А 05213-504
М-601

ԵՐԵՎԱՆԻ ՄԱԹԵՄԱՏԻԿԱԿԱՆ ՄԵՔԵՆԱՆԵՐԻ
ԳԻՏԱՀԵՏԱԶՈՏԱԿԱՆ ԻՆՍՏԻՏՈՒՏ

Միլեթբաշյան Արտակ Սեդրակի

ԱՎԵԼՑՈՒԿԱՅԻՆ ԻՆՖՈՐՄԱՑԻԱՅԻ ՖԻԼՏՐԵՐԻ ՍՏԵՂԾՄԱՆ ԾՐԱԳՐԱՅԻՆ ՄԻՋՈՑՆԵՐ՝ WEB ՀԱՄԱԿԱՐԳԵՐԻ ՕԲՅԵԿՏԱ-ԿՈՂՄՆՈՐՈՇՎԱԾ ՆԱԽԱԳԾՄԱՆ ՀԱՄԱՐ

Ե 13.04-«Հաշվողական մեքենաների, համալիրների, համակարգերի և ցանցերի մաթեմատիկական և ծրագրային ապահովում» մասնագիտությամբ

տեխնիկական գիտությունների թեկնածուի գիտական աստիճանի հայցման ատենախոսության

ՍԵՂՄԱԳԻՐ

ԵՐԵՎԱՆ 2005

ЕРЕВАНСКИЙ НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ
ИНСТИТУТ МАТЕМАТИЧЕСКИХ МАШИН

Милетбашян Артак Седракович

ПРОГРАММНЫЕ СРЕДСТВА ГЕНЕРАЦИИ ФИЛЬТРОВ ИЗБЫТОЧНОЙ ИНФОРМАЦИИ ПРИ ОБЪЕКТНО-ОРИЕНТИРОВАННОМ ПРОЕКТИРОВАНИИ WEB-СИСТЕМ

АВТОРЕФЕРАТ

диссертации на соискание ученой степени кандидата технических наук

по специальности: 05.13.04 – «математическое и программное обеспечение вычислительных машин, комплексов, систем и сетей»

ЕРЕВАН-2005

© National Library of Armenia OCR by Portmind

Ատենախոսության թեման հաստատվել է Հայաստանի պետական ճարտարագիտական համալսարանում:

Գիտական ղեկավար՝ ֆ.-մ.գ.դ. Ս. Կ. Շուքուրյան

Պաշտոնական ընդդիմախոսներ՝ տ.գ.դ. Ա. Ք. Պալյան

տ.գ.թ. Մ. Վ. Մարկոսյան

Առաջատար կազմակերպություն՝ ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ

Պաշտպանությունը տեղի կունենա 2005թ. նոյեմբերի 30-ին ժամը 15-ին, 043 մասնագիտական խորհրդի նիստում, Երևանի մաթեմատիկական մեքենաների գիտահետազոտական ինստիտուտում (հասցեն՝ 375033, ք. Երևան Հ.Հակոբյանի 3):

Ատենախոսությանը կարելի է ծանոթանալ ինստիտուտի գրադարանում:

Սեղմագիրն առաքված է 2005թ. հոկտեմբերի 29-ին

Մասնագիտական խորհրդի
գիտական քարտուղար, տ.գ.դ. Վ.Վ.Մարտինով

Тема диссертации утверждена в Государственном инженерном университете Армении

Научный руководитель: д.ф.-м.н. С. К. Шукурян

Официальные оппоненты: д.т.н. А. Х. Палян

к.т.н. М. В. Маркосян

Ведущая организация: Институт проблем информатики и автоматизации НАН РА

Защита состоится 30 ноября 2005г. в 15 часов на заседании специализированного совета 043 Ереванского научно-исследовательского института математических машин по адресу: 375033, Ереван, ул. А.Акопяна, 3.

С диссертацией можно ознакомиться в библиотеке института.

Автореферат разослан 29 октября 2005г.

Ученый секретарь
специализированного совета д.т.н. В.В.Мартынов

3972-2005

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность. Развитие Интернет приложений привело к появлению определенных негативных побочных явлений. Одно из них – это получение избыточной незапрашиваемой информации, которую часто называют *спамом (spam)*. Сегодня доля спама составляет 30-80% трафика электронной почты*) и имеет тенденцию к увеличению разновидностей и интенсивности. Учитывая сложившуюся ситуацию, актуальность защиты от незапрошенных электронных рассылок приобретает особенное значение. Спам постоянно меняет свои формы, отличается высокой изменчивостью внешних признаков и широким арсеналом лингвистических и технических уловок для обхода спам-фильтров. Перед разработчиками антиспам систем встала необходимость непрерывной работы по созданию защиты от все новых и новых образцов спама. В то же время, антиспам должен обеспечить пользователей эффективной системой противостояния всем уже существующим видам спама. Таким образом, необходимо постоянное обновление технологий распознавания и фильтрации спама в системах антиспама, при сохранении совместимости с антиспам системами предыдущих версий (backward compatibility). Высокая эффективность некоторых систем антиспама достигается благодаря регулярному пополнению базы контекстной фильтрации. Пополнение базы новыми компонентами для эвристического анализа может осуществляться как автоматически через Интернет, так и "вручную"**). Обновление вручную позволяет администратору сети добавлять в базу свои собственные образцы нежелательной информации. Другой возможностью защиты от спама является встроенный механизм анализа сообщений, основанный на нейронных сетях, способных обучаться и отличать обычные письма от спама.

Ограничения на время разработки фильтров часто приводят к необходимости использования одинаковых или подобных фильтров при решении различных задач, декомпозиции разработки сложного фильтра на разработку ряда более простых фильтров с возможностью одновременного проектирования составляющих фильтров, а также использования уже разработанных фильтров или их модификаций в процессе разработки. Определенные типы модификаций фильтров

---

*) ISIPP's Second Annual "SPAM AND THE LAW" CONFERENCE January 28, 2005, San Francisco Westin Hotel at SFO

**) Brian E. Burke, Mark Levitt, "Worldwide Antispam Solutions 2004-2008 Forecast and 2003 Vendor Shares: What Is Being Done About Spam", IDC Research 2004

© National Library of Armenia

описываются следующим образом. Фильтр представляется в виде последовательности фрагментов, которые можно разделить на два подмножества. Фрагменты первого подмножества используются без изменений во всех подобных фильтрах. Фрагменты второго подмножества зависят от определенных условий и в зависимости от их значения содержимое фрагментов различно. Если вместо Фрагментов второго подмножества в описании фильтра использовать некоторые переменные, значения которых отражает значения отмеченных выше условий, то получается некоторый шаблон. По нему, в зависимости от значения переменных, можно сгенерировать тот или иной конкретный фильтр. Каждый такой шаблон можно рассматривать как базовый элемент – примитив фильтрации, из которых можно далее строить более сложные шаблоны фильтрации.

Однако, в доступной нам литературе нет ссылок на возможность параметризации фильтров. Отсутствует также возможность задания последовательности фильтров и, тем более, организации условных переходов внутри последовательности к составляющим фильтрам. Между тем наличие таких средств позволило бы сделать легко наращиваемой и значительно более гибкой систему фильтрации избыточной информации.

Поэтому разработка подходов к проектированию антиспам систем, гибко и быстро настраиваемых на новые формы спама, представляется важной и актуальной. Одним из возможных способов решения указанной проблемы является построение расширяемой инфраструктуры фильтрации, основанной на специальных примитивах проектирования фильтров и обеспечивающей настройку на новые формы спама путем наращивания функциональных возможностей инфраструктуры с помощью комбинаций примитивов.

Цель диссертационной работы – обоснование и разработка инструментальных программных средств фильтрации избыточной информации в условиях динамического изменения ее типов. Эти средства должны гибко настраиваться на новые формы избыточной информации, обеспечивать автоматизацию проектирования и верификации примитивов проектирования и удовлетворять стандартам проектирования объектно-ориентированных WEB систем.

Объекты исследования. Объектами исследования являются примитивы формального представления среды фильтрации: формы представления избыточной информации, система обработки примитивов проектирования фильтров, далее называемых шаблонами, базовые алгоритмы синтеза и верификации шаблонов.

Методы исследования. В диссертационной работе использованы методы объектно-ориентированного проектирования, теории графов, теории автоматов и автоматического синтеза программ.

Научная новизна.

- Модель иерархической среды фильтрации избыточной информации и ее реализация;
- Язык описания шаблонов фильтрации избыточной информации;
- Алгоритм генерации полной системы примеров для верификации шаблонов;
- Алгоритм синтеза корректных фильтров для определенных классов избыточной информации.

Практическая ценность и реализация. Предложенные инструментальные средства разработаны с ориентацией на современные объектно-ориентированные среды проектирования, на базе единой объектной модели, что обеспечивает возможность их использования в виде стандартных настраиваемых блоков, построения новых блоков на их основе, гибкого изменения и расширения полученных программных систем.

Практическое использование результатов диссертационной работы и их значимость отражены в соответствующем акте внедрения в компании Lycos CJSC. Применение инструментальных программных средств фильтрации избыточной информации для конкретной WEB системы – системы переговоров Chat, позволило обеспечить фильтрацию спама на 12-31% и тем самым повысить эффективность функционирования системы Chat.

Возможность эффективного использования предложенной инфраструктуры фильтрации и для других WEB систем зависит от того, насколько можно отразить их специфику в шаблонах фильтрации.

Публикации и апробация. Основные результаты и положения диссертационной работы обсуждались и докладывались на семинарах кафедры программного обеспечения вычислительной техники и автоматизированных систем Государственного инженерного университета Армении (2004-2005 гг.), конференции молодых ученых и аспирантов Государственного инженерного университета Армении (2005 г.).

© National Library of Armenia

Основные результаты работы отражены в 4 публикациях, список которых приведен в конце автореферата.

Структура и объем работы. Диссертационная работа состоит из введения, трех глав, списка использованной литературы и 1 приложения. Объем работы 112 стр., включая 22 рисунков, 9 стр. цитируемой литературы, насчитывающей 114 наименований, и 2 стр. приложений.

СОДЕРЖАНИЕ РАБОТЫ

Во введении обосновывается актуальность, формулируется цель диссертационной работы и приводится краткое описание содержания работы по главам.

В первой главе рассмотрены проблемы защиты от избыточной информации при проектировании WEB-систем. Выделены основные требования к проектированию средств фильтрации избыточной информации. Определены проблемы и обоснованы преимущества использования объектно-ориентированных технологий проектирования для решения поставленных задач.

В §1.1 рассмотрены процессы возникновения и эволюции спама.

Проведен сравнительный обзор антиспам программных систем и динамика изменения типов спама, включая разновидности несанкционированных рассылок.

Проанализированы принципы работы антиспам программ и отдельные программы фильтрации.

В §1.2 проведен анализ архитектур распределенных информационных WEB-систем, организации обработки информации по модели "клиент-сервер" и разработки приложений на базе этой архитектуры.

Рассмотрены принципы интеграции информационных ресурсов для эффективного управления информационной системой на примере гетерогенных прикладных систем. Проведен анализ методов реконфигурации и трансформации информационной системы в распределенную систему.

Рассмотрены архитектура и технологии разработки WEB-систем, особенности их реализации для различных платформ. Приведены особенности объектно-ориентированных технологий проектирования WEB-систем преимущества их использования и, в частности, использования XML-технологий. Выделены стандарты проектирования, которым должна удовлетворять разрабатываемая программная система для интеграции с объектно-ориентированными WEB-системами.

В §1.3 проведена постановка задачи: разработка программных средств для построения расширяемой инфраструктуры фильтрации избыточной информации, основанной на специальных шаблонах генерации фильтров и обеспечивающей настройку на новые формы спама путем добавления новых шаблонов или использования новых комбинаций шаблонов. Эти средства должны удовлетворять рассмотренным стандартам интеграции с объектно-ориентированными WEB-системами.

Основой разрабатываемых средств является программная оболочка фильтрации в виде настраиваемого с помощью управляющих скриптов интерпретатора, основанного на системе обработки шаблонов и обеспечивающего:

Общее решение. Использование единой методологии при каждом дополнении или изменении программной оболочки фильтрации.

Разделение и независимое выполнение логики применения и интерпретации. Определение и разработка логики шаблонов и управляющих скриптов независимо друг от друга.

Выделение частей проекта. Возможность разделения проекта на отдельные части, что облегчает его поддержку и дает возможность одновременной разработки нескольких шаблонов и использования одних и тех же шаблонов в разных контекстах.

Глобальный контроль над оболочкой. Возможность определения шаблонов общего пользования или глобальных шаблонов. Изменения в подобном шаблоне в одном из проектов автоматически производятся и в остальных проектах, что сокращает время разработки и снижает вероятность ошибок при проектировании оболочки.

Повторное использование правил. Правила интерпретации и генерации фильтров могут комбинироваться разработчиками при создании различных фильтров.

© National Library of Armenia

Далее рассмотрен язык, позволяющий описывать подобные механизмы фильтрации в виде шаблонов и показана возможность его применения для генерации фильтров в рамках предлагаемой системы.

Во второй главе описан подход к генерации среды фильтрации избыточной информации с использованием шаблонов. Подход построен на инфраструктуре фильтрации, основанной на системе обработки шаблонов. Эта инфраструктура позволяет гибко настраиваться на новые формы и тем самым эффективно фильтровать избыточную информацию.

В §2.1 описана иерархическая модель среды фильтрации избыточной информации, архитектура системы обработки шаблонов и методология генерации шаблонов.

Подход основан на инфраструктуре фильтрации, приведенной на рис. 1, и называемой далее FI (Filtering Infrastructure). Она включает: фильтр-процессор FP(Filter Processor), генератор фильтров FG(Filter Generator), систему интерпретации управляющих скриптов SC(Script Controller), систему генерации результатов, расширяемые наборы правил интерпретации и генерации результатов.

С точки зрения реализации система является настраиваемым интерпретатором, использующим расширяемое множество правил интерпретации. Реализация основана на XML-технологии, а для описания шаблонов фильтрации разработано специальное расширение языка XML – язык FTML (Filter Template Mark-up Language).

Фильтр-процессор производит фильтрацию входной информации, по фильтрам, сгенерированным из шаблонов фильтрации (filtering templates). Шаблоны фильтрации описывают различные ограничения на формат и содержание входной информации.

Генератор фильтров выдает фильтры для данного шаблона в соответствии с значениями задаваемых параметров фильтрации.

На вход FI поступают шаблоны фильтрации, управляющий скрипт и исходный файл. На выходе FI получаются отфильтрованные файлы и сообщения об ошибках.

Система интерпретации управляющих скриптов (обработчик скриптов) обеспечивает управление очередностью применения шаблонов фильтрации для входной информации.

Обработчик скриптов контролирует очередность строк скрипта, входы для каждой строки и перенаправляет выход интерпретации текущей строки скрипта либо в библиотеку данных, либо в библиотеку истории ошибок. Каждая строка скрипта содержит информацию о шаблоне и параметрах фильтрации, определяет куда должен быть перенаправлен выход фильтрации, а также очередную строку для выполнения.

Одни и те же шаблоны фильтрации и правила генерации результатов могут входить в состав разных шаблонов.

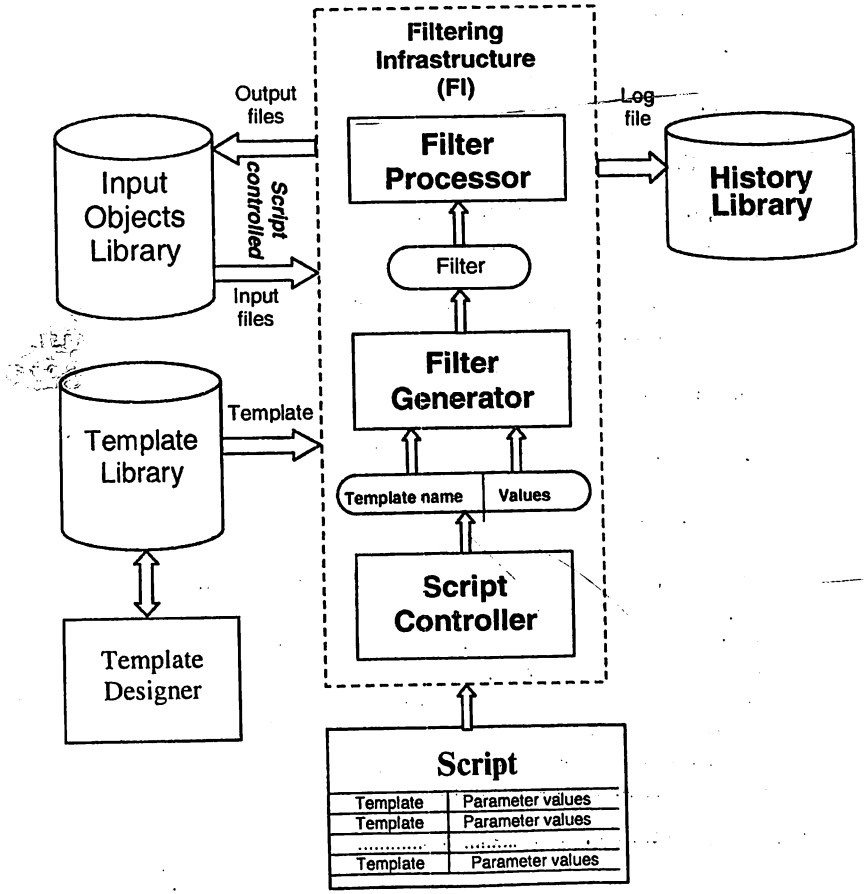


Рис. 1 Архитектура системы обработки шаблонов

© National Library of Armenia

Конфигурируемость с помощью шаблонов обеспечивает необходимую гибкость и адаптируемость к изменениям в спецификации среды фильтрации с помощью языка описания шаблонов FTML. При помощи различных значений параметров одни и те же шаблоны фильтрации могут быть применимы для разных видов спама.

Новые среды фильтрации могут генерироваться на базе уже имеющихся шаблонов или посредством создания новых. Изменение среды будет отражаться только в шаблонах без изменения самой инфраструктуры FI.

Использование шаблонов позволяет также разделить проектирование на максимально независимые части для параллельной реализации, позволив нескольким группам разработчиков независимо работать над одним и тем же проектом. Разработчики имеют возможность изменять среду, используя единую методологию, не делая изменений в FI.

Дизайнеры шаблонов фильтрации и управляющих скриптов имеют возможность определять логику независимо друг от друга.

В §2.2 описан язык описания шаблонов в среде фильтрации избыточной информации. Система фильтрации является настраиваемым интерпретатором. Описаны основные элементы языка FTML. По выполняемым функциям они разделены на следующие группы:

1. **Декларативные тэги:** определяют необходимые переменные и структуры.
2. **Тэги просмотра:** задают необходимые функции для сравнения однотипных объектов.
3. **Тэги обработки:** определяют специальные фильтры для операций, производимых над входными объектами
4. **Тэги ввода/вывода:** задают необходимые механизмы для ввода/вывода информации.
5. **Управляющие тэги:** определяют объекты, используемые в процессе трансляции, задающие последовательность обработки информации и контролирующие последовательность производимых операций в процессе фильтрации.

Описаны также системные переменные.

В языке FTML имеется набор системных переменных, значения которых можно изменять при помощи тэга set:

- _SOURCE_POS – текущая позиция в исходном тексте. Значение переменной может быть получено с помощью функции getSrcPos.
- _CASE_SENSITIVE – флаг, указывающий, зависит ли процесс проверки исходного текста от регистра символов. Принимает значения true и false. Значение переменной может быть получено с помощью функции isCaseSensitive.
- _SPACE_SENSITIVE – флаг, указывающий, являются ли значимыми пробелы между любыми двумя значимыми элементами в процессе обработки. Значение переменной может быть получено с помощью функции isSpaceSensitive.
- Все выражения в FTML имеют форму <операнд1> <оператор> <операнд2>. Все поддерживаемые операторы являются операторами сравнения: = , != , > , < , >= , <= . Все операторы оперируют только с символьными строками. Для операций с другими типами данных должны быть созданы дополнительные функции, которые производят конвертацию типов. К примеру, для произведения целочисленного суммирования создается функция sum(value1, value2), которая преобразует символьные значения value1 и value2 в их целочисленные значения, суммирует их, конвертирует результат обратно в символьную строку и возвращает ее.

FTML поддерживает следующие специальные знаки:

- #SPACE – знак возможного пробела(ов). Используется для указания транслятору пропустить все пробелы в исходном тексте в случае, если системная переменная _SPACE_SENSITIVE установлена в true (в обратном случае эти пробелы являются незначимыми и пропускаются автоматически).
- @ - знак извлечения значения переменной. Синтаксис: @<имя переменной>@.
- $ - знак начала функции.

Все стандартные XML знаки поддерживаются подобно знакам комментариев <!-- ... -->.

В §2.3 рассмотрены примеры использования тэгов для реализации фильтрации.

В среде Chat фильтрация состоит из двух этапов. На первом этапе определяются пользователи Chat-a, которые во время разговоров набрали наибольшее количество штрафных очков. На втором этапе

© National Library of Armenia

фильтруются все тексты, которые поступают в систему от этих пользователей.

На первом этапе входными данными для инфраструктуры фильтрации служат тексты разговоров, представленные в следующем формате: usernick, timestamp, messagetext. Для расчета штрафных очков из файла разговоров последовательно выбирается очередная строка и проверяется совпадение текста строки со списком «запрещенных» слов (проверка производится до первого совпадения). Список «запрещенных» слов представляет собой список регулярных выражений. С каждым выражением связывается количественное значение штрафных очков. При совпадении в специальном списке добавляется пара пользователь-штрафные очки. А при наличии пользователя в списке добавляются штрафные очки на счет пользователя (в соответствии со связанным значением штрафных очков). Таким образом, получается список в следующем формате: usernick, points. Полученный список служит входной информацией для второго этапа.

На втором этапе из списка пользователей последовательно выбирается очередная строка и число штрафных очков сравнивается с допустимым значением. Если пользователь превысил допустимое значение, выдается сообщение. Аналогичным образом можно построить шаблоны фильтрации для других WEB приложений.

Далее рассмотрено построение шаблонов для среды E-mail.

Рассмотрены методы фильтрации, описанные в §1.1.

Для каждого из методов выделены базовые шаги, для которых построены шаблоны на языке FTML. Показано, как с помощью этих примитивов можно реализовать рассмотренные методы, а также их различные комбинации.

В третьей главе предложен алгоритм построения полной системы примеров для верификации шаблонов фильтрации, построен прототип программной оболочки фильтрации избыточной информации, на котором опробована эффективность предложенных решений, и разработан алгоритм автоматического синтеза фильтров для некоторых классов избыточной информации.

В §3.1 описан процесс проектирования тестов для шаблонов фильтрации, состоящий из следующих шагов:

1) разработка тестов для каждой ситуации и возможности, содержащейся в описании шаблона, для каждой границы областей

допустимых значений всех входных данных, областей изменения данных и для всех недопустимых условий;

2) проверка охвата тестами всех условий в алгоритме функционирования шаблона и добавление новых тестов при необходимости;

3) проверка полноты покрытия тестами путей в алгоритме функционирования и добавление новых тестов при необходимости;

4) проверка охвата тестами особых случаев функционирования шаблона и добавление новых тестов при необходимости.

Когда задано формальное описание алгоритма функционирования шаблона, можно использовать подход, основанный на работах по построению полной системы примеров для тестирования программ. Дальнейшие построения основаны на использовании дерева развертки алгоритма функционирования шаблона аналогично построению дерева развертки алгоритмических описаний аппаратуры. *)

Систему тестов для заданного шаблона будем называть полной, если каждый линейный участок (ветвь) алгоритма функционирования шаблона покрывается тестом данной системы **).

По описанию алгоритма шаблона строится дерево развертки, обладающее следующими свойствами:

- все пути, ведущие из начальной вершины дерева развертки в одну из заключительных, реализуемы, т.е. для каждого такого пути существует набор значений входных переменных, при которых выполнение алгоритма пойдет по данному пути;
- ветвь алгоритма, принадлежащая реализуемому пути, принадлежит пути, ведущему из начальной вершины дерева развертки в одну из заключительных вершин.

При построении дерева развертки составляются и анализируются условия реализуемости путей. Путь с тождественно ложным условием реализуемости отсекается. Получение конечного дерева развертки для алгоритма функционирования шаблона не всегда возможно или требует прохождения большого числа ветвей дерева. В этих случаях на практике можно отсекать не только пути с тождественно ложными условиями реализуемости, но и часть реализуемых путей (например, разворачивать циклы не более одного раза), получая тем самым систему примеров, степень полноты которых определяется степенью отсечения реализуемых путей.

---

*) И.Б.Мкртумян, Б.М.Каган, Основы эксплуатации ЭВМ,издательство «Энергия», 1986 г.

**) Барздинь Я. М., Бичевский Я. Я., Калниньш А. А., Построение полной системы примеров для проверки программ. – Учен. зап. латв. Ун-та, 1974, № 210, с. 152-187.

© National Library of Armenia

Техника построения полной системы тестов для шаблона показана на примере описания шаблона выполнения фильтрации избыточной информации в системе chat, описанном в **§2.3**.
Для удобства изложения, алгоритм выполнения преобразования информации, описанный на языке FTML, представлен в несколько упрощенной форме. В частности, из описания алгоритма исключены переменные, значения которых не определяются значениями входных переменных, а также циклы.

Конечное дерево развертки приведено на рис. 2.

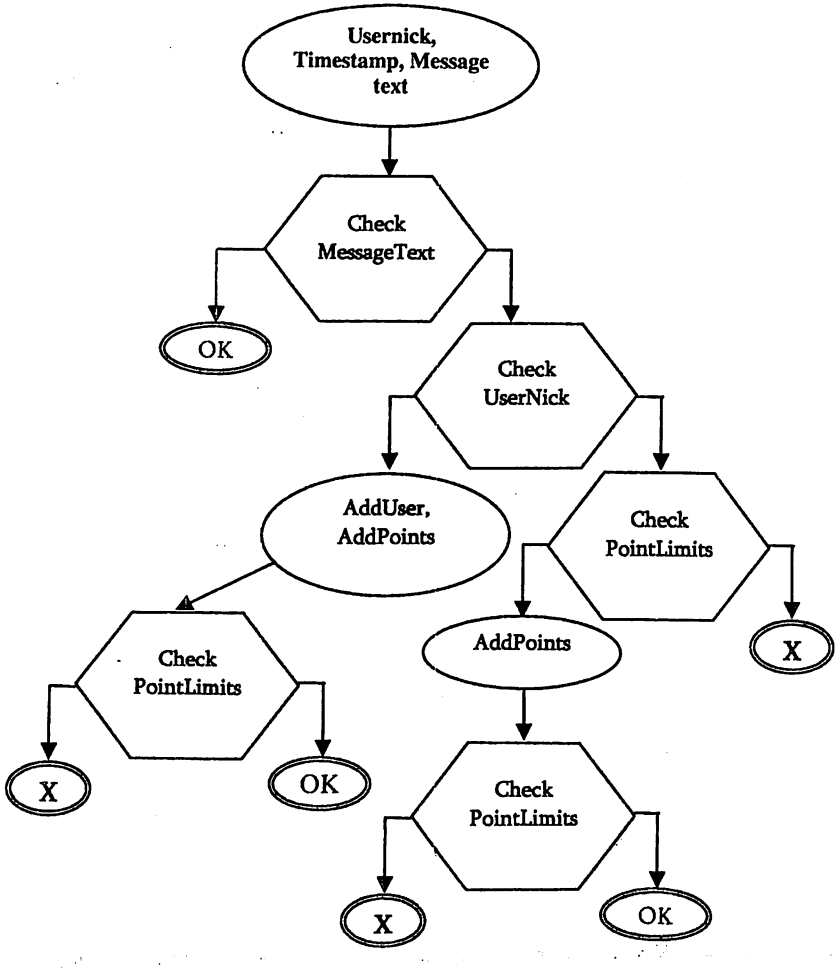


Рис. 2 Дерево развертки алгоритма функционирования

В §3.2 рассмотрен прототип системы фильтрации, на котором опробована эффективность предлагаемых решений.

В качестве основы системы обработки шаблонов взята система, разработанная на кафедре алгоритмических языков ЕГУ *). Расширение интерпретации тэгов на специальные тэги, задающие операции фильтрации избыточной информации, позволяет использовать уже построенную систему обработки шаблонов для генерации фильтров избыточной информации.

Для автоматизации построения фильтров избыточной информации, которые могут быть далее преобразованы в шаблоны фильтрации, разработана специальная программная система REFS (Regular Expression Filter Synthesis). Она внедрена в компании Lycos CJSC и проведены эксперименты, подтверждающие эффективность ее использования.

Система REFS обеспечивает обнаружение нежелательных с точки зрения пользователя лексических оборотов и выполнение соответствующих действий в зависимости от степени нежелательности. Функционирование системы можно описать следующей последовательностью шагов:

1. Задание пользователем потенциально нежелательного лексического оборота;
2. Преобразование заданного оборота в регулярное выражение;
3. Синтез фильтра лексического оборота по полученному регулярному выражению и его запоминание в базе данных;
4. Присвоение рейтинговых баллов накопленным в базе фильтрам лексических оборотов;
5. Выбор критерия фильтрации оборотов;
6. Фильтрация оборотов в соответствии выбранным критерием.

Есть возможность уточнения фильтров в базе данных, путем изменения соответствующих регулярных выражений.

В течение 6 месяцев выполнялось апробация разработанной системы в компании Lycos CJSC для фильтрации избыточной информации в клиент-серверной системе переговоров Chat. Ее результаты приведены в таблице 1 и на рис. 2. Отсутствие данных литературы по процентному отношению спама в сообщениях Chat не позволяет выполнить точную оценку эффективности предложенного

*) S.Baghdasaryan, S.Shoukourian, **An approach for CTL Implementation**, Proceedings of IEEE 6th International Workshop on Testing of Embedded Core Systems, Monterey, USA, pp. 43-49, May 2002.

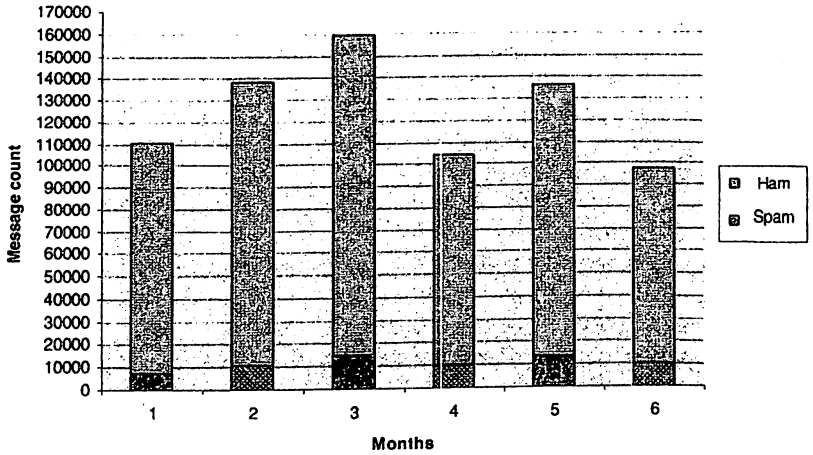
© National Library of Armenia

решения. Однако, если предположить, что процентное отношение спама в систем Chat совпадает процентным отношением спама в систем E-mail (по данным литературы 30-80%), то можно утверждать, что в процессе эксперимента было успешно отфильтровано от 12 до 31% спама.

Некоторое монотонное увеличение процента выявленных спам сообщений в последние месяцы объясняется использованием возможности коррекции регулярных выражений в базе данных.

Таблица 1

All	110537	138323	159396	104401	136273	97862
Spam	7553	10924	15277	10216	13686	10175



Для 50% доли спама в трафике

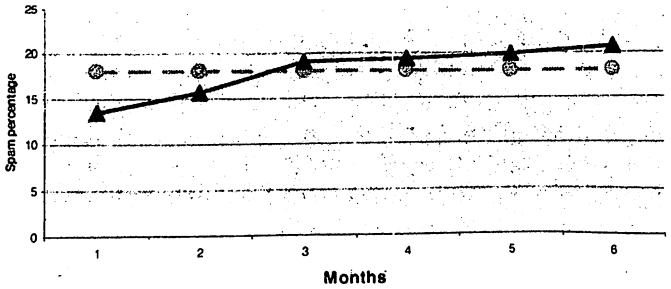


Рис 2. Результаты апробации системы фильтрации

В § 3.3 описан алгоритм синтеза фильтров для определенного класса избыточной информации, который обобщает алгоритм синтеза, использованный в системе REFS.

Предположим, что сообщения, поступающие на вход фильтра, а также формы спама можно представить в виде регулярных выражений. Обозначим множество входных сообщений через X, множество рассматриваемых форм через Q и P. Пусть поступающие сообщения полностью состоят из спама. Это можно представить в виде X=Q.

Предположим теперь, что поступающие сообщения содержат спам как суффикс. Тогда X= XP.

В случае наличия двух альтернатив, описанных выше, одновременно, получается что: X=QvXP.

Это уравнение в алгебре регулярных языков называется каноническим леволинейным уравнением**). Оно имеет минимальное решение X=QP*.

Пусть A – регулярное выражение, e∈A. Тогда AQP* также является решением рассмотренного уравнения.

Таким образом, для определения рассмотренных форм спама достаточно построить (синтезировать) фильтр для регулярного выражение QP*. Многократное его применение позволит полностью отфильтровать сообщения от рассмотренных форм спама. Алгоритм синтеза фильтра следует из алгоритма канонического синтеза автомата по регулярному выражению.

Рассуждения можно обобщить для случаев, когда сообщения состоят из отдельных частей, каждая из которых может быть подвержена действию спама. Например, в случае электронной почты такими частями могут быть: предмет сообщения, адреса рассылки, тело сообщения и т.д. При этом форма спама, использованная в одной части сообщения, может зависеть от формы спама использованной в другой части сообщения. Тогда приходим к системе уравнений в алгебре регулярных выражений

$$X_i = \bigvee_{j=1}^{n} X_j P_{ij} \vee Q_i, i=1,\ldots,n$$

Методом исключения неизвестных можно определить наименьшее решение этой системы. Алгоритм синтеза шаблонов здесь основан на

**) А.А.Летичевский, Ю.В.Капитонова Математические основы проектирования ЭВМ, «Наука», 1985 г.

© National Library of Armenia

алгоритме решения системы уравнений методом исключения неизвестных и алгоритме канонического синтеза автоматов.

Приведем пример применения описанного подхода для системы электронной почты.

$\Sigma = \{a_1 a_2 a_3 \ldots a_n \mid a_i \in A,\ n \in N\}$

Предмет разговора - X_1

Q_1 = 'Viagra' $\Rightarrow Q_1 \subseteq \Sigma$

$P_1 = \Sigma^k[V\ v]\Sigma^k[I\ i]\Sigma^k[A\ a]\Sigma^k[G\ g]\Sigma^k[R\ r]\Sigma^k[A\ a]\Sigma^k$

$k=0,1,2,\ldots$

Тело сообщения – X_2

Q_2 = 'Kill'

$P_2 = \Sigma^*$'kill [you me them him]'$\Sigma^* \cup \Sigma^* Q_2 \Sigma^*$

Адрес отправителя – X_3

Q_3 = 'armen.kirakosyan@yahoo.com'

P_3 = 'armen.'Σ^*@Σ^*.com'$\cup$'armen@'$\Sigma^* \cup$ 'kirakosyan@'Σ^*.com'

Адрес получателя – X_4

Q_4 = 'artak.miletbashyan@lycos-europe.com'

P_4 = 'artak.miletbashyan@Σ^*.com' $\cup$ 'artak@'Σ^*.com'

Шаг 1:

$X_1 = X_1 P_1^+ \vee Q_1^*$	*- взаимосвязь с предметом разговора*
$X_1 = X_3 P_3^+ \vee X_4 P_4^+ \vee Q_3 \vee Q_4$	*- взаимосвязь предмета разговора с адресами получателя и отправителя*
$X_2 = X_2 P_2^* \vee Q_2^*$	*- взаимосвязь с телом сообщения*
$X_2 = X_1 P_1^+ \vee Q_1^*$	*- взаимосвязь предмета разговора с телом сообщения*
$X_3 = X_3 P_3 \vee Q_3$	*- взаимосвязь с адресом отправителя*
$X_4 = X_4 P_4 \vee Q_4$	*- взаимосвязь с адресом получателя*
$X_3 = X_4 (P_3 \vee P_4)^1 \vee (Q_3 \vee Q_4)$	*- взаимосвязь адреса отправителя с адресом получателя*

Шаг 2: *После трансформации имеем следующую систему уравнений регулярных языков*

$X_1 = X_1 P_1^+ \vee X_3 P_3^+ \vee X_4 P_4^+ \vee (Q_1^* \vee Q_3 \vee Q_4)$

$X_2 = X_1 P_1^+ \vee X_2 P_2^* \vee (Q_1^* \vee Q_2^*)$

$X_3 = X_3 P_3 \vee X_4 (P_3 \vee P_4)^1 \vee (Q_3 \vee Q_4 \vee Q_5)$

$X_4 = X_4 P_4 \vee Q_4$

Шаг 3:

$X_4 = X_4 P_4 \vee Q_4 \quad \Rightarrow \quad X_4 = Q_4 P_4^*$

Продолжая таким образом, окончательно получаем:

Шаг 9:

$X_1 = Q_4 P_4^* P_3^+ P_1^+ \vee Q_3 P_3^+ P_1^+ \vee Q_4 P_3^+ P_1^+ \vee Q_5 P_3^+ P_1^+ \vee Q_4 P_4^+ P_1^+ \vee Q_1^* P_1^+ \vee Q_3 P_1^+ \vee Q_4 P_1^+$

$X_2 = Q_4 P_4^* P_3^+ P_1^+ \vee Q_3 P_3^+ P_1^+ \vee Q_4 P_3^+ P_1^+ \vee Q_5 P_3^+ P_1^+ \vee Q_4 P_4^+ P_1^+ \vee Q_1^* P_1^+ \vee Q_3 P_1^+ \vee Q_4 P_1^+ P_2^* P_1^+ \vee Q_1^* P_1^+ \vee Q_2^* P_1^+$

$X_3 = Q_4 P_4 P_3^+ \vee Q_4 P_4^+ P_3^* \vee Q_3 P_3^* \vee Q_4 P_3^* \vee Q_5 P_3^*$

$X_4 = Q_4 P_4^*$

Аналогичные рассуждения проводятся и для синтеза шаблонов соответствующих фильтров.

В заключении перечислены основные результаты работы.

В приложении приведены справки о внедрении результатов диссертации.

© National Library of Armenia

ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ

1. Разработана методология генерации сред фильтрации избыточной информации, основанная на системе обработки шаблонов.
2. Построена модель иерархической среды фильтрации избыточной информации и предложена ее реализация в виде иерархии шаблонов и системы ее обработки.
3. Предложен и реализован язык описания шаблонов фильтрации избыточной информации.
4. Разработан алгоритм построения полной системы примеров для верификации шаблонов.
5. Разработан алгоритм синтеза корректных фильтров и шаблонов фильтров для определенных классов избыточной информации.
6. Разработана система синтеза фильтров избыточной информации в виде объектно-ориентированного Web-приложения для клиент-серверной системы переговоров Chat. Выполнено практическое внедрение системы и получена оценка ее эффективности.

Перечень публикаций по теме диссертационной работы

1. А. Авагян, А. Милетбашяан. О генерации среды фильтрации избыточной информации с использованием шаблонов. Математические вопросы кибернетики и вычислительной техники 24, Ереван 2005, 120-124.
2. А. Авагян, А. Милетбашян. О расширении языка XML для описания шаблонов в среде фильтрации избыточной информации. Сборник материалов международной молодежной конференции «Информационные технологии». Ереван 2005, 183-187.
3. А. Милетбашян. Построение и верификация шаблонов фильтрации избыточной информации. Препринт 05-005 Института проблем информатики и автоматизации НАН РА, Ереван 2005, 17 стр.
4. А. Милетбашян. О построении системы примеров для верификации шаблонов фильтрации. Информационные технологии и управление № 4, Ереван 2005, 8-17.

© National Library of Armenia

ԱՄՓՈՓԱԳԻՐ

Արտակ Սեդրակի Միլեթբաշյանի

«Ավելցուկային ինֆորմացիայի ֆիլտրերի ստեղծման ծրագրային միջոցներ՝ WEB համակարգերի օբյեկտա-կողմնորոշված նախագծման համար»

Ատենախոսությունը նվիրված է ավելցուկային ինֆորմացիայի ֆիլտրացման ծրագրային գործիքային միջոցների հիմնավորմանը և նախագծմանը, նրա տիպերի դինամիկ փոփոխման պայմաններում: Այդ միջոցները պետք է ճկունությամբ կարգաբերվեն ավելցուկային ինֆորմացիայի նոր ձևերին, ապահովեն ավտոմատացված նախագծում և կարգավորման պրիմիտիվների վերիֆիկացում, բավարարեն օբյեկտա-կողմնորոշված WEB-համակարգերի նախագծման ստանդարտներին:

Ստորև բերված են ատենախոսությունում ստացված հիմնական արդյունքները՝

1. Մշակված է շաբլոնների մշակման համակարգի վրա հիմնված ավելցուկային ինֆորմացիայի ֆիլտրացիայի միջավայրերի գեներացման մեթոդոլոգիա:
2. Կառուցված է ավելցուկային ինֆորմացիայի ֆիլտրացիայի հիերարխիկ միջավայրի մոդել և առաջարկված է նրա իրականացումը շաբլոնների հիերարխիայի և դրա մշակման համակարգի տեսքով:
3. Առաջարկված և իրականացված է ավելցուկային ինֆորմացիայի ֆիլտրացիայի շաբլոնների նկարագրման լեզու:
4. Մշակված է շաբլոնների վերիֆիկացիայի համար օրինակների լրիվ համակարգի գեներացման ալգորիթմ:
5. Մշակված է ավելցուկային ինֆորմացիայի որոշակի դասի համար կոռեկտ ֆիլտրերի և ֆիլտրերի շաբլոնների սինթեզման ալգորիթմ:
6. Խոսակցությունների կլիենտ-սերվերային Chat համակարգի համար մշակված է ավելցուկային ինֆորմացիայի ֆիլտրերի սինթեզման համակարգ, օբյեկտ-կողմնորոշված Web կիրառության:

15.05.2014

© National Library of Armenia

Объем – 1п.л. Тираж 100 экз.
Лаборатория компьютерной полиграфии ЕрНИИММ

© National Library of Armenia