САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет информационных технологий и программирования
Кафедра компьютерных технологий

С. Ю. Канжелев

Назначение технологии *Windows Workflow Foundation* – планы и реализация

Обзор

Санкт-Петербург
2009

Введение

Данная работа призвана провести обзор технологии *Windows Workflow Foundation (WWF)*. В работе приведены общие сведения об этой технологии, ее возможностях и преимуществах, а также проведен анализ задач, для реализации которых она применима. Рассматриваются различные источники – документация, книги и статьи, а также записи в блогах ведущих разработчиков *Microsoft*.

Актуальность работы является следствием новизны описываемой технологии и слабой ее освещенностью, особенно в русскоязычной литературе. Однако учитывая возрастающую популярность визуального и декларативного программирования и широкую распространенность технологий *Microsoft*, следует ожидать роста числа переводов англоязычных статей и книг, а также новых русскоязычных публикаций по этой теме.

Технология *WWF*, наряду с *Windows Communication Foundation* и *Windows Presentation Foundation*, является подсистемой *.NET Framework 3.0*. Во многих источниках, например в книге [1], термин *Workflow* переводится как «рабочий поток» или «последовательность выполняемых действий». Поэтому термин *WWF* можно перевести как «фундамент реализации рабочих потоков для Windows».

Цели *WWF*

В 2005 году Дэйв Грин в своем блоге [2] в статье «Что такое рабочий поток и зачем с ним возиться?» (What is Workflow, and why bother?) описал основополагающие принципы, реализация которых являлась целью при создании технологии *WWF*.

Понятие рабочего потока в *WWF*

В начале своей статьи Дэйв Грин пытается объяснить, что такое рабочий поток и зачем необходима его визуализация. Он справедливо отмечает, что слово *workflow* или «рабочий поток» одно из таких определений, которое из-за своей перегруженности зачастую осложняет понимание самой сути. Поэтому поначалу определяет рабочий поток как нечто, реализующее две главные идеи: работа и поток. При этом работа,

представляется рабочему потоку как некоторый черный ящик, составляющий единицу поведения; поток – описывает когда и какая работа будет выполняться.

При таком подходе примером рабочего потока может послужить простой код на языке C#, поскольку он определяет какие функции и в каком порядке выполнять. Однако, многочисленные поставщики реализаций рабочих потоков не удовлетворяются кодом на традиционных языках программирования. Это дает основания Дэйву Грину полагать, что в первичном определении рабочего потока отсутствует ключевая идея.

Такой ключевой идеей, по мнению Дэйва Грина является описание рабочего потока способом простым для анализа, нахождения причин некорректного поведения и внесения изменений. Одним словом, рабочему потоку необходима модель.

Конечно, можно просто определить стандарты программирования для традиционного языка программирования, например, *C#.* Затем, зная эти стандарты, рисовать по ним графы потоков для последующего анализа. Или можно было бы объявить набор атрибутов, которыми программисты могли бы отмечать части кода для описания последовательности выполнения отдельных частей программы. Однако технология *WWF* пошла по другому пути – по пути создания графических моделей. При таком подходе, программисту предлагается изображать логику работы его приложения с помощью набора готовых блоков.

Задачи *WWF*

Возникает вопрос о целесообразности затрат времени и усилий на создание графических моделей. Ответом на него может послужить список преимуществ, приведенный Дэйвом Грином. Перечислим основные преимущества построения графических моделей, предоставляемые *WWF*.

Визуализация полезна как для разработчика в процессе создания и поддержки программного продукта, так и для пользователя рабочего потока, который хочет быть уверен в том, что все происходит в том порядке, в котором и должно происходить. Также визуализация может быть полезна системным администраторам (IT Ops) для поиска причин некорректного поведения или выявления некорректного поведения.

Выразительность. Модель рабочего потока – это язык предметной области, приспособленный под конкретную решаемую задачу. Для примера можно рассмотреть процесс рецензирования, при котором, три положительных голоса из пяти означают, что документ отрецензирован вне зависимости от других двух рецензий, которые в таком случае можно не делать. Описывать такое поведение с помощью классических языков программирования довольно утомительный процесс, в то время как в *WWF* включены конструкции, которые позволяют легко решить такую проблему.

Выполнение. Среда выполнения может исполнять модель непосредственно, без преобразования в промежуточный вид. Кроме того, в технологии *WWF* реализована поддержка таких известных типовых задач для долго выполняющихся рабочих потоков (long running workflow), как поддержка состояний, транзакции. Включение их в модель и работа с ними контролируется простыми и выразительными элементами модели.

Мониторинг. Использование моделей дает возможность без каких-либо дополнительных затрат получать набор событий с понятной семантикой. Этот набор в свою очередь, может быть использован для мониторинга экземпляров рабочих потоков или агрегирования информации о них. *WWF* позволяет также декларативно описывать события, которые интересуют пользователя.

Различные интерпретации. Модели порождают модели. В качестве примера можно привести преобразование модели рабочего потока в модель, построенную по паттерну обмена сообщений, необходимых для общения с ней.

Композиция. Если приложение разработано в виде потока и набора элементарных работ, тогда атомарные элементы работы с понятными интерфейсами могут быть использованы повторно другими рабочими потоками. В свою очередь, рабочие потоки сами по себе подходят под определение работы и могут быть использованы другими рабочими потоками.

Здесь автор статьи отмечает, что в целом гибкость для конкретных задач и преобразование вместе создают систему, в которой определение работы и потока смешиваются. Работа может быть потоком, а поток – работой.

Манипуляция. Часто возникает необходимость создать или модифицировать рабочий поток на лету. Если такое изменение затрагивает код, могут возникнуть проблемы с безопасностью, сложно отследить какой код вставлять можно, а какой – небезопасно Использование моделей позволяет динамически создавать и изменять функциональность. *WWF* позволяет динамически изменять как типы потоков, так и выполняющиеся экземпляры потоков.

Обзор технологии *WWF*

Модель выполнения в рассматриваемой технологии

Книга [3] наверняка покажется интересной тем, кто хочет понять механизм работы среды исполнения рабочих потоков в технологии *WWF*. Первая глава этой книги по построению напоминает детектив, в котором начиная от простейшей программы «Hello, World!» посредством последовательных итераций автор приходит к среде исполнения, похожей на среду используемую в *WWF*.

Основное описание работы среды выполнения технологии *WWF* расположено в третьей и четвертой главах книги.

Авторы считают основой рабочего потока – действия (activity). Реализовано большое число различных действий, действия могут включаться одно в другое. Если построить дерево действий по вложенности, то корнем этого дерева будет сам рабочий поток, который сам по сути, тоже является действием. Поэтому для того, чтобы понять, как работает рабочий поток, достаточно понять, как работает действие.

Среда выполнения рабочих потоков описывает жизненный цикл каждого конкретного действия в виде конечного автомата из пяти состояний. Такое описание позволяет определить в каком состоянии находится каждое конкретное действие и весь рабочий поток в целом.

Перечислим возможные пять состояний, в которых может находиться действие: «Начальное» (Initialized), «Выполнение» (Executing), «Отмена» (Cancelling), «Ошибка» (Faulting), «Закрытое» (Closed). Переходы между состояниями изображены на рис. 1.

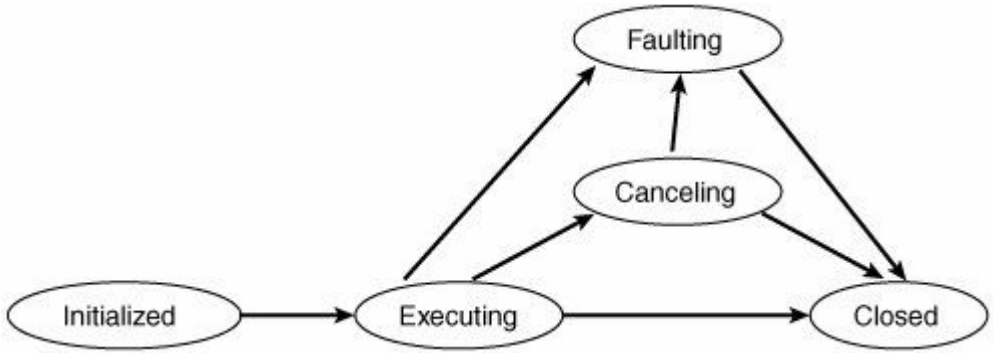

Рис. 1. Конечный автомат, описывающий состояние действия (activity)

Подробное описание вышеперечисленных состояний и возможных переходов приведено в книге [3].

Модель выполнения действий предполагает поочередное быстрое эпизодическое выполнение определенных шагов, прерываемое относительно длинным периодом ожидания внешних событий. Для эффективности, на время ожидания внешних событий, среда выполнения, может не держать в памяти рабочие потоки, а сохранять их во временном хранилище. Это позволяет освободить ресурсы для выполнения в это время других рабочих потоков. Впоследствии, по приходу внешнего воздействия, рабочий поток, его ожидающий, может полностью восстановить свое состояние. Такое сохранение во временном хранилище называется пассивация (passivation).

Следует отметить, что восстановление рабочего потока из временного хранилища и продолжение его работы может произойти как через несколько минут, так через несколько дней. В связи с этим восстановление может происходить в другом процессе и даже на другом компьютере. Поэтому состояние рабочего потока не должно содержать никаких привязок к конкретному программному окружению.

В пятой главе книги [3] описываются дополнительные возможности модели выполнения в технологии *WWF*. Представление рабочего потока в виде набора конечных автоматов позволяет реализовать приостановку, полную остановку и остановку с сохранением текущего состояния в хранилище рабочего потока в любой момент времени. Эти возможности значительно увеличивают сферу применения рабочих потоков.

Рабочие потоки на основе конечных автоматов

Как уже отмечалось, рабочий поток сам является действием. Библиотека *.NET Framework* предоставляет стандартный набор действий, среди которых только два могут представлять рабочий поток – это последовательный рабочий поток и рабочий поток, основанный на конечных автоматах.

В статье [4] приведены рассуждения о том, когда использовать модель, основанную на конечных автоматах, а когда модель последовательных действий. Этот выбор, по словам Аллена Скотта, весьма важен при создании нового рабочего потока. Выбор того или иного типа модели зависит от характера решаемой задачи.

Модель последовательного рабочего потока описывает предсказуемый процесс выполнения работ. Процесс выполнения может иметь разветвления, циклы или ожидания внешних событий. Однако в конечном итоге последовательная модель рабочего процесса использует все эти условия и циклы для того, чтобы двигаться вперед. Одним словом такая модель контролирует процесс и не зависит от внешних воздействий.

Рабочий поток, основанный на конечных автоматах, является рабочим потоком, контролируемым событиями. Конечный автомат полагается на внешние события для того, чтобы изменять свое состояние и в конечном итоге прийти в завершающее состояние. В случае использования такой модели необходимо задать набор состояний и переходов между ними, инициирующимися внешним событием. Предполагается, что рабочий поток в любой момент времени находится в одном из состояний, ожидая внешнего события, для того, чтобы осуществить переход в новое состояние. Можно сказать, что такая модель задает структуру процесса, а контролируют его внешние события.

Таким образом, можно сказать, что используется последовательная модель в случае, если решение, куда переходить принимается самим рабочим потоком, в противном случае, если решение принимается вне модели, следует использовать модель на конечных автоматах.

Следует отметить, что модель на конечных автоматах больше подходит для реализации сложной логики [5] и может применяться для самых разных задач. Также очевидно, что рабочие потоки, основанные на

последовательных действиях, могут быть реализованы с помощью конечных автоматов. Поэтому рассмотрим детальнее реализацию рабочих потоков на основе конечных автоматов.

Технология *WWF* реализует поддержку моделей, основанных на конечных автоматах, с помощью набора классов и интерфейсов. Перечислим основные из них.

Класс `StateActivity` представляет собой состояние в конечном автомате. Рабочий поток задает начальное состояние, с которого начнется выполнение. Также для рабочего потока могут быть заданы заключительные состояния. При переходе в одно из заключительных состояний выполнение потока завершается.

Класс `EventDrivenActivity` представляет собой перехватчик событий в конечном автомате. Экземпляры такого класса – действия, помещаются внутрь состояний, обозначая тем самым список событий, которые может обработать данное состояние. В каждом экземпляре `EventDrivenActivity` может находиться последовательность других действий, которые будут выполнены, когда ожидаемое событие произойдет. Последним действием в этой последовательности обычно является `SetStateActivity`, которое указывает в какое состояние автомат должен перейти.

Использование *WWF*

Характеристики производительности

Использование любой технологии или готовой библиотеки сопряжено с потерями производительности получаемой программы. Величина таких потерь и возможность их минимизации определяют применимость технологии для конкретного программного продукта.

В статье [6] Марк Мецкито приводит основные характеристики производительности программ, построенных с использованием *WWF*. Статья освещает общие вопросы производительности и принципы моделирования, имеющие большую ценность при разработке приложений с использованием *WWF*. Описаны характеристики производительности нескольких показательных сценариев, которые включают многие ключевые особенности рабочих потоков. Также работа содержит рассмотрение вопросов производительности индивидуальных компонент, что может помочь отдать

предпочтение тому или иному из них, модифицируя дизайн приложения или настройки его размещения, в целях улучшения производительности или оптимизации конкретного приложения. Приведенные в статье характеристики производительности не должны рассматриваться как бенчмарки (benchmark), которые могут реализовать все системы. Только эмпирическое тестирование на реальной системе может предоставить корректные значения производительности.

Приведенная статья не описывает как конкретные особенности, компоненты или конфигурация влияют на общую производительность каждого конкретного сценария использования. Статья является только описательным пособием; она не носит предписательный характер и не содержит рекомендаций по оптимизации конкретного сценария использовании *WWF*.

Статья написана в предположении, что читатель хорошо знаком с начальными аспектами *WWF*. Она содержит четыре основных раздела: общие рассуждения о производительности рабочих потоков, сценарии использования рабочих потоков, тематическое исследование производительности *WWF* и покомпонентное описание скорости работы потока.

Раздел, посвященный общим рассуждениям о производительности рабочих потоков, описывает самые важные аспекты для настройки и улучшения производительности приложений, основанных на рабочих потоках. В разделе сценариев использования рабочих потоков представлены три типичных приложения, основанных на рабочих потоках, объясняются характеристики производительности, характерные для этих сценариев и соответствующие настройки, а также приведены результаты тестирования производительности. Раздел тематического исследования включает в себя ключевые характеристики производительности, представленные в виде таблиц и графиков. Наконец, раздел покомпонентного описания производительности системы содержит результаты тестирования производительности нескольких компонент рабочих потоков.

В статье приведены следующие основные факторы, влияющие на производительность:

1. Сохранение состояния рабочего потока (persistence).

2. Размер сохраненного состояния рабочего потока.
3. Действия потока, требующие клонирования контекста выполнения .
4. Действия потока, поддерживающие транзакцию.
5. Действия потока, поддерживающие компенсацию.
6. Запись выполнения действий потока.
7. Сложность действий потока.
8. Время приведения действия в рабочее состояние.
9. Передача данных и событий в потоке.
10. Правила и политика, описанные декларативно.
11. Изменение рабочего потока динамически.
12. Свойства зависимости рабочего потока.
13. Начальное состояние рабочего потока в виде конечного автомата.
14. Время начальной загрузки рабочего потока.

В статье описываются причины влияния указанных факторов на производительность и способы уменьшения этого влияния.

Опыт практического применения *WWF*

Как уже было показано в данной работе, технология *WWF* имеет набор возможностей, а также обладает рядом преимуществ. Однако все дополнительные возможности несут определенную дополнительную нагрузку на скорость выполнения и производительность системы. Стоит задуматься, в каком случае применение описываемой технологии оправдано, а в каком избыточность будет чрезмерна и помешает нормальной работе. Технический руководитель проекта *SharePoint Developer Platform* компании *Microsoft* в своем блоге написал статью «Для каких целей использовать *WWF*?» [7], описывающую области возможного применения технологии *WWF*.

Автор пишет, что эта технология *WWF* не является продуктом, предназначенным для конечного пользователя. Данную технологию должны использовать разработчики программного обеспечения в своих проектах. Статья основывается на опыте автора, который работал со многими разработчиками, использующими рабочие потоки в своих приложениях. Обобщая свой опыт, автор отмечает, что многие, но не все, способы использования предполагались уже при первом представлении технологии.

Создание серверов управления бизнес-процессами

Создается впечатление, что наиболее очевидным использованием рабочих потоков является создание серверов управления бизнес-процессов. Разработчики серверов управления процессами давно знакомы с особенностями разработки процессов и разницей такой разработки с написание классического процедурного кода. Среда исполнения рабочих потоков – общая часть всех серверов управления процессами, которая характеризуется удобством использования этой среды. Поэтому неудивительно, что многие производители серверов управления процессами уже перешли на использование технологии *WWF*. Такой переход позволил этим производителям не заботиться более о поддержке и исправлении ошибок в коде среды исполнения, а сосредоточиться на реализации функциональности более высокого уровня.

Однако среди всех разработчиков, использующих *.NET Framework* число разработчиков серверов управления процессами очень мало. Поэтому действительно интересным вопросом является вопрос использования данной технологии в своих проектах обычными разработчиками *.NET Framework*.

Долго выполняющаяся бизнес-логика

Технология *WWF* хорошо приспособлено для реализации долго выполняющейся бизнес-логики. Некоторые программные продукты не нуждаются ни в какой бизнес-логике, однако большинству промышленных приложений она необходима. Причем часто бизнес-логика занимает большую часть программного продукта и много времени тратится разработчиками в обсуждениях с владельцами бизнеса, что же должен делать программный продукт. Таким образом, работа разработчика промышленных программных продуктов во многом сводится к переводу требований бизнеса в строки процедурного кода. Проблемы начинают возникать, когда необходимо реализовать долго выполняющуюся бизнес-логику. Приходится разделять логически единый бизнес-процесс на атомарные участки кода, каждый из которых выполняет свою короткую часть работы. Также необходимо реализовывать сохранение текущего состояния бизнес-процесса, хранение имеющейся информации о текущем состоянии всех процессов и

возможность возобновления работы любого из бизнес-процессов после получения очередного внешнего события.

При процедурном подходе к программированию бизнес-процессов нет возможности просто приостановить выполнение в ожидании следующего внешнего воздействия. Технология *WWF* решает эту проблему с помощью моделирования процесса на более высоком уровне, чем непосредственно в коде. Код все-таки необходимо писать, однако сохранение состояния, приостановка выполнения до следующего внешнего воздействия и другие проблемы решаются с помощью среды выполнения рабочих потоков.

Регулярно изменяющаяся бизнес-логика и правила

Рабочие потоки также решают еще одну проблему, часто возникающую перед разработчиками. Проблема заключается в том, что программа, написанная однажды, устанавливается у различных заказчиков. Каждый из этих заказчиков имеет свои требования к бизнес-логике, поэтому программа проходит длительный процесс настройки под каждого из них перед тем как конечный пользователь сможет ей воспользоваться. Это увеличивает стоимость и снижает конкурентоспособность поставляемого приложения. Технология *WWF* позволяет решить и эту проблему. Для таких ситуаций возможен сценарий, когда программный продукт содержит в себе набор действий, которые могут выполняться в рабочем потоке. Сами же модели рисуются для каждого заказчика по-своему с использованием этих действий. Таким образом, программирование, как и раньше, происходит с помощью классических языков программирования. Настройка же и создание моделей выполняются с помощью редактора моделей рабочих потоков.

Таким же образом можно легко изменять программы, бизнес-логика и различные правила которых их меняются очень часто. Технологий *WWF* позволяет проводить такое изменение в кратчайшие сроки, при этом изменения могут быть внесены даже в работающие программы.

Необходимость визуализации бизнес-логики или выполнения моделей

Использование модели рабочих потоков для реализации программных продуктов позволяет использовать всю мощь функциональных возможностей

среды выполнения рабочих потоков. Среди основных функциональностей, видимых конечному пользователю, можно выделить возможность автоматического мониторинга и визуального отображения моделей. При этом включение и выключение мониторинга может производиться любым системным администратором стандартным для всех рабочих потоков способом. Визуальное отображение моделей весьма полезно для демонстрации реализованной бизнес-логики людям, мало понимающим в программировании, или для включения ее описания в документацию.

Новая парадигма программирования

Рассмотренная статья [7] вызвала большой интерес и собрала много интересных комментариев. В одном из комментариев, некто Ratham, справедливо отметил, обращаясь ко всем разработчикам, что неправильно было бы рассматривать технологию *WWF* как некий черный ящик, использование которого может помочь в разработке приложений. Правильнее было бы, по мнению автора комментария, видеть в этой технологии новую парадигму, новый подход к программированию. Использование технологии может быть очень широко. Если подумать, мир программирования уже близок к тому, чтобы писать приложения, используя лишь редактор моделей с минимальной необходимостью изменять код. Возможно, скоро любой человек сможет написать простое приложение самостоятельно, не обращаясь за помощью к программисту. В свое время электронные таблицы разрабатывались программистами для бухгалтеров, а теперь сами бухгалтеры способны создать необходимую ему электронную таблицу. По мнению автора, следует думать не о том, что способна дать *WWF*, а о том, что мы можем сделать с помощью этой технологии.

Заключение

Работа содержит обзор довольно новой технологии *WWF*. Приведены не только официальные материалы, но и размышления о целях, преследовавшихся разработчиками технологии и отчеты о практическом опыте людей, использовавших ее. В связи с этим данная работа может быть полезен как специалистам в области программирования с использованием рабочих потоков, или конечных автоматов, так и новичкам.

Важно отметить, что на данный момент описываемая технология только набирает популярность. Поэтому материалов по данной теме недостаточно, а те, что есть, часто освещают лишь некоторые аспекты ее применения. Поэтому подобная работа является важной.

Литература

1. **Уотсон К., Скиннер М., Ивьен Б. и др.** *C# 2005 и платформа .NET 3.0 для профессионалов.* М. : Вильямс, 2008.

2. **Green D.** What is Workflow, and why bother? *Блоги на Microsoft Developers Nework.* [В Интернете] 17.09.2005. http://blogs.msdn.com/davegreen/archive/2005/09/17/470704.aspx

3. **Shukla D., Schmidt B.** *Essential Windows Workflow Foundation.* Upper Saddle River, NJ : Addison-Wesley, 2007.

4. **Scott A.** State Machines In Windows Workflow. [В Интернете] 24. 09. 2006. http://www.odetocode.com/Articles/460.aspx

5. **Shalyto A.** Technology of Automata-Based Programming. [В Интернете] 18. 08. 2004. http://www.codeproject.com/KB/architecture/abp.aspx?print=true

6. **Mezquita M.** Performance Characteristics of Windows Workflow Foundation. *Статьи на Microsoft Developer Network.* [В Интернете] 11. 2006. http://msdn2.microsoft.com/en-us/library/aa973808.aspx

7. **Andrew P.** What to use Windows Workflow Foundation for? *Блоги на Microsoft Developers Network.* [В Интернете] 01.02.2007. http://blogs.msdn.com/pandrew/archive/2007/02/01/what-to-use-windows-workflow-foundation-for.aspx