УДК 004.05:004.4'422

В.М. ГОДУНКО[1], В.О. МИЩЕНКО[2], М.М. РЕЗНИК[3], Д.В. ШТЕФАН[2]

[1]*ИП Годунко В.М., Ростов-на-Дону, Россия*
[2]*Харьковский национальный университет имени В.Н. Каразина, Украина*
[3]*ActForex Inc., Запорожье, Украина*

КАЧЕСТВО ТРАНСЛЯТОРА ШАБЛОНОВ ДИНАМИЧЕСКИХ HTML СТРАНИЦ ДЛЯ ADA WEB СЕРВЕРОВ

Успех известных технологий Java Server Pages и Active Server Pages подсказал разработать инструмент для аналогичной Ада технологии – транслятор шаблонов динамических HTML страниц в генераторы таких страниц для Web серверов, используемых приложениями на языке Ада. Созданный язык шаблонов и разработанный на языке Ада транслятор обеспечивают платформенную независимость (за счёт высокой портабельности Ада программ) и установку на любой из указанных серверов (благодаря полноте соответствующих средств использованной системы библиотек Matreshka). Статья посвящена обоснованию прогноза достаточности качества данного продукта и методу обоснования. Для этого принята определённая модель качества и использованы метрики энергетического анализа кода, который позволяет в современных условиях решать задачи, близкие к тем, для решения которых некогда создавались научные метрики М. Холстеда.

Ключевые слова: *язык Ада, трансляция, HTML, модель качества, метрики, энергетический анализ.*

Введение

Технология Java Server Pages (JSP) [1] предоставляет удобное и надёжное средсво для создания динамических Web-страниц на стороне сервера. Её обновления выпускаются с 1999 г. систематически. Параллельно с JSP развивалась аналогичная технология Active Server Pages (ASP) [2]. Обе технологии хорошо представлены в учебной и профессиональной литературе (см., напр., [3 – 5]). Данный подход оказался удачен, несмотря на свою простоту. Он согласуется с общей современной концепцией получения нужных артефактов путём последовательного преобразования моделей [6].

В настоящее время развёрнуты средства разработки Web приложений, на языке Ада, например Ada Web Server (AWS) [7] или Matreshka [8, 9]. Естественно возник вопрос о создании и испытании Ада технологии, аналогичной JSP. Основным её инструментом должен быть транслятор шаблонов динамических HTML страниц в Ада код, который бы эти страницы генерировал. Этот код предназначается для AWS или для Ада библиотек, ориентированных на протокол CGI или FastCGI.

Поскольку, в отличие от JSP, ASP, новая Ада технология находится на этапе первых испытаний, то её преимущества, в том числе связанные с транслятором, нельзя пока вывести из опыта применений. Однако можно анализировать код в соответствии с выбранной моделью качества программ. Подходящую модель, основанную на характеристиках исходного кода, предложила в своё время наука о программах М.Холстеда [10, 11, 14]. Для сложных многомодульных программ её заменяет энергетический анализ программных систем [12].

1. Постановка задачи

Обычная техника создания динамических HTML страниц состоит в непосредственном написании генерирующей программы на процедурном языке (в нашем случае – на языке Ада), который обеспечивает формирование страниц из заготовленных строк и динамически вычисляемых элементов.

Для замены этой техники была поставлена задача разработки формата шаблона и транслятора так, чтобы получать формирующий HTML страницы Ада код трансляцией.

При этом требуется, чтобы составлять шаблон было проще, чем непосредственно писать программу создания динамических страниц.

Другое важное условие учитывает то, что в системе Ада библиотек Matreshka имеется парсер XML документов (библиотека с корневым модулем XML.SAX).

Поэтому было выставлено требование, чтобы формат шаблонов основывался на XML.

Цель настоящей работы – выработка метода, позволяющего контролировать разработку таких приложений, как транслятор шаблонов динамических HTML страниц в Ада генераторы страниц, на основе простой, но согласованной со стандартами модели.

2. Реализация транслятора

Решение создавать шаблоны в форме XML документов повлекло разработку системы тегов:

<asp:root> – корневой элемент, являющийся контейнером для всех остальных тегов.

<asp:with> – элемент, который содержит обязательный атрибут package, с помощью которого можно подключать необходимые для работы пакеты.

<asp:declare> – элемент, определяющий начало пакета подпрограммы-генератора HTML страницы.

<asp:body> – элемент, свидетельствующий о начале тела подпрограммы-генератора HTML страницы.

<asp:expression> – элемент, внутри которого текст воспринимается как код программы, а не разметка.

Эти базовые теги (которые принадлежат отдельному пространству имен) позволяют выделять в тексте шаблона прообразы определённых частей программы генерации страниц, тогда как все другие воспринимаются как часть создаваемого HTML документа.

Когда XML парсер встречает открывающийся (закрывающийся) программный тег – он заменяет его на соответствующий Ада-код. Результат работы DHTML транслятора это – модуль сгенерированного кода, готовый для включения в проект Web приложения на языке Ада (рис. 1).

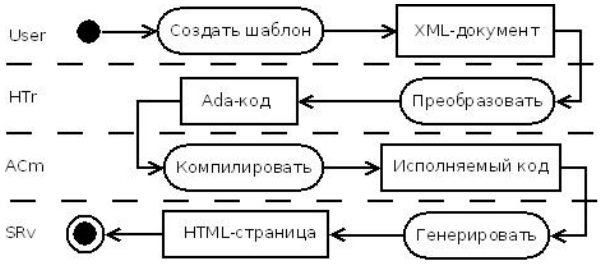

Рис. 1. Схема динамической генерации HTML страниц: User - Web программист, HTr – HTML транслятор, ACm – Ada компилятор, Srv – сервер

DHTML транслятор (см. рис. 2) состоит из Ада пакета Events_Printers, который содержит операции для работы с входным xml-файлом, и главной процедуры, которая анализирует параметры вызова и формирует выходной Ада код. Подробности, полезные для использования и модификации даны в [13].

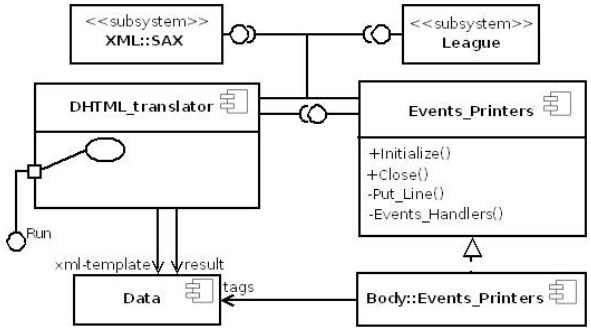

Рис. 2. Архитектура DHTML транслятора

3. Особенности метода

При метрическом исследование исходных кодов разработанной программной системы мы используем такие «научные метрики» [10,11]:

V – объём программного модуля – произведение длины исходного текста на количество информации в словаре, который использует этот текст (размерность sym·bit);

D^ – безразмерная оценка трудности исходного текста - прямо пропорциональна и главным образом определяется числом разных символов-операндов в тексте; в этой роли обычно (но не всегда) выступают идентификаторы и литералы;

A^=D^·V – оценка работы по кодированию модуля (размерность sym·bit).

Эти метрики применялись и применяются при анализе одномодульных программ и модулей, чьей зависимостью от других модулей в контексте исследования можно пренебречь. Иначе мы применяем энергетические метрики [12, 14]:

W – объём разработки модуля ≥V и равен ему, если модуль не зависит от других модулей (чем больше признаков зависимости от других модулей, тем объём разработки больше, он также зависит от порядка разработки модулей);

A = – работа программирования (sym·bit), которая для модуля равна W2/V*, где V* – потенциальный объём модуля, определяемый в [12] правилами, которые формализуют идеи [10];

E – спецификционная энергия (sym·bit), которая для модуля определяется с учётом его структуры в терминах потенциального объёма и уровня языка программирования.

Работа A и энергия E программной системы корректно определяются суммированием [12].

Мы просмотрим стандартное дерево характеристик и подхарактеристик (оно одинаково для внешнего и внутреннего качества программных систем, см., напр., [15]). Для тех, для которых это возможно, обоснуем их наличие.

Для гарантии достаточной ***функциональности*** разработанного DHTML транслятора решающее значение имеет простота спецификации функциональных требований к нему, а также возможность использовать отлаженные библиотеки системы Matreshka. Это обусловило простоту и успех тестовой проверки *пригодности, точности, операционного взаимодействия* (в смысле внешнего качества). *Согласованность* функциональности продукта в смысле соответствия xml технологии и контекстам Matreshka проверена непосредственно, а требований по *безопастности* не было. Остаётся, грубо говоря, ещё в 5 раз больше черт качества для проверки, но к функциональности они не относятся. Обычно оценивание этих черт требует более творческого подхода и больших затрат времени.

4. Систематический анализ качества

Надёжность программных систем прежде всего заключается в *зрелости* разработки, что можно интерпретировать как вероятнось того, что в исходных текстах найдены и исправлены все ошибки, угрожающие операционной способности продукта. Много ли ошибок могло попасть в исходные тексты? Примем гипотезу о том что эта (разумеется, нечёткая) величина пропорциональна объёму разработки [12]. Это позволяет сравнить оценку числа фактически выявленных ошибок – 15 с априорной оценкой:

$$B=(W_{events\_printers}+W_{body}+W_{HTML\_transl})/3000= 7,4 \quad (1)$$

Использование библиотек (Matreshka) наряду с сокращением объёма исходных кодов, которое снижат риск ошибок, отчасти их провоцирует, поскольку требует правильного использования модулей этих библиотек. Отметим, что этот риск, в нашем случае пренебрежимый, можно оценить как

$$(W_{all}-V_{all})/V_{all} = 0,015. \quad (2)$$

Ещё один критерий зрелости – соответствие работы программирования спецификационной энергии системы. Принимая довольно приближенно уровень языка Ада $\lambda = 1,5$, получим, что эти величины вполне согласованы друг с другом [12]:

$$E_{all} = 3,22\cdot10^6,\ A_{all} = 4,83\cdot10^6 \ \text{(bit·sym)} \quad (3)$$

Аргумент за наличие у данной разработки *устойчивости* к ошибкам и *согласованности* по надёжности (с точки зрения внутреннего качества) состоит в следующем. При ошибках пользователя в составлении шаблона транслятор, как положено такому средству, сообщает их тип, место по тексту и дополнительные данные, полезные для исправления. К *восстановимости* при сбоях требований нет

О ***полезности*** DHTML транслятора говорит *понятность* его назначения (конкретность задачи и среды использования) и возможность *изучения* по руководству (9 стр.) и по статье в Интернет [13]. Мы прогнозируем *привлекательность* этого приложения для Ада программистов в силу популярности аналогичных средств, ранее созданных для других сред [1-5]. Об удобстве *применения* не говорим ввиду его простоты, а наличие *согласованности* – это соблюдение стандартов Ада программирования.

Эффективность транслятора с точки зрения поведения **системы** трансляции во *времени* нужно понимать как отношение затрат времени на ручное создание генератора HTML страниц, который произвёл транслятор, ко времени создания соответствующего шаблона. Принимая гипотезу пропорциональности времени кодирования малых программ оценке работы A^ [10,11], мы на примерах (см. табл. 1) получили, что шаблоны экономят работу программирования в среднем в 85 раз и не меньше 27 в каждом случае.

Таблица 1

Оценки научных метрик для шаблонов и генераторов динамических HTML страниц (1 Hd = 1000 bit·sym, 1 Khd = 1000 Hd).

Имя страницы		request_auth	request_logout	request_post	request_upload
Шаблон	V, Hd	0,84	0,90	3,74	1,56
	D^	38,0	46,1	96,0	37,6
	A^, Hd	31,9	41,6	363,6	58,5
Генератор	V, Hd	5,3	5,18	50,07	6,18
	D^	225,5	221,7	1787	225,8
	A^, KHd	1,19	1,15	90,5	1,58

Экономия *аппаратных* ресурсов для нас не актуальна, требований по *согласованности* нет.

Сопровождаемость, первое, требует *анализируемости* продукта. Мы оценим её оценкой трудности D^, которая считается индикатором понятности исходного текста модуля [11]:

D^(events_printers.ads) = 128;

D^(events_printers.adb) = 545;

D^(dhtml_translator.adb) = 42.

Из исследований программ на разных языках известно ([11]) о наличии важных рубежей D_1, D_2. При $D<D_1$ код воспринимается человеком легко, а при D>D2 он является громоздким. Эти рубежи хорошо исследованы для языка PL/1 [11]: $D_1 = 119$, $D_2 = 160$. Предположим, они приблизительно такие же для языка Ада. Следовательно, анализируемость третьей компоненты транслятора хорошая, первой удовлетворительная, а второй – плохая. Очевидным недостатком модуля events_printers является его длина при том, что состоит он из тел полутородесятка разных подпрограмм.

Если 12 тел вынести в субмодули, то для обновлённого проекта из 15 модулей получим:

D^(events_printers.adb) = 159;

D^(events_printers-initialize.adb) = 160,

а для остальных субмодулей $D < D_1/1,5$, и теперь анализируемость транслятора удовлетворительна. Это преобразование проекта не меняет его спецификационной энергии, но показывает меньшую работу: $A_{all} = 1,6\cdot E_{all}$ (согласованность сохраняется).

Модифицируемость также обеспечена. Для данного приложения её смысл наверняка будет за-

ключаться в расширении словаря шаблонов. Для добавления возможности обрабатывать новые теги или атрибуты предусмотрено расширение списков допустимых тегов, атрибутов со вписыванием Ада кода реакции на них в качестве новой альтернативы соответствующего оператора case. Например, можно добавить тег <KhNU/>, который при трансляции должен быть заменён строкой «ХНУ имени В.Н. Каразина». Для этого в файле Events_Printers.adb в 61-й строке в перечень типа ASP_Tags необходимо добавить имя нового тега:

```
type ASP_Tags is (Root_Tag, With_Tag, … Declare_Tag, Khnu_Tag);
```

а в теле процедуры Start_Element пополнить состав альтернатив оператора case реакцией на новый тег:

```
case Tag is
  when Khnu_Tag => Put_Line (+" Writer.Characters
    (+"" ХНУ имени В.Н. Каразина "");");
```

Детерминированность и локальность таких действий означает *стабильность* модификации (с точки зрения внутреннего качества). *Тестируемость.* Трансляторы не требуют оперативной проверки при эксплуатации. Если же добавлен новый тег, то тест, проверяющий правильность трансляции, составить легко. Требований к *согласованности* по сути нет.

Проверяем ***мобильность***. Основой высокой *адаптируемости* является высокая стандартизация и ориентация на полиплатформенность языка разработки – Ада вместе с доступностью исходных кодов транслятора и требуемых библиотек. Специфика проявляется при переносе транслятора в среду разработки Web приложений, которая ориентирована на сервер (или протокол) своего типа. Сейчас трансляция ориентирована на AWS или варианты реализации протоколов CGI и FastCGI. Это регулируется специальной опцией. Для иного варианта нужно, во первых, вписать в инструкцию пользователя, какая используется версия xml и кодировки (он это должен указывать в заголовках шаблонов) и какой путь ведёт к файлам базовых тегов (это является опцией при запуске транслятора). Второе, необходимо создать файлы базовых тегов на основе конкретной информации об используемом сервере, но по образцам, которыми служат такие файлы для реализации протоколов CGI и FastCGI.

Устанавливаемость транслятора равносильна устанавливаемости Matreshka [13]. *Заменопригодность* пока не актуальна. *Сосуществование* обеспечивается автономностью работы и скромными запросами памяти. Требований к *согласованности* нет.

5. Первый опыт применения

Транслятор был использован в состав системы Matreshka при экспериментальном обновлении страниц сайта www.ada-ru.org. В табл. 1. приводятся факторы риска и работы программирования, которая затрачена на разработку шаблонов и на создание генераторов соответствующих страниц, как если бы такой генератор создавался вручную.

Объёмы V для шаблонов в среднем в 7 раз меньше, чем для Ада процедур, генерирующих страницы. Это характеризует снижение риска ошибок кодирования. Трудность D, которая характеризует другой риск - недостаточной понятности кода, для шаблонов меньше, чем для генераторов в среднем 8,6 раза. Тут, однако, важнее другое. Далее, для страховки, примем более жесткие рубежи трудности, скажем, $D_1 = 100$, $D_2 = 200$. При $D < D_1$ код воспринимается человеком легко, и можно думать, что ошибки были сразу замечены и удалены. При D > D2, напротив, код громоздок, и риск остаточных ошибок велик. Для шаблонов согласно табл. 1 риска обсуждаемого типа не было, а для генераторов в данных примерах он был всегда в рассмотренных примерах). Это лучше, чем результаты трансляции тестовых примеров: там экономия объёма кода была в 3,5 раза, работы в 8, трудность уменьшалась в 2 раза, но и для генераторов страниц она оказывалась меньше, чем 80.

Выводы

Показан и испытан на примере актуальной задачи метод системного оценивания признаков качества малых инструментальных приложений для поддержки Web программирования. Он предполагает применение стандартных, но редуцированных моделей качества, научных и энергетических метрик [10, 12, 15]. Прогноз достаточности (для целей практического использования) качества созданного DHTML транслятора является важным этапом в процессе его включения в систему Matreshka, поскольку она предназначена для разработки комплексных приложений на языке Ада (языке критических приложений и больших проектов). В дальнейшем можно исследовать качество в использовании.

Литература

1. Java Server Pages Specification Version 2.1 [Электронный ресурс]. – Режим доступа: download.oracle.com/otn-pub/jcp/jsp-2.1-fr-eval-spec-oth-JSpec/jsp-2_1-fr-spec.pdf. – 3.02.2012 г.

2. Active Server Pages [Электронный ресурс]. – Режим доступа: msdn.microsoft.com/en-us/library /ie/aa286483.aspx. – 3.02.2012 г.

3. Murach, J. Murach's Java Servlets and JSP 2nd Edition [Text] / J. Murach, A. Steelman. – Mike Murach & Assotiates, Inc. - 2008. – 744 p.

4. Гери, Д.М. Java Server Pages. Библиотека профессионала [Ткст] / Д.М. Гери. - М.: Издательский дом «Вильямс». – 2002. – 448 с.

5. Mitchell, S. Designing Active Server Pages [Text] / S. Mitchell. - O`Reilly Media, 2000. – 362 p.

6. MOFM2T 1.0 [Электронный ресурс]. – Режим доступа: http://www.omg.org/spec/MOFM2T/1.0. – 3.02.2012 г.

7. Ada Web Server User's Guide [Электронный ресурс]. – Режим доступа: http://www.adacore.com/wp-content/files/auto_update/aws-docs. – 3.02.2012 г.

8. Матрёшка: использование FastCGI в Web приложениях [Электронный ресурс]. – Режим доступа: http://ada-ru.qtada.com. – 3.02.2012 г.

9. Matreshka [Электронный ресурс]. – Режим доступа: forge.ada-ru.org/matreshka. – 3.02.2012 г.

10. Холстед, М.Х. Начала науки о программах [Текст] / М.Х. Холстед. - М.: Финансы и статистика. – 1981. – 128 с.

11. IEEE Std 982.2-1988. IEEE Guide for the Use of Standard Dictionary of Measures to Produce Reliable Software [Электронный ресурс]. – Режим доступа: http://members.aol.com/geshome/IEEE982/IEEE9822.pdf. – 3.02.2012 г.

12. Мищенко, В.О. Энергетический анализ программного обеспечения с примерами реализации для Ада-программ [Тест] / В.О. Мищенко. – Х.: ХНУ имени В.Н. Каразина, 2007. – 119 с.

13. Матрёшка: DHTML_translator [Электронный ресурс]. – Режим доступа: http://ada-ru.qtada.com/archives/286. – 3.02.2012 г.

14. Мищенко, В.О. Согласование требований при формализации одного метода энергетического анализа программ [Текст] / В.О. Мищенко // Радиоэлектронные и компьютерные системы. – 2008. – № 5 (32). – С. 177 – 182.

15. ISO/IEC TR 9126 2:2003 [Электронный ресурс]. – Режим доступа: http://webstore.iec.ch/preview/info_isoiec9126-2%7Bed1.0%7Den.pdf. – 3.02.2012 г.

Поступила в редакцию 3.02.2012

Рецензент: канд. техн. наук, доцент А.В. Волковой, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина

ЯКІСТЬ ТРАНСЛЯТОРА ШАБЛОНІВ ДИНАМІЧНИХ HTML СТОРІНОК ДЛЯ ADA WEB СЕРВЕРІВ

В.М. Годунко, В.О. Міщенко, М.М. Резнік, Д.В. Штефан

Успіх відомих технологій Java Server Pages та Active Server Pages підказав розробити інструмент для аналогічної Ада технології - транслятор шаблонів динамічних HTML сторінок в генератори таких сторінок для Web серверів, використовуваних додатками на мові Ада. Створена мова шаблонів і розроблений на мові Ада транслятор забезпечують платформену незалежність (за рахунок високої портабельності Ада програм) і установку на будь-який з вказаних серверів (дякуючи повноті відповідних засобів використаної системи бібліотек Matreshka). Стаття присвячена обгрунтуванню переваг даного продукту та процесу його розробки. Для цього використані метричні методи та засоби в рамках підходу, який отримав назву енергетичного аналізу коду, який покликаний замінити в сучасних умовах підхід до оцінки процесу і результатів кодування, відомий як наука про програми М. Холстеда.

Ключові слова: мова Ада, трансляція, HTML, енергетичний аналіз, надійність, супровід

QUALITY OF THE DYNAMIC HTML PAGES TRANSLATOR FOR ADA WEB SERVERS

V.M. Godunko, V.O. Mishchenko, M.M. Reznik, D.V. Shtefan

The success of the well-known technology Java Server Pages and Active Server Pages prompted to develop a similar tool to Ada technology - the translator of dynamic HTML pages templates in the source code of generators of these pages for the Web servers used by Ada applications. Designed and developed a pattern language and DHTML translator provides platform independence (due to the high portability Ada programs) and install on any of the servers (owing to the fullness of the tools used by the system libraries Matreshka). Article are devoted to the justification of the benefits of the product and the process of its development. For this metric used methods and tools in an approach known as the energy analysis of code that is intended to replace the current approach to assessing the conditions of the process and results of coding, known as the science of programs M. Halstead.

Key words: Ada language, DHTML translator, translation, HTML, energy analysis, reliability, maintenance

Годунко Вадим Максимович – директор ИП Годунко В.М., Ростов-на-Дону, Россия.

Мищенко Виктор Олегович – канд. физ.-мат. наук, доцент кафедры моделирования систем и технологий Харьковского национального университета имени В.Н. Каразина, Харьков, Украина.

Резник Максим Михайлович – инженер-программист компании ActForex Inc., Запорожье, Украина.

Штефан Дмитрий Викторович – студент факультета компьютерных наук Харьковского национального университета имени В.Н. Каразина, Харьков, Украина.