УДК 629.78.018

Б.Б. МИХНИЧ, И.Б. ТУРКИН

Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ", Украина

РЕИНЖИНИРИНГ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПОДСИСТЕМЫ ОБРАБОТКИ ТЕЛЕМЕТРИЧЕСКОЙ ИНФОРМАЦИИ ЦЕНТРА УПРАВЛЕНИЯ ПОЛЁТОМ

Показана целесообразность применения технологии Windows Workflow Foundation при решении задач реинжиниринга средств автоматизации обработки телеметрической информации космических аппаратов. Данная технология позволяет технологу непосредственно описать ход вычислительного процесса, используя базовые исполняемые блоки или шаблоны в виде рабочих потоков. Использование конечного автомата, для моделирования процесса управления, позволяет разделить режимы функционирования на отдельные группы и производить их модификацию независимо друг от друга. Применение такого подхода предоставляет возможность накапливать знания об алгоритмах и упрощает дальнейший реинжиниринг разрабатываемой системы

Ключевые слова: *реинжиниринг, Windows Workflow Foundation, state machine, конечный автомат, рабочие потоки, автоматизация испытаний*

Введение

Объем написанного программного обеспечения (ПО) в мире постоянно растет (по разным оценкам от 120 до 700 млрд. строк кода [1]). С одной стороны, это обусловлено тем, что постоянно разрабатываются все новые и новые программы, но еще важнее то, что однажды созданные программы крайне медленно выходят из обращения. Одной из центральных проблем программной инжинерии становится сопровождение и эволюция ПО. Исследования показывают, что от 67 до 80% всех затрат жизненного цикла программы приходится именно на этап сопровождения [1].

В космической отрасли проблема реинжиниринга стоит особенно остро, поскольку существует весомый багаж накопленных решений, но в силу длительного технологического цикла разработки и единичной спецификации продукта требуется постоянная модификация частично устаревшего ПО с обеспечением высокой гарантоспособности.

1. Обзор методов реинжиниринга

В тех случаях, когда программная система становится трудной в сопровождении, но все еще не потеряла своей экономической ценности, необходимо предпринять какие-то действия по ее улучшению. Одним из возможных путей выхода из этого кризиса является реинжиниринг программного обеспечения (software reengineering) [2], т.е. изучение и изменение существующей системы с целью представления ее в новой, улучшенной форме, а также последующей реализации этой формы.

Реинжиниринг представляет собой систематическую трансформацию существующей системы с целью улучшения ее характеристик качества, поддерживаемой ею функциональности, снижения стоимости и трудоемкости ее сопровождения, вероятности возникновения значимых для заказчика рисков.

Можно выделить четыре основных фактора, влияющие на выбор между реинжинирингом, новой реализацией и дальнейшим сопровождением существующей системы:

– соотношение между стоимостью реинжиниринга, стоимостью новой реализации и дальнейшего сопровождения;

– соотношение между ценностью системы после реинжиниринга, новой системы и существующей;

– соотношение между факторами риска реинжиниринга, новой реализации и сохранения системы в прежнем состоянии;

– соотношение между предполагаемым временем жизни системы в случае реинжиниринга, реализованной заново системы и существующей на данный момент.

Одной из наиболее распространенных форм реинжиниринга являются языковые преобразования (language conversion), подразумевающие преобразование устаревших программ в эквивалентные им по функциональности программы на том же или другом языке высокого уровня [3]. Однако такой под-

© Б.Б. Михнич, И.Б. Туркин

ход не всегда позволяет получить программы приемлемого качества на целевом языке.

Одной из задач, решаемых в ходе реинжиниринга: *возвратное проектирование* (reverse engineering) – это процесс анализа рассматриваемой системы с целью идентификации компонент системы и их взаимодействий или с целью создания некоторого представления системы в другой форме на более высоком уровне абстракции [4]. Возвратное проектирование включает в себя такие подзадачи:

– понимание программ (program understanding) – исходящее из неявного предположения, что основным источником информации о системе обычно является исходный текст программ [5];

– редокументация (redocumentation), определяемая как процесс реорганизации системы, результатом которого является семантически эквивалентное представление системы на том же уровне абстракции;

– извлечение архитектуры (architecture extraction), определяемое как определение архитектурных решений по данной программной системе (например, по исходным текстам);

– извлечение проектных решений (design recovery) – идентификация значимых абстракций предметной области, причем более высокого уровня, чем те, что могут быть получены путем изучения самой системы.

Моделирование – обязательный атрибут процессов, в ходе которых реализуется реинжиниринг.

Другой актуальный вопрос реинжиниринга программ – это вовлечение человека в процесс трансформации устаревших систем. Потребность в участии человека связана с тем, что знания об устаревших системах постепенно теряются, и автоматическое восстановление таких знаний обычно не представляется возможным [6].

В космической промышленности, НИИ и КБ проблема реинжиниринга весьма актуальна. Широкий класс программно-технических комплексов, применяемых в космической индустрии, построен на основе опыта экспертов, результатов аналитических расчетов и стендового моделирования [7]. Для ПО, используемого на различных этапах жизненного цикла ракетно-космической техники, характерен широкий спектр программно-аппаратных платформ и зачастую далеко неполное соответствие программной документации существующей версии ПО, что затрудняет его сопровождение и практически делает невозможным его модернизацию.

2. Цели и задачи исследований

Целью данной статьи является обоснование и разработка методов реинжиниринга ПО обработки телеметрической информации (ТМИ) нового спутника микроспутника МС-2-8 с целью:

– повышения гибкости и адаптируемости алгоритмов в случае дальнейшего изменения конфигурации аппаратной части или формата представления пакетов ТМИ;

– перехода на новую программно-аппаратную платформу.

3. Результаты исследований

3.1. Требования к ПО подсистемы обработки телеметрической информации центра управления полетом

Рассматривается программный комплекс, который эксплуатировался на протяжении 7 лет и обладает широким функционалом. Основной задачей подсистемы обработки телеметрической информации наземной станции управления центра управления полетом (ПОТМИ НСУ ЦУП) спутника «Egyptsat-1» является прием и регистрация, автоматизированная обработка (рис. 1) и визуализация в табличном и графическом представлении достоверной информации о функционировании бортовых подсистем в процессе штатной эксплуатации.

Программное обеспечение, подвергаемое реинжинирингу ПОТМИ НСУ ЦУП состоит из двух подсистем:

– сервер ТМИ;

– рабочее место оператора (РМО) анализа ТМИ.

К основным функциональным группам состояний специального программного обеспечения (СПО) *Сервера приёма ТМИ* относятся (рис. 2):

– настройка пользовательского интерфейса;

– конфигурирование базы данных;

– приём и обработка телеметрической информации во время сеанса связи.

При этом каждая из вышеперечисленных групп состоит из строго определённого множества состояний, которое было задано ещё на этапе проектирования автоматизированной системы. Алгоритмы, выполняемые программой в каждом из состояний, разрабатываются отдельно и могут быть изменены в зависимости от конфигурации структурной обработки пакетов ТМИ.

К основным функциональным группам состояний *РМО анализ ТМИ* относятся (рис. 3):

– приём потока пакета ТМ файлов;

– обработка пакетов ТМ файлов;

– режим визуализации ТМ параметров;

– настройка сетевых интерфейсов и режимов отображения.

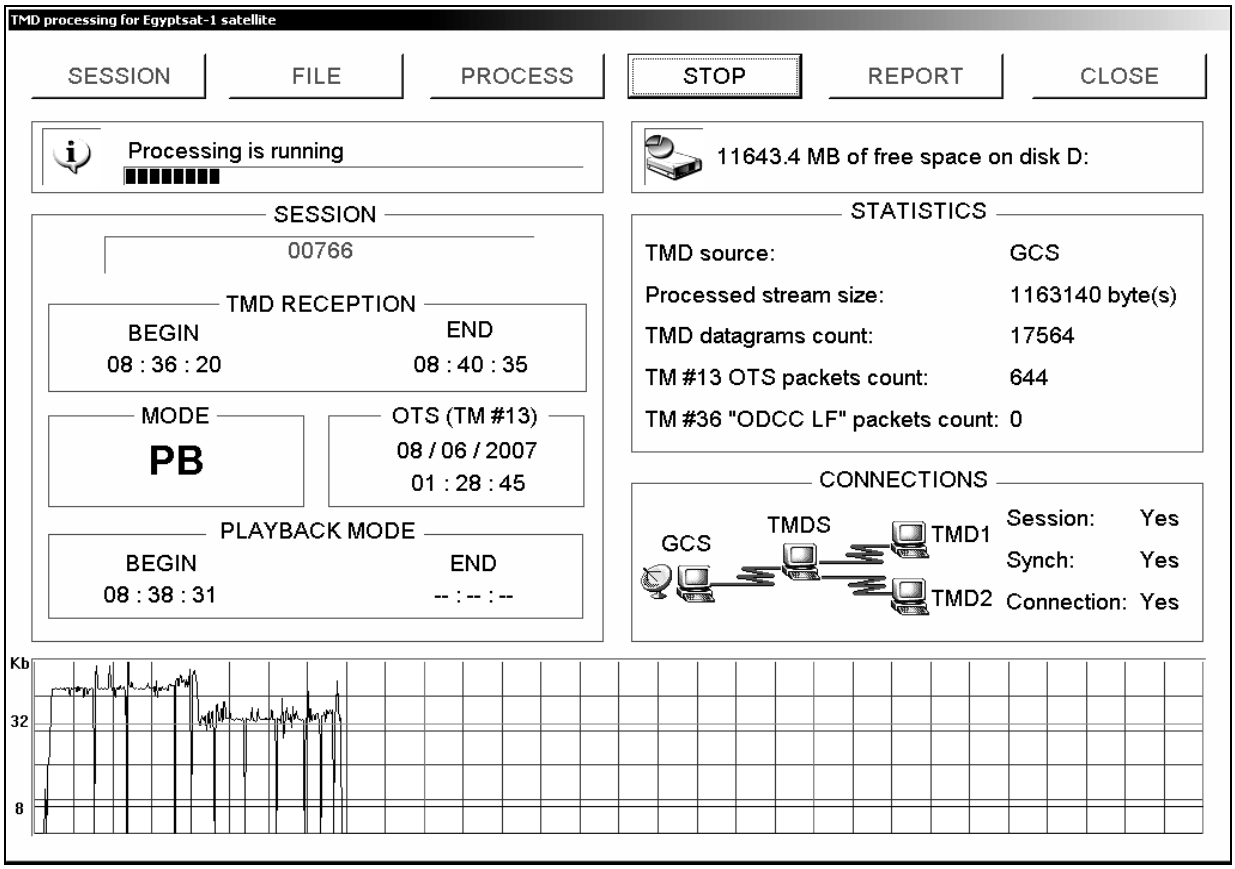

Рис. 1. Процесс приема, регистрации и автоматизированной обработки ТМИ на сервере ТМИ в режиме считывания пакетов телеметрии (ТМ)

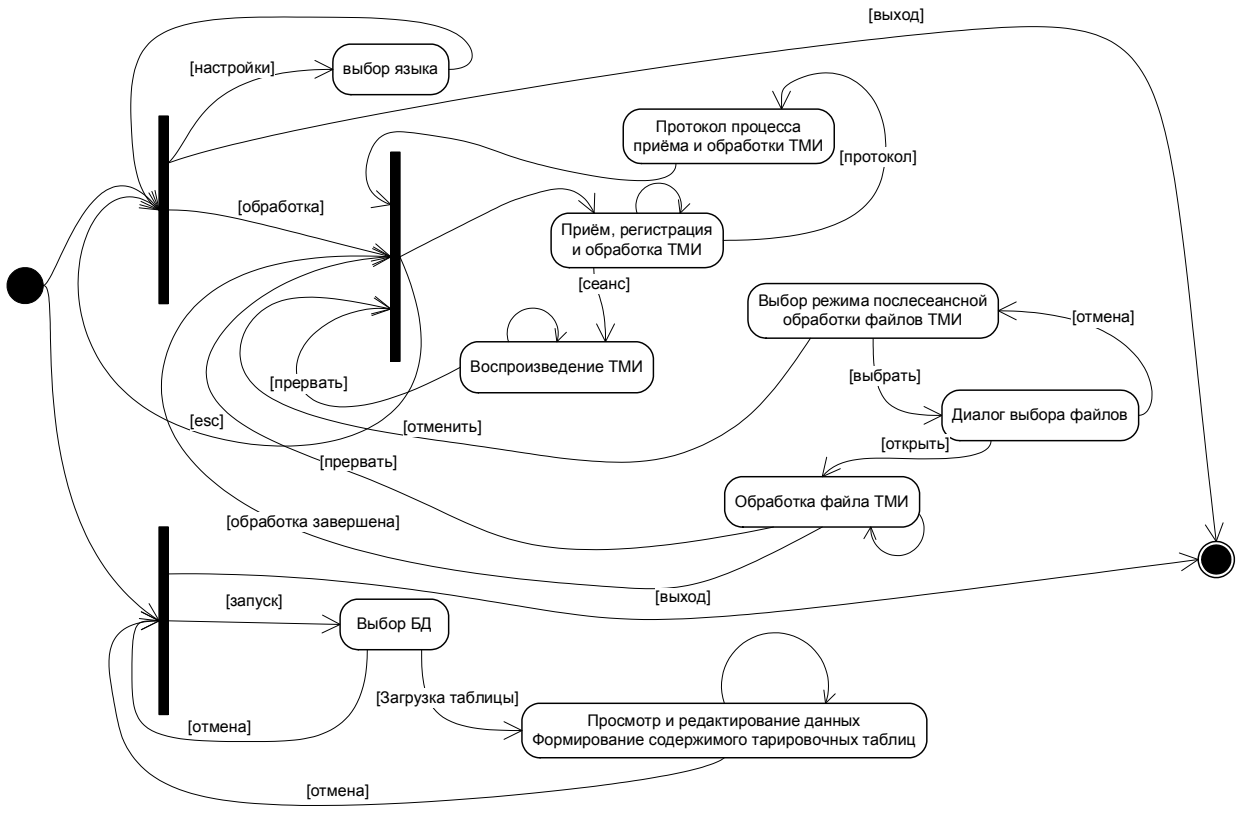

Рис. 2. Диаграмма состояний СПО Сервер ТМИ

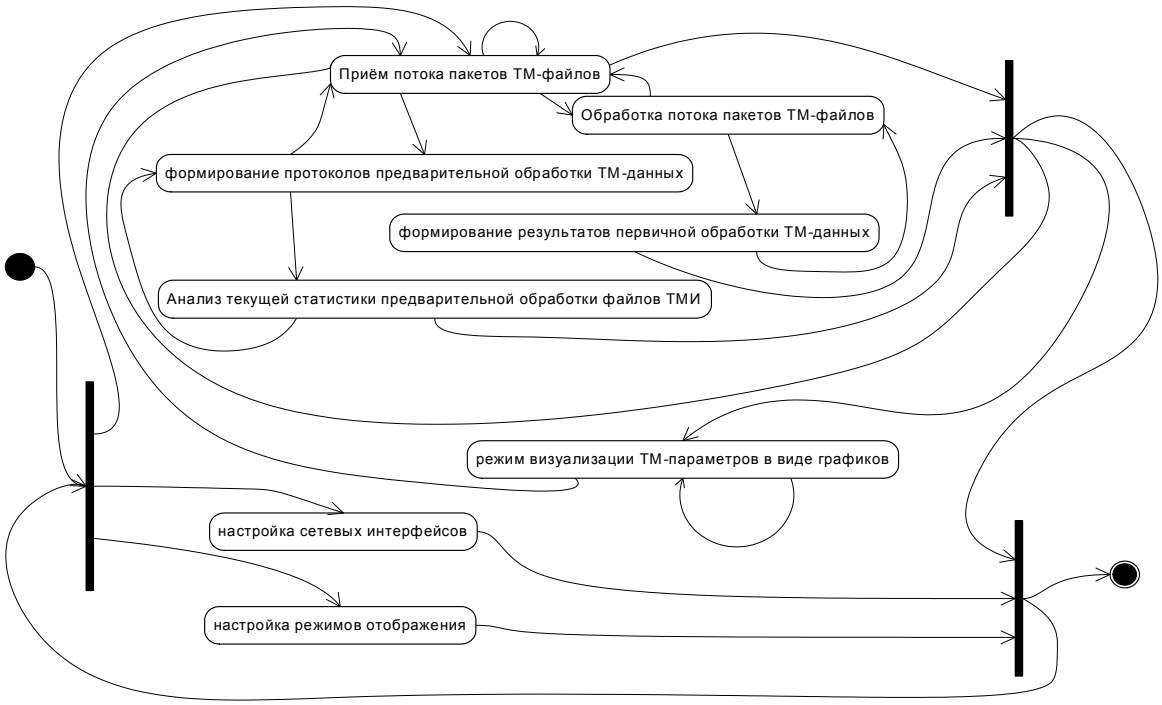

Рис. 3. Диаграмма состояний СПО РМО анализ ТМИ

Принципиальное различие подсистем Сервер ТМИ и РМО ТМИ состоит в источнике и формате получаемых ТМ файлов. Сервер ТМИ осуществляет обработку пакетов, полученных от спутника и сохраняет их в ТМ файлы, для дальнейшего анализа и визуализации подсистемой РМО ТМИ.

3.2. Принципы применения Windows Workflow Foundation

Инструменты для разработки ПО с использованием Windows Workflow Foundation (WF) доступны, начиная с платформы Microsoft .NET Framework версии 3.0 [8].

При помощи WF могут быть описаны три типа процессов:

– последовательный процесс (Sequential Workflow) – переход от одного шага в другой без возвратов обратно;

– конечный автомат (State-Machine Workflow) – переход из одного состояния в другое, возможны и произвольные возвраты в предыдущие состояния;

– процесс, управляемый правилами (Rules-driven Workflow) – частный случай последовательного процесса, в котором переход на следующий шаг определяется набором правил.

Процессы WF формируются из базового набора активностей (состоит из 28 активностей) и которые предоставляют возможность управления исполнением, построения циклов, распараллеливания, работу с исключениями, подсоединение к источникам данных, связывание с Web-службами.

Поддерживается возможность изменение рабочего потока во время работы (on-the-fly) путём создания объекта WorkflowChanges, содержащего все новые действия, которые нужно добавить к рабочему потоку, и вызова ApplyWorkflowChanges, определенный в классе WorkflowInstance, для фиксации этих изменений.

Конструктор Workflow Designer, используемый для проектирования рабочих потоков, не привязан к Visual Studio. При необходимости он может быть развёрнут в среде своего собственного приложения. Это даёт возможность поставлять систему, включающую рабочие потоки, и позволяющую пользователям настраивать ее под свои нужды. Можно предоставить пользователю пустой рабочий поток в качестве шаблона и обеспечить панелью инструментов, включающей специальные действия, которые отвечают предметной области. Затем пользователи самостоятельно смогут конструировать свои рабочие потоки, добавляя эти действия или разрабатывая собственные [8].

При выполнении реинжиниринга были использованы Инструменты Специализированных Языков Предметной Области (Domain-Specific Language-Tools, DSL Tools) [9] входящие в состав Microsoft Visual Studio 2008 SDK.

Специализированный язык предметной области (Domain-Specific Language, DSL) – это язык, разра-

ботанный для того, чтобы быть полезным для решения узкого специфического круга задач, в отличие от языков общего применения. Используя инструменты DSL Tools, могут быть созданы специализированные инструменты моделирования путем определения нового языка моделирования и его реализации. Например, возможно создание специализированного языка для описания интерфейса пользователя, бизнес процесса, базы данных или потоков информации и последующая генерации кода из этого описания. Инструменты Специализированных Языков Предметной Области могут быть использованы для построения индивидуальных визуальных редакторов приспособленных к любой предметной области.

3.3. Применение технологии Workflow для реинжинеринга СПО ПОТМИ ЦПУ

В процессе реинжиниринга СПО ПОТМИ была использована технология *State machine workflow* (.NET Framework 3.5).

В результате реинжиниринга СПО РМО анализа ТМИ был получен конечный автомат, изображённый на рис. 4, все состояния которого описываются последовательными процессами.

После запуска СПО РМО анализ ТМИ система находится в состоянии **выбор_режима** конечное состояние функционирования определено в положении **завершение**.

Из состояния **выбор_режима** конечный автомат может перейти в одно из следующих возможных состояний:

– **настройка_режимов_отображения** – позволяет сконфигурировать набор информационных характеристик ТМИ и способ их отображения в режиме визуализации;

– **настройка_сетевых_интерфейсов** – конфигурация подключений, для получения файлов ТМИ;

– **обработка_ТМИ_и_визуализация** – промежуточный режим, из которого возможны переходы в состояния:

a) **режима_визуализации**;

b) **приёма_потока_пакетов**;

c) **анализ** статистики приёма и протоколы.

Каждое состояние состоит из ряда рабочих потоков, каждый из которых имеет управляемое событием действие в начале, реализующее интерфейс *IEventActivity*, а затем некоторое количество других действий, которые формируют обрабатывающий код внутри состояния. В состоянии **выбор_режима** происходит начальная инициализация модулей управления и ожидание ответа пользователя для перехода в новое состояние (рис. 5).

Переход системы из состояния в состояние происходит посредством инициирования события в системе. Это делается либо за счет использования интерфейса и реализации этого интерфейса, и такая пара объектов называется внешней службой (external service), либо с использованием активности *setStateActivity*, которая инициирует событие перехода на указанное состояние. Переход в состояние **обработка_ТМИ_и_визуализация** происходит после выбора пользователем в состоянии **выбор_режима** соответствующего перехода и инициируется активностью setActivity5: {обработка_ТМИ_и_визуализация}**.**

В режиме **обработка_ТМИ_и_визуализация** реализована аналогичная схема, изображённой не рис. 5, выбора дальнейшего режима функционирования. Из этого состояние возможен переход в режим **приём_потока_пакетов**, Данный режим содержит один *EventDriven:{первичная_обработка}*, в котором выполняется рабочий поток изображёный на рис. 6.

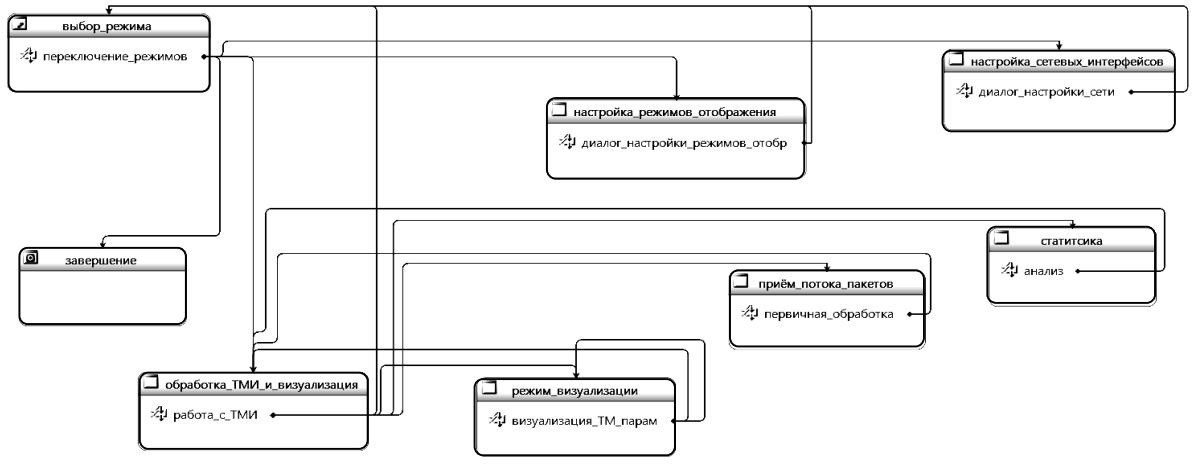

Рис. 4. State-Machine Workflow СПО РМО анализ ТМИ

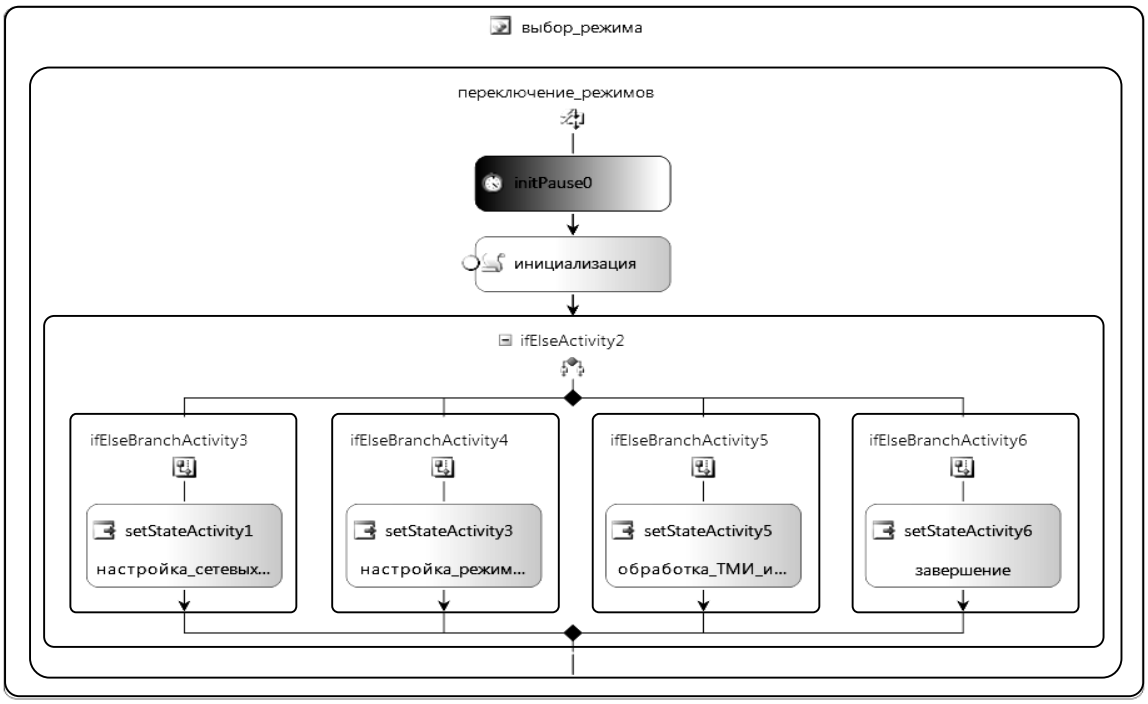

Рис. 5. Sequential Workflow состояния **выбор_режима**

Рабочий поток **первичная_обработка** (рис. 6) ориентирован на работу с пакетами телеметрической информации, которые СПО Сервер ТМИ получает от микроспутника и передаёт их по сети СПО РМО. EventDriven:{первичная_обработка} позволяет проводить распаковку и анализ ТМ-пакетов и состоит из следующих активностей:

– **delayActivity1** – таймер задержки, используется как вспомогательный элемент *EventDriven*, который первым реализует интерфейс *IEventActivity*;

– **whileActivity1** – цикл разбора последовательности пакетных данных, составное действие;

– **sequenceActivity1** – содержит последовательность активностей, используется, если в составном действии необходимо описать более одной активности;

– **распаковка_пакетов**, **анализ_пакетов**, **запись_в_БД** – простые активности, которые содержат набор инструкций, для работы с ТМ-пакетами;

– **set_State_Activity10** – инициирует переход системы в состояние **обработка_ТМИ_и_визуализация**, происходит после завершения цикла.

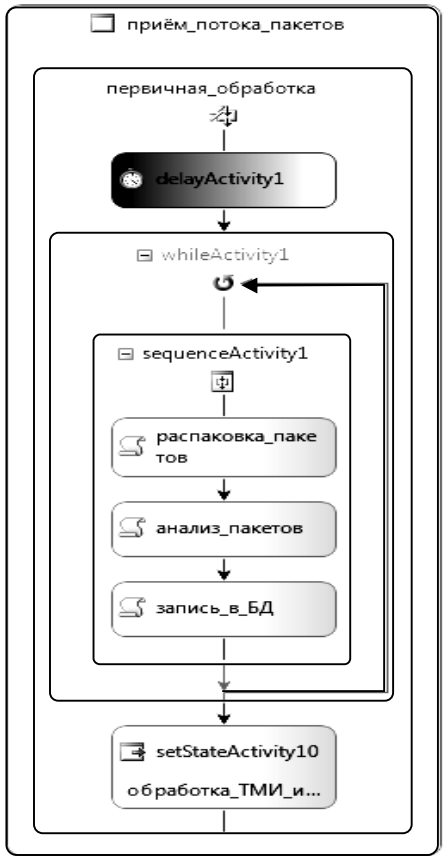

Рис. 6. Sequential Workflow состояния **приём_потока_пакетов**

Заключение

Таким образом, все процессы управления рассматриваемой автоматизируемой системы были разбиты на отдельные состояния, которые описываются набором рабочих потоков. Технология Workflow Foundation позволяет модифицировать данные рабочие потоки без изменения архитектуры системы, даже на этапе выполнения программы. Предложенный подход реинжиниринга программного обеспе-

чения обработки пакетов ТМИ позволяет выносить алгоритм управления во внешнюю базу данных и в дальнейшем легко модифицировать и оптимизировать алгоритмы разбора, адаптируя алгоритм для нового набора компонент КА.

Дальнейшее направление исследований будет проводиться в сторону автоматической генерации управляющего алгоритма.

Литература

1. Ulrich W. The evolutionary growth of reengineering and the decade ahead / W. Ulrich // American Programmer. – 2009. – Vol. 3, No. 11. – P. 14-20.

2. Tilley S.R. Perspectives on Legacy System Reengineering / S.R. Tilley, D.B. Smith // Software Engeneering Industry. – 2009. – Vol. 7, No. 17. – P. 150.

3. Друнин А.В. Построение срезов программ в задачах реинжиниринга / А.В. Друнин // Автоматизированный реинжиниринг программ: сб. С.-Петербургского ун-та. – СПб.:С.-Петербургского ун-та , 2000. – С. 184–205.

4. Фаулер Мартин Рефакторинг. Улучшение существующего кода / Мартин Фаулер. – СПб.: Символ-Плюс, 2008. – 432 с.

5. Biggerstaff T.J. Program Understanding and the Concept Assignment Problem / T.J. Biggerstaff, B.G. Mitbander // Communications of the ACM. – 1994. – P. 72–82.

6. Jones C. Estimating Software Costs / C. Jones. McGraw-Hill. – 1998 – 170 p.

7. Солнечные энергосистемы космических аппаратов. Физическое и математическое моделирование / К.В.Безручко, Н.В.Белан, Д.Г. Белов и др.; Под ред. С.Н. Конюхова. – Х.: Национальный аэрокосм. ун-т «ХАИ», 2000. – 513 с.

8. Нейгел Кристиан C# 2005 и платформа .NET 3.0 для профессионалов / Кристиан Нейгел, Билл Ивьен. – СПб.: Диалектика, 2008. –1376 с.

9. Domain-Specific Development with Visual Studio DSL Tools / Steve Cook, Gareth Jones, Stuart Kent, Alan Cameron Wills. – Addison Wesley, 2007. – 352 p.

Поступила в редакцию 10.11.2010

Рецензент: д-р техн. наук, проф., зав. каф. экономико-математического моделирования В.М. Вартанян, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

РЕІНЖИНІРИНГ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПІДСИСТЕМИ ОБРОБКИ ТЕЛЕМЕТРИЧНОЇ ІНФОРМАЦІЇ ЦЕНТРУ УПРАВЛІННЯ ПОЛЬОТАМИ

Б.Б. Міхнич, І.Б. Туркін

Показана доцільність застосування технології Windows Workflow Foundation при вирішенні завдань реінжинірингу засобів автоматизації обробки телеметричної інформації космічних апаратів. Ця технологія дозволяє технологу безпосередньо описати хід обчислювального процесу, використовуючи базові виконувані блоки або шаблони у вигляді робочих потоків. Використання кінцевого автомату, для моделювання процесу управління, дозволяє розділити режими функціонування на окремі групи і проводити їх модифікацію незалежно один від одного. Застосування такого підходу надає можливість накопичувати знання про алгоритми і спрощує подальший реінжиніринг системи, що розробляється.

Ключові слова: реінжиніринг, Windows Workflow Foundation, state machine, кінцевий автомат, робочі потоки, автоматизація випробувань

REENGINEERING SOFTWARE SUBSYSTEM PROCESSING TELEMETRY INFORMATION CENTER FLIGHT CONTROL

B.B. Mikhnich, I.B. Turkin

This paper shows feasibility of Windows Workflow Foundation technology in solving problems re-engineering tools automate the processing of telemetry data the spacecraft. This technology allows the technology itself to describe the course of the computational process, using the basic executable blocks or patterns in the form of workflows. Using finite-state machine to model the management process allows you to divide the modes of operation into separate groups and to make their modified independently of each other. Such an approach provides an opportunity to accumulate knowledge about algorithms and facilitates the further reengineering of the system being developed.

Key words: reengineering, Windows Workflow Foundation, state machine, finite-state machine, workflows, automation testing

Михнич Борис Борисович – аспирант каф. инженерии программного обеспечения, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: boris.mikhnich@rambler.ru.

Туркин Игорь Борисович – д-р техн. наук, проф., зав. каф. инженерии программного обеспечения, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: energy@d4.khai.edu.