УДК 004.412: 004.415.5

В.В. СКЛЯР[1], Ю.А. БЕЛЫЙ[2], С.А. МАЛОХАТЬКО[2]

[1] *Государственный научно-технический центр по ядерной и радиационной безопасности, Украина*
[2] *ЗАО «Радий», Украина*

ОЦЕНКА КАЧЕСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ВЕРХНЕГО УРОВНЯ ИНФОРМАЦИОННО-УПРАВЛЯЮЩИХ СИСТЕМ АЭС

Проведен анализ структуры и состава программного обеспечения, разработанного ЗАО «Радий» для выполнения функций отображения и регистрации информации в составе верхнего уровня информационно-управляющих систем АЭС. Рассмотрены результаты статического анализа такого программного обеспечения.

верхний уровень информационно-управляющих систем, статический анализ программного кода

Постановка задачи и обзор публикаций

Информационно-управляющие системы (ИУС), разработанные ЗАО «Радий» [1, 2], в настоящее время успешно эксплуатируются на всех украинских АЭС. Программно-технические комплексы (ПТК), являющиеся ядром указанных систем, включают нижний и верхний уровни.

В состав верхнего уровня ПТК входят автоматизированные рабочие места (АРМ) на базе промышленных компьютеров, выполняющие следующие функции [3]: прием информации от нижнего уровня ПТК; отображение информации; регистрация, хранение и предоставление информации оператору.

Несмотря на то, что программное обеспечение (ПО) АРМ непосредственно не выполняет управляющих функций, оно, тем не менее, относится к элементам, важным для безопасности АЭС. Это связано с тем, что действия операторов ядерных установок влияют на безопасность АЭС и зависят от информации, представляемой посредством человеко-машинного интерфейса АРМ. В свою очередь, оперативность, наглядность и достоверность информации зависит от качества программного обеспечения (ПО) верхнего уровня ИУС, выполняющего соответствующие функции в составе АРМ.

На протяжении жизненного цикла верификация является основным процессом, направленным на обеспечение качества ПО. В ходе верификации выполняется подтверждение соответствия ПО заданным требованиям путем различного рода проверок и обеспечения объективных доказательств [4 – 6].

Традиционно основными методами верификации являются тестирование и обзоры программной документации. Однако, развитие инструментальных средств программной инженерии позволило повысить эффективность статического анализа программного кода (САПК). В ходе статического анализа определяются различного рода параметры программного кода, позволяющие продемонстрировать соответствие кода установленным требованиям либо обнаружить потенциальные проблемы, связанные с проявлением дефектов ПО. Таким образом, САПК может являться существенной составляющей процесса верификации, что подчеркивается в стандартах, содержащих требования к ПО для различных отраслей.

Описание способов выполнения и результатов САПК содержится в работах [7 – 10].

Целью данной статьи является анализ результатов САПК, полученных сотрудниками ЗАО «Радий» в процессе обеспечения качества ПО верхнего уровня ИУС АЭС.

1. Структура программного обеспечения верхнего уровня ИУС

ПО верхнего уровня выполняется на рабочих станциях под управлением операционной системы Microsoft Windows XP Professional Edition. ПО верхнего уровня разработано на языке программирования C++. В качестве инструментария используется интегрированная среда разработки Microsoft Visual C++, входящая в комплект Microsoft Visual Studio.NET с использованием библиотеки классов Microsoft Foundation Classes (MFC) версии 7.0.

Методы отработки ПО включают:

– применение средств профайлинга для оптимизации времени выполнения ПО;

– назначение контрольных точек при помощи стандартного макроса Assert, который выдает информацию о выполнении «узких» мест программы;

– динамическая отладка при помощи стандартного макроса Trace, который осуществляет мониторинг выполнения программы в режиме реального времени;

– применение имитатора сигналов для отладки интерфейса с ПО нижнего уровня ИУС.

Структура и состав ПО верхнего уровня ИУС АЭС разработки ЗАО «Радий» представлены на рис. 1, 2 и в табл. 1. ПО верхнего уровня может быть реализовано в виде двух вариантов:

1) ПО ПТК системы аварийной и предупредительной защиты реактора (ПТК АЗ-ПЗ) [1] и ПО ПТК системы автоматического регулирования, разгрузки и ограничения мощности реактора и ускоренной предупредительной защиты (ПТК АРМ-РОМ-УПЗ) [2] – характеризуются небольшим объемом регистрационной и диагностической информации, передача которой осуществляется через Коммуникационный и Информационный сервер (рис. 1);

2) ПО ПТК системы группового и индивидуального управления приводами органов регулирования системы управления и защиты реактора (ПТК СГИУ) и ПО ПТК управляющей системы безопасности (ПТК УСБ) – характеризуются значительным объемом регистрационной и диагностической информации, которая распределяется по разным серверам (рис. 2).

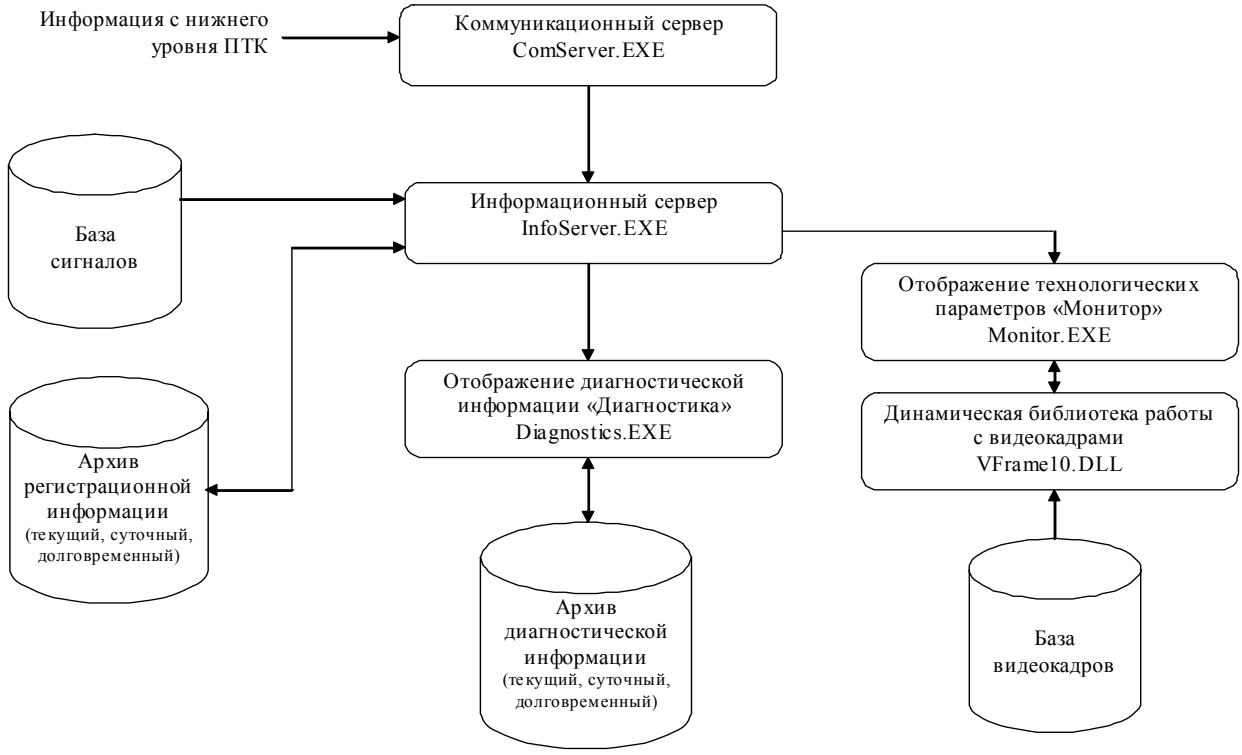

Рис. 1. Структура программного обеспечения верхнего уровня ИУС (вариант 1)

Таблица 1

Структура ПО верхнего уровня ИУС АЭС разработки ЗАО «Радий»

Наименование программы	Наименование модуля	Функции, выполняемые модулем
Коммуникационный сервер	ComServer.exe	Прием регистрационной и диагностической информации от нижнего уровня ПТК (АЗ-ПЗ, АРМ-РОМ-УПЗ); упаковка и передача информации в приложение «Информационный Сервер»; отображение состояния подключений внешних устройств; отображение уровня заполнения буфера преобразователя из оптического последовательного канала в USB-интерфейс (для каждого подключения); отображение скорости приема информации по подключениям.
Информационный сервер	InfoServer.exe	Прием регистрационной (диагностической) информации; передача регистрационной информации приложению «Монитор»; отображение состояния регистрируемых сигналов; архивирование регистрационной информации; отображение информации о текущих размерах долговременного, суточного и текущего архивов регистрационной информации; отображения текущего состояния сетевых подключений приложений «Монитор», «Коммуникационный Сервер», «Диагностика»
	База сигналов	Хранение информации о свойствах сигналов, обрабатываемых в системе
	Архив регистрационной информации	Хранение регистрационной информации
Информационный сервер диагностики	DiagInfoServer.exe	Прием диагностической информации от нижнего уровня ПТК (СГИУ, УСБ); передача диагностической информации приложению «Диагностика»; отображение состояния диагностических сигналов; архивирование диагностической информации; отображения текущего состояния приема информации от блоков нижнего уровня
	Диагностическая база	Хранение информации о конфигурации технических средств и о свойствах диагностических сигналов
	Редактор диагностической базы	Разработка диагностической базы
	База технологических параметров	Хранение информации об эталонных значениях уставок и другой служебной информации
	Архив диагностической информации	Хранение диагностической информации
Монитор	Monitor.exe	Прием регистрационной информации от приложения «Информационный Сервер»; отображение видеокадров в соответствии с текущим состоянием сигналов; отображение и печать регистрационной информации в виде графиков и в табличном виде
	Динамическая библиотека работы с видеокадрами	Загрузка и хранение видеокадров; отображение видекадров в соответствии с установленным состоянием сигналов.
	База видеокадров	Хранение видеокадров
Диагностика	Diagnostics.exe	Прием диагностической информации от приложения «Информационный Сервер Диагностики»; отображения состояния связи со шкафами ПТК; отображения и уведомления оператора о неисправностях блоков, входящих в состав шкафов ПТК; отображения состояния и показаний датчиков шкафов (температура, открытие дверей, пожарная сигнализация); ведение архива диагностической информации; отображения и печати протокола диагностических событий.
	Архив диагностической информации	Хранение диагностической информации

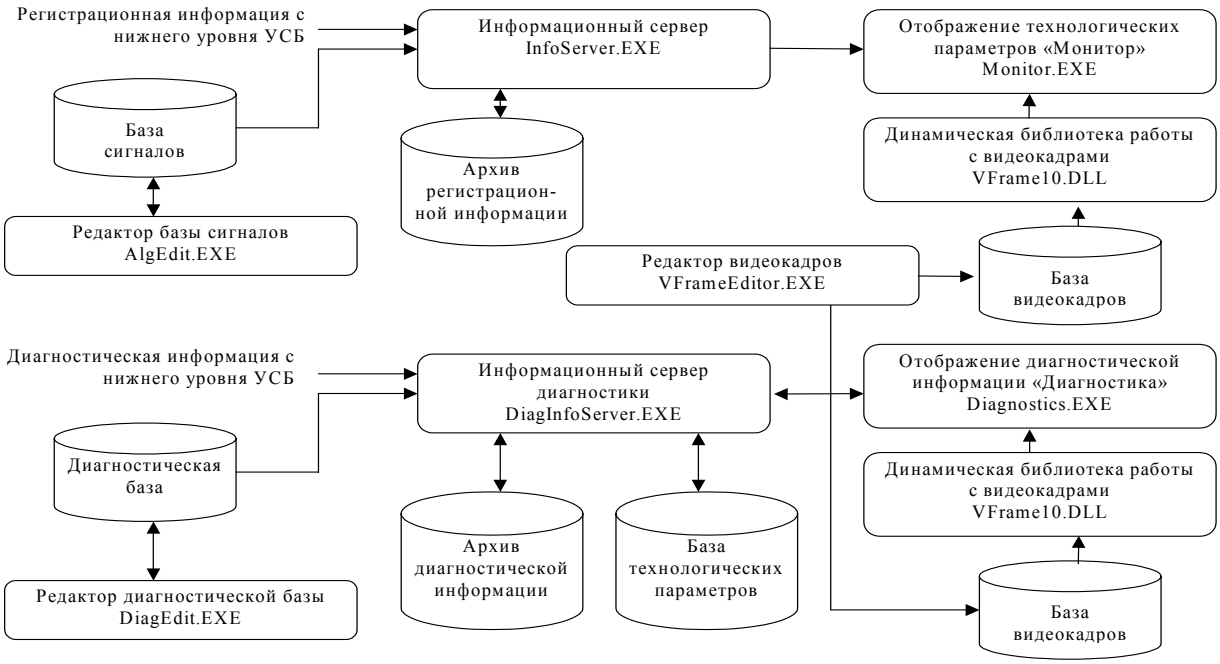

Рис. 2. Структура программного обеспечения верхнего уровня ИУС (вариант 2)

2. Результаты статического анализа программного обеспечения верхнего уровня ИУС

Рассматриваемые результаты были получены по при выполнении государственной экспертизы ядерной и радиационной безопасности для ПО верхнего уровня ИУС разработки ЗАО «Радий». Экспертиза выполнялась Харьковским филиалом Государственного научно-технического цента ядерной и радиационной безопасности.

В ходе САПК применялись следующие методы:

– проверка программного кода на соответствие правилам программирования (табл. 2);

– расчет метрик сложности программного кода (табл. 3).

Для проверки были заданы 14 правил программирования, отклонение от которых считаются недопустимыми, поскольку это создает потенциальную проблему в ходе функционирования программного кода.

Таблица 2

Запрещенные отклонения от правил программирования для проверки программного кода

Наименование отклонения от правил	Краткое описание
Использование конструкции **explicit** для конструкторов с несколькими аргументами	Для конструкторов с многими аргументами не следует использовать конструкцию **explicit**, поскольку такие конструкторы не могут быть использованы для неявного преобразования типов
Использование виртуального фиксированного деструктора	Деструктор должен быть объявленным как виртуальный всегда, когда класс имеет хотя бы одну виртуальную функцию
Не использование ключевого слова **virtual** для виртуальных функций	Все виртуальные функции следует обозначать ключевым словом **virtual**
Связанные операторы	Некоторые операторы должны применяться только в паре со связанным оператором: '==' и '!=' ; '+' и '+ =' ; '-' и '– =' ; '**new**' и '**delete**'
Сравнение для типа с плавающей точкой	Из-за проблемы разрядности тип с плавающей точкой может не всегда равнять заданному числу. Поэтому, такой тип не следует использовать для сравнения
Сложная инициализация или изменение выражения в цикле **for**	Не следует использовать более трех переменных при инициализации цикла **for**
Длинный файл	Длина файла не должна превышать 2000 строк

Наименование отклонения от правил	Краткое описание
Вычисление виртуальной функции из конструктора и деструктора	Нельзя использовать виртуальную функцию и в конструкторе, и в деструкторе. В конструкторе механизм виртуального вызова недоступен, поскольку подмена производным классом виртуальных функций базового класса еще не состоялась
Скрытие унаследованного атрибута	Атрибуты, объявленные в дочернем классе скрывают (повторяют) унаследованные атрибуты
Скрытие унаследованной статической операции (метода)	Статические операции, объявленные в дочернем классе, скрывают (повторяют) унаследованные операции
Скрытие имен	Объявление имен не должно скрывать (повторять) объявление другого имени
Функция с переменным списком аргументов	Вместо функций с переменным списком аргументов следует использовать параметры по умолчанию или перегрузку функций
Использование невиртуального базового класса	Виртуальный базовый класс предоставляет возможность экономить пространство памяти и избегать неопределенности в иерархии классов при использовании множественного наследования. Поэтому невиртуальное двойное (ромбовидное) наследование является нежелательным в тех случаях, когда базовый класс содержит элементы данных
С-стиль распределения памяти	Не следует использовать функции распределения памяти из библиотеки языка C <stdlib.h>, такие как **calloc**, **free**, **realloc**, **malloc**. Следует использовать только операторы **new** и **delete**

Таблица 3

Результаты метрического анализа программного кода

Наименование метрики	Краткое описание	Диапазон допустимых значений	Значение метрики
Количество строк кода	–	[0 – 2000]	149
Количество атрибутов	–	[0-30]	5
Количество операций		[0-50]	6
Количество классов	–	[0-5]	1
Количество конструкторов	–	[0-5]	1
Количество членов класса	Определяется как сумма количества атрибутов и количества операций	[0-80]	12
Цикломатическая сложность	CC = (число связей в графе потока управления) – (число узлов в графе потока управления) + 2 * (число разъединенных частей в графе потока управления)	[0-50]	24
Количество выходов	Учитываются связанные типы данных, такие как атрибуты, возвращаемые формальные параметры, типы, метки и локальные переменные. Простые типы и супертипы при этом не учитываются	[0-15]	7
Длина программы по Холстеду	Длина = (количество операторов) + (количество операндов)	[0 – 2000]	304
Объем программы по Холстеду	Объем = (количество операторов + количество операндів) * Log_2 (количество уникальных операторов + количество уникальных операндов)	[0 – 3000]	2306
Максимальное количество уровней	Глубина вложенности для условных операторов и операторов цикла	[0-7]	1
Максимальное количество параметров для операции	–	[0-4]	1
Максимальный размер операции	Рассчитывается цикломатическая сложность для операции	[0-10]	4

Все отклонения, выявленные в ходе САПК, были проанализированы, их существование в программном коде были признано потенциально опасным, поэтому выявленные отклонения были устранены.

Для выполнения метрического анализа были заданы 13 метрик ПО, для которых были заданы диапазоны допустимых значений. Отклонение значений метрик от заданных диапазонов считаются недопустимыми, поскольку это свидетельствует о повышенной сложности ПО и о повышении риска внесения дефектов в программный код. В ходе выполнения САПК было определено, что значение всех метрик находятся в пределах допустимых диапазонов.

Выводы

Статический анализ программного кода дополняет традиционные процедуры тестирования, поскольку помогает определять все варианты выполнения программы, и поэтому позволяет искать ошибки вне зависимости от входных данных. Благодаря статическому анализу можно находить ошибки, не проявляющиеся во время тестирования или существующие в вариантах выполнения программы, которые не проверяются при тестировании.

Проведение статического анализа для ПО верхнего уровня ИУС АЭС, разработанного ЗАО «Радий», позволило повысить качество ПО и подтвердить соответствие ПО требованиям по безопасности.

Литература

1. Бахмач Е.С., Виноградская С.В., Розен Ю.В. и др. Программно-технические комплексы аварийной и предупредительной защиты ядерных реакторов: обеспечение и оценка безопасности // Ядерная и радиационная безопасность. – 2005. – Т. 8, № 1. – С. 21-50.

2. Бахмач Е.С., Виноградская С.В., Розен Ю.В. и др. Программно-технические комплексы автоматического регулирования, разгрузки и ограничения мощности реактора и ускоренной предупредительной защиты: обеспечение и оценка безопасности // Ядерная и радиационная безопасность. – 2005. – Т. 8, № 1. – С. 67-90.

3. Белый Ю.А., Герасименко А.Д., Головир А.Д. и др. Поддержка деятельности операторов в системах управления и защиты ядерных реакторов АЭС // Збірка наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова. – К.: ІПМЕ, 2006. – Спец. випуск, т. 1. – С. 21-28.

4. Ястребенецкий М.А., Васильченко В.Н., Виноградская С.В. и др. Безопасность атомных станций: Информационные и управляющие системы. – К.: Техніка, 2004. – 472 с.

5. Харченко В.С., Скляр В.В., Гордеев А.А. Верификация программного обеспечения. – Х.: Нац. аэрокосмический ун-т «Харьк. авиац. ин-т», 2006. – 132 с.

6. Скляр В.В., Харченко В.С., Ястребенецкий М.А. Особенности и оценка безопасности программного обеспечения информационных и управляющих систем АЭС Украины // Ядерные измерительно-информационные технологии. – 2006. – № 1 (17). – С. 3-18.

7. Lyu M.R. Handbook of Software Reliability Engineering.– McGraw-Hill Company, 1996.– 805 p.

8. German A. Software Statistic Analysis: Lessons Learnt. 9th Safety-Critical Club Symposium.– Bristol, UK, 2001.– 12 p.

9. Скляр В.В., Ястребенецкий М.А., Карнаухов А.А. Оценка программного обеспечения управляющей системы АЭС с использованием инструментального средства LDRA TESTBED // Ядерная и радиационная безопасность. – 2006. – Т. 9, № 2. – С. 83-98.

10. Скляр В.В. Инструментальные средства для статического анализа программного обеспечения: принципы применения, оценки и выбора // Электронное моделирование. – 2006. – Т. 28, № 2. – С. 29-41.

Поступила в редакцию 7.02.2007

Рецензент: д-р техн. наук, проф. Г.Н. Жолткевич, Харьковский национальный университет им. В.Н. Каразина, Харьков.