УДК 681.5

М.Л. МАЛИНОВСКИЙ

Харьковский национальный технический университет сельского хозяйства им. Петра Василенко, Украина

ОЦЕНКА СЛОЖНОСТИ HDL-ОПИСАНИЙ ЦИФРОВЫХ УСТРОЙСТВ

Рассмотрены принципы интегральной оценки сложности программного обеспечения (ПО). Предложены подходы к выполнению сравнительного анализа сложности табличных и текстовых описаний цифровых устройств. Предложено расширение метода Чепина оценки сложности ПО, в котором учитывается возможность использования готовых программных компонентов и получивший широкое распространение иерархический принцип описания цифровых устройств на языках HDL. Разработана методика сравнительной оценки эффективности использования альтернативных HDL-языков. В результате экспериментальных исследований установлено, что использование табличного языка THDL как альтернативы текстовым языкам описания аппаратуры позволяет снизить количество ошибок ПО в 2 – 5 раз.

Ключевые слова: *метрики Холстеда, табличный язык THDL, метод Чепина, интегральная сложность ПО, гарантоспособность ПО, HDL.*

Введение

Одним из важнейших качественных показателей ПО, который прямо или косвенно влияет на другие показатели, является сложность программы. Оценка сложности ПО позволяет выполнить обоснованный выбор языка программирования для решения данной задачи, оценить трудозатраты на его разработку, надежность и безопасность программных средств и т.д.

Универсальной методики, позволяющей дать оценку сложности ПО, на сегодняшний день не существует; результаты, получаемые с использованием различных методик, зачастую оказываются противоречивыми, поэтому интерпретация получаемых расчетных значений вызывает затруднения.

В данной статье предлагается подход, позволяющий выполнить сравнительную оценку основных показателей сложности ПО, подготовленного на различных языках описания аппаратуры.

1. Интегральная оценка сложности программного обеспечения

Общая сложность ПО может быть оценена некоторой совокупностью метрик, отражающих различные аспекты свойств программы, а интегральную относительную сложность Co можно определить как среднее арифметическое относительных значений всех рассчитанных метрик [1]:

$$\mathrm{Co} = \frac{1}{\mathrm{M}} \sum_{\mathrm{i}=1}^{\mathrm{M}} \mathrm{Cio}\,, \qquad (1)$$

где M – количество используемых метрик.

При наличии статистической информации о степени влияния значений конкретных метрик на интегральную сложность ПО последняя определяется как средневзвешенная сумма значений метрик:

$$\mathrm{Co} = \sum_{\mathrm{i}=1}^{\mathrm{M}} \mathrm{Vi} \cdot \mathrm{Cio}\,, \qquad (2)$$

где Vi – вес i-й метрики сложности, причем

$$\sum_{\mathrm{i}=1}^{\mathrm{M}} \mathrm{Vi} = 1\,. \qquad (3)$$

На сегодняшний день широко применяются три основных группы метрик оценки сложности (Cop – размера программ, Coy – сложности потока управления, Cod – сложности потока данных), поэтому справедливо следующее выражение:

$$\mathrm{Co} = \mathrm{Vp} \cdot \mathrm{Cop} + \mathrm{Vy} \cdot \mathrm{Coy} + \mathrm{Vd} \cdot \mathrm{Cod}\,, \qquad (4)$$

где Vp, Vy, Vd – весовые коэффициенты для метрик Cop, Coy, Cod.

Если предположить, что Coy = Cod = 1 (по сути это означает, что при переходе от одного HDL-языка к другому сложность потока управления и потока данных не меняются, что, как правило, соответствует реальному положению дел), получим:

$$\mathrm{Co} = \mathrm{Vp} \cdot \mathrm{Cop} + \mathrm{Vy} + \mathrm{Vd}; \qquad (5)$$

$$\mathrm{Vp} = \frac{\mathrm{Co} - 1}{\mathrm{Cop} - 1}\,; \qquad (6)$$

$$\mathrm{Co} = \mathrm{Vp} \cdot \mathrm{Cop} + 1 - \mathrm{Vp}\,. \qquad (7)$$

Интегральная сложность Co может быть найдена экспериментально для некоторых типовых задач, таким образом, представляется возможным получить диапазон значений, которые может принимать коэффициент Vp.

2 Сравнительный анализ сложности табличных и текстовых описаний цифровых устройств

В качестве примера выполним сравнительный анализ описаний счетчика на на табличном (THDL) [2] и текстовом (AHDL) языках описания аппаратуры. Будем использовать для этого метрики Холстеда, которые отражают лексический подход к измерению характеристик ПО.

Текстовое описание реверсивного счетчика с реализацией функций сброса, загрузки и разрешения счета на языке AHDL выглядит следующим образом:

```
SUBDESIGN counter
(
clk, reset, load, ena, dir, inp[3..0]    : INPUT;
out[3..0]                : OUTPUT;
)

VARIABLE
state[3..0] : DFF;

Begin
If reset
        then state[].d = 0;
elsif  load
        then state[].d = inp[];
elsif  ena and !dir
        then state[].d = state[].q + 1;
elsif  ena and dir
        then state[].d = state[].q - 1;
else state[].d = state[].q;
End If;
out[] = state[].q;
End;
```

Табличное описание аналогичного счетчика на языке THDL выглядит следующим образом (см. рис. 1).

В соответствии с известными формулами, предложенными Холстедом, выполнен расчет основных метрик, результаты которого сведены в табл. 1.

counter						
Приоритетность			V			
Текущее состояние	**Условия переходов**					**Следующее состояние**
state[3..0]	in[3..0]	ena	reset	load	dir	state[3..0]
			1			0
				1		in[]
		1			0	state[] + 1
		1			1	state[] - 1

Рис. 1 Табличное описание счетчика на языке THDL

Таблица 1

Значения метрик Холстеда

№	Метрика		AHDL	THDL	M^{AHDL}/M^{THDL}
1.	Словарь операторов	n1	29	12	2,42
2.	Общее число операторов	N1	96	31	3,1
3.	Словарь операндов	n2	9	6	1,5
4.	Общее число операндов	N2	25	10	2,5
5.	Словарь программы	n = n1 + n2	38	18	2,11
6.	Длина программы	N = N1 + N2	121	41	2,95
7.	Объем программы	$V = N \log_2 n$	635	170,97	3,71
8.	Потенциальный объем программы	$V^* = (n2^*+2)\log_2(n2^*+2)$	24	24	1
9.	Сложность программы	D = (n1 / 2)(N2 / n2)	40,278	10	4,03
10.	Уровень программы	L = 1 / D	0,0248	0,1	0,248
11.	Усилия на разработку	E = V / L	25 576	1 709,7	15
12.	Количество ошибок	B = V / 3000	0,2117	0,057	3,71
13.	Время разработки	T = E / 18	1420,9	94,982	15

С точки зрения оценки гарантоспособности (надежности и безопасности) ПО наибольший интерес представляет собой ожидаемое количество ошибок B, которое, как показывает расчет, для подобных задач может быть уменьшено в 3,71 раз.

Вместе с тем, реальное относительное значение количества ошибок $Bo = B^{AHDL}/B^{THDL}$ может существенно отличаться от того, которое получено в соответствии с методом Холстеда, поэтому для повышения точности расчетов следует принимать во внимание поправочный коэффициент Vp и пользоваться выражением (7).

Следует ожидать, что значение коэффициента Vp будет зависеть от того, насколько сложным является ПО, т.е. от абсолютного значения сложности. В связи с этим, целесообразно проведение статистических исследований с целью получения экспериментальных значений количества ошибок Boэ для типовых задач различной сложности.

При этом следует использовать для оценки абсолютного значения сложности программ не только метрики Холстеда.

3 Расширение метода Чепина для HDL-описаний

Из всего многообразия существующих методов оценки сложности ПО следует отдавать предпочтение таким, которые наиболее точно соответствуют особенностям выразительных средств HDL – языков и присущему им внутреннему параллелизму. По мнению автора, использование подходов, основанных на применении метрики Чепина, позволит дать максимально точную оценку сложности HDL – описаний, поскольку они ориентированы на оценку информационной прочности отдельно взятого программного модуля (для случая HDL – компонента цифрового устройства), а общая сложность может быть оценена как сумма полученных для каждого

компонента значений. При этом все множество переменных компонента разбивается на 4 функциональные группы:

1. Множество «P» – входные переменные.
2. Множество «M» – модифицируемые или создаваемые внутри программы (внутренние) переменные.
3. Множество «C» – переменные, участвующие в управлении работой программного модуля (управляющие переменные).
4. Множество «T» – не используемые в программе (“паразитные”) переменные.

Поскольку каждая переменная может выполнять одновременно несколько функций, необходимо учитывать ее в каждой соответствующей функциональной группе.

Значение метрики Чепина рассчитывается по формуле:

$$Q = a_1P + a_2M + a_3C + a_4T, \quad (8)$$

где a_1, a_2, a_3, a_4 – весовые коэффициенты.

Весовые коэффициенты использованы для отражения различного влияния на сложность программы каждой функциональной группы. Наибольший вес, равный трем, имеет функциональная группа C, так как она влияет на поток управления программы. Весовые коэффициенты остальных групп распределяются следующим образом: a1 = 1; a2 = 2; a4 = 0,5. Весовой коэффициент группы T не равен нулю, поскольку “паразитные” переменные не увеличивают сложности потока данных программы, но иногда затрудняют ее понимание. С учетом весовых коэффициентов выражение примет вид:

$$Q = P + 2M + 3C + 0,5T. \quad (9)$$

Метод Чепина для HDL-языков позволяет эффективно оценивать сложность одного отдельно взятого компонента программы, но не учитывает возможности использования готовых программных компонентов и получивший широкое распространение иерархический принцип описания цифровых устройств.

Предлагается усовершенствованный метод оценки сложности ПО, представляющий собой расширение метода Чепина с учетом особенностей HDL-языков, который сводится к следующей процедуре.

1. HDL – проект представляется в виде трехуровневого иерархического описания: на нижнем уровне располагаются базовые компоненты: комбинационные схемы, триггеры, триггерные устройства (регистры, счетчики), а также устройства, разработанные и отлаженные ранее, алгоритм функционирования которых известен; на среднем уровне располагаются компоненты, реализованные на основе базовых; на верхнем уровне располагается файл верхнего уровня в иерархии описаний.

В общем случае средних уровней может быть несколько, а для простых проектов верхний и средний уровни могут отсутствовать.

2. Рассчитывается сложность компонентов нижнего уровня Q_l по формуле

$$Q_l = \sum_{i=1}^{n} Q_{li}, \quad (10)$$

где n – количество уникальных компонентов нижнего уровня, Q_{li} – сложность i – го компонента.

3. Рассчитывается сложность связей внутри компонентов среднего уровня Q_m:

$$Q_m = \sum_{i=1}^{k} Q_{mi}, \quad (11)$$

где k - количество уникальных компонентов среднего уровня, Q_{mi} – сложность i – го компонента.

4. Рассчитывается сложность связей на верхнем уровне иерархии Q_h.

5. Общая сложность проекта Q составляет сумму сложностей на всех уровнях:

$$Q = Q_l + Q_m + Q_h. \quad (12)$$

4. Прогнозная оценка эффективности использования альтернативных языков

Как было показано ранее, оценка сложности ПО по абсолютному значению для типовых задач позволит установить зависимость коэффициента Vp и количества ошибок B от сложности Q, оцененной в соответствии с расширением метода Чепина.

Другими словами, экспериментально могут быть получены зависимости Vp = f1(Q) и Boэ = f2(Q), которые позволят прогнозировать эффективность использования альтернативных языков с учетом сложности ПО.

Таким образом, для получения зависимостей Vp = f1(Q) и Boэ = f2(Q) необходимо использовать следующую методику:

1. Выбрать n типовых задач с разным уровнем сложности.
2. Рассчитать сложность выбранных задач Q = {Q1, Q2, … Qn} в соответствии с предложенным расширением метрики Чепина.
3. Рассчитать прогнозируемое количество ошибок для выбранных задач B = {B1, B2, … Bn} в соответствии с метрикой Холстеда при разработке

HDL-описаний на языках AHDL ($B^A = \{B^A1, B^A2, \dots B^An\}$) и THDL ($B^T = \{B^T1, B^T2, \dots B^Tn\}$).

4. Получить по результатам статистических исследований экспериментальные значения количества ошибок при разработке HDL-описаний на языках AHDL ($B^A_э = \{B^A_э1, B^A_э2, \dots B^A_эn\}$) и THDL ($B^T_э = \{B^T_э1, B^T_э2, \dots B^T_эn\}$).

5. Рассчитать относительные значения метрики $Boi = B^Ai / B^Ti$.

6. Рассчитать относительные значения метрики $Bo_эi = B^A_эi / B^T_эi$.

7. Рассчитать значения коэффициентов Vpi = = (Boi – 1)/($Bo_эi$ – 1).

8. Построить зависимости Vp = f1(Q) и Boэ = f2(Q).

В результате экспериментальных исследований и расчетов, проведенных с участием студентов и преподавателей кафедры автоматизации и компьютерных технологий Харьковского национального университета сельского хозяйства, были получены значения качественных показателей для некоторых типовых задач, которые отображены в табл. 2.

Таблица 2

Результаты экспериментальных исследований и расчетов качественных показателей ПО для некоторых типовых задач

Название устройства	Расчетное количество ошибок для программы, написанной на THDL	Расчетное количество ошибок для программы, написанной на AHDL	Относительное значение метрики количества ошибок (расчетное)	Относительное значение метрики количества ошибок (экспериментальное)	Весовой коэффициент	Сложность программы, рассчитанная согласно расширению метода Чепина
	Bp^T	Bp^A	$Bop = Bp^A/Bp^T$	Boэ	Vp = (Boэ-1)/(Bop-1)	Q
Элементарные логические операции	0,009358	0,046727	4,993375	4,3	0,83	9,00
Приоритетный шифратор	0,014793	0,074913	5,064084	4,5	0,86	12,00
Мультиплексор	0,039207	0,14899	3,800087	3,1	0,75	16,00
Шифратор	0,064	0,14415	2,252344	2	0,80	24,00
Счетчик	0,057	0,2117	3,714035	2,9	0,70	42,00
Генератор ШИМ	0,093333	0,25715	2,755188	2	0,57	49,00
Модуль многократного контроля	0,08141	0,57338	7,043115	5,1	0,68	70,00
Генератор АРС	0,116	0,5488	4,731034	3,3	0,62	124,00
Генератор РЦ	0,4823	2,3356	4,842629	3,4	0,62	218,00
Трехфазный генератор	1,175	3,517	2,993191	2,3	0,65	281,00

График зависимости Boэ = f2(Q) приведен на рис. 2. Как показывают экспериментальные исследования, значение Boэ может колебаться в пределах 2 – 5, при этом закономерности в изменении Boэ в зависимости от значения сложности Q не наблюдается.

Анализ графика зависимости Vp = f1(Q) (см. рис. 3) показывает, что коэффициент Vp изменяется в диапазоне 0,57 – 0,86, причем он имеет тенденцию снижаться при росте значения сложности Q. При Q > 40 диапазон изменений коэффициента Vp уменьшается и ограничивается значениями 0,57 – 0,7.

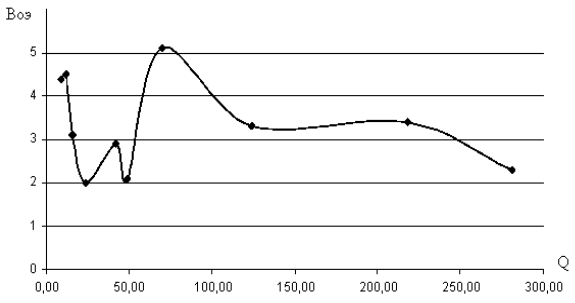

Рис. 2. Зависимость количества ошибок в программе Boэ от сложности Q

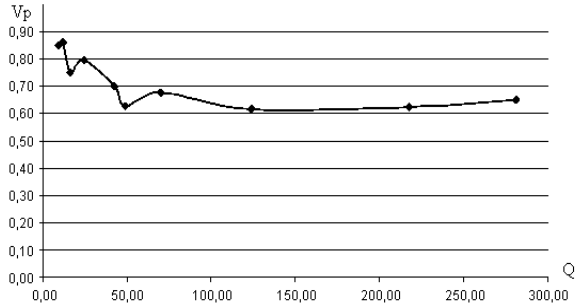

Рис. 3. Зависимость весового коэффициента Vp от сложности Q

Выводы

Таким образом, для оценки снижения вероятности появления ошибок в программе при использовании языка THDL как альтернативы известным текстовым языкам описания аппаратуры, целесообразно использовать коэффициент Vp.

При этом необходимо выполнить расчет метрики сложности Q в соответствии с предложенным выше расширением метода Чепина и метрики количества ошибок (относительного значения) в соответствии с методом Холстеда.

Если сложность программы Q < 40, то диапазон значений Vp должен составлять 0,7 – 0,86; если Q > 40, то диапазон значений Vp должен составлять 0,57 – 0,7. Относительное значение метрики Bo, оценивающей, во сколько раз уменьшится количество ошибок, рассчитывается по формуле (7).

Как показывают экспериментальные исследования, использование табличного языка описания аппаратуры THDL в качестве альтернативы известным текстовым языкам описания аппаратуры (AHDL, VHDL, VerilogHDL) позволяет снизить количество ошибок ПО в 2 – 5 раз.

Литература

1. Бахтизин В. В. Применение метрик сложности при разработке программных средств [Электронный ресурс] / Бахтизин В.В. , Глухова Л.А. – Режим доступа к ресурсу: www.giac.unibel.by /docs/pdf/1-2005/s10-1-2005.pdf.

2. Малиновский М.Л. Управление объектами критического применения на основе ПЛИС: Монография / М.Л. Малиновский. – Х.: Факт, 2008. – 224 с. – ISBN 978-966-637-641-4.

Поступила в редакцию 14.01.2009

Рецензент: д-р техн. наук, проф., зав. кафедрой компьютерных систем и сетей В.С. Харченко, Национальный аэрокосмический университет "ХАИ", Харьков.

ОЦІНКА СКЛАДНОСТІ HDL-ОПИСІВ ЦИФРОВИХ ПРИСТРОЇВ

М.Л. Малиновський

Розглянуто принципи інтегральної оцінки складності ПЗ. Запропоновано підходи до виконання порівняльного аналізу складності табличних і текстових описів цифрових пристроїв. Запропоновано розширення методу Чепина оцінки складності ПЗ, у якому враховується можливість використання готових програмних компонентів і ієрархічний принцип опису цифрових пристроїв на мовах HDL, що одержав широке поширення. Розроблена методика порівняльної оцінки ефективності використання альтернативних HDL-мов. У результаті експериментальних досліджень встановлено, що використання табличної мови THDL як альтернативи текстовим мовам опису апаратури дозволяє знизити кількість помилок ПЗ в 2 – 5 разів.

Ключові слова: метрики Холстеда, таблична мова THDL, метод Чепина, інтегральна складність ПЗ, гарантоздатність ПЗ, HDL.

ESTIMATION OF COMPLEXITY OF THE DIGITAL DEVICES HDL-DESCRIPTIONS

M.L. Malinovsky

The principles of an integrated estimation of software complexity are considered. The approaches to performance of the comparative analysis of the tabulared and textual descriptions complexity of digital devices are offered. The expansion of a Chepin's method of complexity estimation of software is offered, in which the description of digital devices in HDL-languages is taken into account. The technique of a comparative estimation of efficiency of alternative HDL-languages use is developed. Experimental researches are established, that use of tabulared language THDL as alternative to textual languages allows to lower quantity of mistakes in software in 2 – 5 times.

Key words: the metrics of Holsted, tabulared language THDL, Chepin's method, integrated complexity of software, dependability of software, HDL.

Малиновский Михаил Леонидович – канд. техн. наук, доцент, докторант, Харьковский национальный технический университет сельского хозяйства имени Петра Василенко, Харьков, Украина, e-mail: w8l8w@mail.ru.