стандартизации управлений качеством / Б. В. Осипов, Е. А. Мировская. — М. : Узд-во стандартов, 1990. 6. Оценка качества сайта [Электронный ресурс]. — Режим доступа : http://www.myucoz.ru/publ/9-1-0-31/. 7. Электронные издания: Основные виды и выходные сведения: ГОСТ 7.83-2001. / Межгосударственный Совет по стандартизации, метрологии и сертификации. — Минск, 2001. 8. CyberAnalytic [Электронный ресурс]. — Режим доступа : http://cys.ru/.

АВТОМАТИЗАЦИЯ УПРАВЛЕНИЯ КАЧЕСТВОМ ЭЛЕКТРОННОГО ИЗДАНИЯ

Рассматривается качество сайта, которое зависит от многих составляющих и от качества технического задания на разработку сайта. Подается оценка качества сайта, которого можно осуществить лишь на основе тех показателей, которые можно проверить по определенным критериям.

AUTOMATION OF QUALITY ELECTRONIC PUBLICATION

The quality of the site depends on many components and the quality specification for a development site. Assessment of site quality can be achieved only on the basis of those indicators that can be checked according to defined criteria.

Стаття надійшла 19.11.2010

УДК 004

О. В. Овсяк

Львівська філія Київського національного університету культури і мистецтв,
Українська академія друкарства

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ОПЕРАЦІЇ ЕЛІМІНУВАННЯ

Описуються математичні моделі вибору, обчислення розмірів і формування xml - опису операції елімінування.

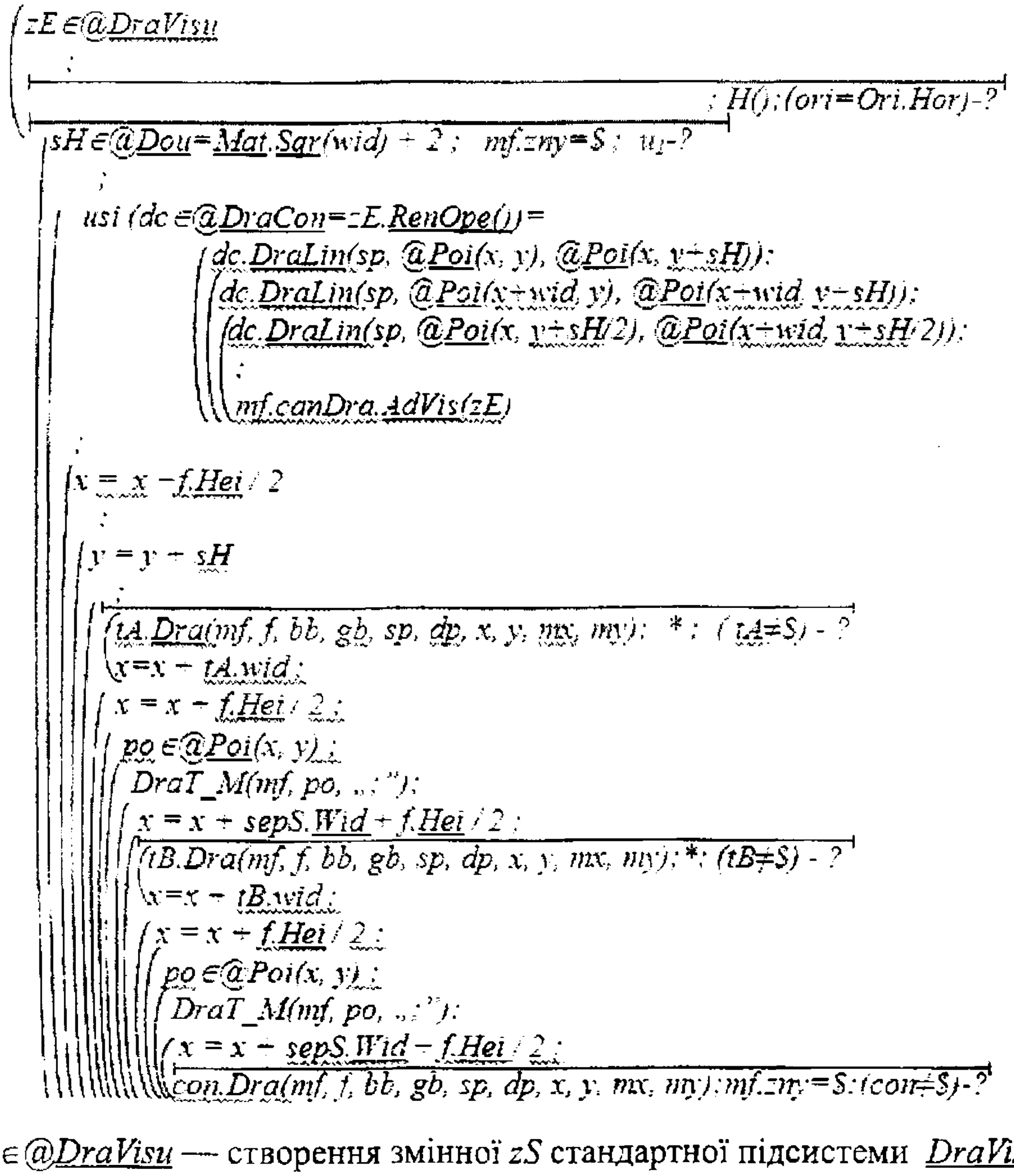
Алгебра алгоритмів, модель, елімінування, технологія

Елімінування є операцією алгебри алгоритмів [4], яка виконується над трьома унітермами і має специфічне графічне позначення, утворене трьома лініями. Розміри знака операції елімінування залежать від довжини трьох унітермів і розділювачів між ними. Використовування універсальних комп'ютерних систем, наприклад, Word та інших, для автоматизованого створення операції елімінування є малоефективним, тому що потребує постійного масштабування, а як наслідок, великих затрат робочого і комп'ютерного часу. Отож було створено спеціалізовані редактори «Модал» [1] і «Абстрактал» [2]. У цих системах використовуються створені спеціальні фіксовані формати даних, у яких зберігаються абсолютні координати графічних знаків операцій алгебри алгоритмів. Зміни фрагментів формул, які необхідно здійснювати у

процесі редагування формул алгоритмів, потребують створення надзвичайно складних алгоритмів, якими описуються ці процеси. У зв'язку з цим, доцільним є створення *xml* - опису операції елімінування, яким суттєво спрощуються алгоритми його створення й аналізу, що досягається використанням вже наявних для виконання цих процесів стандартних процедур.

Оскільки в операції елімінування такі складові частини моделі, як рисування рамки виділення, ідентифікації розділювача унітермів та обчислення його геометричних розмірів аналогічні моделям операції секвентування, то їх опис опускаємо. Лише опишемо загальну модель, яка утворена горизонтальною і вертикальною орієнтацією та модель створення *xml* – опису операції елімінування.

Загальна модель описується такою формулою:



де *zE*∈@*DraVisu* — створення змінної *zS* стандартної підсистеми *DraVisu*, яка реалізується відомим класом `DrawingVisual()` [3, 5], *(ori=Ori.Hor)-?* — перевірка наявності значення змінної *ori*, яке відповідає горизонтальній орієнтації операції секвентування *Ori.Hor*, u_1-? — умовний унітерм, який описується формулою *(mf.zny≠"Del")&(mf.zny≠"dWF")&(mf.zny≠"dWS")&(mf.*

zny≠"dWC") | (mf.zny=$) та описує відсутність режиму видалення елімінування або наявність чотирьох режимів знищення, *sH∈@Dou=Mat.Sqr(wid) + 2* — обчислення висоти *sH*, значення якого належить стандартній підсистемі *Dou* (він реалізується відомим класом `Double` [3, 5]), знака як корінь квадратний *Mat.Sqr()* від довжини *wid* плюс два пікселі, *mf.zny=$* — обнулення ознаки видалення або заміни унітермів,

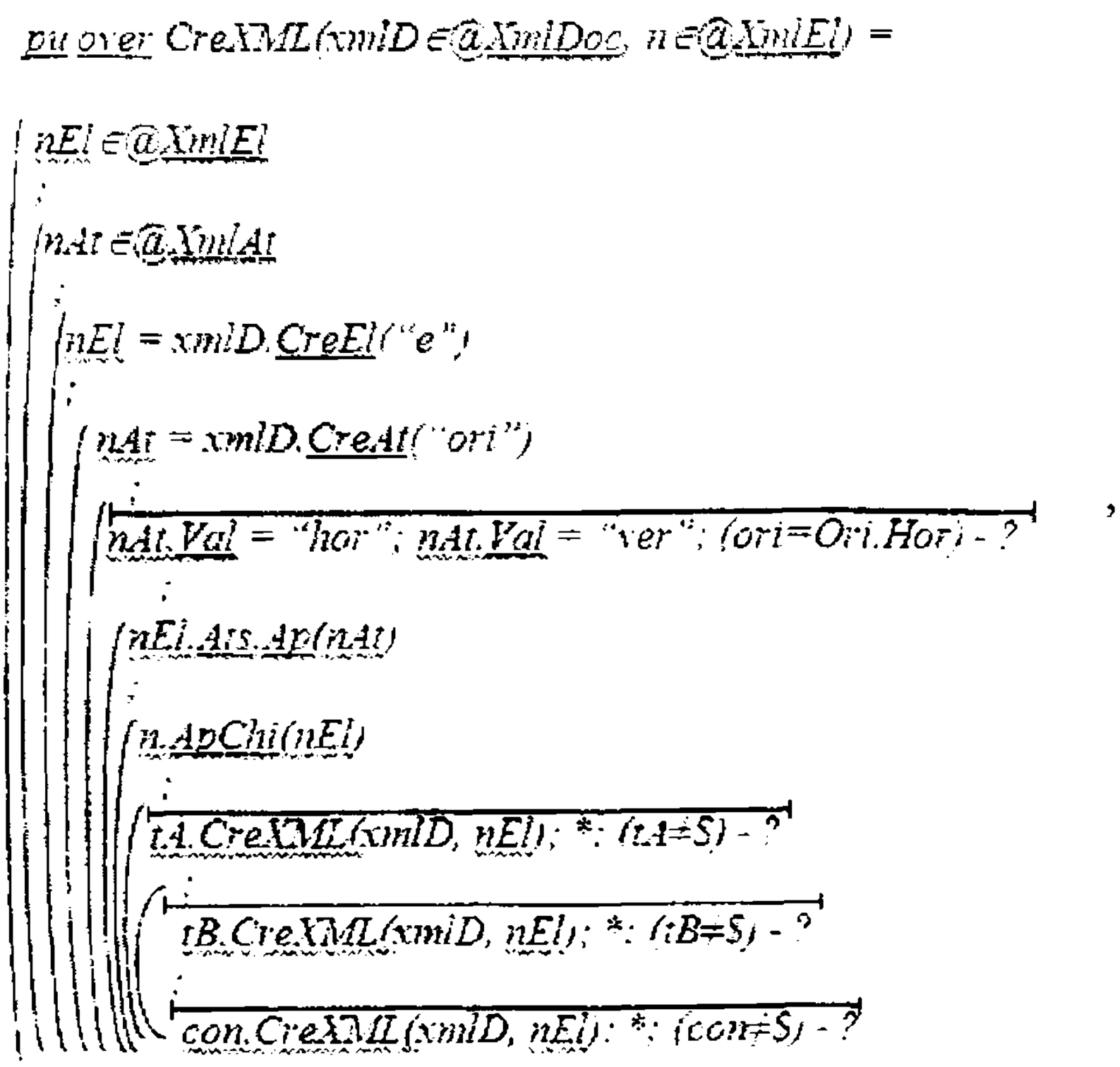
```
H() =
sW∈@Dou=Mat.Sqr(wid) + 2 ;  mf.zny=$ ;  u1-?
  ;
usi(g∈@DraCon=zE.RenOpe()) =
        g.DraLin(sp, @Poi(x, y), @Poi(x, y+sH));
        g.DraLin(sp, @Poi(x+wid, y), @Poi(x+wid, y+sH));
        g.DraLin(sp, @Poi(x, y+sH/2), @Poi(x+wid, y+sH/2));
        ;
        mf.canDra.AdVis(zE)
;
x = x + sW
  ;
y = y + f.Hei/2
  ;
tA.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my);  * ;  (tA≠$) - ?
y=y + tA.hei ;
y = y + 4 ;
po∈@Poi(x, y) ;
DraT_M(mf, po, „: ");
y = y + sepS.Hei + 4 ;
tB.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my): *: (tB≠$) - ?
y=y + tB.hei ;
y = y + 4 ;
po∈@Poi(x, y) ;
DraT_M(mf, po, „: ");
y = y + sepS.Hei + 4 ;
con.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my): *:(con≠$)-?
```

usi(g∈@DraCon=zE.RenOpe()) — стандартний функційний унітерм *usi()*, який описує створення і висвітлення знака операції горизонтального елімінування та включає створення змінної *g* стандартної підсистеми *DraCon*, яка реалізується відомим класом `DrawingContext` [3, 5] та приписування змінній значення змінної, яке отримане стандартним функційним унітермом *RenOpe()*, що реалізується відомою процедурою `RenderOpen()` [3, 5], *g.DraLin(sp, @Poi(x, y), @Poi(x, y+sH))* — рисування стандартним унітермом *DraLin()*, який реалізується відомою процедурою `DrawLine()` [3, 5], однієї із трьох ліній операції елімінування з використанням пера *sp* і кординат початкової *(x, y)* та кінцевої *(x, y+sH)* точок, які є типом стандартної підсистеми *Poi()*, що реалізується відомим класом `Point()` [3, 5], *g.DraLin(sp, @Poi(x+wid, y), @Poi(x+wid, y+sH))* та *g.DraLin(sp, @Poi(x, y+sH/2), @Poi(x+wid, y+sH/2))* — опис рисування

другої та третьої ліній знака операції елімінування, *mf.canDra.AdVis(zE)* — до графічного елемента *canDra* головної підсистеми *mf* додавання стандартним унітермом *AdVis()*, який реалізується відомим методом `AddVisual()` [3, 5], утвореного і збереженого в *zE* зображення операції горизонтального елімінування, *x = x +f.Hei / 2* — збільшення значення абсциси на половину висоти кегля, *y = y + sH* — збільшення значення ординати, *(tA≠$)* — умовний унітерм перевірки наявності першого унітерму, *tA.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my)* — вибір функційного унітерму *Dra()* для рисування першого унітерму операції елімінування, *x=x + tA.wid* – збільшення ординати знака на довжину першого унітерму, *po∈@Poi(x, y)* — створення змінної стандартного класу і приписування їй значень поточних координат, *DraT_M(mf, po, „;")* — функційний унітерм рисування розділювача унітермів, *x = x + sepS.Wid + f.Hei / 2* — збільшення поточного значення абсциси на довжину розділювача з урахуванням кеглю розділювача, *(tB≠$) - ?* — перевірка наявності другого унітерму операції елімінування, *tB.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my)* — рисування другого унітерму, *x=x + tB.wid* — збільшення поточного значення абсциси на довжину другого унітерма, *(con≠$)-?* — перевірка наявності унітерму-умови операції елімінування, *con.Dra(mf, f, bb, gb, sp, dp, x, y, mx, my)* — рисування унітерму-умови, *mf.zny=$* — обнулення ознаки видалення унітермів.

Модель функційного унітерму формування знака операції вертикального елімінування *H()* аналогічна моделі операції горизонтального елімінування.

Модель інформаційної технології xml - опису операції елімінування. Xml — опис операції елімінування має охоплювати ідентифікатори операції елімінування (*e*) та орієнтації (*ori*), а також значення орієнтації (*hor* чи *ver*) і самі унітерми. Формула, яка описує створення подібного опису з такими даними, така:



де *xmlD*∈*@XmlDoc* — вхідна змінна *xmlD* типу підсистеми *XmlDoc*, яка реалізується відомим класом `XmlDocument` [3, 5], *n*∈*@XmlEl* — вхідна змінна *n* типу підсистеми *XmlEl*, яка реалізується відомим класом `XmlElement` [3, 5], *nEl*∈*@XmlEl* — створення змінної *nEl* типу *XmlEl*, *nAt*∈*@XmlAt* — створення змінної *nAt* типу *XmlAt*, яка реалізується відомим класом `XmlAttribute` [3, 5], *nEl* = *xmlD.CreEl("e")* — змінній *nEl* елемент *«e»* приписується типовим функційним унітермом *CreEl()*, який реалізується відомою процедурою `CreateElement()` [3, 5], *nEl.Ats.Ap(nAt)* — змінна *nEl* буде доповнена *Ap()* новими атрибутами *Ats*, *nAt* = *xmlD.CreAt(«ori»)* — до змінної *xmlD* введення *CreAt()* атрибуту *«ori»* і приписування значення змінної *xmlD* змінній *nAt*, *(ori=Ori.Hor)-?* — перевірка наявності задання горизонтальної орієнтації (*Ori.Hor*) операції секвентування, *nAt.Val* = *«hor»* та *nAt.Val* = *«ver»* — приписування атрибуту *nAt* значень *«hor»* та *«ver»*, *n.ApChi(nEl)* — ввести вложений елемент у *nEl*, *(tA≠$)-?* — перевірка наявності першого (*tA*) унітерму елімінування, *tA.CreXML(xmlD, nEl)* — формування *xml* — елемента з першого унітерму елімінування, *(tB≠$)-?* — перевірка наявності другого (*tB*) унітерму елімінування, *tB.CreXML(xmlD, nEl)* — формування *xml* - елемента з другого унітерма елімінування, *(con≠$)* - *?* — перевірка наявності унітерму-умови, *con.CreXML(xmlD, nEl)* — формування *xml* - елемента з унітерму-умови.

Написаний мовою об'єктного програмування **C#** код моделі має такий вигляд:

```
DrawingVisual znakEliminuwannia = new DrawingVisual();
if (orientation == Orientation.Horizontal)
 {
  if (((mf.znyszczyty != "DeleteAll")
        && (mf.znyszczyty != "deleteWithoutFirst")
        && (mf.znyszczyty != "deleteWithoutSecond")
        && (mf.znyszczyty != "deleteWithoutChild"))
          ||(mf.znyszczyty == null))
                    {
double symbolHeight = (Math.Sqrt(width) / 2) + 2;

                 using (DrawingContext dc =
znakEliminuwannia.RenderOpen())
                    {
                     dc.DrawLine(sp, new Point(x, y), new
Point (x, y + symbolHeight));
                     dc.DrawLine(sp, new Point (x + width,
y), new Point (x + width, y + symbolHeight));
                     dc.DrawLine(sp, new Point (x, y +
(symbolHeight / 2)), new Point (x - width, y + (symbolHeight
/ 2)));

mf.canwasDraw.AddVisual(znakEliminuwannia);
                    }
                x += ( f.Height / 2);
```

```
                y += symbolHeight;

                if (termA != null)
                {
                    termA.Draw(mf, f, bb, gb, sp, dp, x, y,
marginX, marginY);
                    x += termA.width;
                }

                x += (f.Height / 2);
                po = new Point(x, y);
                DrawText_M(mf, po, ";");

                x += separatorSize.Width;

                x +=(f.Height / 2);

                if (term3 != null)
                {
                    termB.Draw(mf, f, bb, gb, sp, dp, x, y,
marginX, marginY);
                    x += termB.width;
                }

                x += (f.Height / 2);
                   po = new Point(x, y);
                   DrawText_M(mf, po, ";");

                x += separatorSize.Width;

                   x +=(f.Height / 2);

                   if (condition != null)
                       condition.Draw(mf, f, bb, gb, sp, dp, x,
y, marginX, marginY);
                       }

                      else
                      {
                       mf.znyszczyty = null;
                      }
            }
            else
            {
             if (((mf.znyszczyty != "DeleteAll")
                && (mf.znyszczyty != "deleteWithoutFirst")
                && (mf.znyszczyty != "deleteWithoutSecond")
                && (mf.znyszczyty != "deleteWithoutChild"))
                        ||(mf.znyszczyty == null))
                   {
                int symbolWidth =
```

```
((int)Math.Sqrt(height) / 2) + 2;

                using (DrawingContext g =
znakEliminuwannia.RenderOpen())
                    {
                     g.DrawLine(sp,new Point (x, y), new
Point (x + symbolWidth, y));
                     g.DrawLine(sp, new Point (x, y +
height), new Point (x + symbolWidth, y + height));
                     g.DrawLine(sp, new Point (x +
(symbolWidth / 2), y), new Point (x + (symbolWidth / 2), y +
height));
                        mf.canwasDraw.
AddVisual(znakEliminuwannia);
                    }
                x += symbolWidth;
                y +=(f.Height / 2);

                if (termA != null)
                {
                    termA.Draw(mf, f, bb, gb, sp, dp, x, y,
marginX, marginY);
                    y += termA.height;
                }
                y -= 4;
                po = new Point(x, y);
                g.DrawText(text, x, y);
                    DrawText_M(mf, po, ";");
                y += separatorSize.Height;
                y += 4;
                if (termB != null)
                {
                    termB.Draw(mf, f, bb, gb, sp, dp, x, y,
marginX, marginY);
                    y += termB.height;
                }
                y += 4;
                po = new Point(x, y);
                    DrawText_M(mf, pc, ";");
                y += separatorSize.Height;
                y += 4;
                if (condition != null)
                    condition.Draw(mf, f, bb, gb, sp, dp, x,
y, marginX, marginY);
                    }
                    else
                    {
                        mf.znyszczyty = null;
                    }
                }
        }
```

Отже, засобами розширеної алгебри алгоритмів описані моделі формування і *xml* - опису операції елімінування. Практичною реалізацією і апробацією верифіковано математичні моделі формування і *xml* - опису операції елімінування.

1. Бритковський В. М. Моделювання редактора формул секвенційних алгоритмів: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 «Математичне моделювання та обчислювальні методи» / В. М. Бритковський. — Львів, 2003. — 18 с. 2. Василюк А. С. Підвищення ефективності математичного і програмного забезпечення редактора формул алгоритмів: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 «Математичне та програмне забезпечення обчислювальних машин і систем» / А. С. Василюк. — Львів, 2008. — 20 с. 3. Мэтью Мак-Дональд Windows presentation foundation в .NET 3.5 с примерами на C# 2008 / Мэтью Мак-Дональд. — М.; СПб.; К. : «Apress», 2008. — 922 с. 4. Owsiak W. Rozszrzerzenie algebry algorytmów / W. Owsiak, A. Owsiak //Pomiary, automatyka, kontrola 2. — 2010. — S. 184–188. 5. Petzold C. Programowanie Windows w języku C# / C. Petzold. — Warszawa : RM, 2002. — 1161 s.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ОПЕРАЦИИ ЭЛИМИНИРОВАНИЯ

Описываются математические модели выбора, вычисления размеров и формирования xml - описанию операции елиминирования.

INFORMATION TECHNOLOGY OF OPERATION ELIMINATION

We describe of mathematical models selection, calculation of expression and created xml- described elimination operation.

Стаття надійшла 22.11.2010

УДК 330 46:655

В. Є. Климнюк, О. О. Візір

Харківський національний економічний університет

РОЗРОБЛЕННЯ СИСТЕМИ ПІДТРИМКИ ВПРОВАДЖЕННЯ АСУ ПОЛІГРАФІЧНИМ ВИРОБНИЦТВОМ

Аналізується специфіка вибору, впровадження та функціонування АСУ поліграфічним виробництвом, розглядається розроблена методика впровадження таких систем.

Автоматизовані системи управління, поліграфічне підприємство, інформаційне забезпечення, навчальна підсистема

Значимість автоматизованих систем управління поліграфічним підприємством (АСУПП) важко переоцінити, тому що вони відіграють важливу роль в інформаційному забезпеченні системи управління підприємством.