УДК 629.78.018

Б.Б. МИХНИЧ, И.Б. ТУРКИН, Е.В. СОКОЛОВА

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Украина

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ИСПЫТАНИЙ СИСТЕМ КОСМИЧЕСКИХ АППАРАТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ WINDOWS WORKFLOW FOUNDATION

Рассматриваются особенности применения технологии Windows Workflow Foundation при разработке программного обеспечения для автоматизации стендовых испытаний подсистем космических аппаратов таких, как спутники. Показано, что применение данной технологии позволяет технологу непосредственно описать ход вычислительного процесса, используя базовые исполняемые блоки или шаблоны в виде рабочих потоков, что повышает информативность и упрощает дальнейшие модификации системы.

Ключевые слова: *Windows Workflow Foundation, рабочие потоки, автоматизация испытаний, космический аппарат.*

Введение

Одна из главных особенностей программного обеспечения (ПО) для автоматизации стендовых испытаний космических аппаратов (КА) заключается в необходимости частой модификации алгоритмов и типов проверок аппаратуры [1].

В настоящее время появляется много новых технологий разработки программного обеспечения, нацеленных на то, чтоб процесс разработки стал менее трудозатратен, а конечный продукт наиболее надежен и прост в сопровождении. Следует выделить на фоне всего многообразия технологию Windows Workflow Foundation (WWF), которая является ключевым компонентом платформы Microsoft .NET Framework, начиная с версии 3.0, поставляемой в составе Visual Studio. Технология WWF обеспечивает визуальное и декларативное построение потоков операций на основе управляемого кода.

Целью этой статьи является анализ возможности использования технологии WWF в задачах разработки ПО для автоматизации стендовых испытаний КА.

1.Специфика испытаний КА

Современные КА представляют собой сложнейшие автоматические комплексы и включают в себя 20...25 систем различных наименований и назначений: системы управления, телеметрии, ориентации, терморегулирования, антенно-фидерная и др. Сложность испытаний как этапа жизненного цикла КА объясняется еще и тем, что в это время не только контролируется правильность сборки изделия, проверяется исправность и логика функционирования его систем и компонентов, но и одновременно вводится в эксплуатацию контрольно-измерительная и контрольно-проверочная аппаратура, проверяется эксплуатационная документация, создается собственно технологический процесс испытаний [2].

Состояние современного КА определяется тысячами параметров, характеризующих тепловые, пневматические, гидравлические, химические, электрические явления, и сотни - тысячи команд требуется для управления. Интенсивность информационных потоков при испытаниях может достигать несколько мегабайт в секунду. Динамика переходных процессов различных систем КА колеблется от долей секунд до суток. В некоторых случаях скорость принятия решений по управлению изделием должна быть практически мгновенной.

Автоматизация стендовых испытаний ракетно-космической техники – одно из наиболее важных средств повышения ее надежности, наряду с резервированием, технической диагностикой и аварийной защитой.

2. Описание WWF

Инструменты для разработки ПО с использование WWF доступны, начиная с платформы Microsoft .NET Framework версии 3.0.

Рассматривая архитектуру WWF можно выделить несколько уровней (рис. 1). На нижнем находится родительский процесс (Host Process). Взаимодействие с родительским процессом осуществляется

посредством родительского уровня (Hosting Layer - Workflow). Его интерфейсы для ключевых процессов уровня исполнения WWF должны реализовать разработчики, интегрирующие WWF в свои приложения. Ядром технологии WWF является уровень исполнения (Runtime Engine), который отвечает непосредственно за workflow-моделирование и содержит необходимые для этого службы. Он же управляет жизненным циклом (менеджмент состояний и активация) моделей. На самом верху архитектуры WWF – уровень workflow-модели (Workflow Model Layer). Он отвечает за поддержку различных типов моделей, содержит предопределенные действия (activities) и реализует соответствующий API.

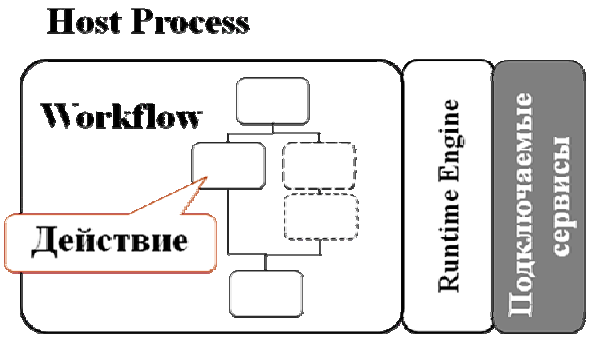

Рис. 1. Архитектура WWF

Характеризуя пользовательский интерфейс среды, отметим, что она включает встроенный визуальный редактор (рис. 2), благодаря которому можно конструировать модели, автоматически сохраняемые в XOML-файлах. При необходимости модели рабочих потоков могут быть реализованы исключительно посредством программного кода, поскольку фактически представляют собой обычные классы.

Одной из главных особенностей, характеризующих технологию WWF, является гибкая способность к расширению как уже существующих функциональных моделей, так и пут ем создания новых активностей. Отдельные элементы workflow-модели – действия (activities) – выполнены в виде компонентов, которые могут создаваться сторонними поставщиками с помощью Visual Studio. Помимо этого, WWF поддерживает динамическое обновление моделей во время исполнения, а также позволяет встраивать редактор моделей в собственные приложения.

WWF поддерживает два типа моделей – последовательные – sequential (рис. 2) и конечные автоматы – state machine. Первые представляют собой набор последовательно исполняемых действий и в наибольшей мере соответствуют классическим блок-схемам. Вторые позволяют моделировать переходы между несколькими предопределенными состояниями, что особенно актуально при конструировании сложных программных систем, например управления бизнес-процессами [3].

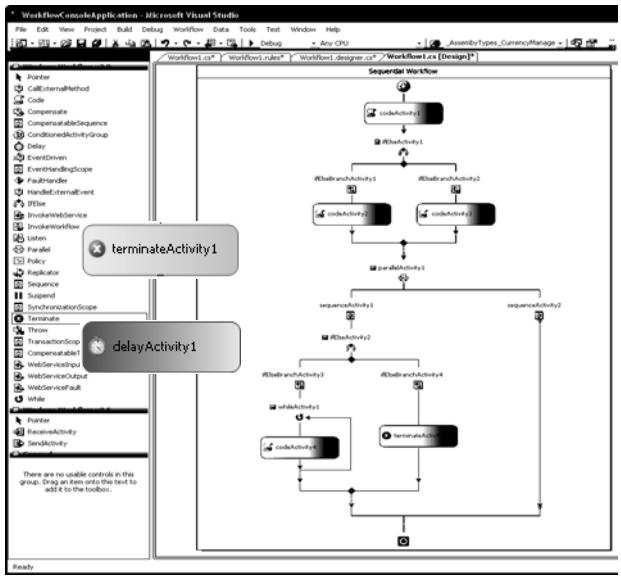

Рис. 2. Последовательная модель в WWF

Базовый набор активностей состоит из 28 активностей (на рис.2 слева) и включает в себя возможности управления исполнением, построения циклов, распараллеливания, работу с исключениями, подсоединение к источникам данных, связывание с Web-службами.

Исполняющая система Runtime Engine, предоставляет следующие службы запущенным рабочим потокам:

– *служба постоянства* – позволяет сохранять текущее состояние рабочего потока, длительное время находящегося в состоянии простоя с последующим возобновлением активностей;

– *служба отслеживания* – записывает события рабочего потока, которые были определены пользователем для отслеживания;

– *пользовательские службы* – создание пользовательских служб пут ем реализации интерфейса IDoorService.

Поддерживается возможность изменение рабочего потока во время работы (on-the-fly) пут ем 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 разв ернут в среде своего собственного приложения. Это да ет возможность поставлять систему, включающую рабочие потоки, и позволяющую пользователям настраивать ее под свои нужды. Можно пре-

доставить 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.

Таким образом, технология WWF позволяет разрабатывать как законченные пользовательские приложения, так и заготовки, внутренняя логика которых в дальнейшем может быть изменена самим пользователем без перекомпиляции всего проекта через встроенный редактор рабочих потоков. Эффективность применения данной технологии обеспечивается многими факторами:

– возможностями создание действий, характерных предметной области, и использования шаблонов действий;

– использованием графического редактора на этапе разработки и, при необходимости, встраиваемого в пользовательский интерфейс;

– визуализацией хода исполнения, возможностями сохранения состояний и отслеживания событий и т.д.

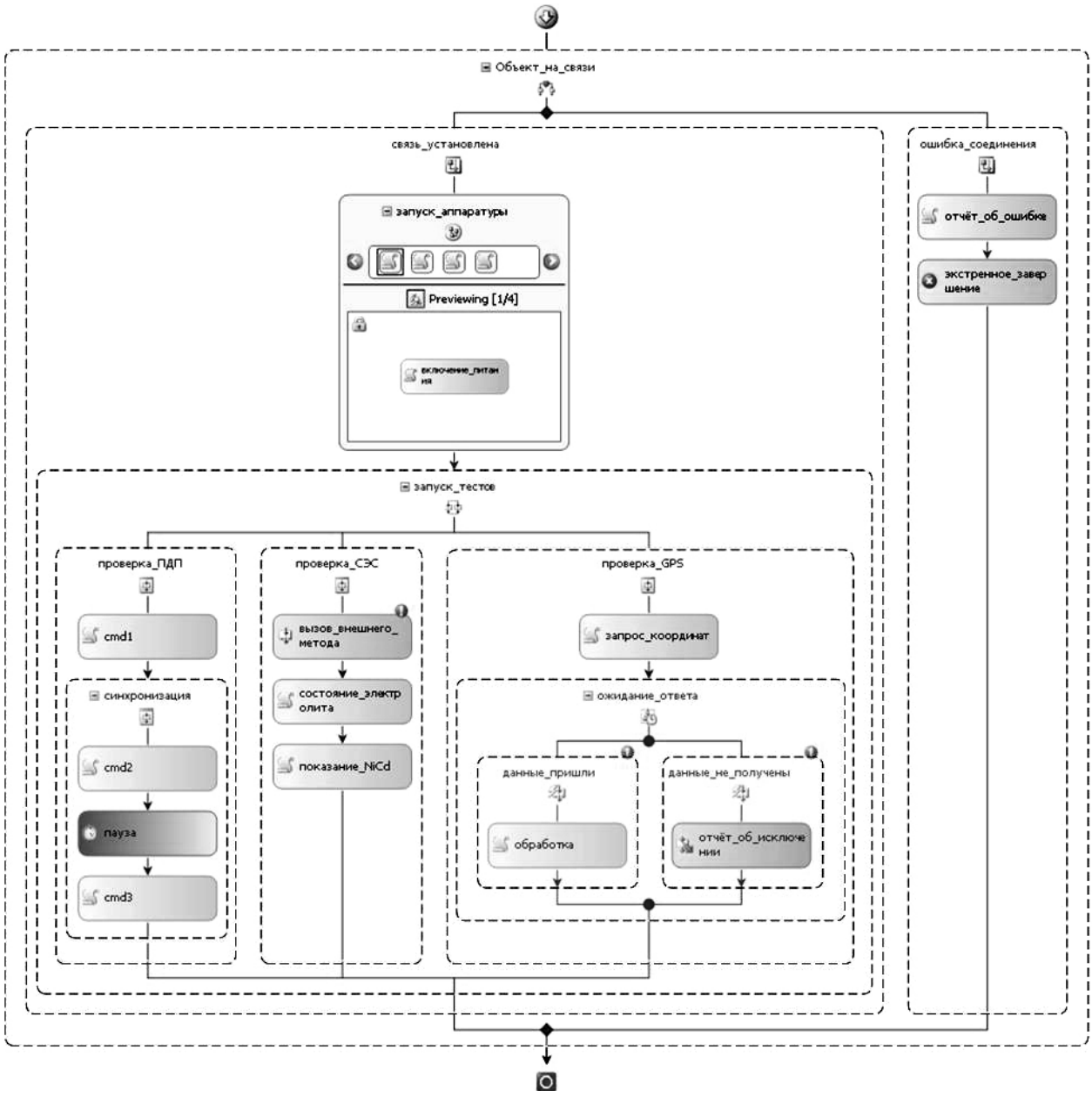

Рис. 3 Описание алгоритма стендовых испытаний КА с помощью WWF в Visual Studio 2008

3. Пример разработки ПО для автоматизации стендовых испытаний КА с использованием WWF

На рис. 3 представлен упрощенный фрагмент ПО, реализующий проведение стендовых испытаний КА. Сначала проводится проверка связи с текущим объектом испытаний, если связь установить не удается, то происходит формирование отчета об ошибке и генерация исключения; иначе запускается процесс испытаний, состоящий из нескольких этапов: сначала запуск аппаратуры (с ожиданием включение всех подсистем), потом параллельное обращение к различным подсистемам спутника – подсистеме данных платформы (ПДП), системе энергоснабжения (СЭС), GPS навигатора. Опишем логико-структурные элементы в той последовательности, в которой они представлены на схеме рабочих потоков (рис. 3):

– *объект_на_связи* – составное действие типа **IfElseActivity**, реализует выбор между ветвями *связь_установлена* и *ошибка_соединения* типа IfElseBranchActivity. Каждая ветвь также является составным действием, унаследованным от SequenceActivity – класса, выполняющего последовательность действий одно за другим;

– *отчет_об_ошибке* – элемент типа **CodeActivity**, содержит набор инструкций, для генерации отчета;

– *экстренное_завершение* – активность типа **TerminateActivity** осуществляет выход из приложения;

– *запуск_аппаратуры* – составное действие типа **ConditionedActivityGroup**, содержит ряд дочерних действий (возможны составные действия и установка дополнительных условий выполнения на каждое дочернее действие) и выполняет их, пока глобальное условие не станет истинным, т.о комбинирует поведение действий While и IfElse;

– *запуск_тестов* – составное действие типа **ParallelActivity** позволяет определить набор действий, выполняемых параллельно в ветках типа **SequenceActivity** (*проверка_ПДП, проверка_СЭС, проверка_GPS*), которые включают в себя набор других действий;

– *синхронизация* – элемент типа **SynchronizationScopeActivity**, обеспечивает непрерывность части рабочего процесса, при входе в этот блок будут непрерывно выполнены: команда *cmd2,* выдержана необходимая *пауза* (элемент типа DelayActivity) и *cmd3;*

– *вызов_внешнего_метода* – активность типа **CallExternalMethodActivity** обеспечивает вызов внешних методов или внешних служб;

– *ожидание_ответа* – активность типа **ListenActivity** обеспечивает организацию ожидания одного из набора возможных событий с указанием максимального времени ожидания;

– *отчет_об_исключении* – активность типа **ThrowActivity** используется для генерации исключения, для перехвата исключений, выданных действиями рабочего процесса, можно добавить действие FaultHandler конструкторы рабочего процесса предоставляют контейнер для всех обработчиков ошибок.

Выводы

Технология WWF открывает широкий спектр возможностей, для реализации которых ранее приходилось создавать сложные программные комплексы.

Применение WWF при создании ПО для автоматизации стендовых испытаний малых КА позволило:

предоставить технологу возможность самостоятельного формирования и повторного использования тестовых наборов команд без дополнительного привлечения программиста;

вынести описательную часть хода испытаний во внешний источник информации;

обеспечить модификацию алгоритма на этапе проведения испытаний;

предоставить возможность сохранения состояний рабочих потоков с последующим возобновлением испытаний; повысить наглядность разработанного алгоритма тестирования; улучшить документируемость хода испытаний пут ем отслеживания событий.

При использовании WWF имеется возможность создать словарь действий и шаблонов, ориентированных на конкретную проблему, при этом трудозатраты на составление ПО значительно уменьшаются, повышается качество конечного продукта и не требуется особой квалификации программиста от разработчика, он может быть экспертом той отрасли, для которой создается программное обеспечение.

Литература

1. Михнич Б.Б. Языково-ориентированное проектирование программного обеспечения для автоматизации стендовых испытаний подсистемы данных платформы спутника МС-2-8 / Михнич Б.Б.

Олейник С.В., Соколова Е.В. // Радиоэлектроные и компьютерные системы. – 2008. – №5 (32) – С. 173-176.

2. Солнечные энергосистемы космических аппаратов. Физическое и математическое моделирование / К.В. Безручко, Н.В. Белан, Д.Г. Белов, С.В. Губин, В.И. Драновский, В.С. Кривцов, И.Т. Перекопский, И.Б. Туркин; под ред. акад. НАН Украины С.Н. Конюхова. – Х: ХАИ, 2000. – 516 с.

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

Поступила в редакцию 14.01.2009

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

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ ВИПРОБУВАНЬ СИСТЕМ КОСМІЧНИХ АПАРАТІВ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ WINDOWS WORKFLOW FOUNDATION

Б.Б. Міхнич, І.Б.Туркін, Є.В. Соколова

Розглядаються особливості застосування технології Windows Workflow Foundation при розробці програмного забезпечення для автоматизації стендових випробувань підсистем космічних апаратів таких, як супутники. Показано, що застосування даної технології дозволяє технологу безпосередньо описати хід обчислювального процесу, використовуючи базові блоки або шаблони у вигляді робочих потоків, що підвищує інформативність і спрощує подальші модифікації системи.

Ключові слова: Windows Workflow Foundation, робочі потоки, автоматизація випробувань, космічний апарат.

SOFTWARE DEVELOPMENT FOR SPACECRAFT SYSTEM TEST AUTOMATIZATION USING WINDOWS WORKFLOW FOUNDATION TECHNOLOGY

B.B. Mikhnich, I.B. Turkin, E.V. Sokolova

Features of application of Windows Workflow Foundation technology on software development for test bench automation of such spacecraft subsystem as satellites. There was shown that application of given technology allows the technologist to describe directly computational process flow, using base run-units (executed units) or templates in the form of working streams. This increases clearness and simplifies further system modification.

Keywords: Windows Workflow Foundation, workflow, test bench automation, spacecraft.

Михнич Борис Борисович –аспирант каф. 603, 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. 603, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина.

Соколова Евгения Витальевна –ст. преп. каф. 603, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: evskhai@gmail.com.