УДК 004.4’273 : 005.311.2

Г.С. Яма, Н.Г. Кеберле

Запорізький національний університет, Запоріжжя

ВІЗУАЛЬНИЙ РЕДАКТОР ДІАГРАМ ДЛЯ ДЕКЛАРАТИВНО ВИЗНАЧЕНИХ ПРОБЛЕМНО-ОРІЄНТОВАНИХ МОВ ДІАГРАМ

Використання засобів загальнодоступних візуальних редакторів діаграм для побудови синтаксично коректних діаграм на довільній проблемно-орієнтованій мові діаграм вимагає від користувача експертних знань з такої мови, що не завжди можливо. В даній роботі розглядаються рішення, що були покладені в основу програмного інструментарію візуального редактора діаграм, який дозволяє малювати синтаксично коректні діаграми за умови існування опису проблемно-орієнтованої мови діаграм.

Ключові слова: *діаграма, візуальний редактор діаграм, проблемно-орієнтована мова діаграм.*

Вступ

Подання відомостей у вигляді діаграм є зручним способом відобразити структурні, змістові, часові та інші залежності між елементами деякої предметної області. Створення діаграм є поширеною задачею і для її розв’язку створено багато рішень. Відомі візуальні редактори діаграм можна умовно поділити на три класи. До першого класу відносяться редактори діаграм загального призначення, які не прив’язані до стандарту моделювання діаграм, і мають певний перелік популярних елементів діаграм загального призначення. До другого класу відносяться вузькоспеціалізовані редактори діаграм, які розробляються під конкретний тип діаграм, такий як UML, блок-схеми, діаграми сімейства IDEF. До третього класу належать візуальні редактори, які пропонують як попередньо визначені спеціалізовані палітри елементів діаграм, так і можливість створити власну палітру елементів діаграм, та стандартними засобами відображати діаграми, що включають елементи з нестандартних палітр.

Розвинений інструментарій візуалізації діаграм, включаючи алгоритми автоматичного розміщення елементів, масштабування зображень, копіювання частин діаграм, налаштування та обробку додаткових властивостей (кольору, типу ліній, типу з’єднань, інших особливостей елементів), імпорту/експорту діаграм в деякий текстовий формат (наприклад, GraphML [1]) надає можливості людині, що створюватиме діаграми, зосередитися на досягненні наочності діаграм, але не контролює змісту та коректності його дій і рішень при обранні тих чи інших елементів та типів з’єднань між елементами.

Створення принципових схем інженерних мереж та електричних ланцюгів, організаційних діаграм, діаграм Ганта, мережевих діаграм PERT, UML діаграм та багатьох інших вимагають від користувача експертних знань відповідної проблемно-орієнтованої мови та використаної нотації.

Метою даної роботи є опис рішень, що були прийняті при розробці візуального редактора діаграм для декларативно визначених проблемно-орієнтованих мов діаграм та побудови синтаксично коректних діаграм на таких мовах.

1. Джерела дослідження та огляд відомих рішень

Прикладом візуального редактора першого класу (діаграм загального призначення) є Microsoft Visio [2] (рис.. 1) Цей редактор має великий набір графічних елементів та дозволяє створювати довільні діаграми. Редактор малює об’єкти у місцях, обраних користувачем, дозволяє їх довільно пересувати та накладати одні об’єкти на інші.

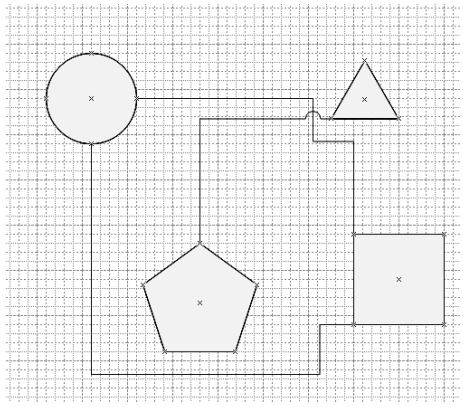

Рис. 1. Приклад діаграми у Microsoft Visio

Завдяки різноманіттю елементів у цьому редакторі можна конструювати будь-які діаграми, але, з іншого боку, велика кількість елементів та неспеціалізованість редактора ускладнює роботу з ними, особливо у випадку, коли користувачеві потрібен один конкретний стандарт, і коректність створеної діаграми цілком залежить від людини, що її створювала.

При малюванні зв’язків редактор використовує контрольні точки для приєднання, що є на кожному

об'єкті. Як видно на рис. 1, лінії між об'єктами малюються автоматично, за складним алгоритмом, що дозволяє уникати перетину лінії з іншими об'єктами та створює особливі місця перетину ліній між собою. У редактора Microsoft Visio немає автоматичних способів розташування об'єктів. З точки зору розширення існуючих або створення нових палітр Microsoft Visio дозволяє описувати власні шаблони заповнення (fill patterns [3]), власні лінії та закінчення ліній, а також додаткові властивості існуючих елементів (line and end patterns [4]). Але редактор Microsoft Visio не відстежує відповідність діаграми до певної мови діаграм, оскільки редактор розраховано на малювання будь-яких діаграм.

Одним з відомих вузькоспеціалізованих редакторів другого класу є ArgoUML [5] (рис. 2). Цей редактор дозволяє створювати діаграми згідно зі стандартом UML. Редактор малює об'єкти у місцях, обраних користувачем, та дозволяє їх довільно пересувати, за винятком випадків накладання об'єктів один на одного. Зв'язки між об'єктами встановлюються як найкоротша відстань між границями об'єктів. За бажанням користувача можна встановити додаткові опорні точки на зв'язках, що дозволить створювати ламані зв'язки.

Редактор дозволяє автоматично вирівнювати діаграми різними способами.

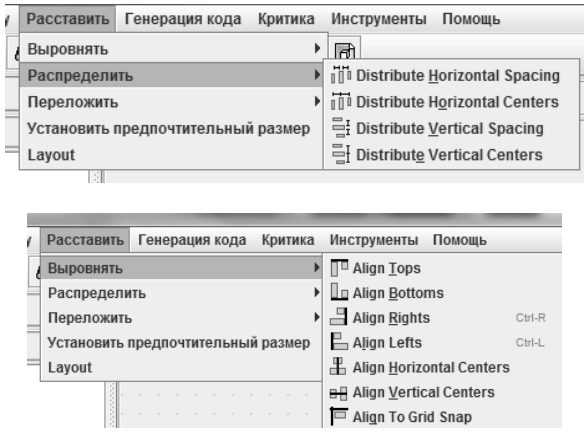

Рис. 2. Можливості ArgoUML щодо вирівнювання елементів діаграми

Як видно на рис. 2, ArgoUML дає можливість вирівнювати об'єкти за висотою та шириною, вирівнювати за границями, а також нормувати розподілення відстаней між об'єктами. Також редактор має функцію Layout, яка дозволяє автоматично розмістити об'єкти на діаграмі. Однак цей спосіб є не ефективним і допускає помилки, а також не може переставляти опорні точки для ламаних ліній зв'язків.

Редактор ArgoUML слідкує за коректністю та не дозволяє створювати невідповідні мові UML діаграми. Таким чином, перевагою вузькоспеціалізованого редактора діаграм можна вважати гарантоване отримання синтаксично коректних діаграм. Досягається це вбудуванням у програмний код перевірки коректності дій користувача при побудові діаграми та забороні некоректних поєднань елементів та зв'язків.

Типовими представниками редакторів третього класу можна вважати yEd [6] та DIA [7]. Обидва редактори мають можливість створювати нестандартні (custom) елементи в діаграмах. Редактор yEd дозволяє додавання нестандартних елементів з файлів векторної (SVG) та растрової графіки [8], які можна завантажити з файлів або з Інтернет. Додатково, редактор дозволяє додавати у палітру довільну комбінацію вже існуючих елементів діаграм.

У редакторі DIA [9] нестандартні елементи можна задати у XML-файлі за допомогою базових SVG специфікацій (line, polyline, polygon, circle та інші) та їх параметрів.

Водночас, в обох редакторах контроль за коректністю отриманих діаграм, з урахуванням нестандартних елементів, відсутний.

2. Архітектура візуального редактора

Візуальний редактор, інтерфейс якого можна побачити на рис. 3, призначено для малювання діаграм у довільній наперед заданій мові діаграм. За умовчанням, жодної палітри не встановлено. Редактор дозволяє завантажити набір файлів, в якому описано потрібну мову діаграм, включаючи графічні зображення її елементів, дозволені комбінації елементів та підписи, та малювати синтаксично коректні графічні діаграмі на цій мові.

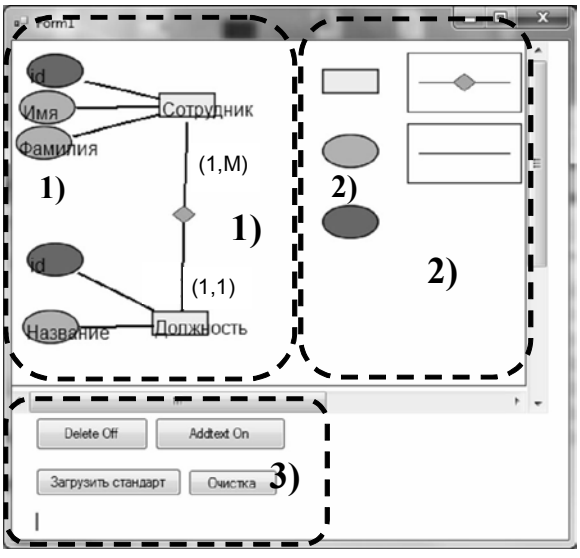

Рис. 3. Загальний вигляд вікна редактора (використовується палітра ER-діаграм)

Інтерфейс програми виглядає наступним чином. Область 1 - область малювання діаграм, у області 2 знаходиться панель інструментів, у області 3

вказані додаткові функції, що застосовуються до елементів діаграм.

При старті редактору, а також при зміні вибору мови опису діаграм, завантажуються відповідні файли. З деталями та правилами опису мови діаграм можна ознайомитися у роботі [10].

Для розробки редактора була використана мова C++.

Малювання панелі інструментів та області для діаграми виконується завдяки класу Graphics [11] з простору імен System.Drawing [12] Microsoft Visual Studio, який створює поверхню для малювання GDI+ [13]. Метод DrawImage[14] дозволяє намалювати зображення з урахуванням прозорості. Метод DrawLine[15] разом із класом CustomCap[16] дозволяє малювати довільні лінії з різними видами кінців.

Архітектура візуального редактора показана на рис. 4.

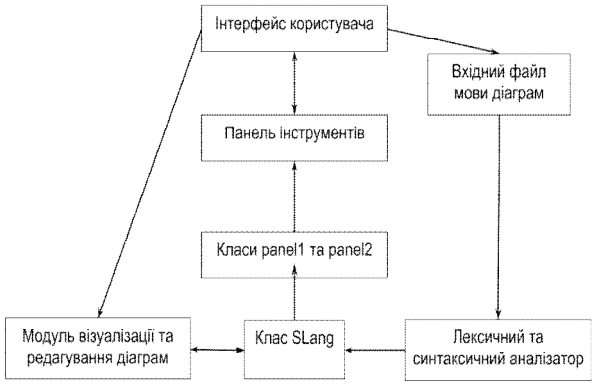

Рис. 4. Архітектура візуального редактора діаграм

Клас Slang відповідає за повне програмне представлення зчитаної з вхідного файлу мови моделювання. Об'єкт класу SLang створюється в разі успішного проходження синтаксичного та лексичного аналізу вхідного файлу мови. Після цього на основі класу SLang формуються набори зображень для об'єктів та допустимі види зв'язків. На наступному етапі формуються об'єкти класів panel1 та panel2, що відповідають за панель об'єктів та панель зв'язків. На основі цих класів створюється інтерфейс редактора, та надається можливість користувачу малювати необхідні йому діаграми.

3. Методи обробки дій користувача при роботі з діаграмою

Користувач може маніпулювати об'єктами та зв'язками у робочій області редактора діаграм.

Будь-який об'єкт можна перемістити в довільне місце шляхом натискання на нього лівою кнопкою миші та подальшому перетаскуванні. При цьому всі наявні з цим об'єктом зв'язки перемальовуються згідно з новим розташуванням об'єктів.

Будь-який кінець зв'язку можна від'єднати від одного об'єкта та приєднати до іншого, за умови допустимості такого з'єднання згідно обраній мові графічного моделювання. Ця операція виконується шляхом натискання на необхідну частину зв'язку лівою кнопкою миші та подальшому перетаскуванні. Зв'язок намагається приєднатися до об'єкта, над яким було відпущено кнопку миші.

За відсутності на цьому місці об'єкта або за наявності неприпустимого для даного зв'язка об'єкта зв'язок повернеться до попереднього положення.

Розглянемо особливості реалізації процесу малювання об'єктів діаграми. Для візуалізації набору елементів використовуються два масиви структур – для зв'язків та об'єктів.

Масив об'єктів відсортований за часом останньої зміни об'єкту.

При необхідності відслідкувати **клік по об'єкту** виконується наступний алгоритм 1:

Алгоритм 1.
***Вхід**: масив об'єктів, кількість об'єктів, координати курсору, де був зроблений клік мишею.*
***Вихід**: номер об'єкту, по якому зроблено клік, або повідомлення про те, що кліку не відбулося.*
***Початок**:*
***Крок 1**. **i** = (кількість об'єктів -1)*
***Крок 2**. якщо **i<0**, то Кінець*
***Крок 3**. якщо координати курсору не потрапили в прямокутник, що відповідає **i**-му об'єкту, то*
***i--**,*
повернутися на Крок 2
***Крок 4**. якщо координати курсору потрапили на прозору область об'єкту, то*
***i--**,*
повернутися на Крок 2
***Крок 5**. запам'ятати **i** - номер поточного об'єкту*
***Кінець**.*

Для **малювання зв'язків** спочатку беруться дві опорні точки: центр С і точка перетину лінії та границі зображення x0 (рис. 5). Після цього виконується алгоритм 2:

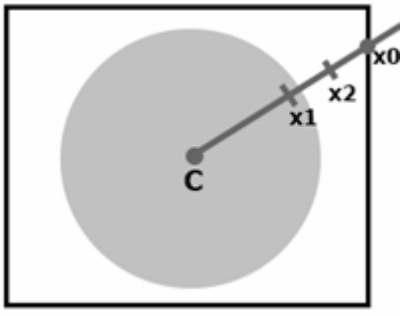

Рис. 5. Схема приєднання зв'язку

Алгоритм 2.
x0 *– початкова точка,* ***xm*** *– дальній кінець відрізку,*
c *– ближній кінець відрізку,* ***xн*** *– середина відрізку.*
Вхід*: малюнок об'єкту, координати центру об'єкту.*
Вихід*: координати на границі об'єкту.*
Початок*:*
Крок 1. ***xm = x0***
Крок 2. *якщо колір в точці* ***xm*** *не прозорий, то Кінець*
інакше
Крок 3. *розраховується точка* ***xн*** *– як середина*
відрізку ***[c,xm]***.
якщо колір в точці ***xн*** *не прозорий то* ***c = xн***,
інакше ***xm = xн***.
якщо довжина відрізку ***[c,xm]<1*** *то кінець,*
інакше перехід на Крок 3.
Кінець.

Під час виконання алгоритму 2, шляхом послідовного наближення точка xt отримає значення координат першого непрозорого пікселя на відрізку. До точки з цими координатами буде приєднано кінець зв'язку.

У разі складної структури зображення, з наявністю значної неоднорідності прозорих та непрозорих частин, наприклад зображення з прозорим центром, алгоритм обере центральну точку для приєднання зв'язку.

У редакторі передбачена можливість перетаскувати об'єкти та зв'язки, тому при роботі з редактором постійно виникає необхідність відслідковувати кліки по об'єктам та по зв'язкам.

Також необхідно відслідковувати кліки по панелі інструментів.

Зв'язки, як і об'єкти, представлені масивом. Однак на відміну від об'єктів, кожен елемент цього масиву – зв'язок – може мати декілька початків і кінців. Тому при визначенні кліку суттєвим є не тільки номер зв'язка, по якому відбувся клік, а також і місце. Допускаються кліки по кожному з кінців зв'язка, а також по центру зв'язка. Клік по центру не дозволяє маніпулювати об'єктом, а необхідний для того щоб ввести текст в центральне поле зв'язку.

При необхідності **відслідкувати клік по зв'язку** (рис. 6), виконується алгоритм 3:

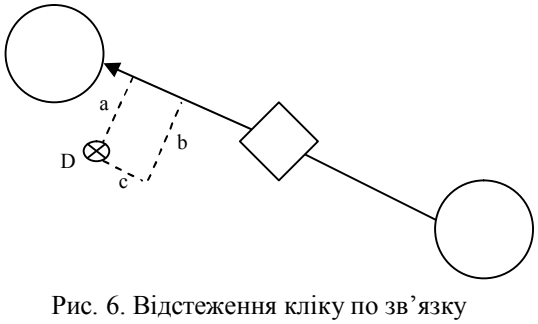

Рис. 6. Відстеження кліку по зв'язку

Алгоритм 3:
Вхід*: масив зв'язків, кількість зв'язків, координати курсору (точка D), де був зроблений клік.*
Вихід*: номер зв'язку, по якому зроблено клік, або повідомлення про те, що клік не відбулося.*
Початок*:*
Крок 1. ***i = 0***.
Крок 2. *якщо* ***i >*** *(кількість зв'язків-1) то Кінець.*
Крок 3. *знаходиться відстань* ***a*** *від точки кліку D до і-й лінії.*
Крок 4. *якщо* ***a > 4*** *то*
i++,
повернутися на Крок 2.
Крок 5. *знаходиться відстань* ***c*** *від точки кліку D до лінії* ***b***, *що є перпендикуляром до лінії зв'язку, проведеним з її центру.*
Крок 6. *якщо* ***c*** *> половини довжини лінії зв'язку, то i++,*
повернутися на Крок 2,
інакше запам'ятати ***i*** *- номер лінії, яка отримала клік.*
Кінець.

Розрахунок відстані від точки до відрізку виконується за рахунок рішення системи лінійних алгебраїчних рівнянь.

$$\begin{cases} \dfrac{x - x_1}{x_2 - x_1} = \dfrac{y - y_1}{y_2 - y_1}; \\ \dfrac{x - x_1}{x_2 - x_1} = \dfrac{y - y_1}{y_1 - y_2} + \dfrac{x_3 - x_1}{x_2 - x_1} - \dfrac{y_3 - y_1}{y_1 - y_2}. \end{cases} \quad (1)$$

Після чого за формулою розрахунку відстані між точками на площині

$$s = \sqrt{(x'-x_3)^2 + (y'-y_3)^2}\ , \quad (2)$$

де (x', y') - розв'язок СЛАР, можна отримати величину s^2, що є квадратом відстані і використовується для подальших розрахунків:

$$s^2 = \frac{\left(-x_1y_3 + x_1y_2 + x_2y_3 + x_3y_1 - x_2y_1 - x_3y_2\right)^2}{\left(y_2^2 - 2y_1y_2 + y_1^2 + x_2^2 - 2x_1x_2 + x_1^2\right)}, \quad (3)$$

де (x_1, y_1) та (x_2, y_2) – координати початку та кінця відрізку, (x_3, y_3) – координати точки, від якої шукається відстань.

4. Застосування редактора

Запропонований редактор діаграм можна використовувати у двох напрямках: по-перше, для створення синтаксично коректних діаграм для вузькоспеціалізованих проблемно-орієнтовних мов; по-друге – для навчання таким мовам.

Наприклад, частина 2 стандарту ISO 51926 [17] передбачає 102 типи елементів діаграм. У вузькоспеціалізованому редакторі [18], що створює та реда-

гує дані згідно зі стандартом ISO 15926, ці дані подаються у текстовому вигляді та у вигляді ієрархії елементів відповідно до частини 2 стандарту, а діаграмне подання таких даних можливо лише шляхом генерації діаграми за допомогою пакету GraphViz [19], що не дає можливості відредагувати дані безпосередньо у режимі діаграми. Для полегшення навчання користувачів даних, оформлених згідно частини 2 стандарту ISO 15926, запропонований редактор діаграм дозволяє створювати приклади діаграм, корегуючи дії користувачів при створенні власних навчальних діаграм.

До факторів, що утруднюють роботу з редактором, відноситься необхідність створювати опис проблемно-орієнтованої мови діаграм, що вимагає експертних знань з неї. Можливим удосконаленням запропонованого редактора можна вважати розробку додаткового редактора опису таких проблемно-орієнтованих мов діаграм: допустимих елементів, їх зовнішнього вигляду, допустимих зв'язків та обмежень на поєднання елементів.

Список літератури

1. GraphML – язык описания графов [Електронний ресурс] / Режим доступу : http://citforum.ck.ua/internet/xml/graphml/ .

2. Профессиональное программное обеспечение для построения схем - Microsoft Visio [Електронний ресурс] / Режим доступу : http://office.microsoft.com/ru-ru/visio/ .

3. Microsoft Visio. Custom Patterns [Електронний ресурс] / Режим доступу: http://msdn.microsoft.com/en-us/library/office/aa200997(v=office.10).aspx .

4. Microsoft Visio. Custom Properties [Електронний ресурс] / Режим доступу: http://msdn.microsoft.com/en-us/library/office/aa200980(v=office.10).aspx .

5. Open Source Software Engineering Tools [Електронний ресурс] / Режим доступа : http://argouml.tigris.org/

6. yEd Graph Editor [Електронний ресурс]/ Режим доступу: http://www.yworks.com/en/ products_yed_applicationfeatures.html .

7. DIA diagram editor [Електронний ресурс] / Режим доступу: http://dia-installer.de/ .

8. Palette' Tool Window [Електронний ресурс] / Режим доступу: http://yed.yworks.com/ support/manual/palette.html .

9. DIA diagram manual: Chapter 11. Custom Shape Module [Електронний ресурс] / Режим доступу: http://dia-installer.de/doc/en/custom-shapes-chapter.html .

10. Кеберле Н.Г. Метод визначення граматики проблемно-орієнтованої мови діаграм / Н.Г. Кеберле, Д.В. Чесановський. – Системи обробки інформації. – Х. ХУПС, 2014. – Вип. 5(121). – С. 112-116.

11. Graphics класс (System.Drawing) [Електронний ресурс]/ Режим доступу: http://msdn.microsoft.com/ru-ru/library/system.drawing.graphics%28v=vs.110%29.aspx .

12. System.Drawing пространство имен [Електронний ресурс] / Режим доступу: http://msdn.microsoft.com /ru-ru/library/system.drawing%28v=vs.110%29.aspx .

13. Управляемый код GDI+ [Електронний ресурс] / Режим доступу: http://msdn.microsoft.com/ru-ru/library/d420az6e%28v=vs.110%29.aspx .

14. Graphics.DrawImage метод [Електронний ресурс]/Режим доступу:http://msdn.microsoft.com/ru-ru/library/system.drawing.graphics.drawimage%28v=vs.110%29.aspx.

15. Graphics.DrawLine метод [Електронний ресурс]/Режим доступу: http://msdn.microsoft.com/ru-ru/library/system.drawing.graphics.drawline%28v=vs.110%29.aspx .

16. LineCap перечисление [Електронний ресурс]/ Режим доступа: http://msdn.microsoft.com/ru-ru/library/system.drawing.drawing2d.linecap%28v=vs.110%29.aspx .

17. Системы промышленной автоматизации и интеграция. Интеграция данных жизненного цикла для перерабатывающих предприятий, включая нефтяные и газове производственные предприятия. Ч. 2. Модель данных. (ISO 15926:2010) : ГОСТ Р 15926:2010. – [Чинний від 2010-21-12]. – М. : Стандартинформ 2013. –257 с. – (Национальный стандарт Российской федерации).

18. Data integration standard ISO 15926 : .15926 Editor [Електронний ресурс] / Режим доступу: http://techinvestlab.ru/dot15926Editor .

19. GraphViz – Graph Visualization Software [Електронний ресурс] / Режим доступу: http://www.graphviz.org/ .

Надійшла до редколегії 12.06.2014

Рецензент: д-р техн. наук, проф. С.І. Гоменюк, Запорізький національний університет, Запоріжжя.

ВИЗУАЛЬНЫЙ РЕДАКТОР ДИАГРАММ ДЛЯ ДЕКЛАРАТИВНО ОПРЕДЕЛЕННЫХ ПРОБЛЕМНО-ОРИЕНТИРОВАННЫХ ЯЗЫКОВ ДИАГРАММ

Г.С. Яма, Н.Г. Кеберле

Использование средств общедоступных визуальных редакторов диаграмм для построения синтаксически корректных диаграмм на произвольном проблемно-ориентированном языке диаграмм требует от пользователя экспертных знаний этого языка, что не всегда возможно. В данной работе рассматриваются решения, положенные в основу программного инструментария визуального редактора диаграмм, который позволяет создавать синтаксически корректные диаграммы при условии существования проблемно-ориентированного языка диаграмм.

Ключевые слова: *диаграмма, визуальный редактор, проблемно-ориентированный язык диаграмм.*

VISUAL DIAGRAM EDITOR FOR DECLARATIVELY DEFINED DOMAIN-SPECIFIC DIAGRAM LANGUAGES

G.S. Yama, N.G. Keberle

Usage of common visual diagram editors to build syntactically-correct diagrams for arbitrary domain-specific diagram language requires an expert level knowledge of the notation and the meaning of such language, which is not always achievable for a user.This paper is dedicated to the overview of the solutions taken during visual diagram editor software development, enabling syntactically-correct diagram creation in presence of domain-specific diagram language description.

Keywords: *diagram, visual editor, domain-specific diagram language.*