На правах рукописи

Долбня Николай Алексеевич

РАЗРАБОТКА И ИССЛЕДОВАНИЕ МЕТОДОВ И ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ СЕРТИФИЦИРУЕМЫХ ДРАЙВЕРОВ АВИАЦИОННЫХ БОРТОВЫХ ИНФОРМАЦИОННО-УПРАВЛЯЮЩИХ СИСТЕМ

Специальность: 05.13.12 – Системы автоматизации проектирования (промышленность)

Автореферат

диссертации на соискание ученой степени
кандидата технических наук

Ульяновск – 2012

Работа выполнена на кафедре «Измерительно-вычислительные комплексы» Ульяновского государственного технического университета.

Научный руководитель –	кандидат технических наук, доцент, **Шишкин Вадим Викторинович**
Официальные оппоненты:	**Негода Виктор Николаевич** доктор технических наук, профессор, УлГТУ, кафедра «Вычислительная техника», профессор кафедры
	Стецко Александр Алексеевич доктор технических наук, главный технолог ФНПЦ ОАО «НПО «Марс»
Ведущая организация –	**Открытое акционерное общество «Московский институт электромеханики и автоматики» (ОАО «МИЭА»), г. Москва**

Защита состоится 26 декабря 2012г. в 15:00 на заседании диссертационного совета Д212.277.01 при Ульяновском государственном техническом университете по адресу: 432027, г. Ульяновск, ул. Северный Венец, 32, ауд.211

С диссертацией можно ознакомиться в научной библиотеке Ульяновского государственного технического университета.

Автореферат разослан __ ноября 2012 г.

Ученый секретарь диссертационного совета
доктор технических наук,
профессор Смирнов Виталий Иванович

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность. Особую роль в современных авиационных бортовых информационно-управляющих системах (БИУС) играет системное программное обеспечение, в частности, комплект драйверов, обеспечивающих взаимодействие системы с внешними входными и выходными устройствами. Повышенные требования к надежности драйверов обусловлены следующими факторами: сбой в драйвере влечет за собой отказ устройства и всего канала связи, реализуемого этим устройством; сбой в драйвере режима ядра способен повлечь за собой отказ всей операционной системы, а следовательно, и всего бортового вычислителя; некорректная работа драйверов в составе операционной системы жесткого реального времени (ОСЖРВ) влечет за собой слабо детерминированное изменение основной характеристики ОСЖРВ – максимального времени реакции БИУС на внешние факторы, что способно привести к отказу БИУС, плохо поддающемуся диагностированию.

Надежность БИУС гарантируется ее сертификацией по принятым в отрасли стандартам. Для программного обеспечения (ПО) БИУС в гражданской авиации принят стандарт КТ-178В. В соответствии с ним драйвер БИУС определяется не только как программный код, а как совокупность десяти типов связанных сертификационных артефактов, порождаемых на всех этапах жизненного цикла ПО БИУС.

В настоящее время номенклатура БИУС составляет несколько сотен и она постоянно увеличивается. Также имеется тенденция перевода БИУС на новые высокопроизводительные вычислительные платформы и модификации внешних устройств БИУС. В связи с этим только в ОАО «УКБП» существует необходимость разработки и модификации в среднем 200 драйверов БИУС в год. При этом пример ОАО «УКБП» является типичным для авиаприборостроительной отрасли. С учетом необходимости разработки всех десяти типов сертификационных артефактов для этих драйверов, а также сокращения времени на разработку БИУС в целом поставленная задача обладает несомненной актуальностью. Особо актуальной является задача автоматизации проектирования требований низкого уровня к ПО, исходного кода драйверов и их модульных тестов, так как проектирование именно этих типов артефактов является наиболее трудоемким и сложно автоматизируемым.

В настоящее время задачей автоматизации проектирования драйверов занимаются фирмы-разработчики операционных систем такие, как Microsoft, LynuxWorks, Wind River, QNX и др. Однако ими задача решается на уровне шаблонов исходного кода, а остальные артефакты по стандарту КТ-178В не рассматриваются. Кроме того, инструментальные средства этих фирм не обеспечивают достаточную эффективность проектирования сертифицируемых драйверов.

Вопросами проектирования, верификации и сертификации системного ПО и, в частности, драйверов, на данный момент занимались Солдатов В.П., Бурдонов И.Б., Косачев А.С., Пономаренко В.Н., Тихонов А.Ю., Щербаков

А.Ю., Martin Timmerman, Bart Van Beneden, W. Mason, A. Baker, J. Lozano, P. Dabak, S. Phadke, Edward N. Dekker, J.M. Newcomer. Вопросы повышения эффективности проектирования с использованием паттернов на данный момент рассматривались Шалыто А.А., Richard Helm, Erich Gamma, Christopher W. Alexander, Ralph Johnson, John Vlissides, J. Kirievsky, M. Fowler, Sherif M. Yacoub, Hany H. Ammar, G. Hohpe, B. Woolf, C. Larman, Scott W. Ambler.

Целью диссертационной работы является разработка методов и инструментальных средств автоматизированного проектирования драйверов БИУС как множеств сертификационных артефактов.

Следуя цели, в диссертационной работе были поставлены и решены следующие задачи:

1. Анализ драйверов БИУС как объектов проектирования и инструментальных средств, обеспечивающих их автоматизированное проектирование.
2. Разработка моделей типового драйвера БИУС.
3. Разработка методики автоматизированного проектирования драйверов БИУС.
4. Разработка инструментальных средств автоматизации проектирования драйверов БИУС.
5. Разработка методики и средств виртуализации стенда для отладки системного, функционального и диагностического ПО БИУС.

Объектом исследования в работе является процесс проектирования драйверов БИУС, предметом исследования служат применяемые для этого модели, методы и инструментальные средства.

Методы исследования базируются на теории алгоритмов, теории системного анализа, теории программирования, теории конечных автоматов, алгебраических методах и строятся на сочетании формальных и содержательных методов.

Научная новизна работы:

1. Модели типового драйвера БИУС, позволившие разработать множество паттернов проектирования для формализации и использования накопленного в предыдущих проектах опыта.
2. Методика автоматизированного проектирования драйверов БИУС как множеств сертификационных артефактов с использованием паттернов, позволяющая повысить эффективность проектирования.
3. Методика создания виртуального стенда БИУС на базе программных моделей процессорных модулей, модельных версий драйверов устройств, а также программно-диагностического комплекса (ПДК), позволяющая распараллелить процессы разработки системного, прикладного и диагностического ПО.

Практическая значимость работы:

1. Система автоматизированного проектирования драйверов БИУС как множеств сертификационных артефактов, позволяющая повысить эффективность проектирования за счет использования формализованного

накопленного опыта работы над завершенными проектами в форме базы паттернов проектирования.

2. Библиотеки программных моделей процессорных модулей, модельных версий драйверов, а также виртуальных устройств ПДК на базе модельных версий драйверов, позволяющие конфигурировать различные виртуальные стенды БИУС.
3. Множество из 41 паттерна для 237 исполнений драйверов БИУС.

Достоверность научных положений, выводов и рекомендаций подтверждена непротиворечивостью применяемых моделей и методов, результатами экспериментальных исследований и результатами успешной эксплуатации разработанных инструментальных средств.

Реализация и внедрение результатов. Разработанные в рамках данной работы методы и инструментальные средства автоматизированного проектирования применяются в ОАО «Ульяновское конструкторское бюро приборостроения» (УКБП) при разработке ПО индикаторов ИМ-8, ИМ-16-1, ИМ-16-1, ИМ-16М-1НЛ, ИМ-16М-4НЛ, ИМ-16-3, ИМ-14, ИМ-33Н, ИМ-44, ИМ-55, ИМ-50, ИМ-24-3 и модификаций блока БПВ-6 авиационного применения, в проектах систем EIU-100, КСЭИС-В1, БСК-26, БСК-28, БСК-17В-5, БСК-38, БИСК-А-1,БИСК-А-1В, СЭИ-32, СЭИ-226, КСЭИС-148, КСЭИС-148Е, КСЭИС-226, ИСРП-3, ИСРП-4-1, ИСРП-5, ИСРП-7, КИС-27СМ, ВВД-1, ИКСВП-42, СУОВО-В1-1, ВВС-А, ВВС-226 для вертолетов Ансат, Ка-226, Ка-32, Ми-38, Ми-171, и самолетов SSJ-100, Ту-204СМ, Ан-148, а также во вновь разрабатываемых проектах ИКБО ИМА,БСТО-ПМ для самолетов МС-21.

Акт, подтверждающий внедрение результатов работы, приведен в приложении 1 диссертационной работы.

Апробация работы проведена на конференциях:

1. Системы искусственного интеллекта и нейроинформатика. Международная конференция «Континуальные логико-алгебраические исчисления и нейроматематика в науке, технике и экономике – КЛИН–2006 г.».
2. II международная научно-практическая конференция «Опыт и проблемы внедрения систем управления жизненным циклом изделий авиационной техники», Ульяновск, 2010.
3. Всероссийская научно-практическая конференция «Устройства измерения, сбора и обработки информации в информационно-управляющих комплексах», Ульяновск, 2011.
4. Третья и четвертая российские научно-технические конференции «Информатика и вычислительная техника» ИВТ-2011, ИВТ-2012. Ульяновск.
5. Информационные технологии. Радиоэлектроника. Телекоммуникации (ITRT-2012): II международная научно-техническая конференция. Поволжский государственный университет сервиса. Тольятти.

6. Симпозиум с международным участием. Самолетостроение России. Проблемы и перспективы. Самара, 2012.

Публикации результатов работы. По теме диссертации опубликовано 16 печатных работ, в том числе: две в журнале списка ВАК, получены два свидетельства об официальной регистрации программ для ЭВМ №2006611950 и №2009610501.

Структура и объем работы. Диссертационная работа состоит из введения, четырех глав и заключения, содержит 197 страниц, 20 таблиц, 18 рисунков, список литературы и приложения.

СОДЕРЖАНИЕ РАБОТЫ

Во введении рассмотрена актуальность темы диссертации, указаны цели и задачи работы, объект, предмет и методы исследования, обоснована достоверность научных положений, выводов и рекомендаций.

Первая глава содержит анализ БИУС реального времени авиационного применения, анализ драйвера как объекта проектирования, анализ существующих инструментальных средств разработки драйверов БИУС, а также анализ жизненного цикла драйверов БИУС в соответствии со стандартом КТ-178В.

С учетом необходимости сертификации ПО БИУС по стандарту КТ-178В, призванного обеспечить необходимое качество ПО, драйвер рассматривается как взаимосвязанная совокупность десяти типов сертификационных артефактов: требования высокого уровня; описание архитектуры; требования низкого уровня; исходный код программных компонентов драйвера; таблицы трассируемости требований низкого уровня на описание архитектуры и требования высокого уровня; таблицы трассируемости требований низкого уровня на исходный код; исходный код модульных тестов и результаты модульного и комплексного тестирования. По экспертным оценкам около 70% всех трудозатрат приходится на разработку архитектуры, требований низкого уровня, исходного кода программных компонентов и модульных тестов. При этом данные артефакты являются результатами проектной деятельности, и в настоящее время их разработка менее всего автоматизирована.

По результатам анализа самых распространенных из существующих инструментальных средств разработки драйверов таких, как

- Windows Driver Development Kit (DDK), поставляемый Microsoft
- LynxOS-178 Cross Development Kit (CDK), поставляемый LynuxWorks
- Wind River Workbench, поставляемый WindRiver
- QNX Momentics, поставляемый QNX

был сделан вывод о том, что на современном международном рынке ПО отсутствуют системы автоматизированного проектирования драйверов, удовлетворяющие следующим требованиям:

- поддержка интеграции автоматизированных средств разработки драйверов с системами управления требованиями, системами управления версиями файлов, системами учета запросов на изменение;

- возможность накопления и использования формализованного успешного проектного опыта;
- поддержка проектирования сертификационных артефактов в соответствии со стандартом КТ-178В.

Соответственно, необходимо разработать методы и средства, устраняющие выявленные недостатки и, как следствие, повышающие эффективность проектирования драйверов БИУС. Согласно стандарту КТ-178В разработанные инструментальные средства должны быть квалифицированы.

Во <u>второй главе</u> разработана структурная модель драйвера как множества сертификационных артефактов. Сертификационные артефакты, порождаемые в процессе проектирования драйвера по КТ-178В с учетом реальной практики проектирования, можно представить следующей структурной моделью (рис. 1):

$$\mathbf{D_i = <AD_i, DD_i, SC_i, TS_i, TE_i, TR_i>},$$

где i – идентификатор драйвера в ПО БИУС;

AD_i – архитектурные требования к поведению драйвера устройства;

DD_i – требования низкого уровня к поведению драйвера устройства;

SC_i – исходный код драйвера устройства;

TS_i – типовые решения, используемые в требованиях и коде драйвера;

TE_i – тестовое окружение драйвера устройства (модульные и комплексные тесты);

TR_i – результаты тестирования драйвера устройства.

Данная модель позволяет выделить типы артефактов, процессы преобразования артефактов одного типа в артефакты другого, возможные переходы между процессами и критерии этих переходов.

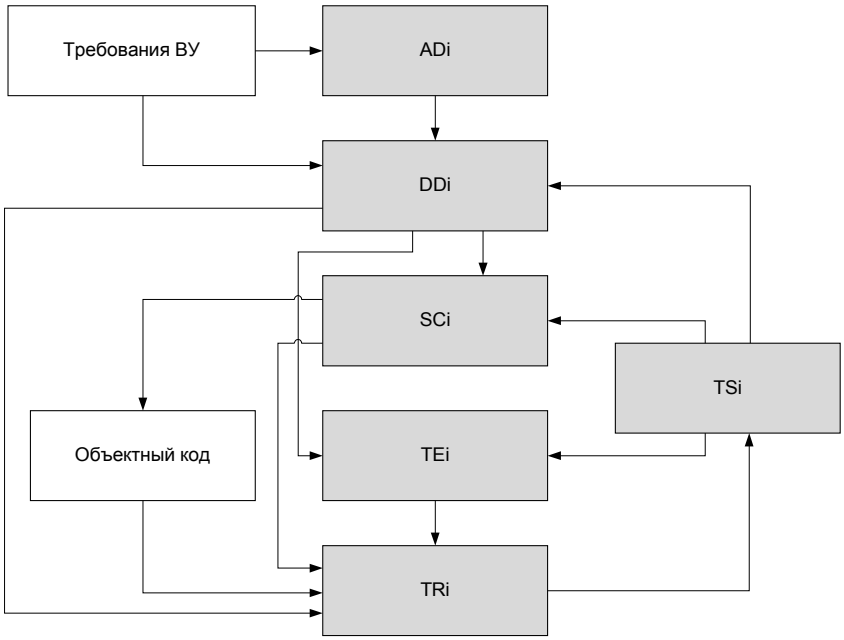


Рисунок 1. Модель типового драйвера как множества артефактов

Для автоматизации проектирования требований низкого уровня, исходного кода драйвера и его модульных тестов сформирована полная классификация существующих драйверов БИУС по таким критериям, как: тип обслуживаемого устройства; режим функционирования драйвера; уровень обслуживаемого устройства в общей топологической иерархии; уровень обслуживаемого устройства в логической иерархии; спецификация целевой операционной системы; спецификация целевой процессорной архитектуры; уровень реализуемого драйвером протокола.

Проведенные классификации драйверов и типов внешних устройств легли в основу спецификации драйвера БИУС в виде кортежа:

$$<\mathbf{EP}, \mathbf{IH}, \mathbf{RS}, \mathbf{MS}, \mathbf{IS}, \mathbf{RV}>,$$

где

EP (Entry Points) – точки входа драйвера;

IH (Interrupt Handlers) – обработчики прерываний;

RS (Register States) – состояния регистров обслуживаемого устройства;

MS (Memory States) – состояние памяти обслуживаемого устройства;

IS (Internal Statuses) – поля управляющей структуры драйвера;

RV (Returned Values) – возвращаемые точками входа значения;

PT (Processing Time) – время нахождения управления в драйвере.

Полная классификация точек входа **EP** драйвера БИУС состоит из точки входа инициализации устройства и драйвера; точки входа настройки режимов работы отдельных составляющих обслуживаемого устройства; точки входа чтения данных; точка входа записи данных; точки входа установки обработчика событий. Точки входа драйвера представляют собой реентерабельные функции независимого выполнения, что подразумевает возможность передачи управления в асинхронном и отложенном режимах.

Множество значений полей управляющей структуры **IS** имеет вид:

$$\mathbf{IS} \subseteq \mathbf{I} \times \mathbf{M} \times \mathbf{P} \times \mathbf{E} \times \mathbf{T},$$

где

I (Initialization Status) – степень инициализированности (по отдельным составляющим устройства);

M (Modes) – режимы работы отдельных каналов устройства;

P (Thread Priority) – приоритет выполнения кода драйвера;

E (Errors Statistics) – статистика событий, определяемых как неожиданное поведение устройства.

T (Processing Time) – множество значений времени нахождения управления в драйвере.

При этом спецификация используемой ОСЖРВ определяет однозначное соответствие элементов множества **P** элементам множества **T**, т.е.

$$p = F(t), \forall\, \mathbf{t} \in \mathbf{T}\ \exists\, \{\mathbf{p}\} \subseteq \{\mathbf{P}\}: |\{\mathbf{p}\}| \equiv 1.$$

Спецификация драйвера позволила построить функциональную модель драйвера БИУС как конечного автомата, состояниями которого являются элементы множества $\mathbf{RS} \times \mathbf{MS} \times \mathbf{IS}$, воздействиями – элементы множества

EP × **IH** × **RS** × **MS**, реакциями – элементы множества **RS** × **MS** × **RV**. Особенностью модели являются следующие ее свойства:

- воздействия в общем случае являются асинхронными (логические потоки: основной управляющий, обработки прерываний, изменения состояний регистров и памяти устройства внешними факторами без генерации прерываний);

- множество возможных состояний исчислимо, но мощность его слишком велика и специфична для каждого исполнения драйвера, что лишает смысла применение общих дискретных методов и полного статического анализа, т.е. представление функциональной модели драйвера в виде связного графа состояний автоматной модели;

- время функционирования точки входа является составляющей частью внутреннего состояния драйвера, но имеет особое значение, так как оно оказывает непосредственное влияние на важнейшую характеристику БИУС – время реакции на воздействия извне.

С учетом описанных особенностей автоматной модели драйвера ее функции перехода имеют следующий вид:

$$\mathbf{F}: \mathbf{f_i} \in \mathbf{F}: \mathbf{f_i} = \mathbf{RS} \times \mathbf{MS} \times \mathbf{IS} \times \mathbf{EP} \times \mathbf{IH} \rightarrow \mathbf{GS},$$

где

$$\mathbf{GS} = \mathbf{RS} \times \mathbf{MS} \times \mathbf{IS}.$$

Предложена трехзвенная структурная модель типового драйвера (рис. 2), что позволило типизировать его программные модули и сопутствующие им сертификационные артефакты (требования, исходный код модуля, исходный код модульных тестов). В данной модели определены следующие уровни программных модулей:

1) Прикладной уровень, ответственный за реализацию интерфейсов, предоставляемых прикладному ПО БИУС.

2) Уровень регистровой модели, ответственный за взаимодействие обслуживаемым устройством на уровне обращения к его регистрам и памяти.

3) Уровень физической среды, ответственный за реализацию механизмов доступа к адресам шины, на которой расположено обслуживаемое устройство.

Возможным подходом к автоматизации проектирования драйвера БИУС является использование параметризуемых кода и требований, однако, это приводит к содержанию неиспользуемого кода и не покрываемых требований в конкретном исполнении драйвера, что в соответствии со стандартом КТ-178В недопустимо. Предлагается в качестве метода автоматизации проектирования драйверов БИУС применить механизм паттернов.

Рисунок 2. Структурная модель типового драйвера

На основании полученных функциональной и структурной моделей типового драйвера БИУС, а также классификации его точек входа разработано множество паттернов проектирования, структурированных в виде трехуровневого дерева с типизацией вершин «Драйвер (1) – Программные модули (2) – Точки входа (3 - 17)»:

1. Паттерн «Драйвер».
2. Паттерн «Программный модуль драйвера».
3. Паттерн «Требование. Точка входа Инициализация».
4. Паттерн «Исходный код. Точка входа Инициализация».
5. Паттерн «Модульный тест. Точка входа Инициализация».
6. Паттерн «Требование. Точка входа Настройка режима».
7. Паттерн «Исходный код. Точка входа Настройка режима».
8. Паттерн «Модульный тест. Точка входа Настройка режима»
9. Паттерн «Требование. Точка входа Чтение».
10. Паттерн «Исходный код. Точка входа Чтение».

11. Паттерн «Модульный тест. Точка входа Чтение».
12. Паттерн «Требование. Точка входа Запись».
13. Паттерн «Исходный код. Точка входа Запись».
14. Паттерн «Модульный тест. Точка входа Запись».
15. Паттерн «Требование. Точка входа Установка обработчика».
16. Паттерн «Исходный код. Точка входа Установка обработчика».
17. Паттерн «Модульный тест. Точка входа Установка обработчика».

Представленные паттерны содержат информацию трех типов: конфигурация множества выгружаемых сертификационных артефактов (паттерны 1, 2), текст требований низкого уровня (паттерны 3, 6, 9, 12, 15), исходный код программных модулей и модульных тестов (паттерны 4, 5, 7, 8, 10, 11, 13, 14, 16, 17).

Каждый паттерн представляет собой параметризуемый расширяемый шаблон, из которого в автоматизированном режиме получаются соответствующие сертификационные артефакты. Разработанные структуры трех типов паттернов: драйвер, программный модуль, точка входа, представлены на рисунке 3. Для того чтобы не загружать рисунок, на нем представлена детализация только одной из дочерних вершин в каждом кусте. Остальные вершины имеют подобную же детализацию.

При этом структурное наполнение паттернов третьего типа «точка входа» основано на автоматной модели драйвера, функции выхода которой соответствуют его состоянию. Паттерн представляет собой совокупность матрицы переходов по состояниям в точке входа и множества функций перехода в виде параметризованного исходного кода или текста требований низкого уровня.

Методика проектирования драйвера от момента получения требований высокого уровня до момента перехода к интеграции ПО с оборудованием БИУС с использованием множества разработанных паттернов представлена на рисунке 4. Она покрывает следующие процессы жизненного цикла драйверов БИУС: разработка описания архитектуры ПО, разработка требований низкого уровня к ПО, разработка исходного кода ПО и модульных тестов.

В связи со значительной трудоемкостью и длительностью разработки и отладки БИУС требуется максимально распараллелить процессы жизненного цикла БИУС. На основании структурной модели типового драйвера (рис. 2) предложена методика виртуализации БИУС. Она основывается на замене в структурной модели типового драйвера компонентов, начиная с уровня физической среды и ниже, на их программные модели. При этом остальные компоненты драйвера не меняются.

Комплект драйверов БИУС с виртуальным уровнем физической среды, дополненный моделями бортовых вычислителей, может быть сконфигурирован в виртуальный стенд БИУС для разработки и отладки функционального ПО. Физической средой виртуализации может выступать локально-вычислительная сеть на персональных компьютерах. Схема виртуализации стенда БИУС

представлена на рисунке 5. В составе виртуального стенда используется ПДК с виртуальными устройствами, скомпонованными с модельными версиями драйверов, вместо интерфейсных плат, используемых в составе ПДК полунатурного стенда. При этом на виртуальном и полунатурном стендах БИУС используется один и тот же комплект диагностического ПО.

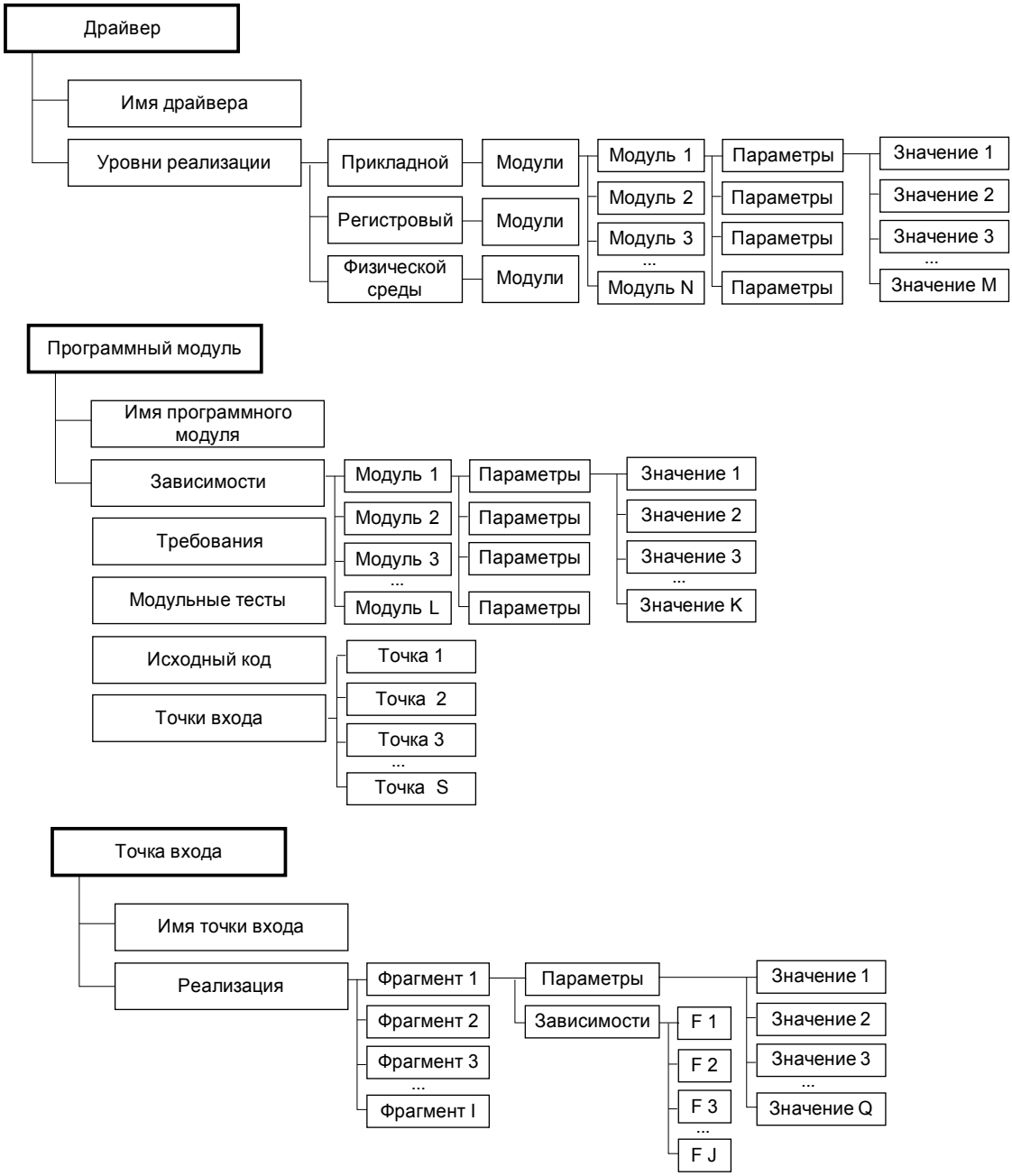


Рисунок 3. Структура паттернов трех типов

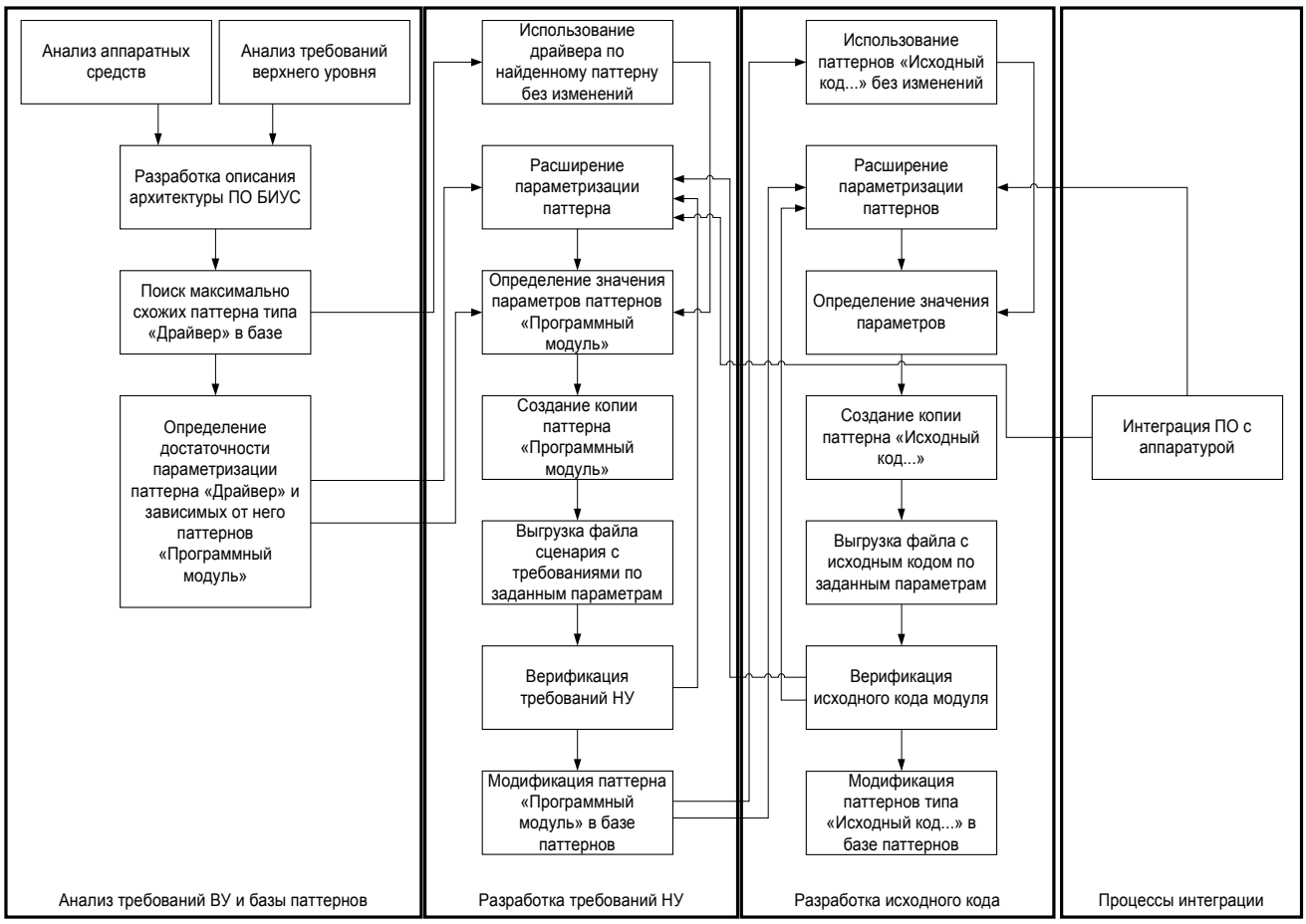


Рисунок 4. Методика разработки драйвера с использованием разработанных инструментальных средств и базы паттернов

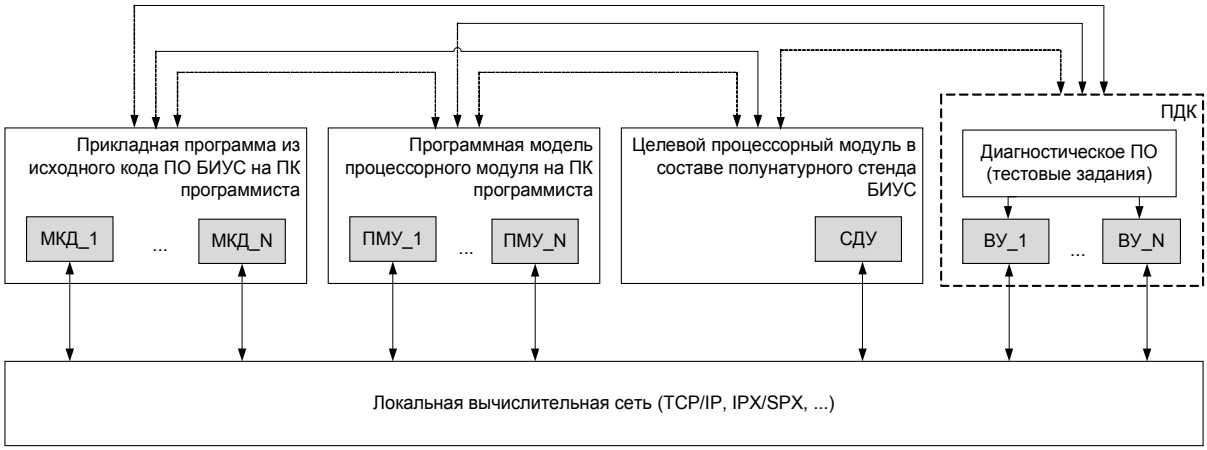


Рисунок 5. Схема виртуализации стенда БИУС

На рисунке 5 используются следующие обозначения: МКД – модельная конфигурация драйвера; ПМУ – программная модель устройства, реализующая логику его работы; СДУ – сервер доступа к корневой шине, на которой расположены устройства процессорного модуля; ВУ – виртуальное устройство – частный случай ПМУ, моделирует интерфейсные платы программно-

диагностического комплекса. Методика виртуализации позволяет распараллелить процессы разработки системного, функционального и диагностического ПО, существенно сокращает время разработки и отладки.

В третьей главе представлена реализация системы автоматизированного проектирования драйверов БИУС как множества сертификационных артефактов.

Разработанные функциональная модель типового драйвера БИУС, модель драйвера как множества сертификационных артефактов, структурная модель драйвера БИУС и иерархическая система паттернов позволили получить перечень требований, предъявляемых к свойствам представления проектных решений для драйвера БИУС:

- Формализованное описание проектных решений на уровне программных модулей и сопутствующих им сертификационных артефактов;
- Формализованные описания паттернов трех типов: требования низкого уровня, исходный код и конфигурация;
- Структура формализованного описания содержит деревья и списки;
- Необходимость поддержки наращиваемости механизмов описания структуры;
- Верифицируемость разветвленной базы формализованных описаний;
- Поддержка опциональных атрибутов формализованных описаний с возможностью определения значений атрибутов по умолчанию.

Информационной основой инструментальных средств является иерархическое множество паттернов, имеющих разработанные структуры (рис. 3). В качестве лингвистического обеспечения для представления разработанных паттернов используется подмножество языка XML без возможности определения дополнительных типов и использования расширений. В связи с тем, что база паттернов является по сути базой XML-файлов, в качестве системы управления используется система управления версиями файлов SubVersion.

Для быстрого поиска необходимых паттернов используется семантическое представление, учитывающее следующие факторы:

- условное имя протокола верхнего уровня согласно структурной модели драйвера;
- условное имя регистровой модели обслуживаемого устройства;
- условное имя физической среды функционирования драйвера.

Таблица соответствия имени файла в системе управления версиями семантическому имени хранится в специальном индексном файле под управлением той же системы управления версиями. Степень соответствия необходимого драйвера паттерну оценивается разработчиком вручную.

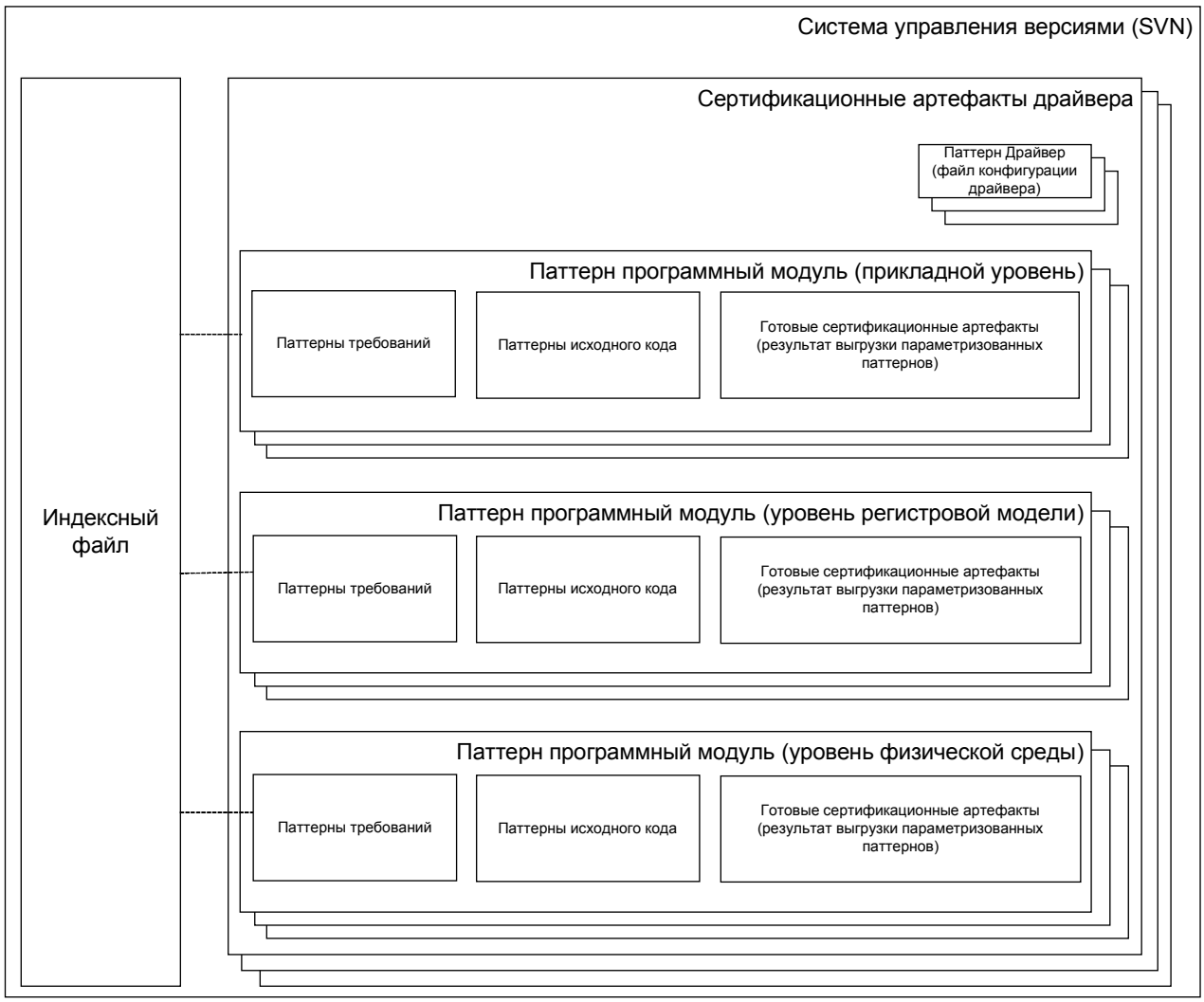


Рисунок 6. Структура базы паттернов

Инструментальные средства (рис. 7) разработаны согласно полученной методике и включают в себя следующие подсистемы:

- подсистема, осуществляющая доступ к индексному файлу паттернов в системе управления версиями посредством использования клиента системы управления и интерфейса командной строки и позволяющая разработчику просматривать список имеющихся паттернов по семантическому имени, стоящему из обозначения протоколов всех трех уровней модульной структуры драйвера согласно его функциональной модели;
- подсистема, позволяющая выгружать файлы паттернов для указанного модуля по его имени файла или семантическому обозначению и типу паттерна и создавать копию паттерна для конкретного проекта;
- подсистема, позволяющая средствами графического пользовательского интерфейса менять параметры для выбранного паттерна;
- подсистема, выгружающая файлы исходного кода модулей драйвера, а также файлы сценариев для системы управления требованиями IBM DOORS на языке DXL (IBM DOORS eXtensions Language).

Для данных подсистем была разработана библиотека анализа и генерации конструкций на выбранном подмножестве языка XML. Анализатор синтаксических конструкций выполнен в виде программного инициального конечного автомата.

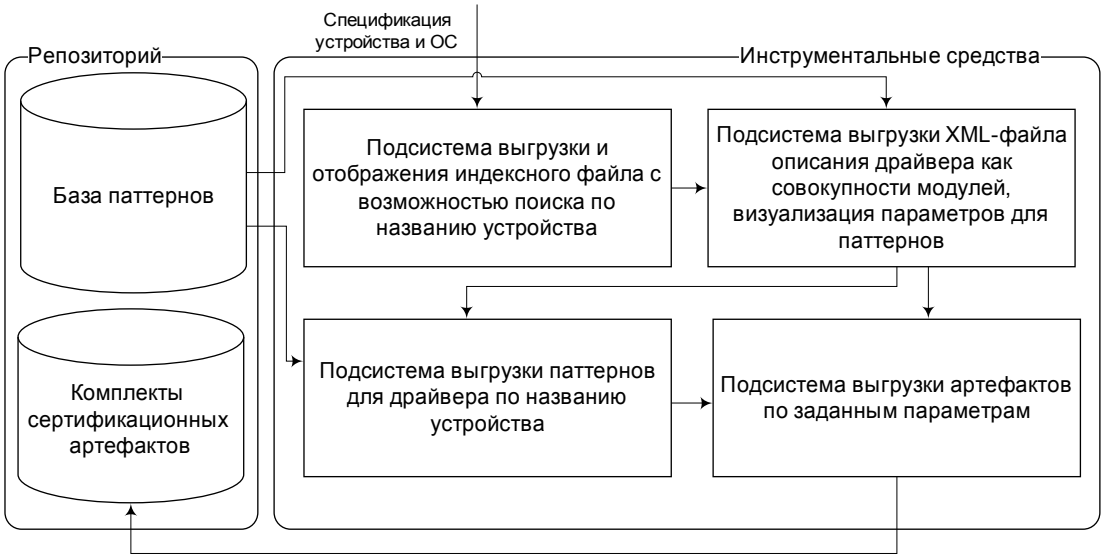


Рисунок 7. Схема разработанных инструментальных средств

Все подсистемы функционируют под управлением графической оболочки. Эти инструментальные средства автоматизируют следующие процессы: процесс разработки архитектуры ПО, процесс разработки требований низкого уровня, процесс разработки исходного кода драйверов и их модульных тестов. Эти процессы порождают артефакты из следующих множеств: $\mathbf{AD_i}$, $\mathbf{DD_i}$, $\mathbf{SC_i}$, $\mathbf{TS_i}$, $\mathbf{TE_i}$. На данный момент система инструментальных средств полностью подготовлена к квалификации в качестве средств автоматизированного проектирования БИУС.

В четвертой главе показана эффективность разработанных методик и инструментальных средств, подтвержденная результатами пяти групп экспериментов.

В рамках первых четырех групп экспериментов группа разработчиков проектировала комплекты драйверов разных уровней с разной полнотой множества необходимых сертификационных артефактов: полный набор (архитектура, требования, исходный код, модульные тесты), сокращенный набор (исходный код, модульные тесты). Эксперименты проводились в два этапа – на первом этапе проектирование производилось в ручном режиме, на втором этапе спустя месяц тот же комплект драйверов БИУС проектировался с использованием существующей базы паттернов в автоматизированном режиме. В качестве объекта проектирования были выбраны комплекты драйверов, указанные в таблице 1. В каждой клетке таблицы представлена разработка драйверов в хронологическом порядке. Оценка эффективности проводилась по временным затратам и количеству итераций.

В качестве оценки временных затрат на разработку рассматривалась комплексная величина T, формируемая следующими составляющими:

- T_1 - отношение количества пройденных при проектировании итераций к количеству этапов проектирования комплекта драйверов;
- T_2 - отношение времени, затраченного на получение первой версии исходного кода к всему времени разработки драйвера, нормированное по минимуму в группе;
- T_3 - отношение времени, затраченного на проектирование от момента получения первой версии требований высокого уровня до момента окончания приемо-сдаточных испытаний, показавших отсутствие ошибок в комплекте драйверов, к объему работ, нормированное по минимуму в группе.

	Полный набор сертификационных артефактов	**Сокращенный набор сертификационных артефактов**
Комплект драйверов уровня блока/индикатора	Группа I ИМ-14-Х, ИМ-16-Х, ИМ-33-Х, ИМ-55-Х	Группа II ИМ-8-Х, ИМ-16М-Х, ИМ-24-Х, ИМ-44-Х, ИМ-50-Х
Комплект драйверов уровня процессорного модуля	Группа III МФВ-1-01, МФВ-2-01, МФВ-2-02, МФВ-3	Группа IV МФВ-1-02, МФВ-1-04, МФВ-1-05, МФВ-1-06, МПР-26-Х

Таблица 1. Комплекты драйверов как объекты проектирования, используемые в летательных аппаратах Ансат, Ан-148, Ту-204СМ, Ми-171 и др.

Одно требование низкого уровня в соответствии с экспертными оценками считалось эквивалентным десяти строкам кода.

По результатам экспериментов были получены графики (рис. 8-10) и по результатам анализа каждой составляющей были сделаны следующие выводы:

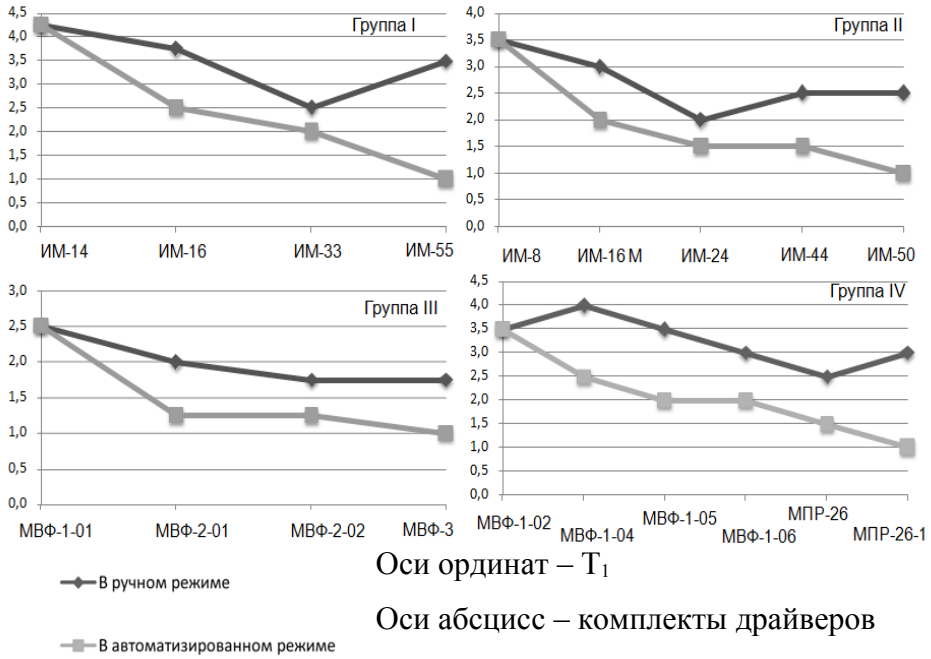
Для $\mathbf{T_1}$ (отношение количества пройденных при проектировании итераций к количеству этапов проектирования комплекта драйверов): уже на втором проекте эта величина для автоматизированного режима существенно меньше, чем для ручного. Более того, очевидно, что динамика убывания этой величины для автоматизированного режима по сравнению с ручным проектированием, во-первых, носит более постоянный характер, и, во-вторых, скорость убывания существенно выше при автоматизированном проектировании, чем при ручном.

Рассматриваемая величина для первого проекта в случае автоматизированного и ручного проектирования одинакова, так как отличие

при проектировании заключается только в том, что параллельно проектированию комплекта драйверов происходит наполнение базы паттернов.

В рамках проведенного эксперимента значение этой величины достигло единицы только в случае автоматизированного проектирования, т.е. база паттернов к последнему проекту эксперимента обладает достаточной полнотой.

Для $\mathbf{T_2}$ (отношение времени, затраченного на получение первой версии исходного кода к всему времени разработки драйвера, нормированное по минимуму в группе): во-первых, уже на втором-третьем проекте эта величина для автоматизированного проектирования становится меньше, чем для ручного, и, во-вторых, в случае с автоматизированным проектированием эта величина убывает заметно быстрее.



Оси ординат – T_1

Оси абсцисс – комплекты драйверов

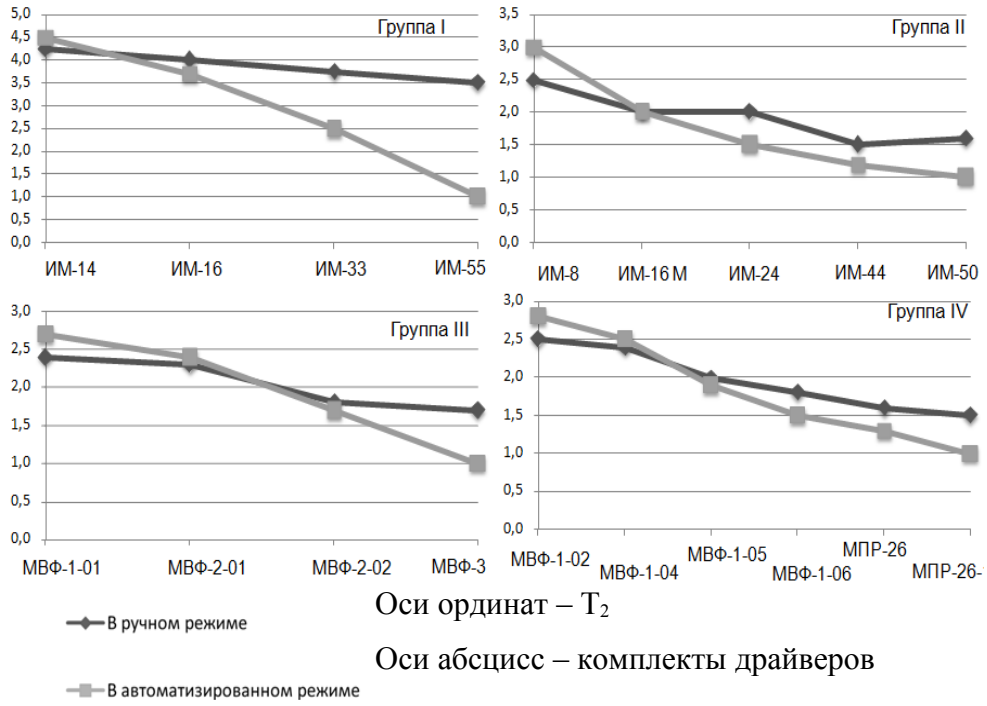
Рисунок 8. Отношение количества пройденных при проектировании итераций к количеству этапов проектирования комплекта драйверов

Издержки стартового режима, связанные с наполнением базы паттернов, начинают компенсироваться на втором проекте для первых двух групп (комплект уровня блоков/индикаторов) и на третьем проекте для последних двух групп.

Полученные графики наглядно демонстрируют одинаковый характер убывания рассматриваемой временной величины при ручном проектировании для всех групп при различном характере убывания для проектирования с использованием инструментальных средств автоматизированного проектирования. При проектировании комплектов драйверов с полным набором сертификационных артефактов рассматриваемая временная величина при автоматизированном проектировании имеет выпуклость вверх для комплектов обоих уровней (вторая производная отрицательна), а при проектировании комплектов с сокращенным набором сертификационных артефактов –

выпуклость вниз (вторая производная положительна). Разумеется, подобная аппроксимация имеет смысл только до достижения рассматриваемой величиной значения единицы.

Это означает, что нормированная скорость наполнения базы паттернов при проектировании комплектов с полным набором сертификационных артефактов выше, чем при проектировании комплектов с сокращенным. Однако даже во втором случае рассматриваемая временная безразмерная величина достигает своего нижнего предела на 4-6 проекте.



Оси ординат – T_2

Оси абсцисс – комплекты драйверов

Рисунок 9. Отношение времени, затраченного на получение первой версии исходного кода ко всему времени разработки, нормированное по минимуму в группе

Значение этой величины при ручном проектировании вообще не опускалось ниже 1.5. Более того, аппроксимация величины позволяет сделать вывод о том, что и в дальнейших проектах она не будет убывать. Значение 1.5 рассмотренной величины обусловлено опытом, накопленным участниками группы разработчиков.

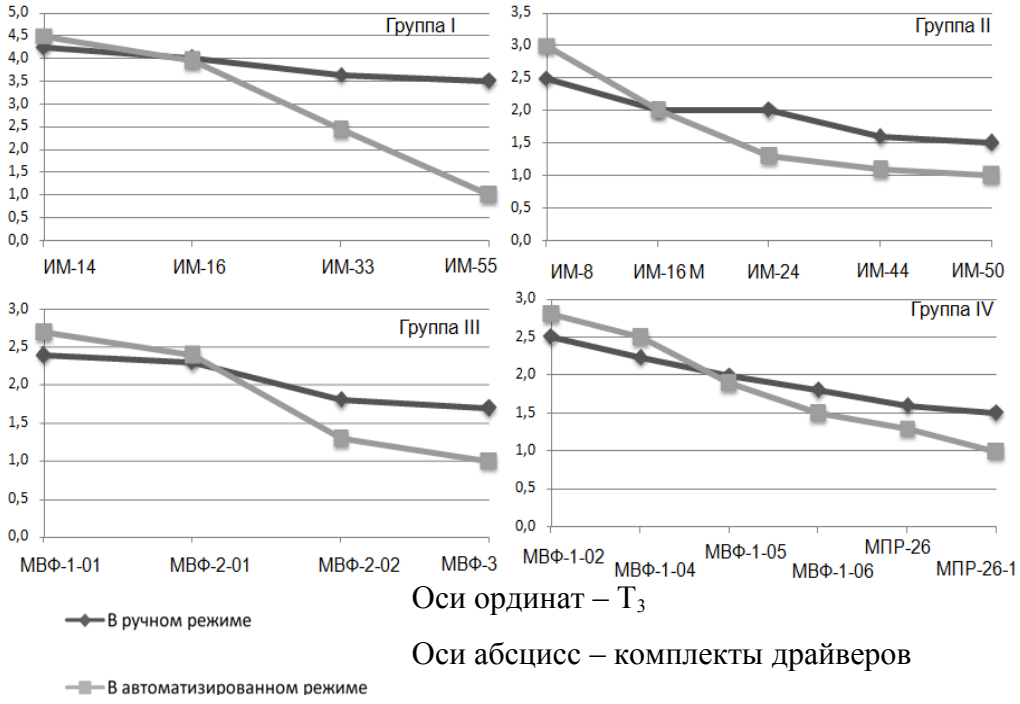
Важно, что это справедливо в равной степени как для комплектов драйверов с полным набором сертификационных артефактов, так и для комплектов с сокращенным набором. Это означает, что рассматриваемая величина свидетельствует об эффективности использования разработанных инструментальных средств для любых комплектов драйверов.

Для $\mathbf{T_3}$ (отношение времени, затраченного на проектирование от момента получения первой версии требований высокого уровня до момента окончания приемо-сдаточных испытаний, показавших отсутствие ошибок в комплекте драйверов, к объему работ, нормированное по минимуму в группе): данная

величина также меньше уже на третьем проекте, и также выше динамика убывания этой величины для автоматизированного режима, чем для ручного.

Также видно, что для комплектов уровня блока/индикатора стартовые издержки, связанные с наполнением базы паттернов, компенсируются уже на втором проекте, а для комплектов уровня процессорного модуля – на третьем.

К концу эксперимента база паттернов для всех четырех групп комплектов была достаточной для того, чтобы рассматриваемая временная величина достигла значения единицы. Для проектирования в ручном режиме эта величина не опускалась ниже 1.5, а в случае с комплектами блока/индикатора с полным набором сертификационных артефактов и вовсе остановилась на 3.5.



Оси ординат – T_3

Оси абсцисс – комплекты драйверов

Рисунок 10. Отношение времени, затраченного на проектирование от момента получения первой версии требований высокого уровня до момента окончания приемо-сдаточных испытаний, показавших отсутствие ошибок в комплекте драйверов, к объему работ, нормированное по минимуму в группе

Результаты экспериментов для всех составляющих показали преимущество использования разработанной методики проектирования драйверов БИУС: на 32%, 14% и 15% в среднем на всем множестве экспериментов соответственно.

Пятая группа экспериментов проводилась с четырьмя БИУС и ПДК «Фрегат», используемым в ОАО «УКБП» для тестирования и диагностирования всех проектируемых БИУС. В рамках этой группы экспериментов ПО БИУС разрабатывалось в два этапа: на первом этапе с использованием полунатурного стенда, на втором этапе другой группой разработчиков с использованием программных моделей процессорного модуля, модельных версий драйверов БИУС, а также ПДК «Фрегат» с виртуальными устройствами на базе модельных версий драйверов. Были получены следующие результаты:

БИУС	Объем исх. кода / строки	Затраты на разраб. ПО / чел-часы	Объем диаг. ПО / строки	Затраты на диаг. ПО / чел-часы	Выигрыш от использ-я вирт. стенда / чел-часы	Выигрыш от использ-я вирт. стенда / %
КСЭИС-В1 (Ми-171А2)	≈530000	≈1000	≈4000	≈250	≈400	32
EIU-100 (SSJ-100)	≈400000	≈770	≈2500	≈200	≈370	38
БСК-17В-5 (Ми-38)	≈500000	≈870	≈3200	≈220	≈380	34
БИСК-А-1 (Ансат)	≈380000	≈640	≈2000	≈180	≈360	44
ИСРП-4-1 (Ту-204СМ, Ми-38)	≈600000	≈1140	≈5000	≈290	≈500	34

В заключении приведены основные результаты и научная новизна диссертационной работы, сведения об апробации и публикациях, внедрении результатов работы.

В приложении содержится акт о внедрении результатов работы.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ

Основными результатами работы являются:

1. Модели типового драйвера БИУС, позволившие разработать множество паттернов проектирования для формализации и использования накопленного в предыдущих проектах опыта.
2. Структуры паттернов, позволившие автоматизировать использование ранее накопленного успешного опыта проектирования драйверов БИУС, а также множество из 41 паттерна для 237 исполнений драйверов БИУС.
3. Методика автоматизированного проектирования драйверов БИУС как множеств сертификационных артефактов с использованием паттернов, повышающая эффективность проектирования на этапах разработки архитектуры ПО, требований низкого уровня, исходного кода драйверов и их модульных тестов.
4. Методика создания виртуального стенда БИУС на базе программных моделей процессорных модулей, модельных версий драйверов устройств, а также программно-диагностического комплекса с использованием виртуальных устройств на базе модельных версий драйверов устройств, позволившая распараллелить разработку системного, функционального и диагностического ПО БИУС.
5. Система автоматизированного проектирования драйверов БИУС как множества сертификационных артефактов, позволяющая накапливать и

использовать формализованный опыт работы над завершенными проектами в форме базы паттернов проектирования.

6. Программные модели процессорных модулей, модельные версии драйверов устройств, а также виртуальные устройства программно-диагностического комплекса на базе модельных версий драйверов, позволяющие организовать виртуальный стенд БИУС с использованием одного комплекта диагностического ПО для виртуального и для полунатурного стендов БИУС, позволившие реализовать методику виртуализации стенда БИУС.

Список публикаций:

В изданиях, входящих в список ВАК:

1. Долбня Н.А., Шишкин В.В., Черкашин С.В. Универсальная система диагностирования бортового радиоэлектронного оборудования. // Известия Самарского научного центра Российской академии наук. Том 11 №3(2) (29), 2009 – Самара: Самарский научный центр РАН, 2009. с. 392-397.
2. Долбня Н.А., Ларин К.В., Шишкин В.В. Методика создания виртуального стенда авиационной бортовой информационно-управляющей системы. // Автоматизация процессов управления. №3 (29) 2012 – Ульяновск: Научно-производственное объединение «Марс». с. 36-41.

В других изданиях:

1. Долбня Н.А., Шишкин В.В., Черкашин С.В. Автоматизированная система создания диагностического обеспечения систем электронной индикации летательных аппаратов Вестник УлГТУ N2, 2006. –Ульяновск: УлГТУ, 2006. с. 55-58.
2. Шишкин В.В., Долбня Н.А. Комплекс диагностирования систем электронной индикации // МНК КЛИН-2006, т.2 «Информатика, системы искусственного интеллекта и моделирование технических систем», Ульяновск, 2006. с. 139.
3. Долбня Н.А. Об одном подходе к разработке программного обеспечения в авиационном приборостроении. // Информационные технологии. Межвузовский сборник научных трудов. – Ульяновск: УлГТУ, 2008. с. 53-54.
4. Долбня Н.А. Обзор подходов и методов разработки драйверов в linux-системах // Тезисы докладов 43-й научно-технической конференции. – Ульяновск: УлГТУ, 2009. с. 266.
5. Долбня Н.А. Обзор подходов и методов автоматизации разработки драйверов в linux-системах. // Корпоративная культура : от теории к практике. Сборник научных трудов. – Ульяновск: УлГУ, 2009. с. 167-169.
6. Dolbnya N.A. How to choose RTOS for embedded systems. // Роль иностранного языка в научной, профессиональной и межкультурной коммуникации. Материалы межвузовской конференции. Ульяновск: УлГУ, 2009. с. 27
7. Долбня Н.А. Критерии оценки безопасности ОСРВ авиационного применения.2009 г. // Проведение научных исследований в области

обработки, хранения, передачи и защиты информации. Сборник научных трудов т.2. – Ульяновск: УлГТУ, 2009. с. 195-198

8. Долбня Н.А. Разработка инструментальных средств для разработки драйверов бортовых встраиваемых систем авиационного применения. // Актуальные проблемы физической и функциональной электроники. Материалы 12-й региональной научной школы-семинара. – Ульяновск: УлГТУ, 2009. с.97-98
9. Долбня Н.А., Акимова Т.Е. Технология разработки и сопровождения программного обеспечения бортовых авиационных систем по стандарту КТ-178В.// Опыт и проблемы внедрения систем управления жизненным циклом изделий авиационной техники. Материалы научно-практической конференции – Ульяновск, 2010. с.26-30
10. Шишкин В.В., Долбня Н.А. Методика автоматизированного проектирования драйверов бортовых информационных систем под управлением UNIX-подобных операционных систем жесткого реального времени // Устройства измерения, сбора и обработки информации в информационно-управляющих комплексах. Тезисы докладов всероссийской научно-практической конференции. - Ульяновск, 2011. с. 120-123
11. Долбня Н.А. Анализ процесса разработки сертифицируемых драйверов бортовых информационных систем.// Информатика и вычислительная техника. Сборник научных трудов 3-й Российской научно-технической конференции ИВТ-2011 – Ульяновск: УлГТУ, 2010. с.199-205
12. Долбня Н. А., Шишкин В. В. Задачи автоматизации проектирования драйверов устройств бортовых информационно-управляющих систем, сертифицируемых по КТ-178В // Информационные технологии. Радиоэлектроника. Телекоммуникации (ITRT-2012): сб. ст. II международной заочной научно-технической конференции. Ч. 2 / Поволжский гос. ун-т сервиса. – Тольятти : Изд-во ПВГУС, 2012. – с. 13-19
13. Долбня Н.А. Система автоматизированного проектирования драйверов устройств бортовых информационно-управляющих систем под управлением операционных систем жесткого реального времени // Информатика и вычислительная техника. Сборник научных трудов 4-й Российской научно-технической конференции ИВТ-2012 – Ульяновск: УлГТУ, 2012. с.171-176
14. Долбня Н.А. Встроенные средства контроля бортовой вычислительной системы под управлением операционной системы реального времени как итеративный агрегированный объект //Самолетостроение России. Проблемы и перспективы: материалы симпозиума с международным участием. – Самара: СГАУ, 2012

Получены свидетельства об официальной регистрации программ для ЭВМ:

1. Программно-диагностический комплекс «Фрегат» / В.П. Деревянкин, Н.А. Долбня, В.И. Кожевников, Н.Н. Макаров, С.В. Черкашин // Свидетельство №2006611950, М.: Роспатент, 07.06.2006.

2. Интерпретатор языка тестовых заданий TMAKE / С.В. Черкашин, Н.А. Долбня, Н.Н. Макаров, В.И. Кожевников, В.П. Деревянкин // Свидетельство №2009610501, М.: Роспатент, 21.01.2009.

ДОЛБНЯ НИКОЛАЙ АЛЕКСЕЕВИЧ

РАЗРАБОТКА И ИССЛЕДОВАНИЕ МЕТОДОВ И ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ СЕРТИФИЦИРУЕМЫХ ДРАЙВЕРОВ АВИАЦИОННЫХ БОРТОВЫХ ИНФОРМАЦИОННО-УПРАВЛЯЮЩИХ СИСТЕМ

Автореферат
Подписано в печать 23.11.2012. Формат 60x84 1/16.
Бумага офсетная. Печать трафаретная. Усл. печ. л.
Тираж 100 экз. Заказ №
Типография УлГТУ. 432027, Ульяновск, ул. Северный Венец, 32.