УДК 681.34

МОДЕЛЬ ФОРМИРОВАНИЯ ОПТИМАЛЬНОЙ ПРОГРАММНОЙ СИСТЕМЫ ПО СХЕМЕ БЛОКА ВОССТАНОВЛЕНИЯ С СОГЛАСОВАНИЕМ

Завьялова О. И., Гриценко С. Н., Тынченко С. В., Царев Р. Ю.

ФГАОУ ВПО «Сибирский федеральный университет», Красноярск, Россия (660074, Красноярск, ул. Академика Киренского, 26, УЛК), e-mail: tsarev.sfu@mail.ru

Введение программной избыточности в структуру программной системы позволяет существенно повысить надежность системы за счет увеличения количества программных компонент, которые вернут результат и обеспечат выполнение программной системой ее целевой функции, даже, несмотря на отказ отдельных компонентов. В статье предложен подход к формированию структуры программной системы, который объединяет достоинства двух широко известных методик создания избыточного программного обеспечения – мультиверсионного программирования и блоков восстановления. Использование программной избыточности с целью повышения уровня надежности программной системы требует дополнительных ресурсов на создание системы. В статье предложена модель формирования оптимальной по составу программной системы, в которой используются как готовые программные компоненты, так и компоненты собственной разработки. В модели учтены особенности избыточной структуры программной системы.

Ключевые слова: программная система, блок восстановления с согласованием, надежность, оптимизация.

OPTIMIZATION MODEL FOR SYNTHESIS OF A SOFTWARE SYSTEM BASED ON THE CONSENSUS RECOVERY BLOCK SCHEME

Zavyalova O. I., Gritsenko S. N., Tynchenko S. V., Tsarev R. Y.

Siberian federal university, Krasnoyarsk, Russia (660074, Krasnoyarsk, street A. Kirenskogo, 26, ULK), e-mail: tsarev.sfu@mail.ru

Introduction of software redundancy into the structure of a software system can significantly improve system reliability by increasing the number of software components providing the results and allows implementation of objective functions by the software system, even in case of failure of individual components. This paper proposes an approach to synthesis of the software system structure, which combines advantages of two well-known methods of redundant software construction such as multiversion programming and recovery blocks. The use of software redundancy for improving the reliability of software systems requires additional resources to build the software system. In this paper the optimization model of the software system synthesis is proposed. For the synthesis both commercial off-the-shelf software components and the components of its own design can be applied. The model takes into account the particularities of redundancy of the software system.

Keywords: software system, consensus recovery block, reliability, optimization.

На сегодняшний день разработано множество методов и средств оценки и повышения надежности программного обеспечения. Существуют несколько подходов к обеспечению необходимого уровня надежности программ, основанных на избыточности: временной, информационной, программной. В свою очередь, основными подходами при обеспечении заданного уровня надежности за счет программной избыточности являются мультиверсионное программирование и блоки восстановления [1, 3].

При обоих подходах для решения критичной задачи разрабатывается или приобретается у сторонних разработчиков несколько функционально эквивалентных программных компонентов. Во втором случае они именуются *COTS*-компонентами (от англ. commercial off-the-shelf – коробочный программный продукт).

Фактически при реализации программного обеспечения по схеме блоков восстановления используются основные идеи метода контрольных точек и рестарта для компонент программного обеспечения таким образом, что в случае ошибки или отказа одного программного компонента исполняется другой функционально эквивалентный ему программный компонент [9]. В рамках схемы блоков восстановления такие программные компоненты называются альтернативами. Оценка корректности результата выполняется в контрольной точке посредством приемочного теста [8]. Именно здесь принимается решение о целесообразности выполнения другой альтернативы в случае ошибки первой.

Мультиверсионное (или N-вариантное программирование) предполагает одновременное выполнение всего набора функционально эквивалентных программных компонентов [4, 10]. В рамках данного подхода программные компоненты называются версиями. Результаты, возвращенные версиями, оцениваются в контрольной точке, например, посредством голосования абсолютным большинством [6]. Именно здесь определяется корректный результат.

Для обоих подходов справедливо предположение, что потенциальные ошибки могут возникнуть в разных точках исполнения программных компонент. Основой данного предположения служит то, что, и альтернативы, и версии реализуют различные алгоритмы и методы решения одних и тех же задач. Это гарантирует, что ошибки отдельных программных компонент, не выявленные на этапе тестирования, не приведут к отказу всей программной системы, даже в случае отказа отдельных ее компонент [4].

Очевидно, что оба подхода, основываясь на программной избыточности, требуют дополнительных ресурсов на разработку и реализацию программного обеспечения [2, 5]. Это приводит к необходимости решения задачи выбора оптимального состава избыточного программного обеспечения, при котором было бы гарантирован максимальный уровень надежности, с одной стороны, учтены существующие ограничения на ресурсы, с другой стороны.

Схема блока восстановления с согласованием

Рассмотрим подход, комбинирующий мультиверсионное программирование и схему блоков восстановления – схему блока восстановления с согласованием. Данная схема объединяет достоинства двух исходных подходов с целью повышения надежности программного обеспечения.

Предположим, что необходимо выбрать оптимальный набор программных компонент, т. е. оптимальный состав программной системы, которая выполняет ряд функций. Выполнение определенной функции программной системы осуществляется модулями данной системы. Обозначим архитектуру программной системы как S, а количество функций, которые должна

выполнять программная система, как L, тогда s_l – множество модулей, необходимых для выполнения функции l, $s_l \in S$, $l \in L$. Обозначим частоту использования функции l как f_l.

Согласно схеме блоков восстановления каждый из n модулей программной системы состоит из m_i альтернатив, $i = 1, 2, \ldots, n$. Однако в схеме блока восстановления с согласованием каждая альтернативу будем рассматривать как мультиверсионную, т. е. каждая альтернатива будет представлена набором версий. Обозначим количество версий, реализующих альтернативу j модуля i, как V_{ij}.

Модель оптимизации выбора компонентов программной системы

Формализуем задачу выбора компонентов программной системы, разрабатываемой согласно схеме блока восстановления с согласованием. Модель оптимизации выбора компонентов может быть записана следующим образом:

$$\max R = \sum_{l=1}^{L} f_l \prod_{i \in s_l} R_i$$

при условиях:

$$R_i = 1 + \left[\sum_{j=1}^{m_i} \frac{1}{(1-R_{ij})^{z_{ij}}} \left(\prod_{k=1}^{m_i} (1-R_{ik})^{z_{ik}} \right) \left(1 - (1-R_{ij})^{z_{ij}}\right) + \prod_{j=1}^{m_i} (1-R_{ij})^{z_{ij}} \right] \cdot$$

$$\cdot \left[\sum_{j=1}^{m_i} z_{ij} \left(\prod_{k=1}^{j-1} P(A_{ik})^{z_{ik}} \right) P(B_{ij}) - 1 \right];\ i = 1, 2, \ldots, n,$$

$$P(A_{ij}) = (1 - p_1)[(1 - R_{ij})(1 - p_3) + R_{ij} p_2],$$

$$P(B_{ij}) = R_{ij}(1 - p_2),$$

$$R_{ij} = p_{ij} y_{ij} + R_{ij};\ i = 1, 2, \ldots, n \text{ и } j = 1, 2, \ldots, m_i,$$

$$p_{ij} = \frac{1 - q_{ij}}{(1 - q_{ij}) + q_{ij}(1 - q_{ij}) N_{ij}^{+}};\ i = 1, 2, \ldots, n \text{ и } j = 1, 2, \ldots, m_i,$$

$$N_{ij}^{+} = (1 - q_{ij}) N_{ij};\ i = 1, 2, \ldots, n \text{ и } j = 1, 2, \ldots, m_i,$$

где R – надежность (вероятность безотказной работы) программной системы; R_i – надежность модуля i; R_{ij} – надежность альтернативы j модуля i; x_{ijk} – булева переменная, равная 1, если выбрана k-я версия j-й альтернативы модуля i, созданная сторонними разработчиками (*COTS*-компонент), 0 – иначе; y_{ij} – булева переменная, равная 1, если j-я альтернатива модуля i разрабатывается собственными силами, 0 – иначе; z_{ij} – булева переменная, равная 1, если j-я альтернатива присутствует в модуле i, 0 – иначе; A_{ij} – событие, соответствующее отклонению результата альтернативы j модуля i; B_{ij} – событие, соответствующее принятию корректного результата альтернативы j модуля i; p_1 – вероятность того, что следующая альтернатива не будет вызвана, несмотря на сбой текущей

альтернативы; p_2 – вероятность неверной оценки корректного результата; p_3 – вероятность принятия неверного результата за корректный; N_{ij}^+ – количество успешно пройденных тестов; N_{ij} – общее количество тестовых испытаний; q_{ij} – вероятность того, что единичное исполнение альтернативы j модуля i не пройдет тест.

Следует отметить, что если альтернатива для модуля i приобретается (т. е. $x_{ijk} = 1$), тогда внутренняя разработка компонента отсутствует (т. е. $y_{ij} = 0$) и наоборот [7]:

$$y_{ij} + \sum_{k=1}^{V_{ij}} x_{ijk} = 1;\ i = 1, 2, ..., n;\ j = 1, 2, ..., m_i.$$

В том случае, если используется *COTS*-компонент, т. е. программный компонент приобретается, то информация о характеристиках компонента и, в том числе, о его надежности предоставляется поставщиком. Таким образом, надежность R_{ij} альтернативы j модуля i программной системы можно представить в виде:

$$R_{ij} = p_{ij} y_{ij} + \sum_{k=1}^{V_{ij}} R_{ijk} x_{ijk} = 1;\ i = 1, 2, ..., n;\ j = 1, 2, ..., m_i,$$

где R_{ijk} – надежность версии k альтернативы j модуля i.

При создании программной системы гарантируется избыточность обоих типов компонентов, как компонентов собственной разработки, так и готовых *COTS*-компонентов:

$$y_{ij} + \sum_{k=2}^{V_{ij}} x_{ijk} = z_{ij},$$

$$x_{ij1} + z_{ij} = 1;\ j = 1, 2, ..., m_i,$$

$$\sum_{j=1}^{m_i} z_{ij} \geq 1;\ i = 1, 2, ..., n.$$

Стоимость самостоятельной разработки альтернативы j модуля i можно выразить как $c_{ij}(t_{ij} + \tau_{ij}N_{ij})$, где c_{ij} – стоимость единичной разработки альтернативы j модуля i; t_{ij} – расчетное время разработки альтернативы j модуля i; τ_{ij} – среднее время, требуемое на выполнение тестового сценария для альтернативы j модуля i. Тогда общие расходы на разработку и закупку компонент программной системы C должны удовлетворять условию:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \left(c_{ij}(t_{ij} + \tau_{ij} N_{ij}) y_{ij} + \sum_{k=1}^{V_{ij}} c_{ijk} x_{ijk} \right) \leq C,$$

где c_{ijk} – стоимость версии k альтернативы j модуля i.

При выборе компонентов программной системы особое внимание стоит уделить сроку поставки готовых компонентов. Срок поставки версии k альтернативы j модуля i d_{ijk} указывается поставщиком. Время же собственной разработки компонентов (t_{ij} + $\tau_{ij}N_{ij}$) рассчитывается группой разработчиков программной системы.

В реальных условиях довольно сложно получить точные значения этих параметров, так как в процессе разработки или при приобретении компонентов необходимо учесть много факторов. Как правило, эти значения рассчитываются на начальных этапах разработки программной системы. Так, время, требуемое для создания программной системы, определяется временем интеграции различных компонентов в состав программной системы, временем тестирования системы, временем поставки *COTS*-компонентов. На время создания программной системы влияют такие факторы, как состав группы разработчиков, стратегия тестирования, среда тестирования, доступность компонент на рынке сбыта, сведения о профессиональной квалификации поставщика т. д. При этом точный срок поставки компонентов для разработки программной системы определить достаточно сложно модульной. Вследствие чего приходится закладывать в срок поставки определенный уровень допустимого отклонения. Таким образом, ограничение на срок разработки программной системы может быть записано следующим образом:

$$y_{ij}(t_{ij} + \tau_{ij} N_{ij}) + \sum_{k=1}^{V_{ij}} d_{ijk} x_{ijk} \le T;\ i = 1, 2, ..., n;\ j = 1, 2, ..., m_i.$$

где T – максимальный срок разработки программной системы, определяемый проектировщиком системы.

Решение задачи оптимизации состава избыточной программной системы, создаваемой согласно схеме блока восстановления с согласованием, может быть получено с помощью хорошо известных методов дискретной оптимизации или методов принятия решений [4]. Следует отметить, что ряд параметров может быть представлен в виде нечетких чисел, например, сроки поставки готовых программных компонент или время тестирования программной системы. В этом случае решение может быть получено при помощи методов и алгоритмов нечеткой оптимизации [7].

Заключение

Проблема обеспечения высокого уровня надежности программных систем всегда остро стоит перед проектировщиками и разработчиками, и успех ее решения во многом определяет эксплуатационные характеристики создаваемых программных систем. Наряду с классическими методами тестирования и отладки программного обеспечения существенный вклад в повышение надежности вносят новые подходы к структурному построению и выбору состава программных систем. Рассмотренный в данной статье подход позволяет повысить надежность программной системы за счет программной избыточности, применение которой, в свою очередь, приводит к дополнительным затратам. В статье предложена модель, формализующая проблему формирования оптимальной программной системы по схеме блока восстановления с согласованием, которая учитывает наиболее важные ограничения

при создании программных систем. Особенностью данной модели является учет возможности формирования программной системы на базе, как готовых программных компонент, так и компонент собственной разработки. Потенциальной областью применения предложенной модели является создание программных систем, к которым предъявляются высокие требования по надежности.

Список литературы

1. Анализ надежности мультиверсионных архитектур аппаратно-программных комплексов / О. А. Антамошкин, А. С. Дегтерев, М. А. Русаков, А. А. Усольцев // Успехи современного естествознания. – 2005. – № 6. – С. 44-45.
2. К вопросу реализации муравьиного алгоритма при выборе состава мультиверсионного программного обеспечения информационно-управляющих систем / И. В. Ковалев, Р. Ю. Царев, А. В. Прокопенко, Е. В. Соловьев // Приборы и системы. Управление, контроль, диагностика. – 2012. – № 2. – С. 1-4.
3. Ковалев, И. В. Архитектурная надежность программного обеспечения информационно-управляющих систем: монография / И. В. Ковалев, Р. Ю. Царев, Д. В. Капулин; Краснояр. гос. аграр. ун-т. – Красноярск, 2011. – 182 с.
4. Царев, Р. Ю. Методология многоатрибутивного формирования мультиверсионного программного обеспечения сложных систем управления и обработки информации: монография / Р. Ю. Царев; Краснояр. гос. аграр. ун-т. – Красноярск, 2011. – 210 с.
5. Царев, Р. Ю. Минимизация межмодульного интерфейса при формировании мультиверсионного программного обеспечения / Р. Ю. Царев, Д. В. Капулин, О. И. Завьялова // Системы управления и информационные технологии. – 2011. – № 3.1 (45). – С. 140-143.
6. Царев, Р. Ю. Мультиверсионное программное обеспечение. Алгоритмы голосования и оценка надежности: монография / Р. Ю. Царев, А. В. Штарик, Е. Н. Штарик. – Красноярск: Сибирский федеральный университет, 2013. – 120 с.
7. Bali, S., Gupta, A., Dinesh Kumar, U. Fuzzy multi-objective build-or-buy approach for component selection of fault tolerant modular software system under consensus recovery block scheme (2012) Advances in Intelligent and Soft Computing, 130 AISC (VOL. 1), pp. 1025-1036.
8. Fiondella, L., Zeephongsekul, P. Recovery block fault tolerance considering correlated failures (2014) Proceedings - Annual Reliability and Maintainability Symposium, art. no. 6798525.
9. Randell, B., Jie, X., The Evolution of the Recovery Block Concept (1995) Software Fault Tolerance, Michael R. Lyu, editor, Wiley, pp. 1–21.

10. Terada, S., Ushio, T. Optimal configuration for multiversion real-time systems using slack based schedulability (2010) IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E93-A (12), pp. 2709-2716.

Рецензенты:

Бронов С. А., д.т.н., профессор, руководитель научно-учебной лаборатории систем автоматизированного проектирования кафедры систем искусственного интеллекта Сибирского федерального университета, г. Красноярск.

Ченцов С. В., д.т.н., профессор, зав. кафедрой «Системы автоматики, автоматизированное управление и проектирование» Сибирского федерального университета, г. Красноярск.