який визначає сукупність вхідних та вихідних портів, сутність обробки даних, що здійснюється процесом цього класу, та реалізацію класу процесу інструментальними засобами. Ідентифікація цих сутностей здійснюється у відповідності до наведених нижче правил, заданих за допомогою EBNF [8]:

```
process ::= process_name ':' (   process_type|
'{'stream'}'
process_name ::= ID
process_type ::= ID
ID ::=('a..z' | 'A..Z' | '_')('a..z' | 'A..Z'
 | '0..9' | '_')*
```

Синтаксичне правило «process» визначає операцію породження екземпляра «process_name» класу процесу «process_type». Альтернативне правило задає специфікацію процесу-константи, для якого необхідно задати

вихідний потік «stream», який асоціюється зі стандартним вихідним портом OUT.

```
stream ::= packet (';' packet)*
packet ::= metadata (',' data)*
metadata ::= 'META{' statement (','
statement)* '}'
data ::= 'DATA{' item (',' item)* '}'
statement ::= ID '=' value
item ::= value
value ::= (INT | FLOAT | STRING | structure)
STRING ::= ' " '  ~ ' \" '  ' " '
structure ::= '{' value (',' value)* '}'
```

Як бачимо з наведених правил, потік даних є послідовністю пакетів, кожен з яких може містити як метадані, так і самі дані. Метадані, в свою чергу, є мапою зі строковими ключами, а дані можуть бути як простими, так і структурованими.

Специфікація з'єднань використовує імена вхідних і вихідних портів, які належать процесам, тому повна специфікація порту включає ім'я процесу.

```
connection ::= input_port'>'output_port
input_port ::= process_name '.'port_name
output_port ::= process_name '.'port_name
port_name ::= ID
```

Остаточно, сценарій описано як мережу, задану множиною примірників процесів і зв'язків між портами.

```
network::= network_name':NETWORK{'g_stat  (';'
g_stat)* '}'
g_stat ::= process | connection
```

Нижче наведено приклад опису процесу стандартизації даних, отриманих в результаті SQL-запиту до бази даних. При цьому над даними виду (3) здійснюється перетворення [9]:

$$C_{norm}(x_{i,j}) = \frac{x_{i,j} - a}{b}, \qquad (4)$$

де $x_{i,j}$ – значення з таблиці (3), a – параметр, який задає зсув; b – параметр, який визначає масштаб нормування. Параметри a і b обчислюються згідно з формулами:

$$a = \overline{X^j} = \frac{1}{n}\sum_{i=1}^{n} x_{i,j}, \qquad (5)$$

$$b = \sigma(X^j) = \sqrt{\frac{\sum_{i=1}^{n}(x_{i,j} - \overline{X^j})^2}{n}} \qquad (6)$$

де $\overline{X^j}$ – середнє значення, а $\sigma(X^j)$ – стандартне відхилення показника X^j.

Лістинг 1. Декларативний опис процесу стандартизації даних.

```
STANDARTIZATION:NETWORK{P1:
{META{url="DBCONNECTION",sql="SQLQUERY"}};
P2:SQL; P3:AVG; P4:STDDEV; P5:AT;P6:TABLE;
P1.OUT>P2.PREFS;
P2.VAR>P3.SERIE;
P2.VAR>P4.SERIE;P2.VAR>P5.IN;
P3.VALUE>P5.SHIFT;P4.VALUE>P5.SCALE;
P5.OUT>P6.VAR;}
```

У наведеному прикладі опису сценарію обробки даних використано наступні процеси: P1 – процес-константа, який генерує вихідний потік, що містить один пакет з метаданими, які описують строку доступу до бази даних (url) та SQL-запит (sql); P2 – процес-джерело, який здійснює SQL-запит та налаштовується через порт PREFS, а його порт VAR містить потік пакетів з перетинами типу X^j, які відповідають стовпцям таблиці (3); P3 та P4 – процеси-перетворення, які обчислюють значення за формулами (5) і (6) відповідно; P5 –процес-афінне перетворення, яке здійснюється за формулою (4); P6 – процес, який з потоку змінних, що надходять через вхідний порт VAR, формує таблицю даних виду (3).

Зрозуміло, що наведений вище опис носить декларативний характер і може бути поданий за допомогою мови XML[10]:

Лістинг 2. XML-опис процесу стандартизації даних.

```
<?xml version="1.0" encoding="utf-8"?>
<network name="STANDARTIZATION">
   <process name="P1">
   <stream>
       <packet>
          <meta key="url">
             DB CONNECTION
          </meta>
          <meta key="sql">
             SQL QUERY
          </meta>
       </packet>
 </stream>
 </process>
 <process name="P2" class="SQL"/>
 <process name="P3" class="AVG"/>
 <process name="P4" class="STDDEV"/>
 <process name="P5" class="AT"/>
 <process name="P6" class="TABLE"/>
 <connection from="P1.OUT" to="P2.PREFS"/>
 <connection from="P2.VAR" to="P3.SERIE"/>
 <connection from="P2.VAR" to="P4.SERIE"/>
 <connection from="P2.VAR" to="P5.IN"/>
 <connection from="P3.VALUE" to="P5.SHIFT"/>
 <connection from="P4.VALUE" to="P5.SCALE"/>
 <connection from="P5.OUT" to="P6.VAR"/>
</network>
```

Найбільш цікавою з точки зору експертів є графічна форма подання сценарію аналітичної обробки даних, яка інструментальними засобами може бути автоматично перетворена до XML-опису. Зображена на рис.2 схема сценарію для процесу стандартизації даних демонструє такі переваги графічної форми подання, як її наочність, простоту та точність. Так на цій схемі кожному процесу відповідає умовне графічне позначення (УГП), яке ідентифіковане ім'ям та класом процесу, вхідні та вихідні порти зображено їх ідентифікаторами, що розташовані в межах УГП процесу з лівого та правого боку відповідно. Зв'язки зображено лініями зв'язку між портами у відповідності до загальних правил побудови структурних та функціональних схем.

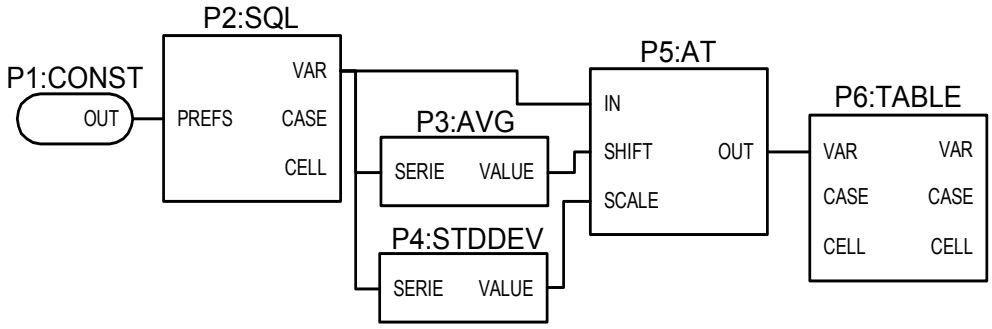

Рис.2. Схема процесу стандартизації даних.

Саме графічна форма подання є найбільш прийнятною до використання експертами з аналітичної обробки даних.

Опис процесу аналітичної обробки даних засобами імперативної інструментальної мови програмування передбачає наявність реалізації класів процесів та класу мережі, який містить всю інформацію про склад моделі та відіграє роль фасаду, в якому передбачені операції для створення процесів та зв'язків. У цьому випадку опис сценарію є лінійним алгоритмом додавання до моделі складових опису, як це показано в Лістингу 3.

Лістинг 3. Імперативна форма опису процесу стандартизації даних.

```
Network n = newNetwork( ”STANDARTIZATION”);
n.add(n.constant(
”P1”,”META{url=\”DBCONNECTION\”,sql=\”SQL
QUERY\”}”));
n.add( n.process(”P2”,SQL.class));
n.add( n.process(”P3”,AVG.class));
n.add( n.process(”P4”,STDDEV.class));
n.add( n.process(”P5”,AT.class));
n.add( n.process(”P6”,TABLE.class));
n.add( n.connection(”P1.OUT”, ”P2.PREFS”));
n.add( n.connection(”P2.VAR”, ”P3.SERIE”));
n.add( n.connection(”P2.VAR”, ”P4.SERIE”));
n.add( n.connection(”P2.VAR”, ”P5.IN”));
n.add( n.connection(”P3.VALUE”, ”P5.SHIFT”));
n.add( n.connection(”P4.VALUE”, ”P5.SCALE”));
n.add( n.connection(”P5.OUT”, ”P6.VAR”));
```

Реалізація засобів автоматизації процесу аналітичної обробки даних

Серед розглянутих засобів опису процесу аналітичної обробки даних найбільш наближеною до потреб експертів є подання у вигляді графічної схеми. З метою забезпечення користувачів інструментальними засобами на основі бібліотеки MXGraph [11] розроблено спеціалізований графічний редактор з набором класів процесів, що може розширюватись. Цей редактор дозволяє створювати схеми процесів та зберігати їх описи у декларативному виді (див. Лістинг 1) та у виді XML (див. Лістинг 2).

В якості засобів моделювання процесу аналітичної обробки даних використано каркас, розроблений на основі бібліотеки FBP [12], яка надає сукупність java-класів для реалізації парадигми програмування, орієнтованого на процеси [13].

Перехід від декларативного опису до імперативної форми (Лістинг 3) здійснюється з використанням синтаксичного аналізатора та семантичного процесора, розробленого засобами ANTLR [14]. Аналогічний перехід від XML-опису реалізовано за допомогою семантичного процесора на основі SAX-парсера [15].

Предметно-орієнтована мова аналітичної обробки даних та інструментальні засоби її реалізації впроваджуються у Світовому центрі даних з геоінформатики та сталого розвитку [16] і використовуються для розрахунку моделей оці-

нювання сталого розвитку в глобальному та регіональному контексті, розрахунку моделі оцінювання загроз для регіонів України, тощо [17].

Висновки

Розроблено проблемно-орієнтовану графічну мову аналітичної обробки даних, використання якої не потребує спеціальних знань та навичок програмування в межах імперативної та об'єктно-орієнтованої парадигми, що дає можливість експертам з аналітичної обробки даних розробляти нові та модифікувати існуючі сценарії безпосередньо, без залучення програмістів. Досвід впровадження розроблених засобів автоматизації аналітичної обробки даних у Світовому центрі даних з геоінформатики та сталого розвитку показав, що можливість подання процесу аналітичної обробки даних у графічному вигляді, автоматична трансформація до виду, що може виконуватися в інструментальному середовищі, наявність інтерактивних засобів налагодження процесу аналітичної обробки даних дозволяють знизити трудомісткість розробки процедур попередньої обробки даних,моделей оцінювання, процедур формування аналітичних звітів з використанням результатів обробки даних великого обсягу.

Список літератури

1. Анализ данных и процессов / А.А.Барсегян, М.С.Куприянов, И.И.Холод и др. 3-е изд. перераб. и доп. СПб.: БХВ-Петербург.– 2009.– 512 с.
2. M.F.Hornick, E.Marcadé, S.VenkayalaJavaDataMining: Strategy, Standard, and Practice.–CA, US:Morgan Kaufmann.– 2007.– 520 p.
3. IMSL Numerical Libraries [Електронний ресурс] – Режим доступу: http://www.roguewave.com/products/imsl-numerical-libraries.aspx
4. OracleDataMining[Електронний ресурс] – Режим доступу: http://www.oracle.com/ru/products/database/options/advanced-analytics/index.html
5. Кельберт М. Я., Сухов Ю. М. Вероятность и статистика в примерах и задачах. Т. II: Марковские цепи как отправная точка теории случайных процессов и их приложения.– М.: МЦНМО.– 2009. – 295 с.
6. Айвазян С.А. и др. Прикладная статистика: Исследование зависимостей: Справ, изд. / С. А. Айвазян, И. С. Енюков, Л. Д. Мешалкин; Под ред. С. А. Айвазяна. – М.: Финансы и статистика, 1985. – 487 с.
7. Буч Г., Якобсон А., Рамбо Дж. UML. Классика CS. 2-е изд. / Пер. с англ.; Под общей редакцией проф. С. Орлова – СПб.: Питер.– 2006. – 736 с.
8. Information technology. Syntactic metalanguage. Extended BNF. [Електронний ресурс] ISO/IEC 14977:1996. – Режим доступу: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26153
9. Згуровский М.З., Болдак А.А. Системное согласование даннях разной природы в мультидисциплинарных исследованиях // Кибернетика и систем. анализ. – 2011., N 4. – С. 51–64.
10. Extensible Markup Language (XML) 1.0 (Fifth Edition)[Електронний ресурс] – Режим доступу: http://www.w3.org/TR/REC-xml/.
11. The most popular Java Graph Visualization Library[Електронний ресурс] – Режим доступу: http://www.jgraph.com/jgraph.html.
12. Java Implementation of FBP Concepts (JavaFBP) [Електронний ресурс] – Режим доступу: http://www.jpaulmorrison.com/fbp/#JavaFBP.
13. J.P.Morrison Flow-based programming : a new approach to application development.–Unionville, Ont.,CA : J.P. Morrison Enterprises.– 2010.– 336p.
14. P.Terence Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages (1st ed.), Raleigh, N.C., US: Pragmatic Bookshelf. –2010.–374p.
15. D.Brownell SAX2.– Sebastopol, CA : O'Reilly.– 2002.– 228p.
16. Світовий центр даних з геоінформатики та сталого розвитку [Електронний ресурс] – Режим доступу: http://wdc.org.ua/
17. Аналіз сталого розвитку – глобальний та регіональний контексти: моногр. /Міжнар. рада з науки (ICSU) [та ін.]; наук. кер. М.З.Згуровський. – К.:НТУУ «КПІ», 2010.– Ч. 2. Україна в індикаторах сталого розвитку.–220 с.