SpiderCNP - интегрирана среда за визуално програмиране чрез управляващи мрежи

Цанко Големанов

Control Network Programming (CNP) *is a style of high-level programming that is especially effective for solving problems that have natural graph-like representation of imperative, declarative, or mixed nature. The report is aimed to describe some basic features of SpiderCNP as a powerful visual Integrated Development Environmen (IDE) for CNP programs authoring, modifying, compiling, deploying and debugging.*

Key words: *Control Network Programming, CNP, SpiderCNP, IDE, AI programming.*

ВЪВЕДЕНИЕ

Реалната ситуация в софтуерната индустрия се определя от тоталната доминация на **императивната** парадигма. Много са факторите определящи водещата роля на този програмен стил, но рядко един сложен софтуерен проект се реализира единствено и само с подобни средства. Все по-често срещана е ситуацията, при която в една конвенционална софтуерна система, наред с текущите обработки, в някои случаи да се налага да се решава и по-сложна и **недетерминирана** задача. Тогава програмистът се сблъсква с нетривиалния проблем да моделира един недетерминиран алгоритъм чрез детерминираните средства, с които разполага. Алтернативният подход е тази част от проекта да се реализира с подходящ недетерминиран инструмент. В този случай обаче обикновено налице е проблем с недостатъчно-гъвкавата интеграция и комуникацията между двете части на проекта.

Програмирането чрез Управляващи Мрежи (**Control Network Programming**) е сравнително нова програмна парадигма, разработена от Костадин Крачанов, позволяваща успешното интегриране на различни програмни стилове. Базовите принципи, касаещи поведението, философията и изпълнението на такава програма, както и връзката с други програмни парадигми са описани в [1,2,3]. Езикът **SPIDER**, като пореден представител на **CNP**, комбинира и разширява възможностите на три основни програмни парадигми:

- процедурно програмиране;
- декларативното програмиране;
- системите, базиращи се на правила.

Фундаменталната част на всяка SPIDER-програма е Управляващата Мрежа (**Control Network**). **CN** по същество представлява ориентиран граф и е крайно множество подмрежи, една от които е главна. Подмрежите могат да се извикват помежду си, потенциално рекурсивно. Всяка подмрежа се състои от етикетирани възли (състояния), и свързващи ги стрелки. По всяка стрелка може да се постави последователност от "примитиви". Примитивите представляват елементарни действия, еквивалентът на които в традиционните езици за програмиране са потребителски-дефинираните функции. Изпълнението на една SPIDER-програма представлява трасировка на графа-**CN** и търсенето на път между началното и финалното състояние. Елементарните действия (примитивите) съдържат код на избран базов език (Pascal, C и др) и се изпълняват при придвижването по стрелките, свързващи отделните възли. На практика се прилага вградена стратегия за извод - разширен BACKTRACKING [4]

В началните версии на SPIDER, **CN** се задаваше само в текстов вид, чрез последователно дефиниране на всички подмрежи, възли и излизащите от тях стрелки с примитиви. Една примерна SPIDER-програма, състояща се от главна мрежа **MainNetwork** и една подмрежа **SubNetwork** е показана на фиг.1. Програмата

включва основните типове състояния (възли) и стрелки, формирани само от два потребителски примитива: **ActionPrimitive** и **ConditionPrimitive**.

```
Main MainNetwork;
        var Sel, Val, Lo, Hi : real
body
        FirstNode:
                ActionPrimitive                                         > NormNode
                ConditionPrimitive, CALL SubNetwork, ActionPrimitive    > FINISH
                                                                        > SelNode;
        NormNode:
                ActionPrimitive                                         > STOP;
        SelNode:
                SELECT BY Sel
                Val:                                                    > NormNode
                3.14:   ActionPrimitive                                 > OrdNode
                ELSE                                                    > RanNode;
        OrdNode:
                ORDER BY Sel
                1.23:                                                   > NormNode
                2.34:                                                   > STOP
                Val:    ActionPrimitive                                 > FINISH;
        RanNode:
                RANGE FROM Lo TO Hi
                6.28:   ConditionPrimitive                              > FINISH
                123:    ActionPrimitive                                 > FirstNode
                ELSE                                                    > OrdNode;
end;

Sub  SubNetwork;
body
        S1:
                ActionPrimitive, ConditionPrimitive                     > RETURN;
end;
```

Фиг. 1 – Примерна SPIDER-програма

Макар да има своите предимства (основно в компактността на записа) този метод на задаване на графа е определено неудобен от потребителска гледна точка:

- Губи се ясната и коректна визуална представа текущото състояние на CN;
- Изкисква се детайлно познаване на синтаксиса и мнемониката на текстовия формат на SPIDER-програмата;
- Затруднено е функционирането на ефективна система за настройка и тестване на SPIDER-програмата, с нагледна визуализация на постъпково обхождане на графа на CN.

Това са и основните причини определящи необходимостта от разработка на **SpiderCNP** - специализирана визуална среда за програмиране чрез управляващи мрежи.

СРЕДА ЗА ВИЗУАЛНО ПРОГРАМИРАНЕ SPIDERCNP

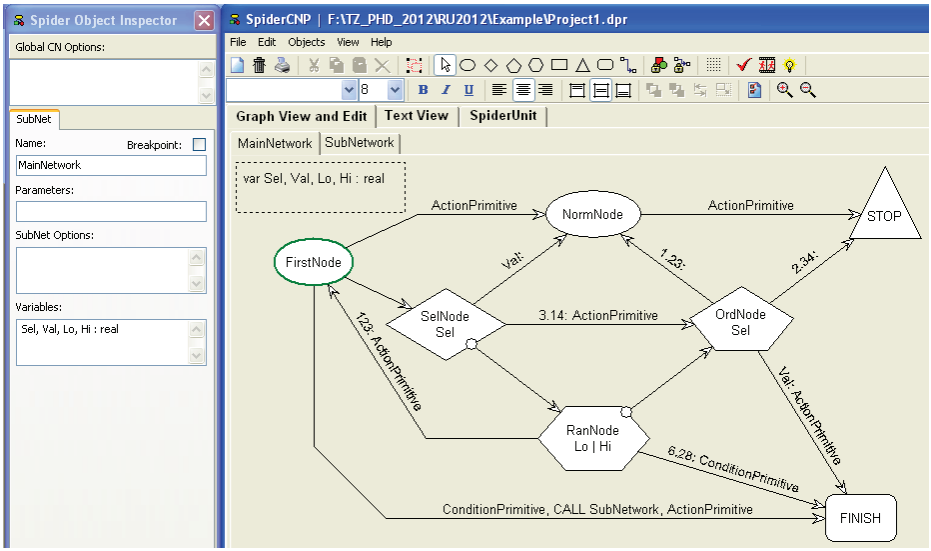
Основните изисквания към средата за визуално CN-програмиране могат да се групират в следните направления [5]:

- Наличие на удобен **визуален редактор на графи**, позволяващ задаването сложни мрежови структури с произволни нива на рекурсивна вложеност;
- Наличие на **компилатор**, с възможност за интегриране на SPIDER-програмата в проекта на базовия език и получаване на комплексно изпълнимо приложение;
- Наличие на **визуален дебъгер** с възможности за детайлен контрол на постъпковото обхождане на графа на CN.

Допълнително предимство би предоставила възможността за автономна работа извън IDE на базовия език. Това е удобна алтернатива за създаване на малки независими приложения със SPIDER-програми.

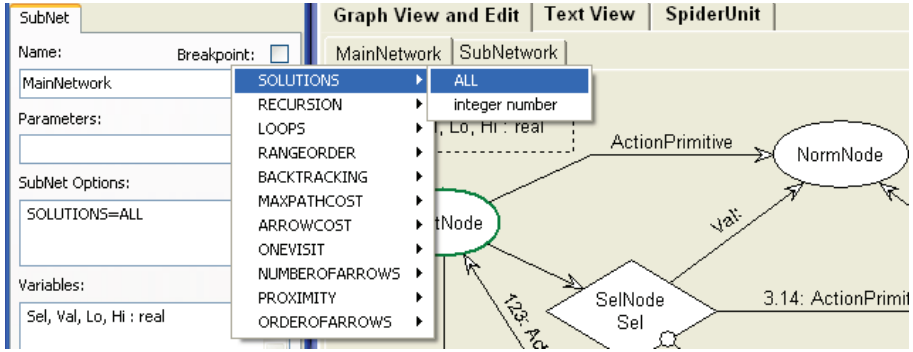
Визуален редактор на графи

На фиг.2 е демонстриран общия изглед на интегрираната среда. В основния оперативен прозорец, в таб **Graph View and Edit**, е показана примерната програма от фиг.1. За визуализиране и модифициране на всяка подмрежа се използва отделен таб (**MainNetwork**, **SubNetwork**) на редакторското поле. В таб **Text View** може да се види автоматично генерираната SPIDER-програма в текстов формат, а въвеждане и редакция на примитиви става в таб **SpiderUnit**. Добавянето на нова подмрежа и изтриването на съществуваща се извършва от най-левите бутони на първия toolbar  , а за добавяне възел и стрелка -  . Всички редакторски операции с обектите на CN са максимално улеснени и се извършват по технологията drag-and-drop. За различните типове възли (състояния) - Normal, Select, Order, Range, RETURN, STOP, FINISH - се използват отделни пиктограми, което прави графа много по-читаем, а логиката на програмата по-разбираема.



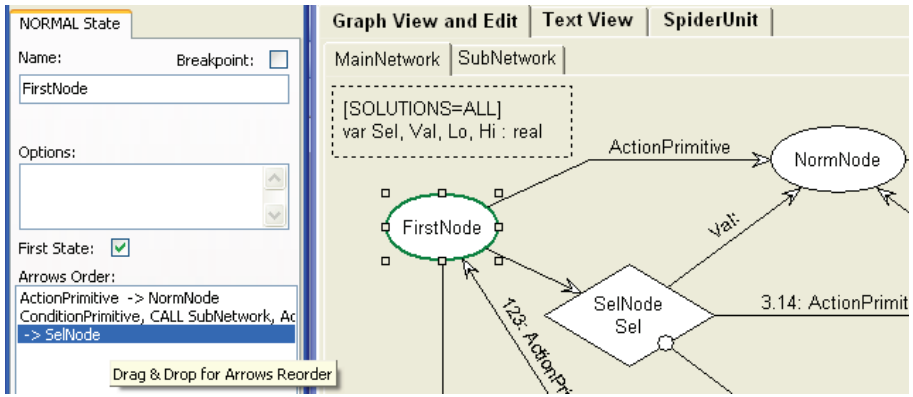
Фиг. 2 – Оперативни прозорци на SpiderCNP

Вляво се е показан втория основен работен прозорец на средата - инспектора на обекти **Spider Object Inspector**. Чрез него могат да се задават и модифицират стойностите на основните свойства на избран обект – подмрежа, състояние или стрелка. Прозорецът на инспектора автоматично променя интерфейса си, в зависимост от вида на избрания обект. В режим **SubNet** може да се редактира името на подмрежата (**Name**), параметрите, с които се извиква (**Parameters**) и локалните й променливи (**Variables**). Определянето на стойностите на системните опции (**SubNet Options**) на ниво подмрежа е улеснено чрез използването на контекстно меню. През него програмистът има достъп и до стойностите, които различните опции могат да заемат. На фиг.3. е демонстрирано задаването на стойност на системната опция за броя на търсени решения: SOLUTIONS = ALL.



Фиг. 3 – Задаване на стойност на системни опции

По аналогичен начин се извършва и редактирането на основните свойства на избрано състояние (възел). Освен стойности на различните системни опции на това ниво, тук може интерактивно да се задават специфичните параметри на управляващите състояния. На фиг.4. е показано и как в отделно поле на инспектора програмистът може да преподрежда последователността на обхождане на излизащите от състоянието стрелки (**Arrows Order**).



Фиг. 4 – Редактиране на състояние

При редактирането на избрана стрелка обработката отново е сведена до избор на съставящите я примитиви от списък с наличните (**Available Primitives**). Този списък се генерира и обновява автоматино чрез анализ на файла потребителските примитиви (**SpiderUnit**).

Компилатор на SPIDER-програми

Стартирането на вградения CNP-компилатор става чрез един от трите бутоните в края на основния toolbar  . Първият от групата дава началото на последователност от обработки, водещи до генериране на изпълним обектен файл (EXE) на целия интегриран проект. Третият бутон води до стратиране на полученото приложение, веднага след успешната му компилация. Чрез втория бутон от групата се стартира вградения визуален дебъгер на средата при необходимост от настройка и тестване на SPIDER-програмата.

Визуален дебъгер на SPIDER-програми

Визуалният дебъгер дава възможност на програмиста да проследи постъпковото изпълнение на програмата и придвижването на управлението по графа на CN. Състоянията и стрелките, през които се преминава, последователно се оцветяват в червено. В отделен toolbar, показан на фиг.5, динамично може да се управлява процеса на трасировка – скорост, спиране на контролни точки, пауза и др.

Фиг. 5 – Настройка и динамично управление на дебъгера

ЗАКЛЮЧЕНИЕ

Практическото използване интегрираната среда при обучението на студенти в Yashar University [6,7] доказа удобството на използването й при разработка на сравнително-сложни приложения. Към настоящия момент SpiderCNP може да се интегрира в две базови среди:

- Borland Delphi - всички версии до Embarcadero RAD Studio XE3 (2012)
- Lazarus 1.0 (2012), която позволява разработката на приложения с CNP освен за Windows, но и за Linux, Mac OS X, FreeBSD и други операционни системи.

ЛИТЕРАТУРА

[1] Golemanov, T., Kratchanov, K, Golemanova, E. SPIDER – A Language for Programming Through Control Networks. In: Proc. CompSysTech 2000, Sofia, June 2000, II.8-1 – II.8-5 (in Bulgarian). Also published by ACM Press, 2081-2085.

[2] Kratchanov, K., Golemanov, T., Golemanova, E., Control Network Programming, In: Proc. 6th EEE/ACIS Conf. on Computer and Information Science (ICIS 2007), July 2007, Melbourne, Australia, 1012-1018.

[3] Kratchanov, K., Golemanova, E., Golemanov, T. Control Network Programming Illustrated: Solving Problems With Inherent Graph-Like Structure, In: Proc. 7th IEEE/ACIS Int. Conf. on Computer and Information Science (ICIS 2008), May 2008, Portland, Oregon, USA, 453-459

[4] Kratchanov, K., Golemanova, E., Golemanov, T, Control Network Programs and Their Execution, In: Proc. 8th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED 2009), February 2009, Cambridge, UK, 417-422.

[5] http://desizntech.info/2010/04/10-free-and-useful-ide-for-programmers-and-developers-2

[6] Kratchanov, K., Golemanova, E., Golemanov, T. and Kulahcioglu, B.: Using Control Network Programming in Teaching Randomness. In: International Conference on Electronics, Information and Communication Engineering (EICE 2012), March 2012, Macau, China, 67-71

[7] Kratchanov, K., Golemanova, E., Golemanov, T. and Kulahcioglu, B.: Using Control Network Programming in Teaching Nondeterminism, In CompSysTech '12, June 2012, Ruse, Bulgaria, 391-398, ACM, NY, USA 2012

За контакти:

Цанко Големанов, Катедра “Компютърни системи и технологии”, Русенски университет “Ангел Кънчев”, тел.: 082-888 681, e-mail: TGolemanov@ecs.uni-ruse.bg

Докладът е рецензиран.