Оценка качества программного обеспечения систем, важных для безопасности АЭС

Е.Ф. Жарко

Аннотация. Автоматизированные системы управления технологическими процессами (АСУ ТП) в последнее десятилетие вышли на качественно новый уровень развития, и одним из следствий данного процесса является существенное усложнение программного обеспечения, включаемого в АСУ ТП, и его жизненного цикла. В ИПУ РАН в 2000 г. были разработаны комплексы программного обеспечения для перспективных АСУ ТП АЭС. Требуемый срок службы всего АСУ ТП АЭС составляет не менее 30 лет, что приводит к необходимости разработки и внедрения значительных по объему и сложности процедур по сопровождению и модернизации программных комплексов. В работе рассмотрен опыт, накопленный ИПУ РАН в процессе сопровождения программного обеспечения для АСУ ТП АЭС по обеспечению качества модернизированной программы и прогнозирование затрат на ее модернизацию и сопровождение.

Ключевые слова: АЭС, системы управления, системы важные для безопасности, программное обеспечение, качество.

Введение

Системы управления технологическими процессами (АСУ ТП) в последнее десятилетие вышли на качественно новый уровень развития, связанный с возросшим уровнем автоматизации объектов управления и, как следствие, ростом числа диагностических и управляющих сигналов, обрабатываемых системой в единицу времени. В то же время, практически линейный рост производительности вычислительных систем, которые могут быть использованы в АСУ ТП, позволил реализовать значительно более сложные алгоритмы управления и анализа данных с использованием сложных программно-технических комплексов. Однако произошедший качественный скачок в составе решаемых задач заставил пересмотреть соотношение составляющих жизненного цикла программ.

Данные изменения хорошо прослеживаются на примере разработки программного обеспечения для АСУ ТП АЭС с требуемым сроком службы всего АСУ ТП АЭС не менее 30 лет. Это значительно превышает средний достигнутый на данный момент срок службы и хранения технических средств и заставляет уделять больше внимания тщательной разработке этапа модификации и сопровождения разработанного программного обеспечения (ПО)

В современных АСУ ТП АЭС программное обеспечение применяется повсеместно, начиная от контроллеров и заканчивая общестанционными системами, предназначенными для организации управления многоблочными АЭС в целом. Не являются исключениями и АСУ ТП АЭС российского производства, которые строятся внутри страны и поставляются за рубеж.

Обеспечение качества программного обеспечения – непрерывный процесс в течение всего жизненного цикла ПО, который охватывает:

- методы и средства анализа, проектирования и кодирования;
- технические отчеты, выполняющиеся на каждом шаге разработки программного обеспечения;
- методику многоуровневого тестирования;

- контроль программной документации и внесенных в нее изменений;
- процедуры обеспечения соответствия стандартам в области разработки программного обеспечения, соответствие которым определено в задании на разработку данного ПО;
- алгоритмы измерений и составления отчетов.

Качество программного обеспечения можно определить как соответствие явно установленным функциональным и эксплуатационным требованиям, явно указанным стандартам разработки и неявным характеристикам, которые ожидаются от профессионально разработанного программного обеспечения. Такое определение качества программного обеспечения подчеркивает три важных обстоятельства:

- требования к программному обеспечению – основа, относительно которой определяется качество ПО;
- указанные стандарты определяют множество критериев проектирования, которое определяет стиль разработки ПО;
- существует множество неявных требований, о которых часто не упоминается (например, сопровождаемость и модифицируемость); если программное обеспечение соответствует явным требованиям к его разработке, но не в состоянии выполнить неявные требования, то качество ПО является сомнительным.

Эти обстоятельства наиболее четко прослеживаются применительно к программному обеспечению систем высокой надежности, к которым относятся и подсистемы АСУ ТП АЭС, так как кроме полной корректности, программное обеспечение обладает и другими характеристиками, представляющими интерес потребителя данного ПО, такими как отсутствие ошибок во время выполнения, целостность данных, временные характеристики, точность, корректность типов, завершенность, функциональная надежность, безопасность, сопровождаемость, понятность, модифицируемость и другие.

1. Классификация систем, важных для безопасности АЭС

При разработке систем для энергетики, где срок эксплуатации основного оборудования исчисляется десятками лет, нужно использовать такие решения в АСУ ТП, которые бы позволяли эксплуатировать, ремонтировать и модернизировать поставленное оборудование без остановки основного технологического процесса. Кроме этого требования, ключевыми требованиями являются обеспечение высокой надежности, живучести и безопасности.

С учетом современных требований, настоящего состояния электроники и программного обеспечения системы энергетики могут строиться на основе либо собственных технологий, либо с использованием заимствованных технологий. При этом заимствованные технологии должны быть подвергнуты процессу адаптации, который бы позволил сделать их прозрачными и управляемыми до такой степени, чтобы поставщик мог распространить на них собственные гарантийные обязательства продолжительностью в несколько десятков лет.

Согласно международной классификации [1] различают системы, важные для безопасности АЭС с точки зрения категорий функций, выполняемых этими системами.

Категория А – функции, которые играют основную роль в достижении или поддержании безопасности АЭС с целью предотвращения развития аварий до недопустимых последствий;

Категория В – функции, которые играют дополнительную роль по отношению к функциям категории А в достижении или поддержании безопасности АЭС, в особенности функции, необходимые для эксплуатации после достижения контролируемого состояния с целью предотвращения развития проектных событий (ПС) до недопустимых последствий или для смягчения последствий ПС.

Категория С – функции, которые играют вспомогательную или косвенную роль в достижении или поддержании безопасности АЭС.

Основные принципы разработки систем управления, важных для безопасности АЭС, нашли отражение в международных стандартах [2, 3]. Не существует единой классификации систем АЭС. В таблице приведено сравнение классов безопасности систем АЭС, приведенных в различных нормативных документах. В зависимости от класса безопасности, на программное обеспечение, разрабатываемое для

Сравнение классов безопасности систем АЭС, приведенных в различных документах

Стандарт или нормативный документ	Классы безопасности (степень важности увеличивается слева направо)				
ПНАЭГ-01-011 [4]	Класс 4	Класс 3		Класс 2	Класс 1
IAEA NS-R-1 [5]	Системы, не важные для безопасности	Системы, важные для безопасности			Нет
		Системы, связанные с безопасностью		Системы безопасности	
IEC 61226	Неклассифицированные	Класс С	Класс В	Класс А	Нет
IEEE 603 [6]	Не класс 1Е			Класс 1Е	Нет

этих систем, накладывают ограничения, связанные с применимостью операционных систем, языков программирования, детальностью документирования и т.д.

В состав АСУ ТП АЭС входят системы 2-го, 3-го и 4-го классов безопасности в соответствии с [4] или в соответствии с международной классификацией [1] системы классов А, В, С. Таким образом, при разработке программного обеспечения для подсистем АСУ ТП АЭС необходимо руководствоваться стандартами [2] (для систем класса А), [3] (для систем класса В, С).

2. Определение качества программного обеспечения

Определение качества программного обеспечения помогает:

- оценить программные изделия;
- оценить принципы организации программного обеспечения;
- улучшить процессы создания программного обеспечения.

Различают следующие этапы оценки качества программного обеспечения:

- определение характеристик качества программного изделия;
- разработка показателей для определения характеристик качества;
- запись значений и сравнение с предыдущими значениями;
- внесение изменений в программное обеспечение для улучшения его качества.

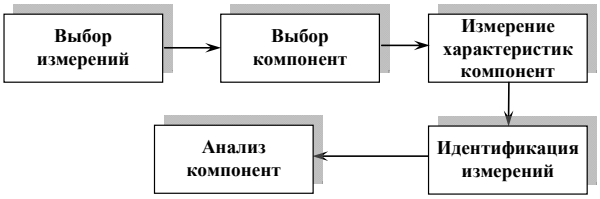

Рис. 1. Процесс «измерения» программного продукта

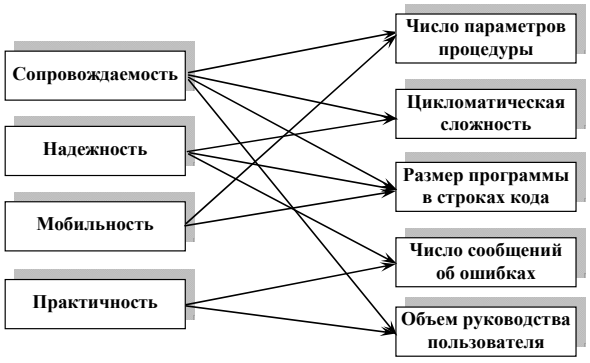

Рис. 2. Взаимосвязь между показателями и характеристиками качества

На Рис. 1 показан процесс «измерения» программного продукта. Невозможно измерить некоторые характеристики напрямую, однако их можно измерить через показатели качества. На Рис. 2 представлена зависимость между характеристиками и показателями качества.

Условия, которые должны быть выполнены при измерении характеристик программного обеспечения через показатели качества:

- показатели качества должны измеряться точно;

- должна существовать зависимость между показателями и характеристиками качества;

- зависимость должна быть выражена в виде формулы или модели.

Показатели качества имеют два типа: показатели управления и показатели прогноза.

Показатели управления используются руководством для управления процессом разработки программного обеспечения. Эти показатели обеспечивают информацией о качестве процесса. Они не являются характерными для программного обеспечения – любое обрабатывающее производство может управляться и контролироваться по подобным показателям. Показателями управления являются, например, объем работ, затраченное календарное время или коэффициент использования для частных задач или действий, процент проверенных операторов и т.д.

Показатели прогноза определяют характеристики изделия, которые предсказывают качество изделия. Характеристики качества предсказаны, если определены соответствующие им показатели. Различают два типа показателей прогноза: динамические и статические.

Динамические показатели собирают измерения, сделанные в процессе выполнения программы для эффективности и надежности. Они тесно связаны с качеством программного обеспечения и удобны для определения. Эффективность может быть рассчитана на основе измерения времени выполнения, а надежность - на основе числа отказов системы и типов отказов.

Статические показатели собирают измерения, сделанные в процессе представления системы для оценки сложности, понятности и сопровождаемости. Эти показатели имеют косвенную связь с качеством программного обеспечения, которая предполагает, что существует зависимость между характеристиками и показателями качества.

Рассмотрим опыт, накопленный в течение более восьми лет модификации и сопровождения сложного высоконадежного программного обеспечения, разработанного в Институте проблем управления им. В.А. Трапезникова РАН для системы верхнего блочного уровня АСУ ТП вновь строящихся и модернизируемых АЭС [7, 8].

Сформулируем основные понятия, используемые в работе.

Модификация (модернизация) - внесения изменений в уже согласованные документы (программу).

Сопровождение - модификация программного изделия, связанная с исправлением ошибок, улучшением функциональности или производительности программного продукта или изменением окружения, в котором происходит выполнение программы.

Сопровождение и, соответственно, модификация ПО могут быть нескольких типов.

- *Коррекция (тип А)* – работы, связанные с необходимостью исправления ошибок в ранее разработанной программе.
- *Адаптация (тип В)* – работы, связанные с изменением условий окружения, в котором выполняется программа.
- *Улучшение (тип С)* – работы, связанные с добавлением новой функциональности в программу.

3. Сопровождение программного обеспечения

Программное обеспечение для подсистем АСУ ТП АЭС является сложным ПО с объемом кода свыше 800 Мб, включающим более 100 программных компонентов. Процедура его модификации и сопровождения должна гарантировать сохранение высокого качества предоставляемой функциональности и совместимости с прикладным ПО после проведения модификации.

Для разработки и сопровождения программного обеспечения подсистем АСУ ТП АЭС, используем «стандартную» модель жизненного цикла [9].

Этап жизненного цикла, связанный с сопровождением ПО, с процедурной точки зрения мало отличается от собственно разработки нового ПО, являясь в определенном качестве рекурсивной процедурой в рамках жизненного цикла и может быть разделен на следующие фазы:

1) идентификация проблемы;
2) анализ проблемы (техническое задание);
3) технический проект;
4) разработка;
5) тестирование;
6) поставка (модифицированного ПО).

Фазы 3, 4 и 6 по своей сути тождественны соответствующим фазам в разработке нового ПО, при необходимости оценки качества модифицированного программного обеспечения используют методы, применяемые для оценки качества вновь разработанного ПО [8, 10]. Далее подробно рассмотрим состав работ, проводимых на фазах 1, 2 и 5.

1) Идентификация проблемы. Действия, проводимые на фазе идентификации проблемы этапа сопровождения, подобны соответствующим работам, проводимым для нового ПО. И в том и в другом случае отправной точкой является наличие некоторой проблемы, которую намереваются устранить в ходе выполнения работ. Однако этап сопровождения обычно характеризуется лучшей специфицируемостью проблемы, так как и у разработчика ПО, и у пользователя уже имеется опыт работы в данной предметной области.

Фазе идентификации следует предварять деятельность, связанную с учетом несоответствий в функционировании ПО (понятие несоответствие надо понимать широко и включить в него все типы сопровождения A-C). Данная деятельность является одной из ключевых для достижения высокого качества программного изделия [11]. Замечания по функционированию ПО оформляется в виде карточки учета замечаний. В оформленный документ с указаниями замечания включены характеристические признаки дефекта: локализация проблемы, автор кода программы, содержащего ошибку, время, требуемое на внесение корректировки в код программы, причина появления данной проблемы, причина ее обнаружения.

Результатом выполнения фазы идентификации должен стать выпуск запроса на модификацию с перечнем замечаний, которые предлагается устранить при модификации ПО. Выпуск запроса на модификацию имеет также цель аккумулировать и группировать выявленные несоответствия по признакам для облегчения проведения следующей фазы сопровождения ПО. В качестве системы классификации использована так называемая «Схема классификации по атрибуту» [12]. Согласно этой схеме данные по карточкам учета замечаний классифицировались по следующим атрибутам:

- категория (управление данными, определение исходных данных, системное (ядро операционной системы));
- тип, поясняющий категорию (пример: определение типа данных);
- наличие кода (несоответствие вызвано наличием лишнего кода, отсутствием кода, ошибкой в кодировании).

Введение данной классификации позволило точнее специфицировать причину и ситуацию появления несоответствия.

2) Анализ проблемы. Фаза анализа проблемы выполняется на основе запроса на модификацию и осуществляет оценку возможности и необходимость устранения каждого замечания, перечисленного в запросе.

На основе анализа запроса на модификацию выпускается техническое задание с перечнем требований, которые необходимо удовлетворить в ходе проведения модификации ПО.

Наибольшую трудность для разработчика во время реализации фазы анализа часто составляет оценка не только технической реализуемости, но и трудоемкости проведения модификации. Кроме непосредственно внесения изменений, связанных с реализуемой (новой) функциональностью, необходимо учитывать затраты на проверку влияния внесенных изменений на смежные области модифицируемого программного обеспечения и уже реализованной в оригинальной (не модифицированном ПО) функциональности. При оценке затрат на модификацию необходимо учитывать следующие факторы.

- Степени модульности (независимости компонентов) модифицируемого ПО. Под модулем понимается программная единица, выполняющая некоторую независимую функцию в составе ПО. Чем более обособленно по группам модулей распределена модифицируемая функциональность, тем меньше затрат на тестирование смежных модулей.
- Объемы кода в каждом из модулей могут существенно отличаться друг от друга (Рис 1) и, следовательно, сложность модификации зависит от того, насколько «сложные» модули вовлечены в процесс модификации. Если модификация вносится в код, обладающий меньшей сложностью, облегчается анализ модифицированного

модуля и уменьшаются и затраты на тестирование реализованной функциональности.

Для оценки сложности программного обеспечения разработано большое количество метрик [13, 14], но их применение в большей мере отражает субъективный выбор эксперта, чем недостатки или достоинства данной метрики. Для оценки сложности модуля (или группы модулей) может быть использована тривиальная метрика, основанная на подсчете суммы линий кода (ЛК). Однако, как было показано в работе [15], объем кода программы по метрике ЛК не является стабильной характеристикой, отражающей ее сложность. Нами были использованы также две другие метрики: цикломатическая сложность [16] и метрика затраченных усилий [17]. Для вычисления последней метрики не требуется проведения глубокого структурного анализа кода и она позволяет оценить число ошибок в коде и затраты, требуемые на его модификацию. Поэтому данная метрика была выбрана основной, хотя для компактных модулей более приемлема, видимо, оценка сложности с использованием метрики, основанной на цикломатической сложности модуля.

5) Тестирование и обеспечение качества. Оценка качества как составная часть тестирования и верификации ПО осуществляется в течение всего его жизненного цикла. В соответствии с международными стандартами [2, 3], одной из важнейших метрик качества ПО в процессе модификации и сопровождения является понятность программного кода.

Любая программа благодаря тому, что реализует некий известный алгоритм работы, может быть полностью проверена при ограничении входного набора данных. Однако это утверждение правомерно только для тривиальных либо некоторых специальных случаев ПО. Ресурсы, выделяемые на тестирование, в большинстве случаев недостаточны для обеспечения полного тестового покрытия программы. Возникает задача классификации частей программного кода для того, чтобы выделить участки, наиболее вероятно содержащие ошибки.

Заключение

Качество программного обеспечения можно считать «достаточно хорошим», когда потенциально-положительные результаты создания или использования разработанного программного обеспечения приемлемо перевешивают потенциально-негативные мнения заказчиков. Такой подход проверяет, с точки зрения традиционного понятия качества программного обеспечения, различные варианты реализации. При таком подходе к качеству программного обеспечения высокие непроверенные требования заменяются оптимальными. Этот подход сфокусирован на идентифицирующих задачах и улучшении возможностей для принятия решений. Таким образом, проект разработки программного обеспечения для систем важных для безопасности, должен быть скорее проблемно-ориентированным, чем целенаправленным на качество ПО.

Рис. 3. Место верификации и валидации программного обеспечения в обеспечении качества

Качество программного обеспечения достигается благодаря эффективной методологии разработки и использованию методов верификации и валидации в течение жизненного цикла разработки ПО для систем, важных для безопасности. На Рис. 3 представлено место верификации и валидации программного обеспечения в контексте обеспечения качества и иерархии стандартов.

Литература

1. IEC 61226 ed3.0 Nuclear power plants - Instrumentation and control important to safety - Classification of instrumentation and control functions. 2009
2. IEC 60880 ed2.0 Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions. 2006.
3. IEC 62138 ed1.0 Nuclear power plants - Instrumentation and control important for safety - Software aspects for computer-based systems performing category B or C functions. 2004.

4. НП-001-97 (ПНАЭ Г-01-011-97). Общие положения обеспечения безопасности атомных станций. ОПБ-88/97. Госатомнадзор России, 1997.
5. Safety of Nuclear Power Plants: Design Safety Requirements. IAEA Safety Standards Series No. NS-R-1. 2000.
6. IEEE Std 603-1998 IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations – Description. 1998.
7. Полетыкин А.Г., Менгазетдинов Н.Э., Бывайков М.Е., Жарко Е.Ф., Промыслов В.Г. Проектирование системы верхнего блочного уровня АСУ ТП с учетом влияния на безопасность АЭС // Труды Международного симпозиума «Измерения, важные для безопасности в реакторах». М., 2002. С. 1-10.
8. Полетыкин А.Г., Жарко Е.Ф., Зуенкова И.Н., Промыслов В.Г., Бывайков М.Е., Менгазетдинов Н.Э. Программное обеспечение для атомной энергетики // Автоматизация в промышленности. 2006. № 8. С. 52-56.
9. ISO/IEC 12207:2008 Systems and software engineering – Software life cycle processes. 2008.
10. Масолкин С.И., Промыслов В.Г., Жарко Е.Ф. и др. Системное программное обеспечение LICS как компонент подсистем АСУТП АЭС // Автоматизация в промышленности. 2004. № 10. С. 21-25.
11. Grady R.B. Software Failure Analysis for High-Return Process Improvement Decisions // Hewlett-Packard Journal. 1996. Vol. 47, No. 4.
12. Ostrand T.S., Weyuker E. Collecting and Categorizing Software Error Data in an Industrial Environment // The Journal of Systems and Software. 1984. Vol. 4. P. 289-300.
13. Davis J.S., Le Blanc R.J. A Study of the Applicability of Complexity Measures // IEEE Transactions on Software Engineering. 1988. Volume SE-14, No. 9. P. 1366-1371.
14. Kafura D., Reddy G.R. The Use of Software Complexity Metrics in Software Maintenance // IEEE Transactions on Software Engineering. 1987. Vol. SE-13, No. 3. P. 335-343.
15. Haapanen P., Pullkinen, U. Licensing process for safety-critical software-based systems. STUK-YTO-TR 171. Helsinki, 2000. 72 p. + Appendices 41 p.
16. Halstead M.H., McCabe T. A Software Complexity Measure // IEEE Trans. Software Engineering. 1976. Vol. 2, No. 12. P. 308-320.
17. Halstead M.H. Elements of Software Science, New York: Elsevier, 1977.
18. IEEE 1012-2004 IEEE Standard for Software Verification and Validation. 2004.

Жарко Елена Филипповна. Старший научный сотрудник Института проблем управления им. В.А. Трапезникова РАН. Окончила Московский инженерно-физический институт в 1989 году. Кандидат технических наук. Автор более 60 печатных работ. Область научных интересов: верификация и валидация программного обеспечения для систем важных для безопасности АЭС, моделирующие комплексы для систем поддержки операторов АЭС. E-mail: zharko@ipu.ru