УДК 681.324.067

ПРИМЕНЕНИЕ СТАТИЧЕСКОГО АНАЛИЗА ДЛЯ ОЦЕНКИ КРИТИЧЕСКОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В.В. Скляр
(Государственный НТЦ ядерной и радиационной безопасности, Харьков)

Выполнен анализ возможностей инструментальных средств для статического анализа программного кода и проведен обзор результатов применения таких инструментальных средств для оценки безопасности программного обеспечения критического применения.

статистический анализ, критическое программное обеспечение

Постановка проблемы. С развитием информационных технологий все больше функций контроля и управления стало возлагаться на компьютерные системы, в том числе и на компьютерные системы технических комплексов критического использования (ТККИ). Увеличение сложности и объема программного обеспечения (ПО) компьютерных систем приводит к повышению вероятности наличия в ПО невыявленных дефектов.

Статическим анализом программного кода (САПК) называется процесс рассмотрения программных листингов без непосредственного выполнения программы, в ходе которого определяются различного рода параметры программного кода, позволяющие продемонстрировать соответствие кода установленным требованиям либо обнаружить потенциальные проблемы, связанные с проявлением дефектов ПО [1-3].

Процедуры САПК дополняют процедуры автономного тестирования программных модулей, поскольку позволяют изучить свойства ПО, которые проявляются при любых входных данных. САПК позволяет обнаружить дефекты ПО, не проявляющиеся во время тестирования или существующие в вариантах выполнения программы, которые не проверяются при тестировании. Имеются данные об устранении до 60% программных дефектов путем САПК [3]. Положительные результаты САПК позволяют сделать выводы о соответствии разработанного программного кода проекту ПО и требованиям к ПО, а также о возможности перехода к этапу интеграции программных модулей. Таким образом, САПК является существенной составляющей процесса верификации, что подчеркивается в стандартах, содержащих требования к ПО ТККИ для различных отраслей [4].

Анализ литературы. Следует отметить, что в отечественной и рус-

скоязычной литературе отсутствует удовлетворительный обзор методик выполнения САПК, а также программных инструментальных средств (ИС) для САПК. Кроме того, в Украине практически не проводятся исследования, направленные на получения новых теоретических результатов в области САПК, а также не ведется разработка соответствующих инструментальных средств. Что касается зарубежных публикаций, то большинство из них посвящены отдельным методикам САПК. Среди немногих обзорных работ следует отметить [1, 2, 5]. Наиболее полные и актуальные обзоры методик САПК даны в стандартах, регламентирующих процессы разработки и верификации ПО для различных типов ТККИ: ***для АЭ***С – в стандарте МАГАТЭ (Международного агентства атомной энергии) NS-G-1.1 (2000) «Программное обеспечение для компьютерных систем, важных для безопасности АЭС. Руководство по безопасности»; ***для ракетно-космической техники*** – в стандарте ECSS (European Cooperation for Space Standardization – Европейской кооперации по космической стандартизации) ECSS-Q-80-03 (2004) «Обеспечение качества космических систем – Методики оценки надежности и безопасности программного обеспечения»; ***для авиационной техники*** – в стандарте RTCA (Radio Technical Commission for Aeronautics – Радиотехнической комиссии по аэронавтике) DO-178B (1992) «Рассмотрение программного обеспечения при сертификации бортовых систем и оборудования»; ***для систем, выполняющих функции безопасности***, безотносительно к области применения – в стандарте МЭК (Международной электротехнической комиссии) МЭК 61518-7 (2000) «Функциональная безопасность электрических, электронных и программируемых систем, важных для безопасности. Часть 7: Обзор методик и измерений».

В результате появления стандартов, содержащих детальные требования к проведению САПК, в странах Западной Европы, США, Канаде и др. выполнение САПК является обязательной процедурой для ПО ТККИ. В качестве примеров можно привести использование САПК для верификации ПО комплекса информационных и управляющих систем в составе АСУ ТП АЭС Темелин (Чехия) [6], а также бортовых компьютерных систем боевых самолетов Jaguar, Hercules, Typhoon, аэробусов A340, A380 и др. [7].

Целью статьи является анализ возможностей инструментальных средств для статического анализа программного кода и обзор результатов применения таких инструментальных средств для оценки безопасности программного обеспечения критического применения.

Методики статического анализа. Ниже приведено краткое описание методик (видов) САПК. Поскольку все термины являются англоязычными, в скобках указаны английские оригиналы названий.

1. В ходе анализа потока управления (Control Flow Analysis) [8, 9] вы-

полняется проверка структуры программы на предмет последовательности выполнения кода; структурированности кода; наличия участков невыполняемого кода; определения входных и выходных точек процедур и циклов.

2. В ходе анализа потока данных (Data Flow Analysis) [8, 9] выполняется проверка используемых в программе переменных на предмет отсутствия переменных, для которых не установлены значения; проверки доступа к локальным и глобальным переменным; использования переменных (отсутствия неиспользуемых переменных); использования формальных параметров.

3. В ходе анализа информационного потока (Information Flow Analysis) [9] выполняется анализ управляющего графа программы; анализ операций, выполняемых с переменными для каждого из участков управляющего графа программы; анализ зависимости выходных переменных программных процедур от входных переменных; сопоставление полученных зависимостей с программной спецификацией и анализ обнаруженных отклонений.

4. В ходе оценки сложности (Complexity Measurement) [10, 11] выполняется определение атрибутов программного кода, необходимых для определения метрик сложности; расчет метрик сложности; анализ свойств ПО, исходя из полученных значений метрик сложности [12].

5. В ходе проверки на соответствие стандартам программирования (Programming Standards Verification) [13] выполняется проверка программного кода на предмет обнаружения заданных отклонений от правил программирования, в том числе, от правил, регламентированных стандартами.

Общая характеристика инструментального средства LDRA Testbed. Рассмотрим возможности ИС для САПК на примере программного продукта LDRA Testbed [14]. Разработчиком Testbed является LDRA Ltd (Ливерпульская ассоциация исследований данных). LDRA основана в Ливерпульском университете в 1975 году.

Целью использования LDRA Testbed является проведение статического и динамического анализа программного обеспечения, что позволяет максимально увеличить глубину оценки ПО, благодаря простоте в использовании и автоматизации. Поэтому LDRA Testbed включает две основные части: ***модуль статического анализа*** для анализа кода программного обеспечения; ***модуль динамического анализа*** для анализа полноты тестирования.

Модуль статического анализа LDRA Testbed обеспечивает выполнение следующих процедур: основной статический анализ; анализ сложности; статический анализ потока данных; анализ перекрестных ссылок; анализ информационного потока. LDRA Testbed позволяет выполнять анализа ПО, разработанного на следующих языках программирования: C; C++; C#; Ada83; Ada95; Java; Visual Basic Intel®; Assemblers Motorola®; Assemblers

Texas Instruments®; Assemblers PowerPC®; Assemblers; Cobol; Coral66; Fortran; Pascal; PL/Mx86; PL/1; Algol. Полный набор функций анализа доступен только для C/C++. LDRA Testbed может выполняться на платформах Windows и UNIX. В качестве входных данных LDRA Testbed используются файлы, содержащие исходный программный код. Этот код должен соответствовать заданному стандарту языка программирования, для которого сконфигурирован LDRA Testbed, кроме того, программный код не должен содержать синтаксические ошибки. Выходные данные LDRA Testbed включают текстовые и графические результаты.

Текстовые результаты включают различные типы отчетов, формируемые по результатам выполнения соответствующих процедур анализа. Основными отчетами, анализируемыми при оценке ПО, являются такие: ***отчеты о качестве*** (Quality Reports) – результаты выполнения основного статического анализа; ***отчеты о метриках*** (Metrics Reports) – результаты выполнения анализа сложности; ***отчеты об анализе статического потока данных*** (Static Data Flow Analysis Reports) – результаты выполнения анализа потока статических данных; ***отчеты о перекрестных ссылках*** (Cross Reference Reports) – результаты выполнения анализа перекрестных ссылок; ***отчеты об анализе информационного потока*** (Information Flow Analysis Reports) – результаты выполнения анализа информационного потока; ***отчеты о планировании тестовых примеров*** по критерию MC/DC (Modified Condition Decision Coverage – покрытие модифицированных условий и решений) (MC/DC Test Case Planner Reports) – результаты выполнения соответствующей части динамического анализа.

Графические результаты включают следующие: гистограммы, отображающие значения всех основных метрик, получаемых по результатам статического анализа; радиальные Kiviat-диаграммы метрик, входящих в модель качества; статические графы потока управления; статические графы вызовов программных процедур.

Пример выполнения статического анализа с использованием инструментального средства LDRA Testbed. Применение ИС для САПК LDRA Testbed рассмотрим на примере оценки ПО управляющей системы безопасности (УСБ) ядерного реактора ВВЭР-1000 [15]. В настоящее время данная система эксплуатируется на энергоблоке №1 Южно-Украинской АЭС.

УСБ предназначена для инициирования действий систем безопасности энергоблока путем выполнения функций технологических защит и блокировок, дистанционного управления и автоматического регулирования в проектных режимах работы энергоблока и в режимах проектных аварий.

В табл. 1 указан объем выполняемых оценок ПО УСБ с учетом оцениваемых свойств, критериев оценки и выполняемого анализа в среде

LDRA Testbed. Далее приведены результаты выполнения анализа сложности ПО УСБ.

Таблица 1

Объем выполняемой оценки ПО УСБ с использованием LDRA Testbed

Оцениваемое свойство ПО	Критерий оценки	Выполняемый анализ в среде LDRA Testbed
Соответствие ПО требованием стандартов по программированию	Отсутствие отклонений от правил программирования	Основной статический анализ (Main Static Analysis)
Сложность ПО	Нахождение численных значений метрик ПО в допустимых интервалах	Анализ сложности (Complexity Analysis)
Структура потока данных	Отсутствие аномалий потока данных	Анализ статического потока данных (Static Data Flow Analysis); анализ перекрестных ссылок (Cross Reference Analysis); анализ информационного потока (Information Flow Analysis)
Полнота тестирования	100% покрытие компонентов программного кода для MC/DC (Modified Condition Decision Coverage – покрытие модифицированных условий и решений) *)	Динамический анализ (Dynamic Analysis) в части касающейся анализа планирования тестовых примеров по критерию MC/DC

*) $MC/DC = \frac{\text{кол-во Булевых операндов, независимо влияющих на результат}}{\text{общее количество Булевых операндов}}$

Результаты выполнения процедуры анализа сложности в среде LDRA Testbed и экспертная оценка результатов. Анализ сложности основывается на вычислении и анализе метрик ПО. Результаты выполнения процедуры анализа сложности содержатся в отчетах о метриках, которые содержит информацию о значениях рассчитываемых метрик ПО.

При определении значений метрик ПО используется модель качества. Следует отметить, что модель качества ПО, реализованная в LDRA Testbed существенно отличается от модели качества ПО, описанной в стандартах серии ISO/IEC 9126 [12]. В LDRA Testbed оцениваются следующие составляющие качества: качество в целом (Common); понятность (Clarity); сопровождаемость (Maintainability); тестируемость (Testability).

Некоторые метрики являются комплексными, то есть могут применяться

для одновременной оценки нескольких характеристик качества ПО. Метрики, используемые для оценки составляющих качества ПО отображаются на соответствующих радиальных Kiviat-диаграммах. На рис. 1 приведен пример общей (Common) Kiviat-диаграммы, полученной с помощью LDRA Testbed в процессе анализа ПО УСБ для интерфейсного программного модуля asmi.c.

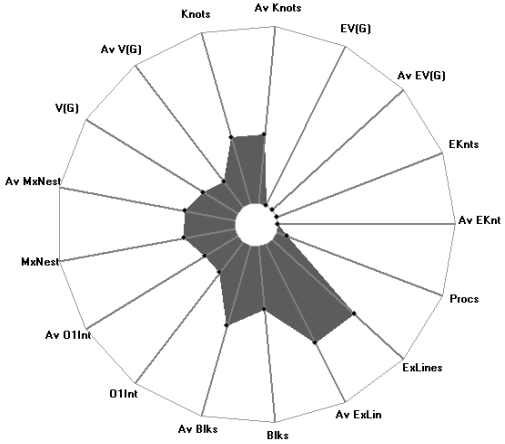

Рис. 1. Общая Kiviat-диаграмма ПМ asmi.c

Для каждой из Kiviat-диаграмм в LDRA Testbed генерируется таблица с численными значениями метрик ПО (табл. 2). Табл. 2 содержит следующие поля: наименование оси Kiviat-диаграммы, по которой отложено значение соответствующей метрики; наименование метрики; начальное значение оси (НЗО) для данной метрики; конечное значение оси (КЗО) для данной метрики; вычисленное значение данной метрики (ЗМ).

Таблица 2

Значения метрик для общей Kiviat-диаграммы программного модуля asmi.c

Наименование оси	Наименование метрики	НЗО	КЗО	ЗМ
Av EKnt	Среднее кол-во существенных узлов в проц.	0	2	0
EKnts	Общее кол-во существенных узлов в прогр.	0	28	0
Av EV(G)	Среднее значение существенной цикломатической сложности в процедуре	1	3	1
EV(G)	Общее значение существенной цикломатической сложности в программе	1	29	1
Av Knots	Среднее кол-во узлов в процедуре	0	5	1,93
Knots	Общее кол-во узлов в программе	0	70	27
Av V(G)	Среднее знач. цикломатич. сложности в проц.	1	10	2,57
V(G)	Общее знач. цикломатич. сложности в прогр.	1	100	23
Av MxNest	Среднее знач. макс. интервала вложения в проц.	1	3	1,57
MxNest	Общее знач. макс. интервала вложения в прогр.	14	42	22
Av O1Int	Среднее знач. интервалов 1 порядка в процедуре	1	5	1,86
O1Int	Общее знач. интервалов 1 порядка в программе	14	70	26
Av Blks	Среднее кол-во базовых блоков в процедуре	1	10	5,21
Blks	Общее кол-во базовых блоков в программе	1	200	73
Av ExLin	Среднее кол-во строк кода в процедуре	1	50	31,57

Наименование оси	Наименование метрики	НЗО	КЗО	ЗМ
ExLines	Кол-во строк исполнимого кода в программе	14	700	442
Procs	Кол-во процедур в программе	1	200	14

Начальные и конечные значения осей на Kiviat-диаграммах одновременно являются границами диапазонов допустимых значений метрик. Для программного модуля asmi.c, результаты вычислений метрик которого приведена на рис. 1 и в табл. 2, проведенная оценка сложности позволила сделан вывод, что все значения метрик соответствуют допустимым диапазонам, поэтому качество ПО в целом соответствует установленным требованиям. Таким образом, метрики ПО соответствуют допустимым значениям, а сложность программного кода не превышает допустимый уровень.

Выводы. Важнейшим фактором внедрения процедур САПК в отечественную практику является накопление опыта применения соответствующих инструментальных средств. Применение САПК, как одного из дополнительных способов обеспечения и подтверждения безопасности ПО для технических комплексов критического использования, является необходимым условием для обеспечения соответствия такого ПО требованиям международных стандартов [6, 7]. Статический анализ дополняет традиционные процедуры тестирования, поскольку помогает определять все варианты выполнения программы, и поэтому позволяет искать ошибки вне зависимости от входных данных. Благодаря САПК можно находить ошибки, не проявляющиеся во время тестирования или существующие в вариантах выполнения программы, которые не проверяются при тестировании [16]. Таким образом, использование САПК позволяет повысить конкурентоспособность ПО, выполняющего функции, важные для безопасности ТККИ [17, 18].

Актуальной задачей внедрения САПК в практику отечественной программной инженерии является разработка методик интерпретации результатов статического анализа для оценки надежности, безопасности и качества ПО. Данный вопрос не освещен в руководствах по применению существующих ИС и решается каждым пользователем заново. Однако, автоматизация процедур интерпретации результатов САПК представляется естественным развитием ИС и ждет своего решения.

ЛИТЕРАТУРА

1. *Lyu M.R. Handbook of Software Reliability Engineering.– McGraw-Hill Company, 1996. – 805 p.*
2. *Leveson N. Safeware: System Safety and Computers. – Addison-Wesley, 1995. – 431 p.*
3. *German A. Software Statistic Analysis: Lessons Learnt. 9th Safety-Critical Club*

Symposium. – Bristol, UK, 2001. – 12 p.

4. *Смит Д., Симпсон К. Функциональная безопасность. Простое руководство по применению стандарта МЭК 61508 и связанных с ним стандартов. – М.: Технологии, 2004. – 208 с.*
5. *Barnes J. High Integrity Ada – The SPARK approach to safety and security. – Addison-Wesley, 2003. – 207 p.*
6. *Piroutek Z., Roubal S., Rubek J. Static Analysis of the Software Used in Safety Critical System on the NPP Temelin // Proceeding CNRA/CSNI Workshop on Licensing and Operating Experience of Computer-based I&C System. – Hluboka, Czech republic, 2001. – Vol. 2. – P. 91 – 98.*
7. *Harrison K. Static Code Analysis on the C-130J Hercules Safety-Critical Software. – Springer, 1999. – 21 p.*
8. *Scott J., Lawrence J. Testing Existing Software for Safety Related Applications. – USA, Lawrence Livermore National Laboratory, 1994. – 67 p.*
9. *Beizer B. Software Testing Techniques. – Van Nostrand Reinhold, 1990. – 592 p.*
10. *Холстед М.Х. Начала науки о программах. – М.: Финансы и статистика, 1981. – 128 с.*
11. *McCabe T. A Complexity Measure // IEEE Transactions on Software Engineering. – 1976, 4, n SE-2. – P. 308 – 320.*
12. *Харченко В.С., Скляр В.В., Тарасюк О.М. Методы моделирования и оценки качества и надежности программного обеспечения. – Х.: НАКУ «ХАИ», 2004. – 159 с.*
13. *MISRA-C:2004. Guideline for the Use of the C Language in Critical Systems. – MIRA Ltd, 2004. – 102 p.*
14. *LDRA Testbed. Manual. C/C++ 7.x (Windows (95\NT)). – Manual Revision 22. LDRA Ltd, 2002. – 860 p.*
15. *TACIS Project U3.02/00 (UK/TS/25). Improvement of scientific and technical support to the nuclear and radiation safety regulation in Ukraine by developing the infrastructure of SSTC NRS and its subsidiaries, including enhancement of training capabilities. Subtask 3.1. Safety review of Control Safety System of NPP South-Ukrainian-1 with use of software tools suite LDRA Testbed. – RISKAUDIT, France, 2005. – 50 p.*
16. *Larus J., Ball T., Das M. et al. Righting Software // IEEE Software. – 2004. – 30, n 3. – P. 314 – 328.*
17. *Ястребенецкий М.А., Васильченко В.Н., Виноградская С.В. и др. Безопасность атомных станций: Информационные и управляющие системы. – К.: Техніка, 2004. – 472 с.*
18. *Ястребенецкий М.А., Васильченко В.Н. Новым энергоблокам АЭС Украины – новые информационные и управляющие системы. // Ядерная и радиационная безопасность. – 2004. – 7, № 4. – С. 5 – 12.*

Поступила 2.11.2005

Рецензент: доктор технических наук, профессор В.С. Харченко,
Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ».