УДК 004.4’232

В. Овсяк, М. Козелко
Українська академія друкарства

МОДЕЛЬ КОНВЕРТАЦІЇ АБСТРАКТНИХ УНІТЕРМІВ ДО ТИПОВИХ ГРАФІЧНИХ УНІТЕРМІВ-АВТОМАТІВ

Засобами алгебри алгоритмів описано модель побудови системи типових графічних унітермів, призначених для конвертації абстрактного унітерма.

Ключові слова: система, модель, графічний унітерм, абстрактний графічний унітерм, алгоритм.

Means algebra of algorithms described created a model of the system default image unitermiv designed to convert abstract uniterm.

Key words: system, model uniterm graphic, abstract graphic uniterm, algorithm.

Вступ і формулювання задачі

Інструментальні засоби інформаційних технологій і систем реалізуються графічними інтерфейсами. Можливі різні типи інструментальних засобів, наприклад, Windows чи Web проектів. Сам процес проектування інструментальних засобів є творчим і складним. Особливо на початкових стадіях проектування є невідомими як загальний вигляд інструментальних засобів, так і його складові (графічні унітерми [1–3]) – кнопки, елементи альтернативного і безальтернативного вибору, текстові поля та інші. Для спрощення проектування інструментальних засобів доцільно вводити абстрактні графічні елементи – унітерми.

Використання абстрактних унітермів надає можливість відмовитись від строго типизованих елементів на етапі розроблення інтерфейсу. Такий підхід є універсальнішим, гнучкішим, передбачає повторне використання спроектованих інтерфейсів для різних типів проектів – таких, як Windows-програма [4, 5] чи інтернет-сторінка [6], а також і в інших проектах. Однак при компіляції готового проекту усі абстрактні унітерми мають бути конвертовані до типових графічних унітермів. У зв’язку з цим існує проблема коректного конвертування абстрактних унітермів до типових графічних унітермів із заданням всіх властивостей та методів їх опрацювання, а також дочірних елементів абстрактного унітерма типовому графічному унітерму.

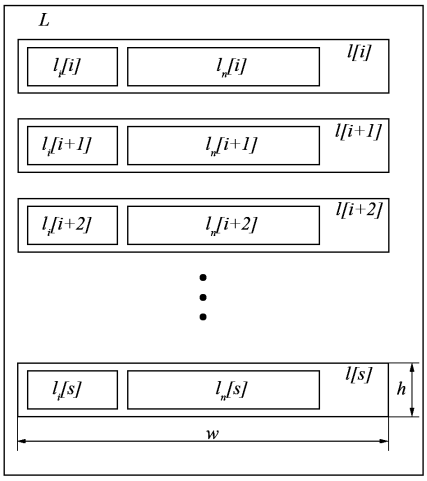

Схема моделі системи типових графічних унітермів, призначених для конвертування абстрактного унітерма

Схематична модель системи типових графічних унітермів

Створену схематичну модель наведено на рисунку. Система типів (L) утворюється із s графічних унітермів контейнерного типу, тобто таких, які можуть містити інші графічні унітерми. Для кожного i-го графічного унітерма задається поле l, яке утворене назвою (l_n) і графічним позначенням (l_i). Для кожного поля задається його висота ($l.h$) і ширина ($l.w$). Кожному полю l задається подія вибору його користувачем для подальшого конвертування абстрактного унітерма до вибраного графічного типу.

Модель побудови системи типових графічних унітермів

Система типових графічних унітермів L формується у разі вибору корисувачем унітерма з розміщених на унітермі-формі (виконання умови u_1, див. наведену нижче формулу алгоритму для побудови якої застосовано модифіковану алгебру алгоритмів [7–10]). Якщо умова u_1 виконується, то за функціональним унітермом L_1 визначається, котрий унітерм вибрав користувач (n_e), а також встановлюється тип вибраного унітерма ($t_e = n_e.type$). Зважаючи на те, що конвертувати можливо лише унітерм абстрактного типу, є доцільним формувати систему L лише при виборі користувачем унітерма абстрактного типу. У зв'язку з цим перевіряють умови $t_e=abstract$ на виявлення абстрактного унітерму. Якщо умова $t_e=abstract$ виконується, то за функціональним унітермом L_2 визначається кількість дочірних унітермів вибраного абстрактного унітерма ($k=n_e.count$), а також перелік усіх доступних графічних унітермів для конвертації абстрактного. Залежно від кількості дочірних унітермів k формується система графічних унітермів L. Якщо $k>0$, тобто обраний абстрактний унітерм містить дочірні унітерми, то формується система L, яка містить графічні унітерми лише контейнерного типу. Для відображення у списку назв типів графічних унітермів, у циклі, поки виконується умова $i<=s.length$ (*length* – типова системна властивість [4, 5, 6], яка використовується для обчислення кількості унітермів s системи), генерується поле відображення типу графічного унітерма для конвертації абстрактного. Щоб реалізувати відбір до системи L

графічних унітермів лише контейнерного типу, перевіряють умову *s[i].child=true*, тобто перевіряється чи може графічний унітерм містить дочірні унітерми. Якщо умова *s[i].child=true* виконується, то генерується поле відображення типу графічного унітерма *l* із заданими розмірами висоти *l.h* і ширини *l.w*. Кожному полю *l* задається подія вибору його користувачем для подальшого конвертування абстрактного унітерма до вибраного графічного типу. Поле відображення типу графічного унітерма *l* містить унітерм l_n назви типу графічного унітерма і унітерм графічного позначення цього унітерма l_i. Згенерованому унітерму l_i задається графічне позначення відповідного графічного унітерма $l_i.v=s[i].ico$ і розміщення його на полі $l.child= l_i$. Унітерму відображення типу графічного унітерма l_n задається назва типу $l_n.n=s[i].type.name$ і проводиться розміщення його на полі $l.child=l_n$. Успішно згенероване поле *l* додається до списку *L.child=l*. Враховуючи кількість визначених контейнерних унітермів, формують висоту списку *L.h=l.h*i*. Якщо ж *k=0*, тоді формується система *L* із всіх можливих типів графічних унітермів. Залежно від кількості доступних для конвертації типів графічних унітермів, у циклі, поки виконується умова *j<=s.lenght*, генерується поле відображення типу графічного унітерма *l*. Полю *l* задається висота *l.h* і ширина *l.w*, а також подія вибору поля користувачем *l.click()*. На полі *l* розміщуються унітерми відображення назви типу графічного унітерма для конвертації $l.child=l_n$ із заданим значенням $l_n.n=l[j].type.name$, а також розміщується графічне позначення відповідного типу $l.child=l_i$ з наступним значенням $l_i.v=s[j].ico$. Успішно згенероване поле відображення типу графічного унітерма *l* додається до списку *L.child=l*. У цьому випадку висота списку залежатиме від кількості доступних для конвертації обраного абстрактного унітерма типів графічних унітермів *L.h=l.h*j*.

Фрагмент програмної реалізації моделі

Реалізація моделі конвертації абстрактних до типових графічних унітермів автоматів проілюстровано фрагментом програмного коду мови програмування C# у вигляді методу `private void Convert_Click(object sender, RoutedEventArgs e)`. Даний програмний код реалізує конвертацію абстрактного унітерма до унітерма типу кнопки, або типу текстового поля у залежності від вибору користувача. Програмно визначається тип унітерма обраного користувачем для подальшої конвертації абстрактного до типопого графічного унітерма. На основі обраного типу генерується екземпляр елемента цього типу із заданням всіх властивостей притаманних абстрактному унітерму: розміри, розміщення, колір і т.д. Задаються всі функціональні особливоті абстрактного новому згенерованому унітерму. Також відбувається перевірка наявності у абстрактного унітерма дочірних унітермів, для розміщення їх на новому графічному унітермі.

```
private void Convert_Click(object sender, RoutedEventArgs e)
        {
            removRes();
            if (contr.GetType().Name == "Abstract")
            {
                FrameworkElement fr = new FrameworkElement();
                SortedList sl = new SortedList();

                Abstract ab = (Abstract)contr;
                if (typStr == "Button")

                        //Кнопка
                {

                        //Створення об’єкта класу типового графічного
унітерма
                    childbutton = new Button();

                        //Передача параметрів абстрактного унітерма
                    childbutton.Name = ab.Name;
                    childbutton.Content = ab.Name;
```

```
                childbutton.Height = ab.Height;
                childbutton.Width = ab.Width;
                Canvas.SetLeft(childbutton, Canvas.GetLeft(ab));
                Canvas.SetTop(childbutton, Canvas.GetTop(ab));

                    //Задання назви методу опрацювання події
                childbutton.Click += new
RoutedEventHandler(childbutton_Click);
                oblist.Add(childbutton);
                fr = childbutton;

                    //Приписування усіх властивостей
                EventL.Add(fr.Name, sl);
            }
            if (typStr == "TextBox")

                    //Текстове поле
            {
                childTextBox = new TextBox();
                childTextBox.Name = ab.Name;
                childTextBox.Height = ab.Height;
                childTextBox.Width = ab.Width;
                Canvas.SetLeft(childTextBox, Canvas.GetLeft(ab));
                Canvas.SetTop(childTextBox, Canvas.GetTop(ab));
                childTextBox.GotFocus += new
RoutedEventHandler(childTextBox_GotFocus);
                oblist.Add(childTextBox);
                fr = childTextBox;
                EventL.Add(fr.Name, sl);
            }

        }

                //Перевірка і врахування дочірніх унітермів
            switch (ab.Parent.GetType().Name)
            {
                case "Abstract":
                    Abstract a = (Abstract)ab.Parent;
                    a.Children.Add(fr);
                    a.Children.Remove(ab);
                    oblist.Remove(ab);
                    break;
                case "Canvas":
                    Canvas c = (Canvas)ab.Parent;
                    c.Children.Add(fr);
                    c.Children.Remove(ab);
                    oblist.Remove(ab);
                    break;
                case "StackPanel":
                    StackPanel s = (StackPanel)ab.Parent;
                    s.Children.Add(fr);
                    s.Children.Remove(ab);
                    oblist.Remove(ab);
                    break;
                default:
                    break;
            }    }
```

Висновки

1. Використання створеної моделі забезпечує коректне і безвтратне перетворення абстрактних унітермів до типових графічних унітермів.

2. Нові графічні унітерми забезпечуються усіма функціональними особливостями, притаманними абстрактному унітерму, які задані коритувачем на етапі розроблення графічного інтерфейсу.

3. Модель перетворення абстрактного до типових графічних унітермів програмно реалізовано і апробовано.

1. Овсяк О.В. Модель інформаційної технології опрацювання формул алгоритмів / О. Овсяк // Вісник Нац. ун-ту "Львівська політехніка": Комп'ютерні науки та інформаційні технології. – 2011. – № 710. – С. 224–234. 2. Овсяк В., Овсяк О., Василюк А. Елементи мови предметних унітермів // Збірник наукових праць „Кваліlогія книги". – 2009. – № 2(16). – С. 4–9. 3. Овсяк О.В. Інформаційна технологія побудови моделей інтерфейсів інформаційних технологій і систем / О.В. Овсяк // Збірник наукових праць „Комп'ютерні технології друкарства". – 2011. – № 26. – С. 105–114. 4. Шилд Г. C# 4.0: Полное руководство. – М.: ООО И.Д. "Вильямс", 2011. – 1056 с. 5. Троэлсен Э. Язык программирования C# 2010 и платформа .NET 4.0. – М.: ООО И.Д. "Вильямс", 2010. – 1392 с. 6. Мак-Дональд М. WPF: Windows Presentation Foundation в .NET 3.5 с примерами на C# 2008 для профессионалов. – М.: ООО И.Д. "Вильямс", 2008. – 928 с. 7. Ovsyak A. The extended algebra of algorithms width multiconditional elimination / V. Ovsyak, A. Ovsyak // Вісник Нац. ун-ту "Львівська політехніка": Комп'ютерні науки та інформаційні технології. – 2010. – № 672. – С. 291–300. 8. Овсяк О. Розширена алгебра алгоритмів і модель зв'язку між класичною і розширеною операцією елімінування / О. Овсяк // Збірник наукових праць „Комп'ютерні технології друкарства". – 2010. – № 23. – С.45–53. 9. Owsiak A. Rozszrzerzenie algebry algorytmów / W. Owsiak, A. Owsiak // Pomiary, automatyka, kontrola. – 2010. – № 2. – S. 184–188. 10. Овсяк О.В. Розширення алгебри алгоритмів аксіомами операцій циклів / О.В. Овсяк // Вісник Нац. ун-ту "Львівська політехніка": Комп'ютерні системи проектування. Теорія і практика. – 2010. – № 685. – С. 12–20. 11. Ovsyak A. The extended algebra of algorithms with additional cycle elimination axiomats / A. Ovsyak, V. Ovsyak // Conference "Intelligent Information and Engineering Systems" (INFOS 2011), September 19–23, 2011, Polańczyk, Poland. – P. 23–34.