

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Московский государственный университет имени М.В. Ломоносова»
Научно-исследовательский вычислительный центр

На правах рукописи

Быстрицкий Николай Дмитриевич

**МЕТОДИКА И ИНСТРУМЕНТАЛЬНОЕ СРЕДСТВО
ОЦЕНКИ КОРРЕКТНОСТИ ФУНКЦИОНИРОВАНИЯ
ИНФОРМАЦИОННЫХ РЕСУРСОВ**

Специальность 05.13.11 – Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель
доктор технических наук,
Н.В. Макаров-Землянский

Москва – 2018

Оглавление

Термины и определения	4
Введение.....	6
Глава 1. Анализ существующих методов и методик для проведения автоматизированной оценки корректности функционирования информационных ресурсов.....	14
1.1. Проблемы корректного взаимодействия пользователя с информационным ресурсом в сети Интернет	14
1.2. Особенности составляющих компонентов информационного ресурса	21
1.3. Анализ проведенных исследований в области корректного функционирования информационных ресурсов	23
1.4. Обзор возможностей существующих программных средств	36
1.5. Постановка задачи для проведения исследований	54
Выводы по главе 1	61
Глава 2. Разработка методики оценки корректности функционирования информационного ресурса	63
2.1. Описание структурной схемы методологического аппарата	63
2.2. Разработка алгоритма анализа исходных текстов интернет-страниц информационного ресурса	64
2.3. Разработка алгоритма анализа «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса.....	71
2.4. Разработка алгоритма анализа «сценарных» исходных текстов «подключаемых» внешних файлов информационного ресурса.....	74
2.5. Методика оценки корректности функционирования информационного ресурса.....	84
Выводы по главе 2.....	89

Глава 3. Анализатор исходных текстов информационного ресурса.....	90
3.1. Формирование требований к возможностям анализатора.....	90
3.2. Описание структуры анализатора	91
3.3. Классификация выявляемых ошибок разработанным анализатором.....	97
3.4. Принципы организации работы анализатора при исследовании сверхбольших информационных ресурсов.....	102
3.5. Алгоритм проведения эффективного анализа информационных ресурсов	104
Выводы по главе 3.....	116
Глава 4. Прикладные исследования по оценке корректности функционирования информационных ресурсов	117
4.1. Общие положения при проведении прикладных исследований	117
4.2. Применение разработанного веб-анализатора при эксплуатации информационных ресурсов	122
Выводы по главе 4.....	127
Заключение	128
Литература	131
Приложение А. Обобщенный алгоритм анализа интернет-страниц информационного ресурса	148

Термины и определения

Информационный ресурс – совокупность структурированной информации, содержащейся в информационных системах общего пользования, доступной посредством информационно-телекоммуникационных сетей.

Интернет-ссылка – адрес URL (Uniform Resource Locator), определяющий однозначное месторасположение интернет-страницы или подключаемого внешнего файла информационного ресурса.

Ошибка – случайное (непреднамеренное) изменение данных информационного ресурса, которое при реализации адекватных алгоритмов обработки и использования информации может привести к изменению логики или результатов решения функциональных задач информационного ресурса.

Потенциально-опасная ошибка – случайное (непреднамеренное) изменение данных информационного ресурса, которое при реализации адекватных алгоритмов обработки и использования информации приводит к неприемлемой логике или недопустимому выполнению функциональных задач информационного ресурса.

URN (Uniform Resource Name) – унифицированное (единообразное) название (имя) ресурса.

URI (Uniform Resource Identifier) – унифицированный (единообразный) идентификатор ресурса, который является URL, URN или URL+URN.

XSS атака (Cross Site Scripting) – межсайтовый скриптинг, внедрение в выдаваемую веб-системой страницу вредоносного кода (выполняемую на компьютере пользователя при открытии им этой страницы) и взаимодействие этого кода с веб-сервером злоумышленника

CSRF/XSRF атака (Cross Site Request Forgery) – межсайтовая подделка запроса, выполнение каких-либо действий на уязвимом интернет-ресурсе от лица жертвы (изменение пароля, секретного вопроса для восстановления пароля, почты, добавление администратора и т.д.)

CDN (Content distribution network) – сетевая инфраструктура, позволяющая оптимизировать отправку и хранение контента для конечных интернет-пользователей.

Введение

Реализовав в 1990 году первый в мире веб-браузер WorldWideWeb [1], Tim Berners-Lee заложил основной принцип организации гипертекстовых документов посредством коммуникационного взаимодействия пользователя с информационной системой через сеть Интернет. Такие достоинства как структуризация информации, простота и привычность интерфейса, возможность удаленной работы и быстрота разработки веб-приложения позволили веб-обозревателю стать одним из обязательных самостоятельных приложений в составе большинства операционных систем, а интернет-ресурсам - одним из стратегически важных и динамически развивающихся видов информационных ресурсов. Современный информационный ресурс сегодня представляет собой не просто статичный набор веб-страниц, а многофункциональный портал с использованием различных средств и технологий, в том числе и применением различных шаблонов для разных уровней вложенности.

Неотъемлемой частью работоспособности информационного ресурса является выполнение всех возложенных на него задач и целей, т.е. корректность его функционирования. Такое понятие содержит довольно широкий спектр задач, таких как:

- предоставление и размещение на информационном ресурсе корректных сведений (контента, информации), не содержащих ложных/закрытых данных или запрещенного материала законодательством РФ;

- соответствие используемых технологий информационного ресурса существующим международным стандартам для обеспечения кроссбраузерного функционирования;

- наличие запрашиваемых источников (интернет-страниц, файлов и т.д.), непосредственно относящихся к информационному ресурсу;

- соответствие государственного информационного ресурса существующим требованиям законодательных и нормативно-методических документов РФ.

В 2013 г. Министерством экономического развития Российской Федерации была разработана «Методика мониторинга официальных сайтов органов государственной власти и местного самоуправления» [2]. Одним из важнейших этапов данного мониторинга является проверка корректности информационного ресурса. Исходя из обозначенного термина, разработанная по данной методике система АИС «Мониторинг государственных сайтов» [3] проводит:

- эвристический анализ предоставляемых сведений (контента, информации) органами государственной власти и местного самоуправления;
- поверхностный анализ заглавной страницы исследуемого информационного ресурса с помощью стороннего программного обеспечения без исследования всей структуры интернет-ресурса.

Однако проверка того, что интернет-ресурс корректно отображается в нескольких веб-браузерах, не дает абсолютно никакой гарантии его правильного отображения в других случаях. Существование такой проблемы подтверждают внесенные в 2014 году при разработке спецификации HTML5 консорциумом W3C предложения по анализу структуры HTML-документа [4, п.п. 8.2.8]. Отчасти это связано с постоянно возрастающей сложностью веб-приложения, которая не позволяет разработчикам информационного ресурса своевременно контролировать качество написанного кода, тем самым вовремя выявлять возникающие функциональные ошибки. Такой периодический мониторинг качества кода информационного ресурса необходимо проводить не только при его разработке, но и при его эксплуатации.

Проблеме исследования корректности функционирования веб-приложений были посвящены многие научные работы, проводимые в Российской Федерации и за рубежом, а также в ряде диссертационных работ по данной тематике [5-8].

Теоретическую базу исследования составили работы известных российских учёных: В.Ф. Шаньгин, А.С. Марков, В.Л. Цирлов, А.В. Барабанов, Д.А. Мельников, И.О. Шелухин, Д.Ж. Сакалема, А.С. Филинова, В.В. Ерохин, Д.А. Погоньшева, И.Г. Степченко [9-13], а также зарубежных учёных: D. Stuttard, M. Pinto, J. Pauli, M. Shema, T. Canavan, S. Purewal, C. Eilers, J.R. Vacca,

S. Davidoff, J. Ham [14-20], которые внесли значительный вклад в получение основополагающих результатов в области исследования корректности функционирования веб-приложений и в смежных областях. В этих работах рассматривались особенности функционирования веб-приложений в различных условиях, разрабатывались практические рекомендации по улучшению их функциональности. В ходе проведения научных работ, а также в ряде диссертационных работ по исследуемой области был создан уникальный научный задел, используемый и в настоящее время.

Однако проведенный в работе обзор ведущих программных средств, таких как Rational AppScan (IBM) [26], Web Vulnerability Scanner (Acunetix) [27], NTOSpider (NT Objectives, Inc.) [28], NetSparker (Netsparker Ltd.) [29], WebInspect (HP) [30], Application Inspector (PT) [31], SkipFish (Google) [32], Validator Suite (W3C) [33] и др. показал, что на сегодняшний день не существует программного средства, которое могло бы предоставить достоверную оценку корректности функционирования всего интернет-ресурса и провести комплекс мероприятий, направленных на устранение функциональных ошибок и повышение общей безопасности интернет-ресурса.

В результате, в сложившихся обстоятельствах, рассматриваемая проблема актуальна как для государственных информационных ресурсов, так и для интернет-ресурсов коммерческих компаний, деятельность которых базируется на функционально-корректном предоставлении информации и услуг через сеть Интернет, что в свою очередь предопределяет необходимость совершенствования методов и методик автоматизированного выявления ошибок в работе информационных ресурсов.

В диссертационной работе проведены новые исследования в области обеспечения корректного функционирования веб-приложений, активно проводимых мировым сообществом в течение последних 15 лет одновременно с развитием и совершенствованием телекоммуникационных технологий, и направленных на теоретическое и экспериментальное исследование проблем функциональной корректности веб-приложений в сети Интернет. Использование разработанного в

диссертации нового методического аппарата позволит не только повысить эффективность функционирования информационного ресурса, но и получить достоверную оценку его корректности за счет использования новых подходов в проведении анализа. Полученные в диссертационной работе результаты будут способствовать более полному решению проблем корректного функционирования интернет-ресурсов. Диссертационный материал и содержащиеся в нем выводы и предложения могут быть использованы в качестве основы для проведения дальнейших научных исследований и практического совершенствования корректности функционирования интернет-ресурсов.

Объект исследования – информационные ресурсы.

Предмет исследования – методы и инструментальные средства по оценке функциональной корректности информационного ресурса.

Целью диссертационной работы является разработка методики оценки корректности функционирования информационных ресурсов и разработка инструментального средства и практических рекомендаций по улучшению их функциональности для пользователя.

Для достижения поставленной цели в работе поставлены и решаются следующие задачи:

- анализ основных результатов существующих исследований, требований законодательных и нормативно-методических документов, определяющих корректное функционирование информационных ресурсов;
- разработка алгоритмов проведения анализа корректности функционирования исходных текстов интернет-страниц информационного ресурса;
- разработка методики оценки корректности функционирования информационного ресурса;
- разработка алгоритма и инструментального программного средства анализа исходных текстов сверхбольших информационных ресурсов с использованием параллельных технологий для получения за приемлемое время объективной оценки корректности его функционирования;

- на основе проведенных прикладных исследований разработка предложений и рекомендаций по повышению функциональной корректности информационных ресурсов.

Основными научными результатами, выносимыми на защиту, являются:

1. Методика оценки корректности функционирования информационных ресурсов.
2. Алгоритмы проведения анализа исходных текстов интернет-страниц информационного ресурса.
3. Алгоритм анализа исходных текстов сверхбольших информационных ресурсов с использованием параллельных технологий.

Научная новизна диссертационного исследования состоит в следующем:

1. Разработана новая методика определения оценки корректности функционирования информационного ресурса, которая, в отличие от существующих методик, предполагает исследование всей структуры информационного ресурса и выявление особенностей взаимодействия между собой составляющих его элементов.
2. Разработаны новые алгоритмы проведения анализа исходных текстов интернет-страниц информационного ресурса, которые, в отличие от существующих, на основе принципов построения интернет-страниц и исследования функциональных связей, учитывают различные неоднозначные трактовки используемых международных интернет-стандартов.
3. Разработан новый алгоритм анализа исходных текстов сверхбольших информационных ресурсов с использованием параллельных технологий, который позволяет, в отличие от существующих алгоритмов, более эффективно использовать все возможности не только процессорной системы, но и пропускную способность имеющегося канала связи для получения за приемлемое время оценки корректности его функционирования.

При выполнении исследования использовалась методология программирования, теория алгоритмов, методологический аппарат синтаксических моделей теории графов и множеств. Достоверность предлагаемого в диссертации подхода

обоснована проведенными теоретическими и экспериментальными исследованиями.

Практическая значимость полученных результатов состоит:

1. В разработке и реализации на основе созданной методики оценки корректности функционирования информационных ресурсов программного комплекса «Анализатор исходных текстов информационного ресурса «Акула» (Свидетельство № 2015616442).

2. В проведении прикладных исследований по оценке корректности функционирования информационных ресурсов, результаты которых показали недостаточное соблюдение требований международных интернет-стандартов, что позволяет сформировать предложения и рекомендации по повышению функциональной корректности информационных ресурсов.

3. В обеспечении корректного функционирования информационных ресурсов для точного выполнения возложенных на него задач с целью функционально-корректного предоставления пользователям информации и услуг через сеть Интернет и подтверждена актами внедрения следующих организаций:

- ООО «ЦСС» (внедрение в Систему регистрации, анализа и мониторинга событий информационной безопасности);

- ФНС России (внедрение для обнаружения и предотвращения компьютерных атак на собственные информационные ресурсы в качестве «Агента мониторинга»);

- Администрация г. Фрязино, ООО «НТЦ «СОТИС» (внедрение в практическую деятельность для сопровождения собственных информационных ресурсов).

Соответствие диссертации паспорту научной специальности. Содержание и результаты диссертационной работы соответствуют паспорту специальностей 05.13.11, а именно следующим областям исследований (пункты 1, 2, 8 и 10):

- модели, методы и алгоритмы проектирования и анализа программ и программных систем, их эквивалентных преобразований, верификации и тестирования;

- языки программирования и системы программирования, семантика программ;
- модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования;
- оценка качества, стандартизация и сопровождение программных систем.

Апробация работы. Основные положения диссертационной работы докладывались и обсуждались на научной конференции «Ломоносовские чтения» (2013, 2016 гг.), на XV международной научно-практической конференции «Современное состояние естественных и технических наук» (2014г.), на XVI международной научно-практической конференции «Техника и технология: новые перспективы развития» (2015г.), на научном семинаре «Проблемы современных информационно-вычислительных систем» под руководством д. ф.-м. н., проф. В.А. Васенина (2015г.), на научно-методологическом семинаре НИВЦ МГУ имени М.В. Ломоносова под руководством д. ф.-м. н., проф. А.В. Тихонравова (2017г.), на семинаре «Оптимальное восстановление по точным и приближенным данным» под руководством д. ф.-м. н., проф. К.Ю. Осипенко (2017г.), на совещании-семинаре работников налоговых органов ФНС России по теме информационной безопасности в Федеральной налоговой службе (2017г.), на регулярных семинарах лабораторий компьютерной безопасности и анализа информационных ресурсов НИВЦ МГУ имени М.В. Ломоносова.

Основные результаты диссертации опубликованы в 9 печатных работах [34-42], среди которых 4 статьи из списка журналов, рекомендованных ВАК [34-35, 41-42].

Личный вклад автора. Все исследования, результаты которых изложены в диссертационной работе, проведены лично автором в процессе научной деятельности. Из совместных публикаций в результаты диссертационной работы включен лишь тот материал, который непосредственно принадлежит автору.

Работы [36-37, 40-42] написаны единолично. В работах [34-35, 38] Быстрицкому Н.Д. принадлежат: подход к проведению анализа веб-приложений, алгоритм

работы программно-аппаратного комплекса «Анализатор исходных текстов информационного ресурса», обзор исследований по безопасному функционированию информационных ресурсов, Макарову-Землянскому Н.В. принадлежит постановка задачи и проверка результатов. В работе [39] Быстрицкому Н.Д. принадлежит обзор существующей концепции безопасного взаимодействия пользователя с информационным ресурсом, Мартьянову Е.А. принадлежит постановка задачи исследования по оценке защищенности информационных ресурсов.

Диссертация состоит из введения, четырех глав с выводами по каждой из них, заключения, списка цитируемой литературы и приложения. Общий объем работы составляет 148 страниц машинописного текста, включая 30 рисунков, 10 таблиц и списка литературы из 155 наименований.

Глава 1. Анализ существующих методов и методик для проведения автоматизированной оценки корректности функционирования информационных ресурсов

1.1. Проблемы корректного взаимодействия пользователя с информационным ресурсом в сети Интернет

Современный веб-браузер уже значительно отличается от того первоначального веб-браузера Тима Бернерса-ли, превратившись из простого средства просмотра текстовой информации в комплексное прикладное программное обеспечение для обработки данных и обеспечения интерфейса для взаимодействия между информационными ресурсами и человеком. Одним из его ключевых особенностей является поддержка управления веб-приложениями [17].

Веб-приложение представляет собой клиент-серверное приложение, логика которого распределена между клиентом (в данной роли выступает веб-браузер) и сервером (веб-сервер, обрабатывающий коммуникационные HTTP-запросы от клиента). Для создания веб-приложений на стороне сервера используются различные технологии и языки программирования, в то время как на стороне клиента – HTML и CSS для графического отображения и JavaScript, Java, ActiveX, Silverlight и др. для формирования и обработки запросов.

С точки зрения корректного функционирования, использование веб-приложений требует соблюдение определенных мер предосторожности. Это связано со следующими причинами:

1. Передача информации между пользователем, использующим веб-браузер, и веб-сервером происходит посредством сети Интернет (передаваемая информация может быть перехвачена злоумышленником);

2. Клиентской частью веб-приложения осуществляется обработка данных при взаимодействии с пользователем (полученная информация со стороны сер-

верной части веб-приложения может быть ошибочно обработана, либо произведена атака со стороны информационного ресурса);

3. Серверной частью веб-приложения осуществляется обработка данных при взаимодействии с пользователем (ошибочная обработка полученной информации со стороны клиентской части веб-приложения, либо произведена атака со стороны пользователя).

Рассмотрим безопасное использование веб-приложений со стороны клиента. Решение данной проблемы возможно несколькими путями. Одним из путей является встроенная защита в самом веб-браузере. Это логичное решение, т.к. пользователь работает напрямую с веб-браузером, совершая различные действия, соответственно при этом взаимодействии необходимо осуществлять контроль над действиями пользователя.

Для удобства проведения дальнейшего анализа необходимо определить веб-браузер. Конечно, в данном конкретном случае исследование будет исходить из определенных специфических фактов рассматриваемого веб-обозревателя, однако все популярные веб-браузеры имеют практически идентичный набор возможностей.

Таким образом, рассмотрим данный путь решения на примере популярного веб-браузера Mozilla Firefox. Для безопасной работы в сети Интернет Firefox обладает следующими особенностями [43]:

1. приватный просмотр интернет-страниц;
2. блокировка всплывающих окон;
3. безопасное хранение паролей и сертификатов;
4. контроль при установлении защищенного соединения;
5. встроенная защита от фишинга и вредоносных программ.

Первый параметр является незначимым, т.к. позволяет не сохранять информацию о посещенных сайтах внутри веб-браузера, однако данная информация все равно как минимум останется у провайдера, предоставляющего доступ в сеть Интернет. Второй параметр является необходимым, потому как в большинстве случаев всплывающие окна носят рекламный характер. Тем не менее, включенная

данная возможность может создать проблемы при работе с информационными ресурсами, использующими, например, всплывающие окна для обеспечения доступа к важным функциям. Дополнительно, блокирование всплывающих окон может не всегда сработать, некоторые всплывающие окна могут носить скрытый характер (открытие нового фонового окна) или иметь неизвестный метод открытия. Таким образом, веб-браузер Firefox может справиться лишь только с большей частью всплывающих окон. Следующий параметр - безопасное хранение паролей и сертификатов - еще одна необходимая особенность при работе с веб-браузером в сети Интернет, т.к. в случае их кражи злоумышленник не должен извлечь из этого выгоду. Существенными параметрами являются последние два показателя. В процессе работы веб-браузер Firefox анализирует действия пользователя и предупреждает о подозрительных ситуациях:

- «это соединение является недоверенным» [44];
- «имеется информация, что этот сайт атакует компьютеры!» [45].

Первое сообщение появляется в случае, если при работе с зашифрованным соединением не удастся точно определить статус сертификата. Второе сообщение показывает, что работа с указанным информационным ресурсом в панели адреса может быть небезопасна. Такая ситуация происходит при обнаружении интернет-ресурса в списках известных фишинговых и вредоносных сайтов. Такой мониторинг, с одной стороны, позволяет обезвредить большинство опасных информационных ресурсов, однако с другой стороны информационный ресурс может попасть в данный список по ошибке, хотя не представляет для пользователя никакой угрозы.

Таким образом, рассмотренный аспект на примере веб-браузера Mozilla Firefox показал, что предоставляемая безопасность пользователя на уровне веб-браузера является недостаточной для обеспечения корректного функционирования.

Другой путь – анализ интернет-трафика веб-браузера пользователя. Реализация данного пути обеспечивается межсетевым экраном или персональным файрволом.

Для семейства операционных систем Microsoft Windows межсетевой экран «Брандмауэр Windows» [46] входит в состав операционной системы. Данный межсетевой экран осуществляет контроль доступа программ в сеть Интернет, таким образом, лишь фиксирует факт наличия доступа. Следовательно, проведение анализа интернет-трафика веб-браузера не представляется возможным.

Рассмотрим этот путь решения на примере персонального файрвола Outpost Firewall Pro [47-48]. Данный программный продукт состоит из следующих ключевых компонентов:

- брандмауэр;
- антишпион;
- проактивная защита;
- веб-контроль;
- защита личных данных.

Компонент Антишпион помогает выявить нежелательные и несанкционированные действия вредоносных программ во время работы в сети Интернет. Следующий компонент – Проактивная защита, представляет собой поведенческий блокиратор, не позволяющий вредоносным процессам осуществлять действия от имени доверенных процессов. Наиболее важным для исследования компонентом является компонент Веб-контроль. Он позволяет контролировать работу интерактивных элементов, встроенных в загружаемую интернет-страницу, блокировать всплывающие окна, рекламные объявления и баннеры, а также проводить блокировку вредоносных информационных ресурсов.

Таким образом, Outpost Firewall Pro позволяет обезопасить пользователя от большинства проблем при навигации в сети Интернет, однако все данные проблемы будут обнаружены только при непосредственном контакте с атакующей веб-страницей. В результате, не представляется возможным заранее утверждать о корректном функционировании информационного ресурса в целом.

Outpost Firewall Pro не позволяет полностью обезопасить персональный компьютер пользователя, т.к. не обладает полноценной антивирусной защитой. Для этих целей разработаны комплексные средства, сочетающие в себе антивирус

и брендмауэр, такие как Outpost Security Suite Pro [47], Dr. Web Security Suite [49] и др. Однако даже в таком случае это все равно не решает указанную выше проблему.

В результате, рассмотренные выше способы помогают повысить безопасность пользователя в сети Интернет, помогая пресечь несанкционированные действия со стороны информационного ресурса.

Необходимо также учитывать, что любой пользователь при использовании интернет-ресурса уже является потенциальным нарушителем. Отсюда следует вывод, что политика безопасности информационного ресурса должна быть выстроена таким образом, чтобы предусмотреть возможные действия ошибочного характера. **Рассмотрим основные проблемы и пути решения при обеспечении безопасного функционирования интернет-ресурса.**

К проблемам безопасного функционирования информационно ресурса можно подходить с разных позиций. Базовой позицией для полноценного функционирования интернет-ресурса является правильный выбор хостинга (площадки для размещения информационного ресурса). В большинстве случаев такой выбор необходимо делать исходя из следующих критериев:

- выбор оптимального хостинг-провайдера, учитывая наличие тех или иных служб и возможностей [16, гл. 2];

- настройка собственного интернет-сервера;

Для большинства потребительских нужд и размещения сравнительно небольшого информационного ресурса первый вариант является наилучшим ввиду того, что хостинг-провайдер помимо размещения интернет-ресурса также должен предоставлять сопутствующий пакет услуг [25, ст. 16; 50, п.п. 5.4; 51, п. XIV, XVI]:

- защита интернет-сервера;

- защита телекоммуникационной сети от атак;

- защита хостинга (интернет-площадки со стороны веб-сервера);

- защита используемого программного обеспечения и приложений, разработанных хостинг-провайдером;

- проведение профилактических мероприятий для поддержания работоспособности веб-сервера.

Второй вариант представляет собой построение собственной политики безопасности в отношении как интернет-сервера, так и площадки для информационного ресурса, а также требует затрат средств не только на программную и аппаратную составляющую, но и решение такой задачи как стабильное подключение к информационно-телекоммуникационной сети Интернет [16, гл. 5]. Чаще всего, такой вариант требуется для решения специфических задач, например, невозможность найти на рынке конкретного хостинг-провайдера с необходимыми или требуемыми возможностями. Таким образом, если в первом случае часть обязанностей на себя берет хостинг-провайдер, то во втором случае вопросами безопасного функционирования необходимо будет заниматься владельцу информационного ресурса самостоятельно.

Следующий вопрос полноценной работы информационного ресурса связан с решением проблем безопасного функционирования. Для обеспечения удобного управления и поддержания актуальной информации разработчиками на стадии проектирования интернет-ресурса на веб-сервер устанавливается система управления содержимым (Content management system, CMS). Выбор одной из CMS зависит от поставленных целей и задач, которые должна решать система управления содержимым. Однако вне зависимости от этого CMS должна отвечать следующим требованиям в области информационной безопасности [16, гл. 1,3]:

- защищенный вход в административную часть (скрытие непосредственного входа);
- стойкая учетная запись пользователя/администратора CMS (неассоциативное имя пользователя и пароль);
- измененный стандартный префикс у используемой БД;
- использование механизма поисковой оптимизации (SEO) для вывода ссылок в виде SEF (Search Engine Friendly) (скрытие запросов к БД);
- защита от вредоносного кода при запросе к БД (защита от SQL-инъекций);

- фильтрация веб-ссылок, выходящих за рамки публичных материалов информационного ресурса (перенаправление на главную страницу или специально созданную веб-страницу «Запрашиваемая информация не найдена»);

- скрывание информации об используемой CMS (для предотвращения атак, направленных на уязвимости конкретной CMS);

- защита (гарантированное скрывание) адреса электронной почты (для пресечения массовой рассылки спама);

- защита встроенного механизма от сброса пароля пользователя CMS;

- правильно выставленные права на папки и файлы CMS.

Однако данных действий для обеспечения полной безопасности информационного ресурса будет недостаточно. Помимо вышеперечисленных проблем, разработанный информационный ресурс также должен [16, гл. 6-7]:

- иметь исправный протестированный исходный код интернет-страниц, используемых сценарных и стилевых файлов;

- исходный код интернет-страницы должен соответствовать стандартам HTML [4], CSS [52] и JavaScript [53], исходный код внешнего сценарного файла – стандарту JavaScript, исходный код внешнего стилевого файла – стандарту CSS.

- разработанный исходный код интернет-страниц, а также код используемых внешних сценарных и стилевых файлов должен полностью удовлетворять вопросам безопасности таким образом, чтобы не нанести вред или не совершить атаку на пользователя, использующего данный интернет-ресурс.

Отмеченные выше требования по обеспечению безопасности информационного ресурса позволяют существенно повысить его защищенность, однако не решают полностью поставленной задачи. Во-первых, использование текущих технологий не всегда явно показывает скрытые существующие проблемы, и лишь со временем приходит дальнейшее переосмысление или выявление некоторых из них. Во-вторых, используемые технологии постоянно совершенствуются, и выявленные сегодня проблемы могут не затронуть проблем, которые возможно возникнут в будущем.

Вышеизложенные рекомендации для безопасности интернет-ресурсов необходимо использовать при разработке и эксплуатации информационного ресурса.

1.2. Особенности составляющих компонентов информационного ресурса

Функционирование любого информационного ресурса зависит от различных составных компонентов (Рис. 1). Список используемых компонентов может варьироваться в зависимости от задач, поставленных перед информационным ресурсом. В большинстве случаев, в состав информационного ресурса входят такие технологии как HTML, CSS и JavaScript.

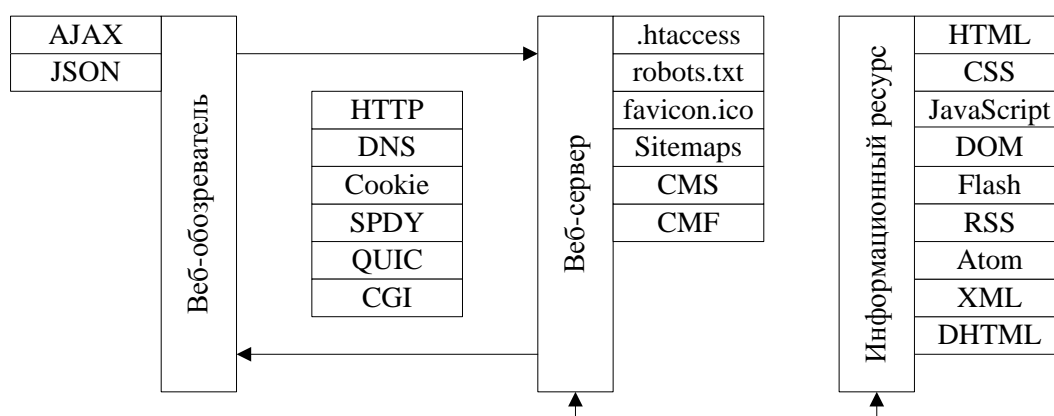


Рис. 1. Составляющие компоненты для работы информационного ресурса

Главная трудность при решении задачи исследования заключается в практической невозможности её выражения в аналитическом виде. Поэтому для её решения целесообразно провести декомпозицию общей задачи исследования на отдельные подчастные задачи, допускающие их точную формализованную постановку и последующее решение, и на этой основе научно обосновать требования к обеспечению функциональной корректности информационных ресурсов.

Исходя из этого, общая задача проведения диссертационных исследований может быть декомпозирована на следующие частные подзадачи (Рис. 2):

- разработка алгоритма анализа исходных текстов интернет-страниц информационного ресурса;

- разработка алгоритма анализа «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса;

- разработка алгоритма анализа «сценарных» исходных текстов «подключаемых» внешних файлов информационного ресурса;

- разработка принципов проведения исследования исходных текстов системы управления содержимым при углубленном анализе информационного ресурса.

Таким образом, сформулированная выше научно-исследовательская задача сведена к иерархической системе взаимосвязанных частных задач, для решения которых необходима разработка следующих алгоритмов и методик, согласованных по входным и выходным параметрам:

- методика оценки корректности функционирования информационных ресурсов,

- алгоритм работы анализатора исходного кода - программно-аппаратного комплекса нового поколения «Анализатор исходных текстов информационного ресурса «Акула»;

- алгоритм проведения эффективного анализа информационных ресурсов с использованием параллельных технологий,

- алгоритм формирования исходных данных для проведения этапов прикладных исследований информационных ресурсов,

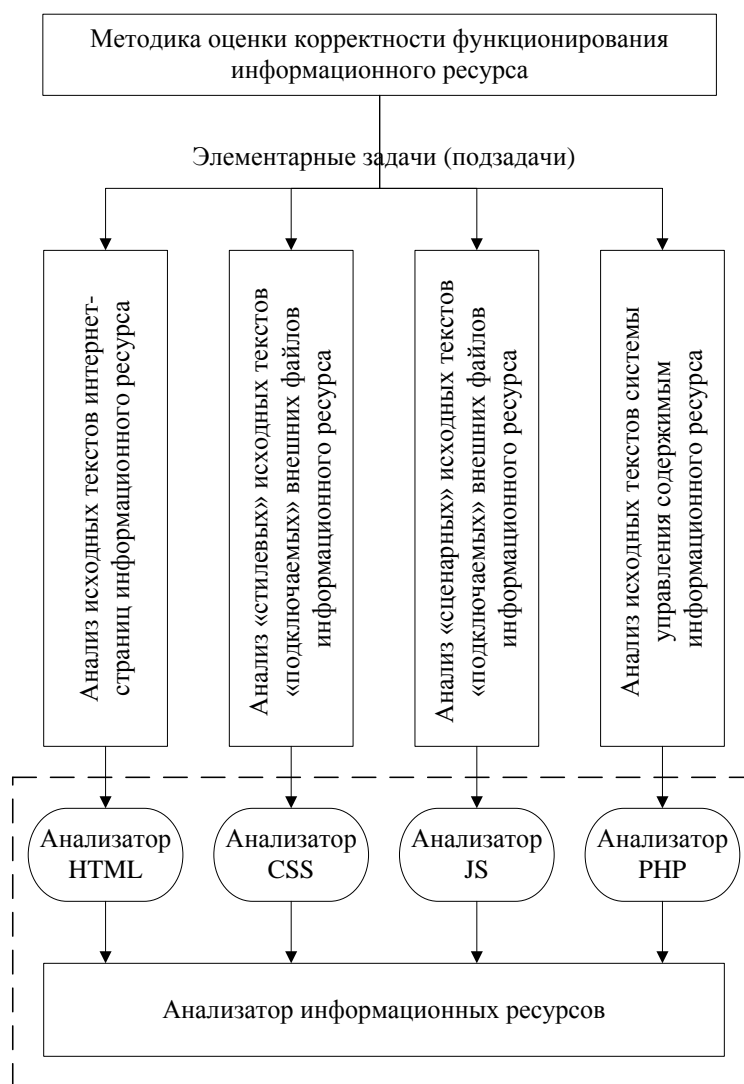


Рис. 2. Декомпозиция задачи исследования на частные подзадачи

1.3. Анализ проведенных исследований в области корректного функционирования информационных ресурсов

Как отметил в 2014 году на конференции IEEE Internet Computing международный эксперт Т. Grandison [64], за последние 10 лет Интернет значительно эволюционировал, и из некогда амбициозного проекта превратился в обычную повседневную часть жизни практически любого человека. Сегодня сеть Интернет представляет собой не просто набор статической информации, а богатый контент с различным пользовательским взаимодействием, в котором уже с 2011 года

наметились тенденции полного сосредоточения государственной деятельности через сеть Интернет [65].

Однако при переходе от формата Web к Web 2.0, Т. Grandison [64] утверждает, что необходимо уделять особое внимание конфиденциальности и безопасности не только каждого пользователя, но и каждого источника, предоставляющего информацию, т.к. киберпреступность и кибератаки на сегодняшний день являются серьезными угрозами безопасности для любого интернет-ресурса. Основными способами таких кибератак является взлом, распространение вредоносных программ и DDoS-атаки. Описание механизмов таких видов кибератак, а также возможные пути их решения подробно рассмотрены в работах исследователей из Воронежского Государственного Технического Университета, Gachon University (Соннам, Корея), Institute for Risk and Insurance (Гамбург, Германия) и Institute of Entrepreneurial Studies and Innovation Management (Берлин, Германия) [66-70]. Использование принципа «Защита от DDoS как сервис», основанного на CDN, позволяет блокировать такие вредоносные запросы [71]. Другой вариант защиты заключается в совместном использовании двух методик – управления доступом на основе «белого» списка (для контроля доверенной загрузки) и определения атаки на основе «оживленного» периода [72]. Еще один вариант защиты заключается в комплексном подходе [73], который основан на применении трех механизмов: только зарегистрированный пользователь всегда имеет доступ к системе, проверка подлинности подключения пользователя, процесс трассировки, что помогает однозначно идентифицировать пользователя.

Одной из существующих проблем безопасного функционирования являются ошибки, допущенные в процессе создания веб-приложения. К сожалению, это происходит из-за того, что большинство разработчиков веб-приложений сосредотачиваются только над функциональностью приложения и пользовательским интерфейсом, оставляя вопросы безопасности «на последний план». Согласно проведенным исследованиям в 2013 году [74-76], понимание разработчиками защиты веб-приложений имеет довольно средний уровень. Как следствие этого, более 70% атак используют уязвимости уже на уровне приложения, а доля информации-

онных ресурсов, содержащих критические ошибки, составляет 45%. Самыми распространенными атаками на интернет-ресурсы являются межсайтовый скриптинг, SQL-инъекция, подделка маркеров (cookie poisoning) и принудительный переход по ссылке. Возможные алгоритмы использования злоумышленниками таких типов уязвимостей, а также эффективные способы борьбы с такими атаками рассмотрены в совместной работе исследователей из Institute of Engineering and Sciences (Индия), Institute of Computer Science (Индия) и University of Patras (Греция) [77]. Отчасти, такое стало возможным благодаря тому, что в последние годы большую популярность приобрели различные информационные системы управления контентом (CMS). Как и любое популярное программное обеспечение, системы управления контентом имеют проблемы в области безопасности [78]. Кроме этого, важно учитывать такой факт, как подтверждение подлинности содержимого веб-страниц. Такая проблема возникает каждый раз при обращении к содержимому веб-страницы, которое в большинстве случаев состоит не только из статических HTML, CSS, JavaScript, но и из динамических данных, полученных при обращении к базам данных или вычисленных в процессе выполнения [79].

Необходимо отметить, что SQL-инъекция используется злоумышленниками как базовая технология для проведения взлома [80], а значит, необходима проверка от возможного типа атак для выявления пользовательских входов, которые могут достичь целевой базы данных [81]. Один из вариантов определения SQL-атаки предполагает анализ размера запрашиваемых данных, сравнивая результаты запросов с ранее полученными входными данными с интернет-ресурса с текущим сконструированным запросом [82]. С другой стороны, необходимо стоять защиту от данного вида атак путем проверки динамически создаваемого запроса, который будет выполняться в базе данных сервера [83].

Другим популярным видом атаки является межсайтовый скриптинг (т.н. XSS-атака). Одним из возможных способов предотвращения такой атаки является методика, использующая совместно два метода для тестирования безопасности, а именно метод внесения неисправностей (проверка системы на отказоустойчивость) и метод испытания проникновением («процесс взлома») [84]. Однако, все

же для эффективного выявления такого рода атак необходимо уже на этапе проектирования веб-приложения использовать заранее заготовленный так называемый «код-шаблон» [85], основным назначением которого является «поглощение в себя» атаки и генерирование внутренней контролируемой ошибки, которую возможно «подавить». Кроме этого, XSS-атака также возможна в интернет-ресурсах с использованием технологии Flash [86].

Поэтому для более глубокого понимания уязвимостей межсайтового скриптинга и SQL-инъекций в 2014 году сотрудниками University of Coimbra (Португалия) [87] был представлен анализ исходных кодов скриптов, используемых злоумышленниками. По замыслу авторов, результаты этого исследования должны помочь при обучении разработчиков веб-приложений и продемонстрировать им основные применяемые опасные конструкции.

С другой стороны, проблемы безопасности в большинстве случаев начинаются еще глубже, не на самой интернет-площадки, а на сервере. Так, для выявления симптомов инъекции кода необходимо использовать из теории информации принцип дивергенции Кульбака-Лейблера (Kullback-Leibler distance, KLD), который во время совершения интернет-атаки отклоняется от ожидаемого значения [88]. Однако, так или иначе атака проводится неавторизованным пользователем, поэтому для гарантированного обмена информацией между пользователем и сервером пересылаемые cookie должны быть зашифрованы. Такой подход предлагает блокирование неавторизованных пользователей в системе и возможен только для использования в корпоративных сетях и сетях интернет-магазинов [89]. Еще одним из вариантов защиты веб-запросов, является использование не только HTTPS, но и так называемых доверенных промежуточных центров, которые имеют корневые сертификаты для проверки содержимого [90-91]. Особенно такая проблема актуальна при передаче конфиденциальной информации [92].

Принципиально другой подход к безопасности лежит в использовании механизма SME (Secure Multi-Execution, безопасное мульти-выполнение), который заключается в контролируемом выполнении веб-скриптов. Для проверки такой методики в Beihang University (Пекин, Китай) [93] был разработан веб-

обозреватель FlowFox. Экспериментальная оценка по международному рейтингу Alexa Top-500 Global Sites показывает, что FlowFox имеет на практике хорошую совместимость с текущим веб-форматом.

Однако возможное решение всех проблем надо искать не в самом веб-обозревателе, а в «доверенных расширениях» [94]. Именно такие решения по наращиванию возможностей самого веб-браузера могут нарушить его безопасность. Это становится возможным благодаря тому, что такие «плагины» имеют не только высокий приоритет выполнения, но и большое сходство с самими веб-приложениями.

Таким образом, учитывая описанные выше проблемы, оценка безопасности должна быть произведена еще на стадии создания информационного ресурса. Это связано с тем, что современные средства тестирования гарантированно не справляются с постоянно растущей сложностью веб-приложений [95], и поэтому представленная методика позволяет заблаговременно обнаружить вредоносные скрипты.

Обзор методик и мер по повышению уровня защиты информационных систем [96] показывает, что, учитывая принципиальное ограничение «стоимость реализации этих мер не должна превышать стоимости защищаемых информационных ресурсов», требования информационной безопасности должны быть направлены исключительно на обеспечение оптимального режима функционирования информационной системы в целом.

Для определения оценки защищенности информационного ресурса Т.А. Биячueвым [5] на основе ГОСТ Р ИСО/МЭК 15408 «Информационная технология. Методы и средства обеспечения безопасности. Критерий оценки безопасности информационных технологий» [97] предложено использовать следующий метод оценки, который включает:

- наличие механизмов идентификации и аутентификации пользователей для доступа к определенным частям интернет-ресурса;
- использование протоколов SSL и TLS для защиты протокола HTTP.

Разработанная Е.А. Проценко [6] методика построения модели угроз системам защиты информации сайтов органов власти на основе описанного выше метода предусматривает сбор и анализ следующих данных:

- формирование выборочной совокупности информационных ресурсов в рамках федерального округа на основе официального государственного информационного ресурса;

- информации об исследуемом информационном ресурсе с использованием общедоступной информации о сетевой инфраструктуре (трассировка маршрута к интернет-ресурсу; анализ информации, выдаваемой службой WhoIs; анализ программного обеспечения веб-сервера с использованием службы NetCraft и т.д.);

- информации, размещенной на самом информационном ресурсе (сведения, которые могут стать источником угроз: структура сайта, система ссылок, размещение файлов, информация о мероприятиях и т.д.);

- информации о доступности URL-адресов информационного ресурса до второго уровня вложенности включительно;

- информации о поддержке информационным ресурсом протокола SSL/TLS.

Однако, требования, предъявляемые законодательными и нормативно-методическими документами Российской Федерации, должны быть учтены и выполнены при разработке информационных ресурсов. Так, А.А. Павлютенковым [7] были разработаны модель и метод логического контроля использования национальных стандартов по информационной безопасности и расширения существующих классов информационной безопасности [98].

Более того, для получения гарантированной оценки защищенности информационных систем на основе накопленных баз данных по их уязвимостям и модели временных рядов разработана двухуровневая система критериев [8], которая, помимо использования активного и пассивного тестирования систем защиты, предлагает использовать метод аналитической оценки и прогнозирования общего уровня защищенности. Такой метод позволяет оценить уровень защиты отдельных элементов информационной системы. В отличие от других методик, пред-

ставленный подход исключает достаточно высокий уровень абстракции, который в каждом конкретном случае даёт большую свободу в интерпретации предписанных шагов алгоритма анализа и их результатов.

Для выявления общих проблем и повышения уровня защищаемого веб-сервера в 2011 году Tom Canavan, высококвалифицированный специалист компьютерного оборудования, в том числе имеющий опыт работы в области компьютерной безопасности и организации работы центров обработки данных, опубликовал книгу [16], основной особенностью которой является подробный анализ защиты сервера от внешних атак. Автор указывает на возможные недостатки и способы их устранения не только при выборе хостинг-провайдера или организации работы веб-сервера, но и при построении функционально правильной работы интернет-ресурса при использовании различных систем управления содержимым (CMS) с точки зрения компьютерной безопасности.

Для понимания механизмов функционирования безопасности клиент-серверных приложений, экспертами Dafydd Stuttard и Marcus Pinto [14], занимающиеся разработкой защищенных веб-приложений, проведено исследование различных типов атак, в котором затронуты не только вопросы уязвимости большинства веб-обозревателей (не конкретизируя достоинства и недостатки отдельного программного обеспечения), но и проблемы корректного реагирования веб-серверов на посылаемые пользователем запросы. Результатом исследования стала необходимость в проведении анализа исходного кода веб-приложений, использующих для работы технологии Java, PHP, Perl, JavaScript и ASP.NET. На основе полученных данных была разработана комплексная методика, которая включает в себя 13 пунктов [14, гл. 21]:

1. Исследование содержимого приложения;
2. Анализ приложения;
3. Тестирование управляющих элементов на стороне клиента;
4. Тестирование механизмов аутентификации;
5. Тестирование механизмов управления сеансом;
6. Тестирование элементов контроля доступом;

7. Тестирование уязвимостей, основанных на вводе;
8. Тестирование уязвимостей, основанных на функционально-специфическом вводе;
9. Тестирование логических недостатков;
10. Тестирование уязвимостей, направленных на использование общего хостинга;
11. Тестирование уязвимостей серверных приложений;
12. Смешанное тестирование (тестирование DOM-атак, локальной конфиденциальности, слабых мест SSL и т.д.);
13. Отслеживание любой утечки информации.

Несовершенство используемых сегодня веб-технологий заставляет рассматривать в процессе функционирования интернет-ресурсов вопросы компьютерной безопасности. Однако необходимо также обращать внимание на проблемы безопасности веб-технологий и со стороны пользователя. В своей работе Michal Zalewski, польский эксперт в области информационной безопасности, занимающийся вопросами защищенности веб-обозревателей, отмечает [99], что безопасность веб-технологий всегда будет предметом споров и обсуждений. Более того, нельзя полностью обеспечить абсолютную безопасность ввиду того, что невозможно устранить все существующие изъяны. Но, тем не менее, автор призывает находить оптимальный уровень защиты пользователя, т.е. как можно более точно описывать и предсказывать веб-уязвимости, с которыми можно столкнуться в текущий настоящий момент времени. Поэтому, необходимо продолжать, несмотря на сегодняшние проблемы в информационной безопасности, развивать и совершенствовать веб-технологии, опираясь одновременно не только на наработанные разработки, но и предлагать новые принципиальные возможности аудита в области обеспечения безопасности. Так, например, в недавно объявленном стандарте HTML 5, опубликованном в окончательной редакции 28 октября 2014 года, по сравнению со стандартом HTML 4.01, предусмотрены рекомендации по возможному поведению HTML-анализатора веб-обозревателя при обработке таких исключений как [4, п.п. 8.2.8]:

- перекрестное закрытие тегов, требующих обязательного открытого и закрытого тега;
- перекрестное закрытие тегов, не требующих обязательного закрывающего тега;
- неправильное использование тегов в таблицах;
- модификация веб-скриптами структуры интернет-страницы «как будто она была уже распознана»;
- выполнение скриптов, которые «движутся» через множество документов;
- незамкнутые элементы форматирования;

Описание таких ситуаций на начальном этапе дает возможность браузерам, во-первых, повысить надежность их функционирования, а, во-вторых, избежать некорректного отображения веб-страницы из-за неправильного разбора HTML.

Исходя из рассмотренных существующих в настоящее время методик для определения безопасного функционирования информационного ресурса в сети Интернет [5-8, 14], проведем качественную оценку обозначенных исследований.

Подход, предложенный Т.А. Биячуевым [5], гарантирует только, что пользователь имеет безопасное соединение с интернет-ресурсом, т.е. обеспечена защита от прослушивания сетевого соединения. Конечно, в таком случае злоумышленнику будет затруднен поиск паролей и личной информации, но, тем не менее, автором не рассматривается вопрос защиты механизма идентификации и аутентификации, на который может быть совершена атака даже при защите протокола HTTP. Однако даже в этом случае автор пытается оперировать требованиями, предъявляемыми стандартом ГОСТ Р ИСО/МЭК 15408 [97] для информационных ресурсов.

Использование данного метода не позволяет гарантировать, что используемые технологии помогут в полной мере обезопасить информационный ресурс. Таким образом, владельцу интернет-ресурса придется полагаться на добросовестность пользователя.

В исследованиях, проведенных Е.А. Проценко [6], автор не конкретизирует, как именно должны учитываться результаты первых двух пунктов при опре-

делении оценки функциональной корректности информационного ресурса, а именно - нет четкого алгоритма анализирования публичной информации. Более того, информация, размещенная для публичного доступа, обычно является общедоступной информацией для предоставления интернет-пользователям. Таким образом, данный пункт актуален в случае поиска конфиденциальной информации, которая в большинстве случаев защищена механизмами идентификации пользователя. Дополнительно, автор не обосновывает необходимость проведения анализа доступности первых двух вложенных уровней URL-адресов информационного ресурса. Такая вложенность для крупных интернет-ресурсов может быть достаточно глубокой ввиду структурирования размещаемой информации.

Учитывая отмеченные выше недостатки этого метода, получить достоверную оценку функциональной корректности информационного ресурса не представляется возможным.

Кроме этого, предложенный метод никак не рассматривает наличие ранее рассмотренного стандарта ГОСТ Р ИСО/МЭК 15408 [97], а лишь использует наработки, полученные в ранее проведенных исследованиях. В то время как, дополнительно, при предъявлении требований к безопасности функционирования информационного ресурса необходимо также учитывать из данного стандарта следующие классы:

- «Класс FAU. Аудит безопасности». Поддержание актуального состояния современного информационного ресурса подразумевает многопользовательский режим доступа, поэтому для безопасного функционирования интернет-ресурса (при попытках доступа, внесения изменений и т.д.) необходимо протоколирование действий пользователей.

- «Класс FRU. Использование ресурсов». Информационный ресурс, как средство для предоставления достоверной официальной информации общего пользования в сети Интернет, в текущий момент должен иметь довольно большой запас прочности для обработки всех запросов пользователей. Степень проработки данного вопроса напрямую зависит от степени значимости интернет-ресурса для его владельца, а также актуальности его существования.

- «Класс FCO. Связь». Помимо предоставления информации в сети Интернет, организованный информационный канал должен иметь достаточную пропускную способность для обеспечения устойчивой связи «пользователь»-«информационный ресурс». Это необходимо, в первую очередь, для предотвращения получения достоверной информации без использования третьих лиц (посредников), а, во-вторых, предотвращения получения официального подтверждения подаваемой информации из других источников владельца (официальное заявление, изданный протокол, заключение и т.д).

- «Класс FDP. Защита данных пользователя». Статья 18, 20 и 21 [57] регламентирует перед операторами обязанности при сборе, обработке и хранении персональных данных. Дополнительно, статья 18.1 и 19 [57] регламентирует меры, которые должны быть приняты оператором для предотвращения неправомерных действий с персональными данными. Таким образом, информационным ресурсам, в общем случае выступающим не только как средство для предоставления информации, но и как средство для оказания различного вида электронных услуг, необходимо учитывать данный нормативный документ.

- «Класс FMT. Управление безопасностью». Для гибкой настройки политики безопасности администратором информационного ресурса системой управления содержимым должна быть предусмотрена возможность «тонкой» настройки. Таким образом, переосмысление и внесение изменений в текущее состояние не должно вызывать особых трудностей.

Кроме этого, в 2008 г. был разработан ГОСТ Р ИСО/МЭК 18045 «Информационная технология. Методы и средства обеспечения безопасности. Методология оценки безопасности информационных технологий» [100], более известный как «Общая методология оценки». Данный стандарт является расширением стандарта ГОСТ Р ИСО/МЭК 15408-3 «Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Компоненты доверия к безопасности» [101], который лишь описывает требования для вынесения общего заключения, в то время как ГОСТ Р ИСО/МЭК 18045 [100] устанавливает процесс действий для проведения комплек-

сов мероприятий. Для заключения оценки безопасности информационных технологий необходимо произвести анализ следующих классов:

- «Класс APE. Оценка профиля защиты»;
- «Класс ASE. Оценка задания по безопасности»;
- «Класс ADV. Разработка»;
- «Класс AGD. Руководства»;
- «Класс ALC. Поддержка жизненного цикла»;
- «Класс ATE. Тестирование»;
- «Класс AVA. Оценка уязвимостей»;
- «Класс ACO. Композиция».

Таким образом, выполнение всех данных «пунктов» подразумевает, практически, проведение на соответствие наличия заявленных функций в разработанном руководстве и поиск «отклонения от нормального состояния», которые формируют класс недокументированных возможностей, а значит, представляют потенциальную угрозу. Такой подход возможен лишь при анализе программного обеспечения еще на этапе проектирования или активной разработки. Кроме этого, применяя данную методологию к информационным ресурсам, необходимо быть уверенным в безопасности используемых веб-технологий. Более того, информационный ресурс в общем понимании обычно представляет довольно сложный объект, функциональными компонентами которого являются не только программная, но и аппаратная составляющая. Стоит также отметить, что дополнительный фактор по трудозатратам накладывает общедоступность информационного ресурса, т.к. программное обеспечение может быть изъято из производства, доработано и выпущено заново на рынок уже по прошествии времени, в то время как при разработке или доработке информационного ресурса его функционирование не может быть приостановлено полностью. В результате, предъявляемые требования, скорее всего, применимы к интернет-ресурсам лишь только на этапе разработки или перед непосредственным запуском его в производство.

Одним из результатов исследований, выполненных А.А. Павлютенковым [7], является разработанный метод, который позволяет на основе открытой

информации определить использование на информационном ресурсе средств защиты. Данная концепция позволяет учесть разнородные общие требования безопасности существующих нормативных документов Российской Федерации, которым должен соответствовать объект без учета разработанных зарубежных методов и методик, а также ранее отмеченных недостатков работ [5-6].

В процессе исследования М.С. Политов [8] отмечает, что разработанный сканер при анализе удаленной машины не должен учитывать тип используемой операционной системы (т.е. должен быть универсальным), однако разработанное программное обеспечение в рамках диссертационной работы функционирует исключительно под ОС Windows, а, следовательно, не является кроссплатформенным. Также автор большое внимание уделяет лишь сканированию портов на информационной системе, но, тем не менее, в рамках программного обеспечения предлагает проводить анализ веб-контента без описания ключевых особенностей проведения анализа. Предложенная двухэтапная концепция помогает сделать более осмысленным подход к оценке защищенности интернет-ресурсов на основе накопленной информации об известных проблемах в используемых технологиях, однако не дает возможности найти недекларируемые возможности применяемых технологий.

Разработанная по результатам совместной работы Dafydd Stuttard и Marcus Pinto методика [14, гл. 21], возможно, помогает выявить большинство уязвимостей. Однако при проведении анализа таких технологий как HTML, CSS и JavaScript авторы предлагают только:

- исследовать потенциально-уязвимые конструкции JavaScript;
- исследовать описание объектов тегов HTML.

Данная методика не предполагает исследование корректного использования стилевых параметров и значений CSS, заданных значений атрибутов HTML, структуры DOM, а также синтаксического разбора JavaScript. Таким образом, анализ безопасного использования этих веб-технологий не производится в полной мере.

1.4. Обзор возможностей существующих программных средств

В зависимости от классов контролируемых уязвимостей существуют несколько типов анализаторов для поиска уязвимостей в программном коде. Одним из типов являются специализированные анализаторы, предназначенные для нахождения уязвимостей в программном коде веб-приложений. Рассмотрим возможности каждого из представленных на компьютерном рынке анализаторов на предмет проведения достоверного анализа исходных текстов информационного ресурса.

Первой группой из представленных на рынке анализаторов являются программные продукты крупных компаний, занимающихся не только производством программного обеспечения, но и аналитическими исследованиями в области информационной безопасности. Так специалисты подразделения российского исследовательского центра Positive Research на основе разработанных программных продуктов занимаются мониторингом и анализом уязвимостей веб-приложений [102]. Американская (Acunetix) и британская (Netsparker) компании имеют персональные информационные блоги с собственным аналитическим взглядом на безопасное использование веб-приложений и веб-технологий [103-104]. Также, к данной группе можно отнести следующие веб-анализаторы: IBM Rational AppScan [26], Acunetix Web Vulnerability Scanner [27], XSpider [105], NTOSpider [28], NetSparker [29] и HP WebInspect [30]. Остальные анализаторы подобного рода – либо имеют существенно худшие характеристики для проведения анализа, либо обладают меньшими функциональными возможностями.

Рассмотрим особенности указанных программных продуктов. В табл. 1, согласно обзору [106-107], проведенному аналитиком и экспертом в области информационной безопасности Shay Chen, приведены сводные характеристики веб-анализаторов.

Таблица 1. Сводная характеристика веб-сканеров

Наименование характеристики	Веб-сканеры					
	HP WebInspect	IBM Rational AppScan	NetSparker	Acunetix WVS	NTOSpider	N-Stalker
Происхождение	USA	USA	GBR	USA	USA	BRA
Технологии	.Net	.Net	.Net	? (Win32)	Java	? (Win32)
WIVET	96%	94%	92%	92%	94%	94%
WAVSEP 2014	74% ¹ /0% ²	74%/13%	73%/5%	64%/0%	65%/2%	54%/2%
Издания	SE ³ , EE	SE, EE, Src ⁴	SE, PE, CE	SE, FE	SE	EE, IE, FE
Cross-platform	–	+ (Src)	–	–	–	–
Анализ JavaScript	+	+	+	+	+	+
Анализ HTML	+	+	+	+	+	+

Значение теста WIVET (Web Input Vector Extractor Teaser) [108], разработанного Bedirhan Urgan, показывает, насколько разработанный веб-сканер может распознать типичные интернет-уязвимости. Однако данный тест не отражает насколько «глубоко» веб-анализатор может провести анализ интернет-ресурса. Для проведения такого рода испытаний Shay Chen разработал комплексный тест WAVSEP (Web Application Vulnerability Scanner Evaluation Project) [109], состоящий из 6 подгрупп тестов, суммарно включающих в себя 1126 тестов для определения явных и скрытых уязвимостей и 31 тест для определения ложных уязвимостей, а также тесты для анализа переадресованных ссылок и скрытых файлов. В результате, данный тест позволяет охватить наиболее широкий спектр существующих веб-уязвимостей.

Таким образом, если по значению теста WIVET рассматриваемые анализаторы имеют примерно равные показатели, то тест WAVSEP уже показывает более существенные различия в качестве проведения анализа данными программными

¹ Процент от общего числа пройденных тестов при определении различного типа веб-уязвимостей

² Процент ошибочных срабатываний при определении различного типа веб-уязвимостей

³ SE – Standard Edition, EE – Enterprise Edition, PE – Professional Edition, CE – Community Edition, FE – Free Edition, IE – Infrastructure Edition

⁴ Издание для анализа исходных текстов, а не предоставления исходных текстов программного обеспечения

продуктами. Однако уже на данном этапе важно отметить, что ни один из представленных веб-сканеров не прошел полностью даже тест WIVET, и это притом, что рассматриваются коммерческие анализаторы, которые традиционно являются более продвинутыми продуктами, по сравнению с некоммерческими аналогами. Это говорит о том, что в настоящий момент существующими средствами невозможно полностью выявить даже все типичные интернет-уязвимости.

Из отмеченных программных продуктов для проведения анализа информационного ресурса лучшими (по сравнению с остальными) можно выделить только HP WebInspect, IBM Rational AppScan и NetSparker. HP WebInspect, в отличие от IBM Rational AppScan, имеет минимальный процент ложных срабатываний, однако не является кроссплатформенным программным обеспечением, а NetSparker – минимальное значение теста WIVET. Остальные же из рассматриваемых программных продуктов в основном «проваливаются» при глубоком анализе возможностей веб-сканера теста WAVSEP.

Вне данного обзора остался такой анализатор как XSpider, который является разработкой российского исследовательского центра Positive Research. Данный веб-сканер является, в первую очередь, одним из программных обеспечений, ориентированных именно на российский рынок, т.к. нередко зарубежные стандарты информационной безопасности не соответствуют национальным стандартам в области защиты информации. Помимо XSpider, Positive Research имеет такой программный продукт как Application Inspector [31], позволяющий проводить статический и динамический анализ исходного кода таких языков программирования, как JavaScript, HTML и PHP. Существенным недостатком данного программного обеспечения является отсутствие поддержки кроссплатформенности.

Второй группой представленных на рынке анализаторов являются некоммерческие программные продукты с открытым исходным кодом. В данную категорию следует выделить такие программные обеспечения как W3AF [110], arachni [111], SkipFish [32] и Wapiti [112]. Учитывая результаты проведенного обзора [106-107], табл. 2 представляет сводную характеристику таких веб-сканеров.

Таблица 2. Сводная характеристика некоммерческих веб-сканеров

Наименование характеристики	Веб-сканеры			
	W3AF	arachni	SkipFish	Wapiti
Технологии	Python	Ruby	C	Python
WIVET	19%	19%	48%	44%
WAVSEP 2014	39%/12%	69%/3%	53%/24%	47%/46%
Cross-platform	± ⁵	± ⁶	+ ⁷	?
Анализ JavaScript	+	–	–	–
Анализ HTML	+	+	+	+

Из рассматриваемых некоммерческих анализаторов, отдаленно-близким аналогом по функциональности коммерческого анализатора можно назвать веб-сканер Skipfish ввиду того, что он имеет средние значения как по тесту WIVET, так и по тесту WAVSEP. Это говорит о том, что данный сканер «покрывает» около половины известных уязвимостей. Отсюда следует, что SkipFish способен решить большинство задач информационной безопасности интернет-ресурса, однако все равно остается необходимость в дополнительном исследовании аудита информационного ресурса. Также данный сканер практически полностью является кроссплатформенным программным обеспечением. Единственным существенным недостатком служит довольно большое число ошибочных срабатываний.

Похожими характеристиками с Skipfish обладает веб-сканер Wapiti. Однако имеет более низкие показатели по результатам тестов WIVET и WAVSEP, в том числе показатель ложных срабатываний составляет около половины, что является довольно высоким результатом.

Стоит также отметить веб-анализатор arachni благодаря тому, что тест WAVSEP смог найти тот скрытый «потенциал», который не мог обнаружить тест WIVET. Это показывает то, что веб-сканер arachni обладает специфическими возможностями для решения узкого класса задач. Также тест WAVSEP показал, что данный сканер имеет минимальное число ошибочных срабатываний. Из недостат-

⁵ Установка для ОС Windows возможна, но не рекомендуется производителем

⁶ Установка возможна только для ОС Linux/MacOS

⁷ Установка для ОС Windows рекомендуется с помощью Cygwin [22]

ков данного программного обеспечения стоит выделить тот факт, что оно не является полностью кроссплатформенным.

Оставшийся веб-сканер W3AF по сравнению с другими рассматриваемыми веб-анализаторами имеет довольно низкие показатели и является близким по функциональности с характеристиками arachni: значения тестов WIVET совпадают, а результат ложных срабатываний составляет больше чем у arachni, но существенно меньше, чем у Skipfish и Wapiti.

Остальные веб-сканеры, не попавшие в данную группу для обзора некоммерческих анализаторов, представляют собой узкоспециализированные средства для решения конкретных задач, а значит результаты вышеуказанных тестов, а также их функциональные характеристики значительно ниже представленных программных продуктов. Так, например, sqlmap [113] является средством для автоматического определения SQL-инъекций, которое по результатам теста WAVSEP имеет 100% нахождение и нулевую погрешность при определении такого типа уязвимостей.

В последнюю – **третью группу**, можно выделить сканеры, которые являются перехватывающими прокси-серверами. Примерами таких программных продуктов являются ZAP (Zed Attack Proxy) [114], Burp Suite [115-116], Paros Proxy [117] и IronWASP [118]. Среди данного типа веб-сканеров присутствуют как коммерческие, так и некоммерческие решения. Данная группа, в отличие от вышерассмотренных групп, представляет собой общее сравнение такого типа анализаторов вне зависимости от финансирования разработок, которая также обуславливается немногочисленностью данного типа сканеров на рынке.

Табл. 3 приводит сводные характеристики обозначенных анализаторов, учитывая исследования [106-107], проведенные Shay Chen.

Таблица 3. Сводные характеристики веб-сканеров, являющихся перехватывающими прокси-серверами

Наименование характеристики	Веб-сканеры			
	ZAP	Burp Suite	Paros Proxy	IronWASP
Технологии	Java	Java	Java	.Net
Лицензия	ASF2	Commercial	CAL	GPL3
WIVET	73%	16%	10%	н/д
WAVSEP 2014	72%/13%	63%/0%	17%/69%	67%/19%
Cross-platform	+	+	+	± ⁸
Анализ JavaScript	+	+	–	–
Анализ HTML	+	+	+	+

Как видно из результатов тестов WIVET и WAVSEP, коммерческие веб-сканеры имеют показатели существенно ниже, чем некоммерческие сканеры. Это обусловлено тем, что коммерческие сканеры обладают такими дополнительными узкоспециализированными возможностями как поиск уязвимостей в flash-приложениях и анализ веб-сервисов, которые не учитываются тестам WIVET и WAVSEP. Стоит также отметить, что рассматриваемый сканер Paros Proxy признан многими экспертами довольно устаревшим ввиду поддержки со стороны производителя только обновлениями баз уязвимостей, но не учитывающий доработку/модификацию и дальнейшую разработку существующего программного обеспечения. Таким образом, Paros Proxy, по сравнению с другими сканерами, имеет довольно низкие показатели и высокую вероятность ошибочных срабатываний.

По результатам тестирования, разработанный анализатор Burp Suite смог выявить лишь 16% типичных уязвимостей и 63% уязвимостей при глубоком исследовании. Данный факт показывает, что приложение, в большей степени, способно решать только узкоспециализированные задачи, несмотря на довольно глубоко проработанную теоретическую основу [14].

⁸ Установка для ОС MacOS рекомендуется с использованием CrossOver [23], ОС Linux – Wine [24]

Среди рассматриваемых сканеров данной группы наилучшие характеристики имеет сканер ZAP. В сравнении со сканерами второй группы (W3AF, arachni, SkipFish и Wapiti), ZAP имеет существенно лучшие показатели, обладая хорошими результатами тестов WIVET и WAVSEP и возможностью кроссплатформенности. Однако имеющий более низкие возможности по сравнению с коммерческими сканерами.

Однако стоит также отметить, что сканеры данной группы являются только программными комплексами. Так, компания Blue Coat Systems предлагает комплекс Blue Coat [119], содержащий не только программную, но и аппаратную составляющую. Данный комплекс не участвовал в исследованиях, проводимых Shay Chen, ввиду изучения в первую очередь программных средств, но может существенно повысить безопасность, поскольку анализируемый трафик (в том числе трафик локальной или корпоративной сети) будет контролироваться дополнительным устройством. Подход, предложенный Blue Coat Systems, помогает осуществлять анализ действий пользователя и выявлять аномально-опасные ситуации, однако данный комплекс мероприятий больше повышает безопасность в области административной составляющей.

Помимо этого, **дополнительным способом** выявления и устранения угроз информационного ресурса является комплексный анализ исходного кода интернет-страниц средствами веб-браузера. Для этих целей разработчиками W3C (являющимися также разработчиками стандарта HTML [4] и CSS [52]) созданы сервисы:

- W3C Validator для проверки исходного текста веб-страниц [62];
- W3C Validator Suite для комплексной проверки веб-ресурса [33];
- W3C Link Checker для проверки ссылок на веб-странице [120].

Существенным недостатком первых двух сервисов является отсутствие поддержки JavaScript, что не позволяет произвести полный анализ интернет-ресурса, а последний сервис осуществляет сбор ссылок только с указанной интернет-страницы.

Другим средством для проведения анализа интернет-страниц является Firebug [121], который может быть установлен как расширение для браузера («плагином»). Firebug позволяет проводить комплексный анализ «на ходу» и поддерживает работу с HTML, CSS и JavaScript [53]. Существенным недостатком данного «плагина» является отсутствие автоматической проверки всего интернет-ресурса, которые может занимать ни одну интернет-страницу. Аналогичный функционал присутствует в веб-обозревателе Internet Explorer [122], который является встроенным средством разработчика и служит для отладки, тестирования и поиска причин утечки памяти.

Однако, многие интернет-ресурсы, в большинстве случаев использующие одну из популярнейших систем управления содержимым, часто имеют один «шаблон» для всех интернет-страниц и таким образом большинство ошибок от «страницы» к «странице» будут повторяться. С одной стороны, это верно ввиду того, что многие организации, в том числе государственные структуры, делают свой информационный портал как можно проще для использования пользователями и стараются не загромождать лишней информацией. С другой стороны, если интернет-ресурс создается как многофункциональный, на котором будет содержаться множество различных сервисов и храниться информация различного типа/рода, такое утверждение будет заведомо неверным, потому как подавляющее большинство интернет-страниц будут иметь собственный «шаблон», а система управления содержимым в большинстве случаев будет «заточена» под сам интернет-ресурс. Таким образом, для получения «общей картины» ошибок на всем интернет-ресурсе так или иначе необходим анализ всех интернет-страниц. Конечно, можно найти ПО, позволяющее обойти весь интернет-ресурс по ссылкам, однако это будет проверка работоспособности всех ссылок данного интернет-ресурса.

Учитывая тот факт, что не все представленные и реализованные характеристики программного обеспечения могут быть одинаково полезными и пригодными при проведении исследований, введем дополнительно так называемый «весовой коэффициент», который в значении от 0 до 1 будет отображать степень значимости характеристики для проведения исследований. Это значит, что численная

оценка реализованных возможностей сравниваемых программных обеспечений вычислима по формуле:

$$h_j^k = \sum_{i=1}^N a_{ij} p_i^k,$$

где k – порядковый частный случай исследования информационного ресурса, N – суммарное число характеристик, $a_{ij} = \{0, 0.5, 1\}$ – степень реализации i -ой характеристики для j -го программного обеспечения, $p_i^k \in [0, 1]$ – степень значимости характеристики для k -го случая исследования информационного ресурса, h_j^k – суммарное численное значение реализованных характеристик для j -го программного обеспечения для k -го случая исследования информационного ресурса.

В итоге, исходя из описанных ранее функциональных возможностей программных обеспечений и поставленных целей и задач диссертационной работы, охарактеризуем частные случаи исследований информационных ресурсов, для которых имеются готовые программные решения:

1. Исследование многофункционального интернет-ресурса, направленное на выявление недостатков безопасного функционирования используемой информационным ресурсом системы управления содержимым, определения возможных классов атак на содержащий информационный ресурс хостинг или веб-сервер;
2. Исследование интернет-ресурса, состоящего из одной и более веб-страниц, содержащего статический или динамический контент и являющегося поддельным («фишинговым»), атакующим или распространяющим вредоносные программы;
3. Исследование функциональной корректности интернет-ресурса, состоящего из одной и более веб-страниц, содержащего статический и динамический контент с открытым, частично открытым или закрытым доступом, рассчитанным на широкую тематическую аудиторию и являющимся корпоративным ресурсом, ресурсом организации или государственным ресурсом общего пользования.

Для проведения корректного сравнения представленных программных обеспечений и выбора наилучшего (приемлемого) решения для каждого случая определим множество групп измеримых характеристик [123], которое будет точно отображать параметры исследуемой задачи и возможности сравниваемых программных обеспечений. Каждая группа характеристик описывается несколькими величинами точное представление о группе. Такое множество групп характеристик может состоять из следующих элементов:

1. Поддержка различных протоколов (6⁹).
2. Используемые методы аутентификации (4).
3. Методы управления сетевым сеансом (6).
4. Параметры сканирования структуры информационного ресурса (15).
5. Исследование элементов структуры информационного ресурса (13).
6. Методика тестирования информационного ресурса (42).
7. Функциональные возможности анализатора (15).
8. Предоставляемая отчетность (2).

Сопоставим каждой величине группы характеристик степень её реализации в программном обеспечении и выделим три состояния:

«—» или «Н/Д» – данная величина группы характеристики отсутствует или не реализована (0% или 0);

«±» – данная величина группы характеристики реализована частично (50% или 0.5);

«+» – данная величина группы характеристики реализована полностью (100% или 1).

Таким образом, табл. 4 отображает сводные результаты проведенных расчетов. По этим результатам можно сделать следующие выводы:

1. Для решения задач первой группы проводимых исследований наиболее приемлемым программным средством является выбор Rational AppScan (IBM) [26]. Данный веб-сканер наиболее полно удовлетворяет предъявляемым требова-

⁹Общее число характеристик в группе

ниям и позволяет провести анализ безопасного функционирования информационных ресурсов.

2. Для решения второй группы проводимых исследований наиболее приемлемыми программными средствами являются Google Search Console [124] и Яндекс.Вебмастер [125]. Такой выбор объясняется как обязательным присутствием ключевой характеристики 6.15 (табл. 4), которая показывает направленность анализа, так и тесного взаимодействия встроенных защитных средств веб-обозревателя с действиями пользователя [43-45].

3. Наиболее близкими решениями к возможностям программных продуктов по исследованию функциональной корректности информационных ресурсов третьей группы являются SkipFish (Google) [32], Rational AppScan (IBM) [26], Application Inspector (PT) [31], Burp Suite (PortSwigger Ltd) [115], Validator Suite (W3C) [33] и АИС «Мониторинг государственных сайтов» (МЭР РФ) [3]. Такой выбор объясняется как необходимостью проведения полного анализа всей структуры интернет-ресурса, так и поиском наиболее сбалансированного решения для данного класса задач. По полученным результатам (табл. 4) можно сделать вывод, что разрабатываемый в диссертационной работе анализатор исходных текстов информационного ресурса обладает рядом характеристик, которые не учтены в иных программных обеспечениях и исследованиях.

Таблица 4. Сравнение возможностей программных обеспечений относительно исследуемых характеристик поставленной задачи

Характеристика	Весовой коэффициент			Программные продукты											
	p^1	p^2	p^3	Акула (НИВЦ МГУ)	SkipFish (Google)	Rational AppScan (IBM)	Application Inspector (PT)	Burp Suite (PortSwigger Ltd)	Validator Suite (W3C)	АИС «Мониторинг гос. сайтов»	Firebug	Google Search Console	Яндекс-Вебмастер	Validator (W3C)	Link Checker (W3C)
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1. Поддержка различных протоколов															
1.1. Транспортная поддержка															
a. HTTP	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. SSL/TLS	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
c. HTTP User Agent Configuration	0.5	0.5	0.5	+	+	+	+	+	-	-	+	-	±	-	-
1.2. Поддержка прокси															
a. HTTP proxy	0	0	0	+	+	+	+	+	-	-	+	Н/Д	Н/Д	-	-
b. Socks proxy	0	0	0	+	+	+	+	+	-	-	+	Н/Д	Н/Д	-	-
c. PAC File Support	0	0	0	-	Н/Д	-	Н/Д	+	-	-	+	Н/Д	Н/Д	-	-
2. Используемые методы аутентификации															
2.1. Схемы аутентификации															
a. HTTP Negotiate (NTLM/Kerberos)	0.5	0.5	0.5	-	+	+	Н/Д	+	-	-	+	-	-	-	-
b. HTML Form-based	0.5	0.5	0.5	+	+	+	Н/Д	+	-	-	+	-	-	-	-
c. Single Sign On	0.5	0.5	0.5	-	+	+	Н/Д	+	-	-	+	-	-	-	-
d. Client SSL Certificates	0.5	0.5	0.5	+	+	+	Н/Д	+	-	-	+	-	-	-	-
3. Методы управления сетевым сеансом															
3.1. Возможности управления сеансом															
a. Начало нового сеанса	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
b. Обновление сеансового идентификатора	0.5	0.5	0.5	+	+	+	+	+	-	-	-	-	-	-	-
c. Определение недействительной сессии	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
3.2. Поддержка типа идентификатора управления сеансом															
a. HTTP Cookies	0.5	0.5	0.5	+	+	+	+	+	-	-	+	-	-	-	±
b. Параметры запроса HTTP	0.5	0.5	0.5	+	+	+	+	+	-	-	+	-	±	-	-
c. HTTP URL Path	0	0	0	+	+	+	+	+	-	-	+	-	-	-	-
4. Параметры сканирования структуры информационного ресурса															
4.1. Конфигурация модуля сканирования интернет-ресурса															
a. Определение заглавной страницы	1	1	1	+	+	+	+	+	+	+	-	±	±	-	-
b. Определение дополнительных имен хостов	1	1	1	+	+	+	+	+	+	+	+	±	Н/Д	-	-
c. Определение исключений	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
d. Определение предела избыточности запросов	0.5	0.5	0.5	+	+	+	+	+	+	+	+	-	-	±	±
e. Поддержка одновременных сессий	1	1	1	+	±	±	±	-	-	+	±	±	±	±	±
f. Определение задержки запроса	0	0	0	+	+	+	+	+	+	+	+	-	-	-	-
g. Поддержка максимальной глубины сканирования	1	1	1	+	+	-	Н/Д	-	+	-	-	±	±	-	-
h. Ручное сканирование	0.5	0.5	0.5	+	+	+	+	+	+	+	+	±	±	+	+
4.2. Функциональность модуля сканирования интернет-ресурса															
a. Определение внешних ссылок	1	1	1	+	+	+	+	+	+	-	±	-	-	-	±
b. Поддержка автоматической отправки форм	0.5	0.5	0.5	-	-	-	-	-	-	-	±	-	-	-	-
c. Определение ошибочных ссылок	1	1	1	+	+	+	+	+	+	-	±	+	+	-	+

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
d. Определение перенаправляемых ссылок	1	1	1	+	+	+	+	+	+	-	±	+	+	-	±
e. Выявление кольцевых ссылок	1	1	1	+	Н/Д	Н/Д	Н/Д	-	Н/Д	-	-	-	-	-	-
f. Определение и поддержка Cookie	0	0	0	+	+	+	+	-	-	-	±	-	+	-	-
g. Поддержка AJAX-приложений	0.5	0.5	0.5	-	-	+	-	-	-	-	+	-	-	-	-
5. Исследование элементов структуры информационного ресурса															
5.1. Тип исследуемого контента															
a. HTML	1	1	1	+	+	+	+	+	+	+	+	+	+	+	±
b. JavaScript	1	1	1	+	+	+	+	+	-	-	+	+	±	-	-
c. VBScript	1	1	1	-	-	-	-	+	-	-	-	-	-	-	-
d. XML	1	1	1	+	+	+	+	-	-	-	-	-	±	-	-
e. RSS/Atom	1	1	1	+	-	-	-	-	-	-	-	-	-	-	-
f. Plaintext	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-
g. ActiveX Objects	1	1	1	+	+	+	-	-	-	-	±	Н/Д	Н/Д	-	-
h. Java Applets	1	1	1	+	+	+	+	-	-	-	±	Н/Д	Н/Д	-	-
i. CSS	1	1	1	+	+	+	-	±	+	-	+	±	Н/Д	-	-
5.2. Поддержка различных кодировок	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
5.3. Устойчивость анализа к ошибкам на веб-странице	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
5.4. Тонкая настройка анализатора	0	0	0	+	+	+	+	+	+	Н/Д	±	Н/Д	Н/Д	±	±
5.5. Анализ динамического контента	1	1	1	+	+	+	+	+	-	-	±	±	±	-	-
6. Методика тестирования информационного ресурса															
6.1. Задание тестовой конфигурации															
a. Имя хоста (или IP)	1	1	1	+	+	+	+	+	+	+	±	+	+	+	+
b. Список дополнительно проверяемых ссылок	0.5	0.5	0.5	+	+	+	+	+	-	-	±	-	-	-	-
c. Файловые расширения	1	1	1	+	+	+	+	+	-	-	±	-	-	-	-
d. Параметры	0.5	0.5	0.5	+	+	+	+	+	+	+	±	-	-	+	+

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
e. Используемый HTTP-Заголовок	0.5	0.5	0.5	+	+	+	+	+	-	-	+	-	±	-	±
f. Cookies	0.5	0.5	0.5	+	+	+	+	+	-	-	+	-	-	-	±
6.2. Тип исследуемого ресурса															
I. По доступности															
a. открытый (общедоступный)	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. частично открытый (необходима регистрация)	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
c. закрытый (локальный)	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
II. По функциональности															
a. статический	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. динамический	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
c. флеш-сайт	0.5	0.5	0.5	-	+	+	-	±	-	-	+	+	+	+	+
III. По размеру аудитории															
a. Тематический	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. Интернет-портал	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
6.3. Вид исследуемого ресурса															
I. По виду представительства организации															
a. «Сайт-визитка»	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. Корпоративный (сайт организации)	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
II. По используемым веб-сервисам															
a. Новостной портал	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
b. Форум	1	1	1	±	+	+	+	+	+	+	+	+	+	+	+
c. Блог	1	1	1	+	+	+	+	+	+	+	+	+	+	+	+
6.4. Синтаксический анализ исходных текстов	1	1	1	+	+	+	+	+	+	±	+	+	±	+	±
6.5. Динамический анализ исходных текстов	1	1	1	+	+	+	+	+	-	-	+	-	-	-	-
6.6. Оценка качества кода	1	1	1	+	+	+	+	+	+	±	-	+	±	±	-
6.7. Исследование полноты ресурса	1	1	1	+	+	-	-	-	+	-	-	±	±	-	-

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
6.8. Анализ избыточности кода	1	1	1	+	-	-	-	-	-	-	-	±	±	-	-
6.9. Анализ корректности структуры интернет-ресурса (анализ DOM)	1	1	1	+	+	+	+	+	+	±	±	±	±	±	-
6.10. Проверка исходного текста на соответствие диалекту веб-обозревателей	1	1	1	+	-	-	-	-	-	-	-	±	-	±	-
6.11. Анализ методов авторизации															
a. Небезопасный идентификатор сессии (Weak Session Identifier)	1	0	0	-	-	+	Н/Д	+	-	-	-	-	-	-	-
b. Фиксация сессии (Session Fixation)	1	0	0	-	-	+	Н/Д	-	-	-	-	-	-	-	-
c. Обход аутентификации (Authentication Bypass)	1	0	0	-	-	+	Н/Д	-	-	-	-	-	-	-	-
6.12. Анализ атак на клиентов (Client-side Attacks)															
a. Межсайтовое выполнение сценариев (Cross-site Scripting, XSS)	1	0	0	-	+	+	+	-	-	-	±	-	-	-	-
b. Расщепление HTTP-запроса (HTTP Response Splitting)	1	0	0	-	-	+	Н/Д	+	-	-	±	-	-	-	-
6.13. Определение возможности выполнения кода (Command Execution)															
a. Переполнение буфера (Buffer Overflow)	1	0	0	-	-	+	Н/Д	-	-	-	-	-	-	-	-
b. Атака на функции форматирования строк (Format String Attack)	1	0	0	-	+	+	Н/Д	-	-	-	-	-	-	-	-
c. Внедрение операторов LDAP (LDAP Injection).	1	0	0	-	-	+	Н/Д	+	-	-	-	-	-	-	-
d. Выполнение команд ОС	1	0	0	-	+	+	Н/Д	+	-	-	-	-	-	-	-

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
(OS Commanding).															
e. Внедрение операторов SQL (SQL Injection)	1	0	0	-	+	+	+	+	-	-	-	-	-	-	-
f. Внедрение серверных расширений (SSI Injection)	1	0	0	-	-	+	Н/Д	-	-	-	-	-	-	-	-
g. Внедрение операторов XPath (XPath Injection)	1	0	0	-	+	+	Н/Д	+	-	-	-	-	-	-	-
6.14. Определение возможности логических атак															
a. Отказ в обслуживании (Denial of Service)	1	0	0	-	-	+	Н/Д	+	-	-	-	-	-	-	-
b. Неверное предоставление привилегий (privilege escalation)	1	0	0	-	-	+	Н/Д	+	-	-	-	-	-	-	-
c. Обратный путь в директориях (Path Traversal)	1	0	0	-	+	+	Н/Д	+	-	-	-	-	-	-	-
6.15. Определение заражения вредоносным ПО и фишинговых атак	0	1	0	-	-	-	-	-	-	-	-	+	+	-	-
7. Функциональные возможности анализатора															
7.1. Возможности анализа															
a. Анализ по расписанию	0	0	0	±	±	±	Н/Д	-	±	+	-	±	±	±	±
b. Остановка и продолжение анализа	1	1	1	+	+	+	Н/Д	+	+	Н/Д	+	+	Н/Д	-	-
c. Статус проводимого анализа	1	1	1	+	+	+	+	+	+	+	+	+	+	-	-
d. Определение неправильно заданной конфигурации	1	1	1	+	+	+	Н/Д	+	+	Н/Д	-	-	-	+	+
e. Проведение нескольких одновременных исследований	1	1	1	+	±	±	Н/Д	-	±	+	+	+	+	+	+
f. Поддержка использования проведения анализа несколькими пользователями	0	0	0	-	-	-	-	-	±	+	±	+	+	±	±

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ми															
g. Прерывание анализа	1	1	1	+	+	+	Н/Д	+	+	Н/Д	+	+	+	-	-
7.2. Возможности интерфейса															
a. Поддержка графического интерфейса	1	1	1	+	-	+	+	+	+	+	+	+	+	+	+
b. Поддержка командной строки	1	1	1	+	+	+	-	+	-	-	±	-	-	-	-
с. Веб-интерфейс (для множественного доступа)	0	0	0	-	-	-	-	+	+	+	-	+	+	-	-
7.3. Расширяемость и функциональная совместимость															
a. Библиотека анализа (API)	0.5	0.5	0.5	-	-	-	-	+	+	±	±	±	+	±	±
b. Интегрирование со средствами отслеживания ошибок	0.5	0.5	0.5	±	±	±	±	+	+	+	±	±	±	±	±
8. Предоставляемая отчетность															
8.1. Формирование отчетности	1	1	1	+	+	+	+	+	+	+	Н/Д	±	±	+	+
8.2. Настройка формирования отчетности	0	0	0	-	-	+	Н/Д	+	-	+	Н/Д	+	+	-	-
Сравнение относительно разрабатываемого ПО (p^1)				57.75	59.75	68.25	45.75	58.75	41	31.75	44.25	37.75	35.25	30.25	30.5
Сравнение относительно разрабатываемого ПО (p^2)				57.75	53.75	53.25	43.75	49.75	41	31.75	43.25	38.75	36.25	30.25	30.5
Сравнение относительно разрабатываемого ПО (p^3)				57.75	53.75	53.25	43.75	49.75	41	31.75	43.25	37.75	35.25	30.25	30.5
Сравнение относительно методик работы ПО (p^3)				42.5	39	37.5	34.5	33.75	29.5	23	29.75	27.75	25.5	21.75	21.75

1.5. Постановка задачи для проведения исследований

Вербальная постановка научной задачи:

Обеспечить корректное функционирование информационного ресурса путём проверки ключевых составляющих частей информационного ресурса и разработать предложения по обеспечению их функциональной корректности в условиях эксплуатации в сети Интернет.

Математическая постановка научной задачи:

Пусть информационный ресурс, находящийся в условиях эксплуатации в сети Интернет, обладает в текущий момент времени некоторыми характеристиками. Пусть все такие характеристики образуют множество $X = \{x_1, \dots, x_n\}$. Разобьём данное множество на непересекающиеся подмножества X_i таким образом, что $X = \bigcup_{i=1}^k X_i$. К разным подмножествам отнесем характеристики, описывающие не связанные напрямую между собой особенности информационного ресурса. Тогда каждое подмножество может быть оценено как

$$x^i = F_i(x_1^i, \dots, x_{n_i}^i),$$

где $\{x_1^i, \dots, x_{n_i}^i\} = X_i, i = \overline{1, k}, n = \sum_j n_j$.

Таким образом, общая оценка информационного ресурса представима в виде

$$A = F(x^1, \dots, x^n).$$

Опишем аналогично абстрактного типичного пользователя такого информационного ресурса в виде множества числовых характеристик. Пусть все такие характеристики образуют множество $Y = \{y_1, \dots, y_m\}$, разбитого на непересекающиеся подмножества $Y = \bigcup_{j=1}^l Y_j$. Тогда, возможности пользователя для использования информационного ресурса можно оценить как

$$y^j = G_j(y_1^j, \dots, y_{m_j}^j),$$

где $\{y_1^j, \dots, y_{m_j}^j\} = Y_j, j = \overline{1, l}, m = \sum_q m_q$.

В результате, общая оценка возможностей пользователя может быть представлена как

$$B = G(y^1, \dots, y^l).$$

Сопоставим оценки информационного ресурса (A) и пользователя (B) в виде одной функции, которая позволит оценить корректность функционирования информационного ресурса для типичного абстрактного пользователя:

$$H(x^1, \dots, x^k, y^1, \dots, y^l).$$

Такая функция позволяет учесть, что разные подмножества характеристик отвечают за принципиально несопоставимые стороны информационного объекта и численные оценки этих подмножеств имеют принципиально разный смысл.

Оценка функциональной корректности информационного ресурса предполагает выбор определенных количественных показателей, объективно отражающих существо рассматриваемой задачи. Разработка системы показателей и оценки функциональной корректности информационного ресурса является важным этапом проведения научных исследований, т.к. только после его завершения возможно проведение оценки функциональной корректности информационного ресурса и выбор методов (способов) для его корректного функционирования. Показатель обеспечения корректной функциональности информационных ресурсов в условиях программно-математического воздействия должен быть:

- выбран с учетом системного подхода к исследованию рассматриваемой проблемы и наиболее полно отражать существо исследуемых характеристик (параметров);

- критичным и чувствительным к изменениям основных параметров процесса функционирования информационного ресурса;

- наглядным и простым в применении.

Опишем исследуемые характеристики информационного ресурса. Пусть к разным подмножествам множества X будут относиться характеристики, описывающие не связанные напрямую между собой особенности исходного кода информационного ресурса. Так, одно из подмножеств будет содержать общую ста-

тистику по найденным ошибкам и несоответствиям стандарту: для закрывающего тега открытый тег не найден, неизвестные атрибуты, некорректные значения свойств и т.д. Другое подмножество будет содержать информацию об ошибочных ссылках, а также информацию о присутствии ссылок, результатом которых является обращение к одному и тому же материалу (т.н. цикличность).

Для объективной оценки качества кода воспользуемся числовым значением метрики исходного кода (подсчета числа ошибок на 1000 строк кода), типичной для большинства программ. При определении таких сравнительных оценок следует учитывать, что на контекст и результирующую плотность ошибок оказывает влияние множество факторов, однако проведенные многочисленные исследования [127] показывают, что для их учета необходимо использовать следующие значения:

$$\text{оценка качества кода} = \begin{cases} \text{отл.}, & \text{если } t \in [0,1), \\ \text{хор.}, & \text{если } t \in [1,5], \\ \text{удовл.}, & \text{если } t > 5, \end{cases}$$

где t - числовое значение метрики исходного кода (подсчета числа ошибок на 1000 строк кода).

Абстрактного типичного пользователя можно описать используемым веб-обозревателем. Такая характеристика является важной ввиду того, что задает множество определенных для пользователя средств отображения и функционирования (тегов, атрибутов, свойств, функций и объектов).

В итоге, к оценке функциональной корректности информационного ресурса можно подойти следующим образом (табл. 5).

Таблица 5. Предлагаемый критерий корректности функционирования информационного ресурса

Оценка корректности функционирования информационного ресурса	Значение метрики m	Типы выявленных ошибок	Ошибочные ссылки	Цикличность (кольцевые ссылки)
Информационный ресурс является некорректным	$m > 5$	В ходе анализа исходного кода данного информационного ресурса выявлены ошибки различных типов (предупреждающие, критические)	Присутствуют	Присутствует
Информационный ресурс является потенциально некорректным	$m \in [1,5]$	В ходе анализа исходного кода данного информационного ресурса выявлены предупреждающие и небольшое число критических ошибок	Присутствуют	Присутствует
Информационный ресурс является в целом корректным	$m \in (0,1)$	В ходе анализа исходного кода данного информационного ресурса выявлены только предупреждающие ошибки	В целом отсутствуют	В целом отсутствуют
Информационный ресурс является корректным	$m = 0$	В ходе анализа исходного кода данного информационного ресурса не выявлен ни один из типов ошибок	Отсутствуют	Отсутствует

Формализуем исследуемые характеристики информационного ресурса:

x_1^1 – числовое значение метрики исходного кода (подсчет числа ошибок на 1000 строк кода);

x_1^2 – число выявленных предупреждающих ошибок исходного кода;

x_2^2 – число выявленных критических ошибок исходного кода;

x_1^3 – число выявленных ошибочных веб-ссылок;

x_2^3 – число доступных веб-ссылок;

x_1^4 – число выявленных циклических ссылок;

$x_i^5 = \begin{cases} 0, P \subseteq W_i \\ 1, P \not\subseteq W_i \end{cases}, i = \overline{1, z}$ – результат принадлежности выявленного множества имен P среди множества имен веб-обозревателей $W_1 \dots W_z$.

Оценим каждое подмножество характеристик как

$$x^1 = F_1(x_1^1) = \begin{cases} 0, & x_1^1 \in [0,1), \\ 0.5, & x_1^1 \in [1,5], \\ 1, & x_1^1 \in (5, \infty), \end{cases}$$

$$x^2 = F_2(x_1^2, x_2^2) = (x_1^2 > 0) \vee (x_2^2 > 0),$$

$$x^3 = F_3(x_1^3, x_2^3) = \frac{x_1^3}{x_1^3 + x_2^3},$$

$$x^4 = F_4(x_1^4) = (x_1^4 > 0),$$

$$x^5 = F_5(x_1^5, \dots, x_z^5) = \sum_{i=0}^z x_i^5.$$

Важно отметить, что в таком виде характеристика x^5 показывает соответствие исследуемого текста веб-ресурса различным пространствам имен веб-обозревателей, тем самым проверяется свойство кроссбраузерности веб-ресурса (веб-ресурс является кроссбраузерным, если x^5 принимает значение 0).

Формализуем исследуемые характеристики пользователя:

$$y_i^1 = \begin{cases} i, & Q \subseteq W_i \\ 0, & Q \not\subseteq W_i \end{cases}, i = \overline{1, z} - \text{результат принадлежности выявленного множе-$$

ства имен Q среди множества имен веб-обозревателей $W_1 \dots W_z$.

Будем считать, что в некоторый момент времени для работы с веб-ресурсом пользователь использует один веб-обозреватель. Это значит, что один из элементов $y_i^1, i = \overline{1, z}$ будет иметь значение, отличное от 0. Оценка такого подмножества характеристик примет вид

$$y^1 = G_1(y_1^1, \dots, y_z^1) = \sum_{i=0}^z y_i^1 \neq 0.$$

Необходимо учесть, что в итоговой формуле характеристика пользователя y^1 должна быть сопоставлена характеристике информационного ресурса x^5 , т.к. эти характеристики являются взаимозависимыми. Таким образом, общий вид функции, которая позволит оценить корректность функционирования информационного ресурса, может быть записана в следующем виде

$$H(x^1, x^2, x^3, x^4, x^5, y^1) = \sum_{i=1}^4 x^i + \left(1 - \sum_{j=1}^z (x_j^5 < 1) \& (y_j^1 > 0) \right).$$

В результате, интернет-ресурс является функционально корректным для пользователя, если характеристики пользователя и информационного ресурса удовлетворяют следующему соотношению

$$H(x^1, x^2, x^3, x^4, x^5, y^1) = 0.$$

Задача диссертационной работы состоит в улучшении и определении новых характеристик информационного ресурса для получения более точной оценки функциональной корректности интернет-ресурса.

Рис. 3 представляет методическую схему проведения исследований по оценке функциональной корректности информационных ресурсов.

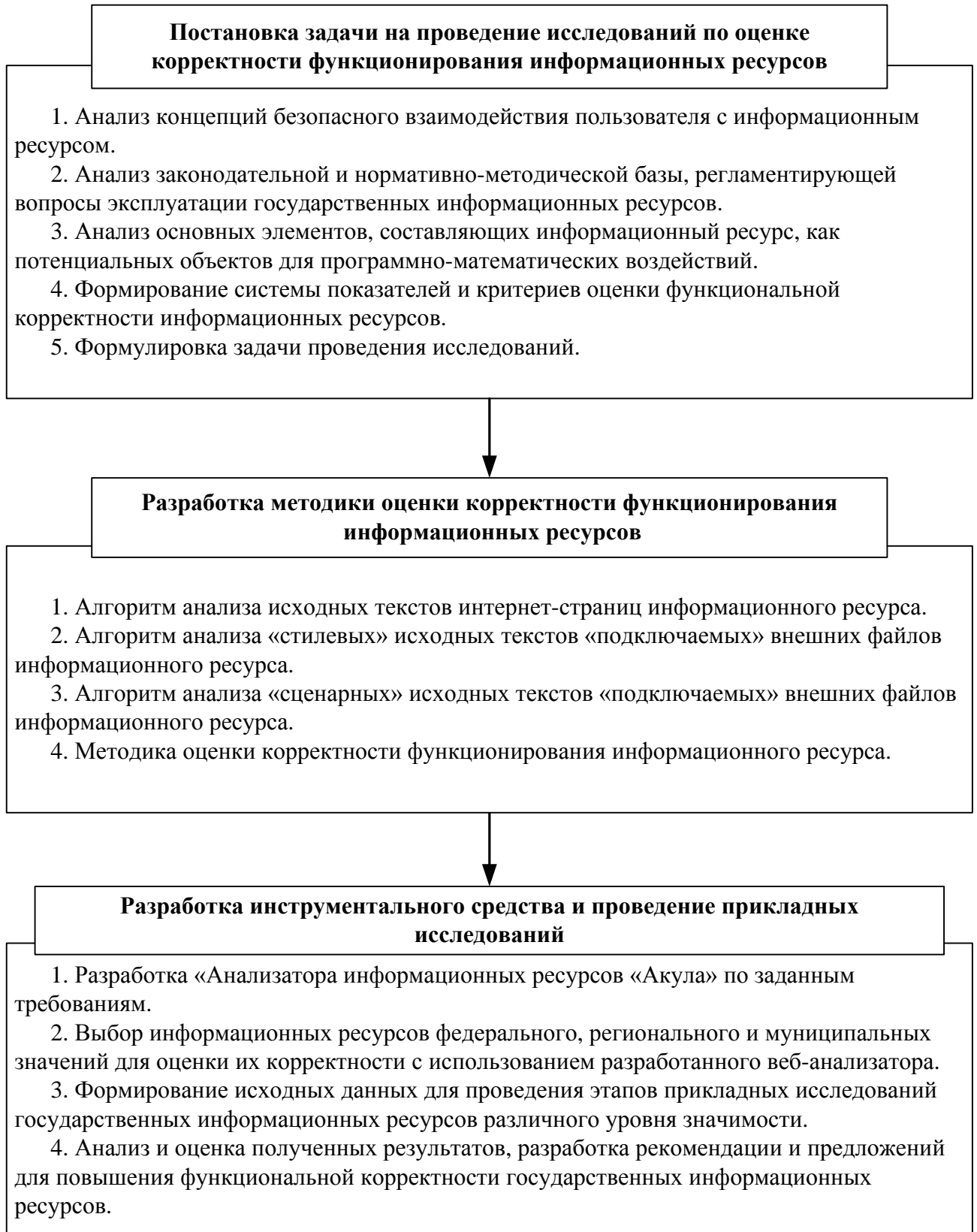


Рис. 3. Методическая схема проведения исследований по оценке функциональной корректности информационных ресурсов

Выводы по главе 1

В данной главе:

1. Проведен анализ взаимодействия пользователя с информационным ресурсом в сети Интернет, который свидетельствует о том, что рассмотренные аспекты помогают повысить безопасность пользователя в сети Интернет и пресечь несанкционированные действия со стороны информационного ресурса, при этом политика полноценного функционирования информационного ресурса должна максимально предсказывать любые возможные действия ошибочного характера. Однако это полностью не решает поставленную задачу, и, более того, интернет-ресурс может порождать некорректности, которые способны повлиять на пользователя. В таком случае, самым правильным решением является дополнительное исследование информационного ресурса со стороны клиента, т.е. обход всего интернет-ресурса и анализ исходного кода интернет-страниц, которые могут быть загружены при появлении пользователя на данной интернет-странице информационного ресурса.
2. Проведен обзор существующих в настоящее время методов и методик, направленных на выявление уязвимостей в информационных ресурсах сети Интернет.
3. Проведенный аналитический обзор на данный момент исследований в области функциональной корректности информационных ресурсов позволил определить, что ни одна из существующих методик не может предложить исчерпывающий набор методов и инструментов для выявления присутствующих в информационном ресурсе функционально-некорректных конструкций. Это значит, что существующие на сегодняшний день методы и методики являются несовершенными, а данная проблема по-прежнему остается нерешенной. Более того, само понятие полной функциональной корректности всего информационного ресурса находится лишь на «интуитивном» уровне, при котором проверка функ-

циональной корректности информационного ресурса фактически происходит исходя из «пользовательского принципа» для решения конкретных задач.

4. Проведенный обзор ведущих программных средств показал, что в настоящее время не существует ни одного программного средства, обеспечивающего гарантированный поиск функционально-некорректных конструкций. Это означает, что невозможно получить достоверную оценку корректности функционирования информационного ресурса и провести комплекс мероприятий, направленных на устранение неполадок и повышение общей безопасности интернет-ресурса. Таким образом, по полученным результатам выполненного расчета численных оценок реализованных возможностей программных обеспечений можно сделать вывод, что разрабатываемый в диссертационной работе анализатор исходных текстов информационного ресурса обладает рядом характеристик, которые не учтены в иных программных обеспечениях и исследованиях.
5. Сформулирована и поставлена задача для проведения исследований функциональной корректности информационных ресурсов.

Глава 2. Разработка методики оценки корректности функционирования информационного ресурса

2.1. Описание структурной схемы методологического аппарата

Структура методического аппарата для проведения оценки корректности функционирования информационного ресурса может быть представлена следующим образом (Рис. 4).



Рис. 4. Структура методического аппарата для проведения оценки корректности функционирования информационного ресурса

Анализ корректности функционирования информационного ресурса необходимо произвести в соответствии с поставленной задачей, которая формируется на основании:

- исходных текстов информационного ресурса;
- исходных текстов системы управления содержимым информационного ресурса;
- выбранного критерия и показателя корректности функционирования информационного ресурса.

Проведение исследований исходных текстов информационного ресурса заключается в применении разработанных алгоритмов анализа основных составляющих информационных технологий информационного ресурса:

- алгоритма анализа исходных текстов интернет-страниц информационного ресурса;
- алгоритма анализа «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса;
- алгоритма анализа «сценарных» исходных текстов «подключаемых» внешних файлов информационного ресурса.

Результатом проведенного исследования является оценка функциональной корректности информационного ресурса с учетом выбранного критерия. Выявленные в процессе анализа ошибки позволяют сформировать состав потенциально некорректных составляющих компонентов информационного ресурса и указать на конкретные функционально-некорректные конструкции.

2.2. Разработка алгоритма анализа исходных текстов интернет-страниц информационного ресурса

Исходный текст интернет-страницы в общем случае представляет собой HTML-документ со вставками JavaScript и CSS кода. Поэтому алгоритм анализа исходных текстов интернет-страниц состоит в проверке следующих пунктов:

- проверка HTML-конструкций интернет-страницы:
 - проверка корректности задания тегов и атрибутов;
 - проверка корректности задания значений атрибутов;
- проверка JavaScript вставок интернет-страницы:
 - поиск и анализ «подключаемых» js-файлов;
 - анализ вызовов из JavaScript-вставок, в т.ч. и в «подключаемые» js-файлы;
 - анализ конструкций JavaScript-вставок атрибутов;
 - анализ конструкций JavaScript-вставок всей интернет-страницы (в «парных» тегах «script»);
- проверка CSS-вставок интернет-страницы:
 - поиск и анализ «подключаемых» css-файлов;
 - анализ конструкций CSS-вставок атрибута «style»;
 - анализ конструкций CSS-вставок при описании интернет-страницы (в «парных» тегах «style»);
 - анализ использования описанных CSS-конструкций, в том числе и из «подключаемых» css-файлов;

Данный алгоритм реализован в HTML-анализаторе исходных текстов интернет-страниц, который имеет общую синтаксическую схему работы, изображенную на Рис. 5.

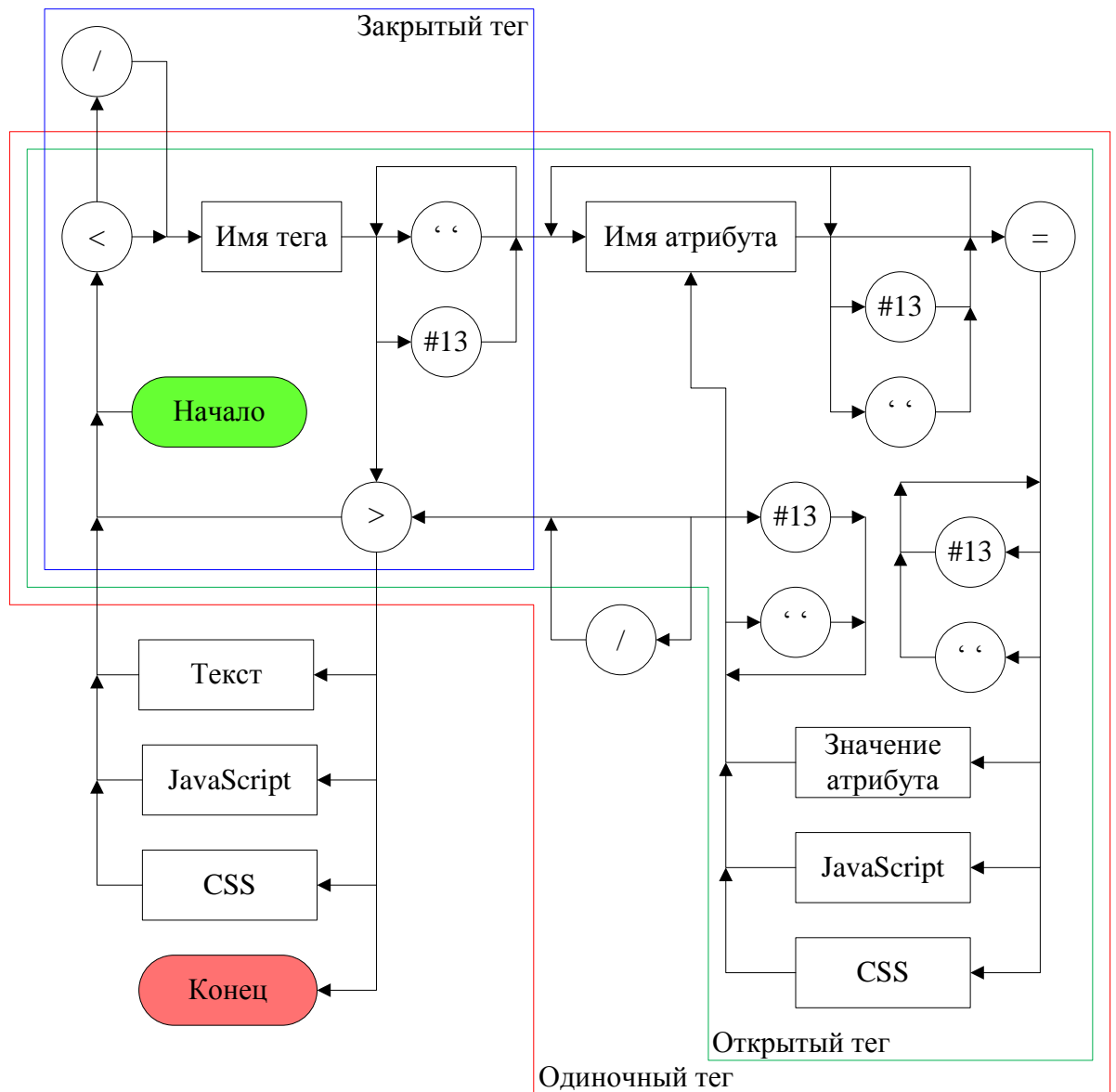


Рис. 5. Синтаксическая схема HTML-анализатора

Важно отметить, что при анализе HTML-документа можно выделить следующие три типа встречаемых тегов:

- «одиночный» тег – данный тип тегов не имеет внутреннего содержимого, действие на который будет распространяться, поэтому вся полезная информация содержится либо в названии тега, либо в названии и дополнительных определяющих его атрибутах;

- «открытый» тег – тип тегов, являющийся «парным», для которого необходимо иметь закрывающий тег, таким образом, определяется область действия тега

для внутреннего содержимого, в дополнительных определяющих атрибутах возможны корректирующие действия;

- «закрытый» тег – тег завершения действий «открытого» тега, совпадает с именем открытого тега и дополнительно вначале имеющий отличительный знак закрытия «/».

Дополнительно, при работе HTML-анализатора учтены встречные возможные посторонние символы пробела, табуляции (код символа - #9) и перевода строки (код символа - #13).

Для проведения анализа JavaScript и CSS вставок управление из HTML-анализатора передается в JavaScript-анализатор (подробнее см. параграф 2.4) или CSS-анализатор (подробнее см. параграф 2.3), который обрабатывает найденный код с учетом специфики размещения кода на интернет-странице и добавляет найденные элементы анализа во внутренне представление исследуемой интернет-страницы. По окончании работы данного вида анализаторов и для продолжения работы HTML-анализатора, управление возвращается обратно.

Найденные в процессе анализа HTML-документа ссылки на «подключаемые» JavaScript или CSS файлы обрабатываются путем добавления внутреннего представления данного файла в текущее внутреннее представление исследуемой интернет-страницы для возможности проведения общего анализа. Таким образом, внутреннее представление HTML-документа в общем случае представляет собой внутреннее представление нескольких составляющих – HTML, JavaScript и CSS.

Для составления **пространства HTML-имен** для анализа исходного текста интернет-страницы информационного ресурса используются следующие материалы:

- спецификация HTML 5 [4];
- спецификация HTML 4.01 [128];
- спецификация XHTML 1.0 [129];
- спецификация XHTML 1.1 [130].

Формируется таблица со следующими параметрами:

- название тега;

- параметры тегов:
 - «парность» тега («парный» или «одиначный» тег);
 - необязательное закрытие тега;
 - вхождение тега в зону действия тега, не требующего обязательного закрытия;
- имена универсальных атрибутов;
- имена допустимых атрибутов для каждого тега (в том числе и специальных атрибутов, допустимых для данного тега);
- допустимые значения атрибутов.

Процесс анализа происходит сравнением распознанного имени тега с найденным именем из пространства HTML-имен. Фиксируется позиция из пространства HTML-имен, которая учитывается при последующем анализе атрибутов. Полученные данные записываются во внутреннее построение и используются при дальнейшем анализе.

Спецификация HTML 4.01 подразумевает также введение нескольких типов HTML:

- Transitional («переходный»)¹⁰ – для совместимости со старыми версиями HTML;
- Strict («строгий»)¹¹ – не содержит элементов, отвечающих за внешний вид, а также «устаревших» или «не одобряемых» элементов;
- Frameset («фреймовый»)¹² – содержит элементы для создания наборов фреймов.

Таким образом, для проведения анализа исходных текстов информационного ресурса, учитывая использованные спецификации, необходимо ввести следующие **диалекты**:

- HTML 4.01/XHTML 1.0 Transitional/Frameset;

¹⁰ HTML 4.01 Transitional DTD [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/html4/loose.dtd>, свободный.

¹¹ HTML 4.01 Strict DTD [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/html4/strict.dtd>, свободный.

¹² HTML 4.01 Frameset DTD [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/html4/frameset.dtd>, свободный.

- HTML 4.01/XHTML 1.0 Strict;
- HTML 5;
- XHTML 1.1.

Однако при работе с интернет-браузерами составленное пространство HTML-имен поддерживается не полностью [131]. Поэтому необходимо дополнительно ввести диалекты для следующих популярных браузеров:

- Internet Explorer (IE);
- Mozilla Firefox;
- Opera;
- Google Chrome;
- Safari.

В результате сформирован список диалектов, позволяющий анализировать пространство HTML-имен со стороны клиента и со стороны сервера.

В спецификациях, отмеченных выше, HTML-документ может допускать так называемые **«опциональные» конечные теги**. Это позволяет при открытом «парном» теге не ставить закрывающий «парный» тег. Стандарт HTML 4.01 не конкретизирует теги, которые могут быть включены при отсутствующем закрывающем «парном» теге, в то время как HTML5 имеет соответствующие рекомендации [4, п.п. 8.1.2.4].

Однако данная предусмотренная возможность не позволяет избежать таких ошибок, как:

- «перекрестное» закрывание тегов (данная ситуация возникает при невозможности включения очередного открытого или одиночного тега в «тело» тега, в то время как предыдущий включенный тег был открыт);
- неправильная «разметка» (данная ситуация возникает при внутренней обработке браузерами - в «тело» тега включаются разное количество тегов).

Анализатор позволяет проводить исследование исходного кода в двух режимах:

- «основной» режим:
 - исследование HTML-текста;

- исследование текста HTML-текста с XML-вставками;
- «вспомогательный» режим:
 - исследование XML-текста;
 - исследование XSL- текста (XSLT/XSL-FO).

XML является расширяемым языком разметки [132]. Исследование данного вида текстов происходит при «объявлении» использования возможностей XML-технологий в начале изучаемого текста. Для «основного» режима XML-текст рассматривается вместе с HTML-кодом (сформируется общее внутреннее представление), для «вспомогательного» режима - формируется отдельное внутреннее представление.

XSL является языком для преобразования и виртуализации XML-документов [133], состоящий из следующих частей:

- XSL Transformations (XSLT) - язык для преобразования XML-документов [134];
- XSL Formatting Objects (XSL-FO) - язык для разметки типографских макетов и иных предпечатных материалов [135];
- XPath - язык путей и выражений для доступа к отдельным частям XML-документа при использовании XSLT [136].

Для данного вида файлов (с расширением .xsl) формируется отдельное внутреннее представление.

С помощью XPath реализуется навигация по DOM в XML. Ввиду того, что в основе XML лежит древовидная структура, фактически XPath – «путь» к элементу. В настоящий момент XPath является составной частью языка XQuery [137], который используется для обработки данных в формате XML.

Для использования XPath в JavaScript реализована библиотека XPath [138], которая реализует большую часть спецификации DOM Level 3 XPath, позволяя использовать XPath как в XML-документах, так и в HTML-документах.

Таким образом, при проведении анализа HTML-документа необходимо в свойствах элементов учитывать XPath-путь для проверки запросов из JavaScript.

2.3. Разработка алгоритма анализа «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса

Формальный язык описания внешнего вида документа CSS на интернет-страницах информационного ресурса, т.е. встроенного в HTML-документ, согласно Главе 1 «CSS и документы» [139], возможно использовать несколькими способами:

- между парой тегов «style»;
- с помощью атрибута «href» тега «link» или директивы «@import» тега «style» для подключения внешнего css-файла;
- с помощью атрибута style для непосредственного описания стиля.

В результате можно выделить два основных типа файлов для анализа CSS:

- HTML-документ со встроенными вставками CSS-кода;
- «подключаемый» внешний css-файл.

В параграфе 2.2. Разработка алгоритма анализа исходных текстов интернет-страниц

информационного ресурса» описаны основные принципы для изучения первого типа файлов. Алгоритм анализа исходных текстов второго типа файлов учитывается при проведении анализа первого типа файлов и заключается в проверке следующих пунктов:

- проверка корректности CSS-конструкций;
- анализ влияния описанного стиля на HTML-документ (в том числе выявление избыточных/неиспользованных описаний).

Данный алгоритм реализован в CSS-анализаторе, который имеет следующую общую синтаксическую схему работы, изображенную на Рис. 6.

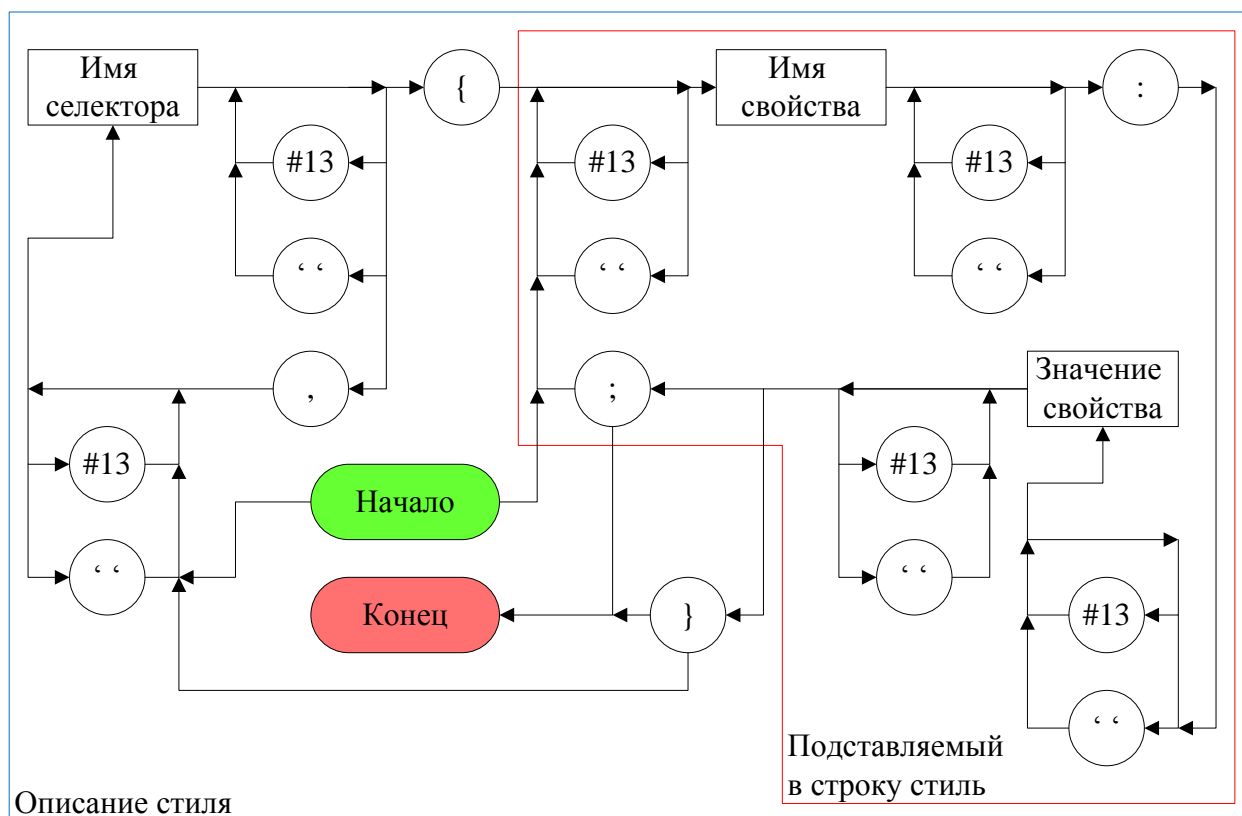


Рис. 6. Синтаксическая схема CSS-анализатора

CSS-анализатор позволяет проводить исследование исходного кода в двух режимах:

- обработка стиля, подставляемого в строку (данный режим применяется при обработке значения атрибута «style»);
- обработка «описательного» стиля (все остальные случаи).

При работе CSS-анализатора учтены встречающиеся возможные посторонние символы пробела, табуляции (код символа - #9) и перевода строки (код символа - #13).

Для составления **пространства CSS-имен** для анализа исходного текста CSS-вставок интернет-страницы информационного ресурса и «подключаемых» внешних css-файлов используются следующие материалы:

- спецификация CSS 2.1 [52];
- спецификация CSS3 [140];
- пространство HTML-имен.

Изучаемые параметры:

- название файла, где было обнаружено описание стиля;
- название свойства;
- название селекторов (для большинства селекторов название заимствуется

или является составной частью из пространства HTML-имен):

- универсальный селектор;
- селектор типа;
- селектор потомков;
- селектор дочерних элементов;
- селектор сестринских элементов;
- селектор классов;
- селектор идентификатора;
- простой селектор атрибутов;
- селектор атрибутов с конкретным значением;
- селектор частичного значения атрибутов;
- селектор начальной/ конечной/произвольной подстроки значения атрибута;
- селектор языковых атрибутов;

- псевдоклассы и псевдоэлементы.

Исследование происходит путем сравнения распознанного имени селектора с найденным именем сформированного пространства CSS-имен (это имя может быть составным, поэтому изучаемый селектор учитывается с дополнительными параметрами). Далее для данного распознанного селектора происходит анализ свойств, названия которых также ассоциируются с пространством CSS-имен. Для каждого распознанного свойства проверяется допустимое значение.

2.4. Разработка алгоритма анализа «сценарных» исходных текстов «подключаемых» внешних файлов информационного ресурса

Как было указано в Главе 12 «JavaScript в веб-браузерах» [141], прототипно-ориентированный сценарный язык программирования JavaScript на интернет-страницах информационного ресурса, т.е. встроенного в HTML-документ, возможно использовать несколькими способами:

- между парой тегов «script»;
- с помощью атрибута «src» тега «script» для подключения внешнего js-файла;
- с помощью атрибута-обработчика событий «onclick» или «onmouseover» как значения HTML-атрибута;
- как URL с помощью специального протокола «javascript».

Таким образом, можно выделить два основных типа файлов для анализа JavaScript:

- HTML-документ со встроенными вставками JavaScript-кода;
- «подключаемый» внешний js-файл.

Базовые принципы исследования первого типа файлов описаны в параграфе 2.2. При проведении анализа первого типа файлов учитывается алгоритм анализа исходных текстов второго типа файлов. Алгоритм анализа исходных текстов второго типа файлов заключается в проверке следующих пунктов:

- проверка корректности JavaScript-конструкций;
- анализ дополненных заданных JavaScript-объектов различными методами и свойствами;
- выявление переопределения заданных переменных внутри локальной области, а также выявление неявной декларации переменных;
- анализ действий в глобальной области и в описанных функциях;
- анализ влияния «прикрепленных» внешних js-файлов на HTML-документ.

Данный алгоритм для отмеченных двух основных типов файлов реализован в JavaScript-анализаторе исходных текстов. Важно отметить, что при исследова-

нии отдельных js-файлов без конкретной интернет-страницы, данный анализ будет иметь общий набор возможных «влияний» на HTML-документ.

Для описания всех конструкций прототипно-ориентированного сценарного языка программирования JavaScript согласно Приложению Г. «Синтаксические диаграммы» [142] необходимо использовать 43 синтаксические схемы. Построим граф, взяв за вершину – имя синтаксической схемы, а ребро – используемые имена на синтаксических схемах для «обработки» данной синтаксической схемы (Рис. 7).

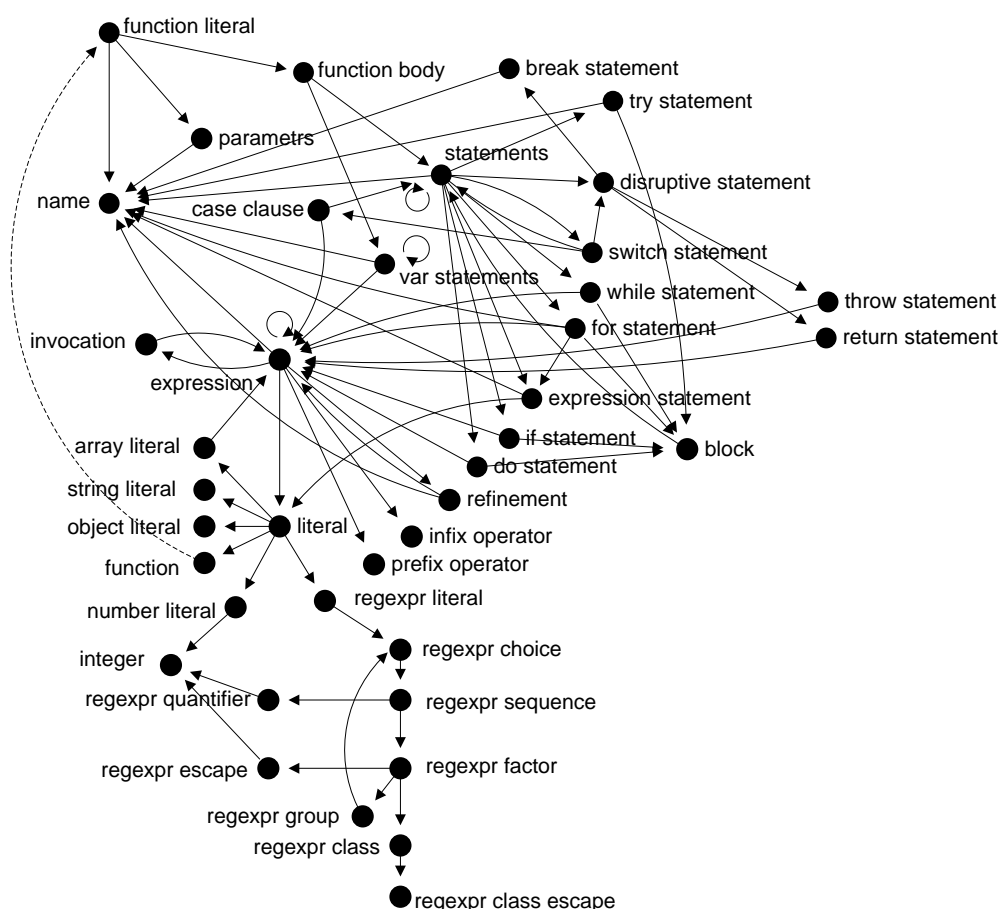


Рис. 7. Система использования синтаксических схем JavaScript

Такие же аналогичные действия справедливы для спецификации ECMA-262 [53]. Данная схема является более обширной ввиду того, что описывает полностью все возможности языка ECMAScript, подмножеством которого является клиентский JavaScript (Рис. 8).

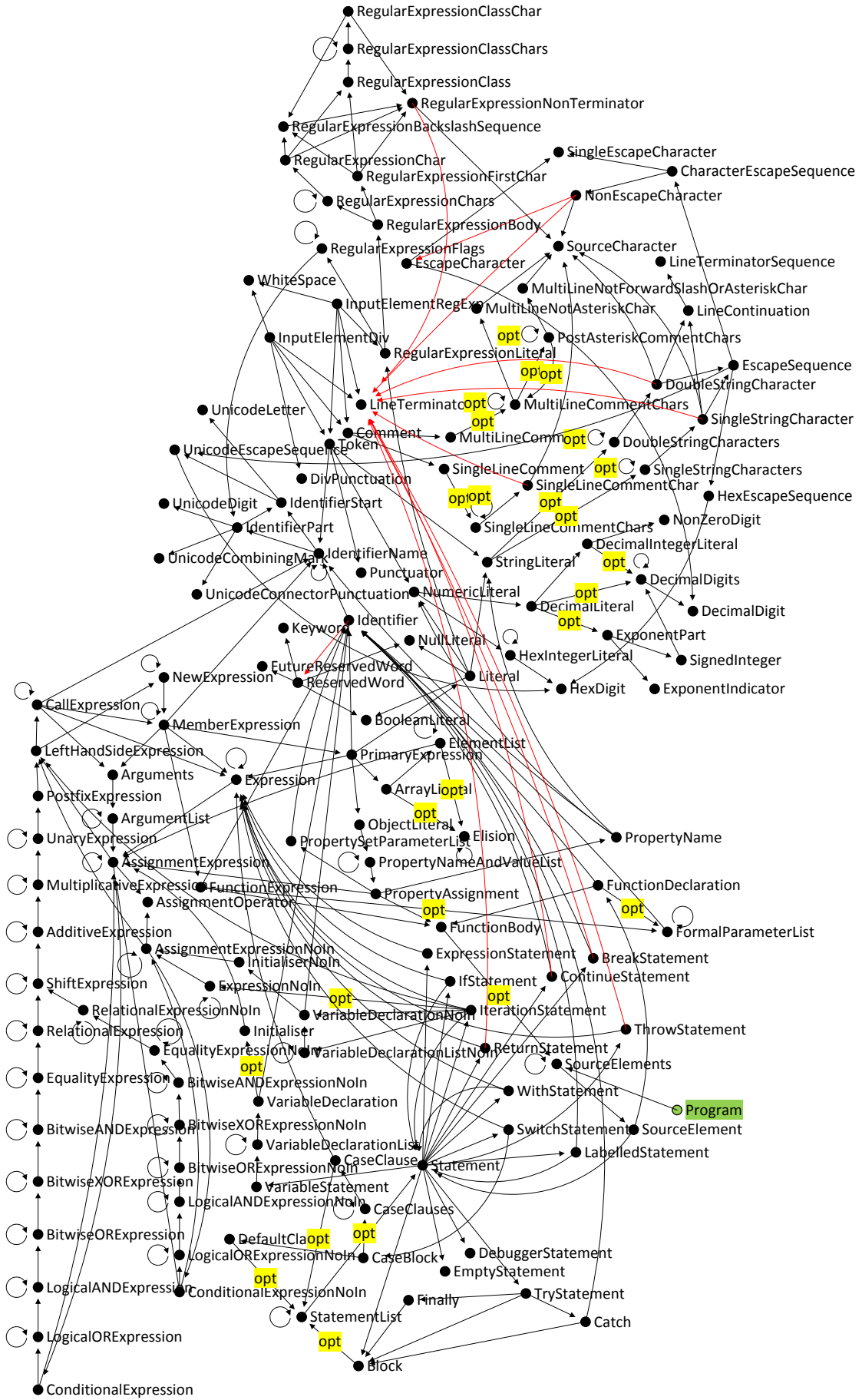


Рис. 8. Система синтаксических схем ECMA 262

Однако в источнике [142] не указана общая синтаксическая схема, справедливая для описания JavaScript-конструкций любой интернет-страницы или «подключаемого» внешнего «сценарного» файла информационного ресурса. Поэтому согласно Главе 12 «JavaScript в веб-браузерах» [141] основными (базовыми) элементами языка следует выделить 4 синтаксические схемы: блок «Пробел», блок «Операторы», блок «Литерал функции» и блок «Объявление переменных». Остальные синтаксические схемы являются дополняющими или раскрывающими. Таким образом, ранее представленный граф можно разбить на 3 различные области действия каждой конструкции (Рис. 9).

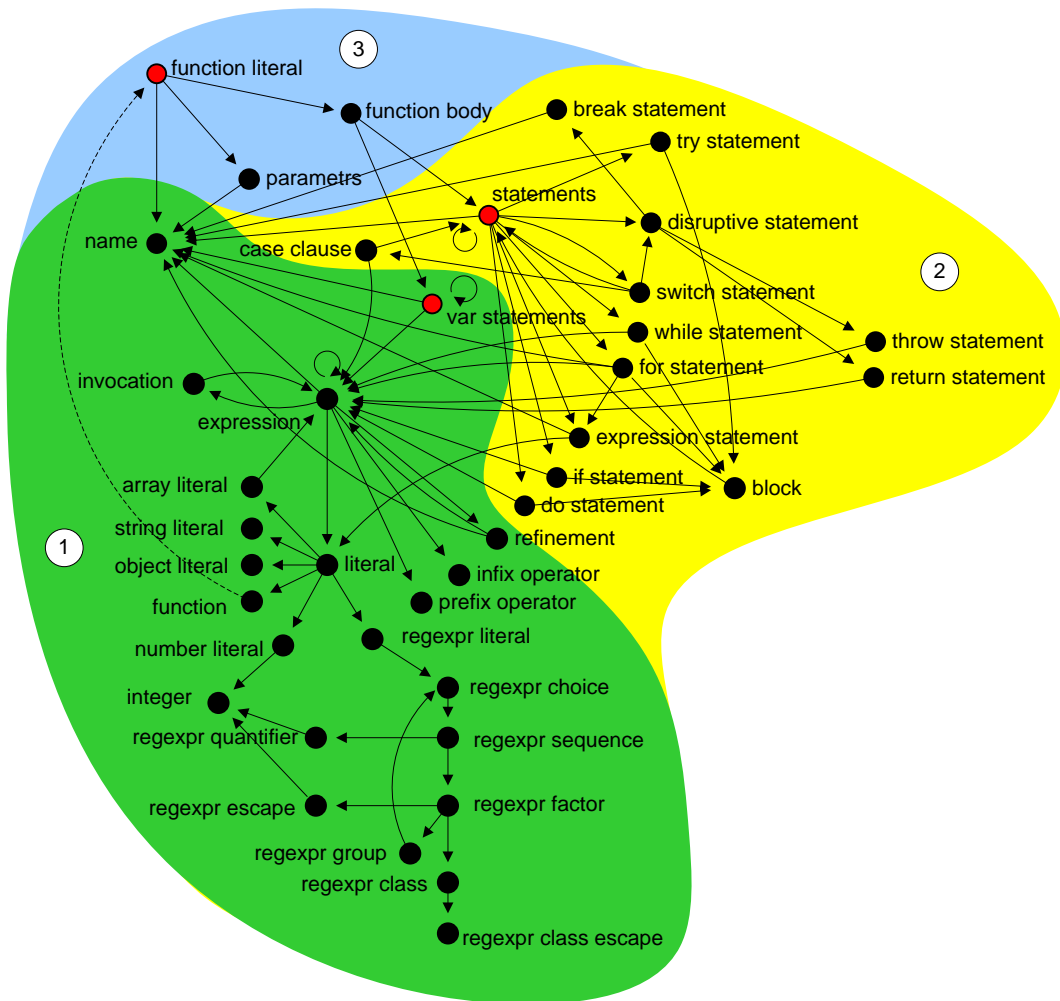


Рис. 9. Области «действия» основных конструкций JavaScript

В сформированных областях можно выделить цепочку включений: блок «Объявление переменных» содержится в блоке «Операторы», блок «Операторы» содержится в блоке «Литерал функции». Важно также отметить, что построенный граф не содержит синтаксическую схему для описания комментариев, т.е. не содержит блок «Пробел», потому как данный блок является функционально значимым в каждой конструкции.

В результате, синтаксическая схема JavaScript-анализатора может быть представлена следующим способом, изображенным на Рис. 10.

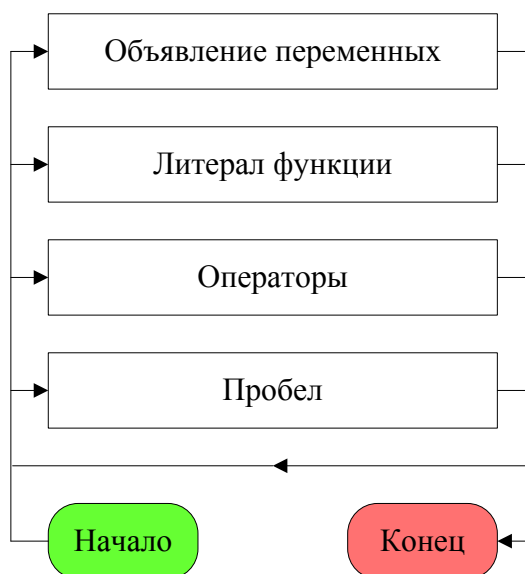


Рис. 10. Синтаксическая схема JavaScript-анализатора

Блок «Пробел» (в оригинальном варианте книги [142] - «Whitespace») содержит проверку таких символов как символы пробела, табуляции (код символа - #9) и перевода строки (код символа - #13), а также проверку конструкций комментариев.

Блок «Операторы» (англ. вариант - «Statements») содержит в себе 9 основных языковых типов операторов (do, for, while, switch, if, try, break, return и throw), а также дополнительный тип оператора – «Оператор выражения», который позволяет формировать выражения. Также перед некоторыми типами операторов (switch, while, for и do) возможно задание «названия метки».

Блок «Литерал функции» (англ. вариант - «Function literal») содержит описание функции: ключевое слово «function», название функции, заданные параметры для вызова и тело функции.

Блок «Объявление переменных» (англ. вариант - «Var Statements») содержит имена используемых переменных на интернет-странице или в «прикрепляемом» внешнем js-файле и их заданные начальные выражения при необходимости.

Таким образом, JavaScript анализатор, работающий на представленной синтаксической схеме, способен исследовать все возможности прототипно-ориентированного сценарного языка программирования JavaScript.

Для составления **пространства JavaScript-имен** при исследовании исходного текста JavaScript-вставок интернет-страницы информационного ресурса и «подключаемых» внешних js-файлов используются следующие материалы:

- спецификация JavaScript от MDN [143];
- спецификация ECMA-262 [53];
- спецификация Microsoft JScript [144].

Каждая из используемых спецификаций определяет свой прототипно-ориентированный сценарный язык программирования JavaScript, однако с выходом очередных версий любой спецификации возможности либо совпадают, либо дополняют друг друга, но в любом случае каждая из спецификаций способна реализовать большинство используемых возможностей JavaScript. Таким образом, под названием «JavaScript» можно понимать любую из спецификаций (подробнее Глава 1 «Введение в JavaScript» п.п. 1.2 «Версии JavaScript» [141]).

Основными составляющими языка JavaScript являются (Глава 1 «Введение в JavaScript» п.п. 1.3 «Клиентский JavaScript» [141]):

- «клиентский» JavaScript (интерпретатор JavaScript, встраиваемый в веб-браузер);
- «базовый» JavaScript (язык JavaScript, определяемый отмеченными выше спецификациями);
- DOM (поддержка отмеченными выше спецификациями рекомендации W3C DOM).

С одной стороны, «клиентский» JavaScript является расширением «базового» JavaScript, т.к. основывается на спецификациях JavaScript и является результатом работы интерпретатора JavaScript, встраиваемого в веб-браузер. С другой стороны, спецификация (или рекомендация) W3C DOM определяет порядок работы с объектной моделью документа и является расширением «базового» JavaScript. Таким образом, соотношение составляющих JavaScript можно представить в следующем виде (Рис. 11).

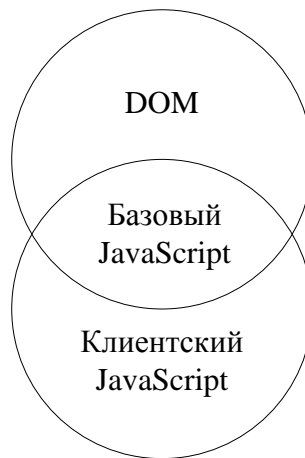


Рис. 11. Соотношение составляющих JavaScript

Однако «клиентский» JavaScript для полноценной работы должен объединять в себе две технологии - расширять возможности (или поддерживать возможности) как «базового» JavaScript, так и уметь работать с объектной моделью документа (в данном случае - это HTML-документа и XML-документа). Таким образом, «клиентский» JavaScript должен сочетать вышеописанные спецификации, а также спецификацию W3C DOM. В результате, соотношение составляющих JavaScript для полноценной работы можно представить в следующем виде (Рис. 12).

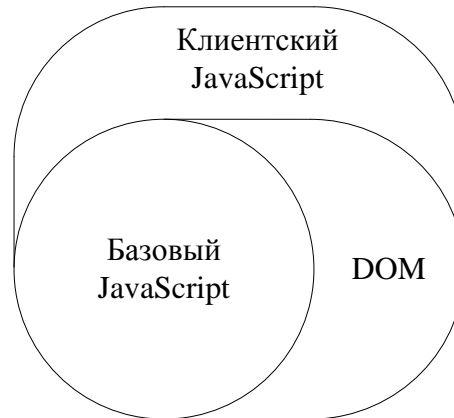


Рис. 12. Соотношение составляющих «клиентского» JavaScript

Поэтому, упоминая про JavaScript, большинство людей подразумевают именно «клиентский» JavaScript.

Спецификации, отмеченные выше, определяют следующие **типы зарезервированных слов** (Глава 2 «Лексическая структура» п.п. 2.8 «Зарезервированные слова» [141]):

- зарезервированные ключевые слова JavaScript (имеют специальное значение для интерпретатора JavaScript, т.к. являются частью синтаксиса языка);
- слова, зарезервированные для расширений ECMA (определены спецификацией для возможного расширения языка);
- идентификаторы, которых стоит избегать (глобальные свойства, переменные и функции, определенные спецификацией);

Кроме того, **среда программирования «клиентского» JavaScript** должна включать в себя три основных компоненты (Глава 12 «JavaScript в веб-браузерах» [141]):

- объект Window (глобальный объект и глобальный контекст исполнения для кода «клиентского» JavaScript);
- объекты «клиентского» JavaScript и объектная модель документа (DOM);
- управляемая событиями модель программирования (независимые, но взаимодействующие между собой обработчики событий, встроенные в HTML-файл).

Рис. 13 отображает иерархию объектов «клиентского» JavaScript, которая иллюстрирует объекты документа, ставшие стандартом, единообразно реализованным всеми основными веб-браузерами.

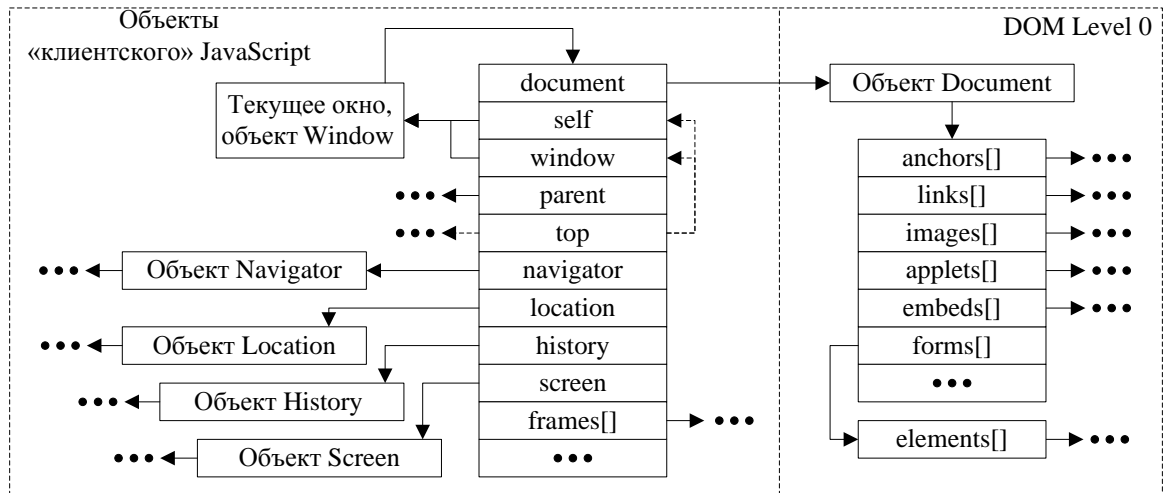


Рис. 13. Иерархия объектов «клиентского» JavaScript и DOM Level 0

Объект Window – ключевой объект в «клиентском» JavaScript, поэтому находится в корне иерархии. Все остальные объекты присоединены к нему и доступны как свойства других объектов. Поддерево объекта Document известно как объектная модель документа (DOM) и образует базовый уровень функциональности документа, т.е. DOM Level 0.

DOM Level 0 не относится к какому-либо формальному стандарту, а служит для неформальной ссылки на общие средства моделей документа, реализованных основными браузерами до стандартизации консорциумом всемирной паутины W3C. Помимо DOM Level 0 имеются 3 «уровня» стандарта DOM. DOM Level 1 определяет базовые интерфейсы DOM в HTML- и XML-документах, такие как Node, Element, Attr и Document. DOM Level 2 расширен за счет определения работы с событиями документа и каскадными таблицами стилей (CSS). DOM Level 3 состоит из 6 спецификаций, которые являются дополнительными расширениями DOM.

Тем не менее, текущим уровнем спецификаций DOM является DOM Level 2, однако некоторые части спецификаций DOM Level 3 являются рекомендуемыми W3C. Учитывая это обстоятельство, даже на момент написания данной работы не существовало веб-браузера, полностью соответствующему стандарту DOM. Однако в настоящее время изменения в сфере поддержки веб-браузерами стандартов происходят слишком быстро, чтобы пытаться выявить конкретный веб-браузер с более полной поддержкой DOM, поэтому для тестирования реализации DOM организация W3C сделала средство для его проверки [145]. Таким образом, при исследовании исходного кода «клиентского» JavaScript необходимо также учитывать поддержку DOM веб-браузером данных возможностей.

В результате, для составления пространства JavaScript-имен для анализа исходного текста интернет-страницы и «подключаемых» внешних файлов информационного ресурса используются следующие материалы:

- спецификации;
- зарезервированные слова, соответствующие данным спецификациям;
- объекты «клиентского» JavaScript.

Параметры при составлении пространства JavaScript-имен представлены следующим образом:

- название объекта;
- принадлежность объекта (составляющая JavaScript, к которой относится объект);
- ссылка объекта на родительский объект (если таковой имеется);
- тип, возвращаемый объектом;
- методы и свойства, определенные для объекта;
- тип, возвращаемый методом;
- тип свойства;
- свойства, доступные только для чтения (константы).

Данные параметры вносятся во внутреннее представление для дальнейшего анализа. Помимо выделенных параметров, анализ JavaScript-кода проходит с учетом зарезервированных имен, определенных JavaScript.

Процесс анализа JavaScript-имен проходит сравнением распознанного имени сначала с именем зарезервированного слова, а затем при необходимости с именами объектов, методов и свойств. В случае обнаружения объекта, данный параметр фиксируется в специальном значении, которое используется при дальнейшем анализе для поиска метода или свойства. При внесении изменений в объекты «клиентского» JavaScript, эти данные отражаются во внутреннем представлении для дальнейшего анализа.

2.5. Методика оценки корректности функционирования информационного ресурса

Основным объектом испытаний при проведении исследований являются информационные ресурсы общего пользования, находящиеся в условиях эксплуатации в сети Интернет.

Проводимые **испытания информационного ресурса:**

- «поверхностный» анализ информационного ресурса подразумевает проведение анализа только исходных текстов информационного ресурса в сети Интернет «со стороны сервера»;

- «глубокий» анализ информационного ресурса помимо проведения «поверхностного» анализа подразумевает проведение дополнительного анализа исходных текстов системы управления содержимым, функционирование которого осуществляется на сервере информационного ресурса

Целью испытаний является проверка соответствия информационных ресурсов общего пользования следующим требованиям:

- обнаружение факторов возможности неправомерных действий на информационном ресурсе:
 - получение несанкционированного доступа;
 - модификация/уничтожение информации;

- выявление возможных действий пользователей, в результате которых может быть нарушено функционирование информационного ресурса:
 - обнаружение уязвимостей на информационном ресурсе;
 - ошибки, допущенные при разработке или проектировании информационного ресурса (в том числе ошибки, внесенные при эксплуатации).

Продолжительность испытаний зависит от:

- типа проведения анализа («поверхностного» или «глубокого»);
- месторасположения/«удаленности» информационного ресурса;
- возможностей канала (скорости передачи данных в сети Интернет между сервером и пользователем);
- объема исследуемых текстов (общего количества интернет-страниц и подключаемых внешних файлов информационного ресурса).

Испытаниям подлежат следующие характеристики информационного ресурса:

- надежность (отсутствие сбоев в работе системы при проведении испытаний);
- полнота (определение отношение числа доступных интернет-страниц и подключаемых внешних файлов информационного ресурса к общему объему найденных интернет-ссылок внутри информационного ресурса);
- качество (определение отношение числа функционально правильно работающих интернет-страниц и подключаемых внешних файлов информационного ресурса к общему объему найденных интернет-страниц и подключаемых внешних файлов информационного ресурса).

Программа испытаний предполагает полную проверку функционирования и выявление всех уязвимостей информационного ресурса. Испытания проводятся в четыре этапа:

1. Проверка исходного состояния информационного ресурса;
2. Статический анализ исходных текстов информационного ресурса;

3. Динамический анализ исходных текстов информационных ресурсов;
4. Выявление и анализ значимых событий.

Проверка исходного состояния информационного ресурса проходит с целью определения доступности всего информационного ресурса. Данный этап производится с помощью программного комплекса «Акула» и включает следующие технологические операции:

- составление полного списка всех интренет-страниц и подключаемых «отдельных файлов» информационного ресурса, необходимых для функционирования интернет-ресурса;

- проверка доступности всех интернет-страниц и подключаемых внешних файлов информационного ресурса, в том числе выявление интернет-страниц и подключаемых внешних файлов информационного ресурса, доступ к которым невозможно получить.

Целью проведения **статического анализа исходных текстов информационного ресурса** является выявление опасных или потенциально опасных участков кода, которые могут быть использованы пользователем-злоумышленником при использовании информационного ресурса.

Статический анализ производится с помощью программного комплекса «Акула» и включает следующие технологические операции:

- построение и анализ объектной модели HTML документа HTML DOM (в том числе проверка допустимых значений HTML DOM);
- проверка описания «внешнего вида» HTML документа (проверка CSS-стиля как в самом HTML документе, так и в прикрепленном к HTML документу файле);
- анализ JavaScript-кода внутри HTML документа (проверка JavaScript-кода как в самом HTML документе, так и в прикрепленном к HTML документу файле):
 - анализ информационных объектов различных типов (локальных переменных, глобальных переменных, внешних переменных и т.п.);

- формирование списка маршрутов выполнения функциональных объектов (процедур, функций, ветвей);
- анализ критических маршрутов выполнения функциональных объектов (процедур, функций) для заданных экспертом списков информационных объектов;

При проведении «глубокого» типа анализа информационного ресурса необходимо произвести дополнительно:

- анализ PHP-кода системы управления содержимым информационного ресурса:
 - анализ информационных объектов различных типов (локальных переменных, глобальных переменных, внешних переменных и т.п.);
 - формирование списка маршрутов выполнения функциональных объектов (процедур, функций, ветвей);
 - анализ критических маршрутов выполнения функциональных объектов (процедур, функций) для заданных экспертом списков информационных объектов;

Исследование данных возможностей предполагается в дальнейших работах автора.

Динамический анализ исходных текстов информационного ресурса проводится с помощью программного комплекса «Акула». Он заключается в том, чтобы определить функционирование информационного ресурса. Данный этап включает следующие технологические операции:

- моделирование модифицированной объектной модели HTML документа HTML DOM посредством воздействия на него JavaScript-кода.

При проведении испытаний информационного ресурса необходимо проводить **выявление и анализ значимых событий**, к которым относятся:

- доступ к управляющим элементам ActiveX;
- экспорт информации об интернет-пользователе (например, адрес электронной почты, история просмотра интернет-страниц и т.д.);

- выявление «межсайтового скриптинга» (т.н. XSS атака);
- выявление «межсайтовой подделки запроса» (т.н. CSRF или XSRF атака);
- подмена прототипа/сессии (при помощи технологий AJAX).

Испытания проводятся в указанном выше порядке. Возможно частичное совмещение первых трех этапов (при наличии программного обеспечения для автоматизированного анализа).

К информационному ресурсу предъявляются следующие требования:

- для проведения «поверхностного» анализа:
 - информационный ресурс должен быть расположен в сети Интернет или исходные тексты информационного ресурса должны быть представлены на носителе¹³.
- для проведения «глубокого» анализа:
 - необходимо дополнительно предоставить исходные тексты системы управления содержимым, при работе с которой возможно получить все интернет-страницы информационного ресурса.

¹³ Под исходными текстами понимаются тексты интернет-страниц и подключаемых внешних файлов, загруженные с информационного ресурса в сети Интернет

Выводы по главе 2

В данной главе:

1. Разработан алгоритм анализа исходных текстов интернет-страниц информационного ресурса.
2. Разработан алгоритм анализа «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса.
3. Разработан алгоритм анализа «сценарных» исходных текстов «подключаемых» внешних файлов информационного ресурса.
4. Предложена методика оценки корректности функционирования информационного ресурса, отличительной особенностью которой является исследование всей структуры информационного ресурса и выявление особенностей взаимодействия составляющих его элементов между собой.

Глава 3. Анализатор исходных текстов информационного ресурса

3.1. Формирование требований к возможностям анализатора

Для исследования защищенности информационных ресурсов согласно разработанной методике (параграф 2.5), создаваемый веб-анализатор должен обладать следующими характеристиками:

- проверять интернет-ресурс на соответствие веб-стандартам используемых технологий (с целью исключения сбоев в функционировании информационного ресурса с различными браузерами);

- анализировать и выявлять потенциально опасные конструкции исходного кода информационного ресурса;

- проводить анализ защищенности информационного ресурса на известные интернет-атаки;

- выполнять сканирование всего информационного ресурса (с целью получения более точного результата оценки корректности функционирования информационного ресурса);

- проверять доступность всех компонентов информационного ресурса;

- выполнять анализ информационного ресурса за обозримое время (с целью обеспечения мониторинга защищенности информационного ресурса);

- проводить комплексный анализ компонентов информационного ресурса согласно разработанным положениям методики (параграф 2.5);

- позволять производить масштабируемость (одновременное проведение анализа нескольких информационных ресурсов с целью получения оценки корректности функционирования в определенной части сегмента сети Интернет или отрасли);

- обладать свойством кроссплатформенности (для возможности проведения исследований на различных ОС).

3.2. Описание структуры анализатора

Для разработки программного обеспечения использована свободная среда разработки программного обеспечения Lazarus [146], в состав которого входит компилятор FPC (Free Pascal Compiler) [147], позволяющий получить кроссплатформенное программное обеспечение на всех популярных аппаратных и программных платформах на сегодняшний день.

Разрабатываемое программное обеспечение предназначено для исследования исходных текстов информационного ресурса и состоит из следующих анализаторов:

- HTML-анализатор (для проведения анализа исходных текстов интернет-страниц информационного ресурса);
- CSS-анализатор (для проведения анализа CSS-вставок интернет-страницы и «стилевых» исходных текстов «подключаемых» внешних файлов информационного ресурса);
- JavaScript-анализатор (для проведения анализа JavaScript-вставок интернет-страницы и «сценарных» исходных текстов «подключаемых» внешних файлов).

Данное программное средство предоставляет следующие возможности:

- производить автоматизированный анализ и загружать веб-страницы заданного веб-ресурса с сохранением структуры сайта;
- анализировать загруженные веб-страницы с помощью веб-анализаторов;
- составлять отчет о найденных ошибках по каждой странице веб-ресурса и вести диагностическую историю.

Процесс выполнения работы заключается в последовательном выполнении следующих действий (Рис. 14). Перед началом работы программы пользователю предлагается ввести основные параметры для запуска: исходный адрес ресурса в интернете, директорию для сохранения web-сайта и настройки интернет-соединения. После запуска происходит сохранение исходного состояния настроек и подготовка к началу работы - инициализации окружения для загрузки веб-

страницы. Загрузка происходит в 2 этапа: для начала запрашивается код ответа веб-страницы методом HEAD и в зависимости от ответа происходит загрузка содержимого веб-страницы методом GET. После окончания загрузки происходит подготовка к анализу исходного кода web-анализатором. Целью проведения анализа является нахождение всех веб-ссылок, относящихся к заданному веб-ресурсу, а также проверка кода на ошибки. После завершения анализа веб-страницы происходит сортировка найденных веб-ссылок и формирование URL-списка для дальнейшей загрузки. По окончании работы происходит составление отчета и запись диагностической истории.

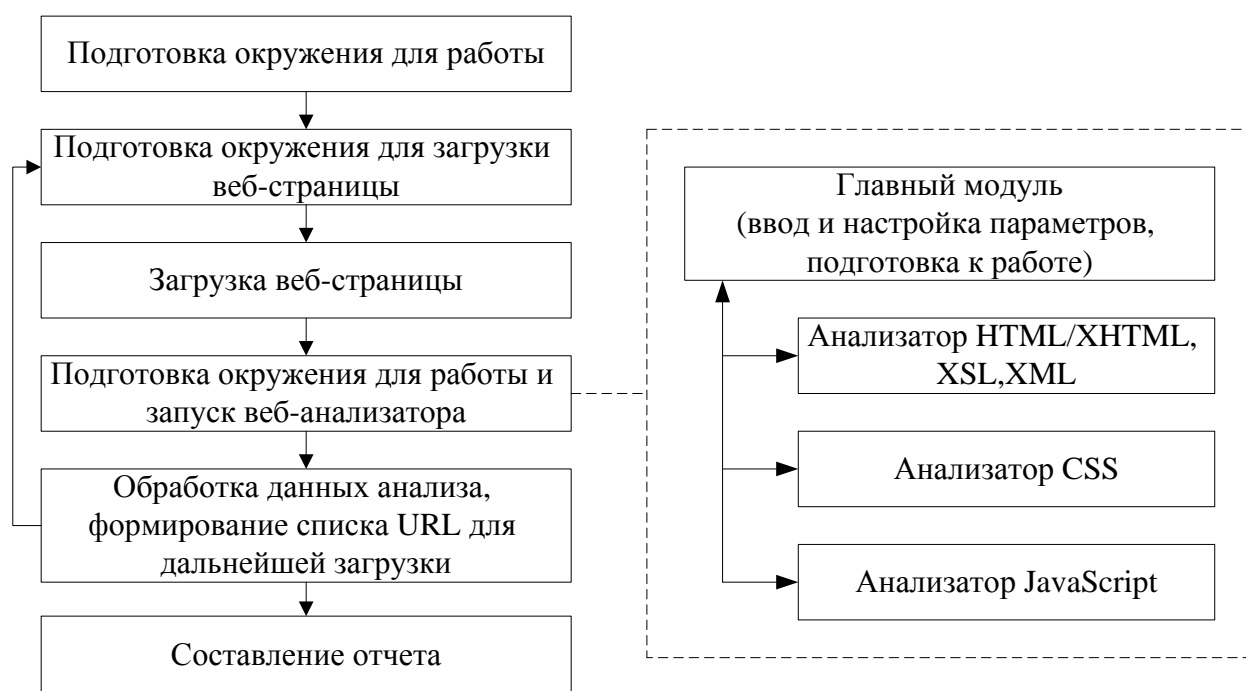


Рис. 14. Структура анализатора веб-ресурсов

Таким образом, алгоритм анализа информационного ресурса разработанным программным средством заключается в проведении **трех ключевых этапов**:

- загрузка интернет-страницы;
- проведение анализа интернет-страницы;
- анализ найденных на странице ссылок.

Первый этап состоит в правильном выборе, настройке и использовании сетевого модуля или компонента. В рамках разработанного программного обеспечения использованы следующие компоненты:

- InDy (Internet Direct) [148];
- ICS (Internet Component Suite) [149];
- Synapse [150].

Широкий выбор компонентов объясняется их различным спектром возможностей. Каждый из используемых компонентов является свободным программным обеспечением с открытым исходным кодом.

Второй этап заключается в проведении процесса анализа исходного кода обозначенными выше веб-анализаторами. Дополнительно производится поиск и формирование веб-ссылок интернет-страницы.

Последний этап заключается в проведении анализа отобранных веб-ссылок и переформировании списка веб-ссылок для дальнейшего исследования. Каждая отобранная веб-ссылка должна обладать следующими критериями:

- 1) не содержать символы перевода строки (#10, #13) и пробел;
- 2) не содержать многократные символы «\»;
- 3) не содержать вызов функции JavaScript;
- 4) не содержать почтовую ссылку;
- 5) не содержать «якорь»;
- 6) определять однозначно файл с расширением «.js», «.css» и «.html».

Алгоритм проверки найденной веб-ссылки на интернет-странице заключается в последовательной проверке обозначенных критериев, а также поиске исследуемой ссылки в ранее сформированном списке веб-ссылок для загрузки. Веб-ссылка считается отобранной, если она удовлетворяет указанным условиям. Рис. 15 содержит представленный алгоритм.

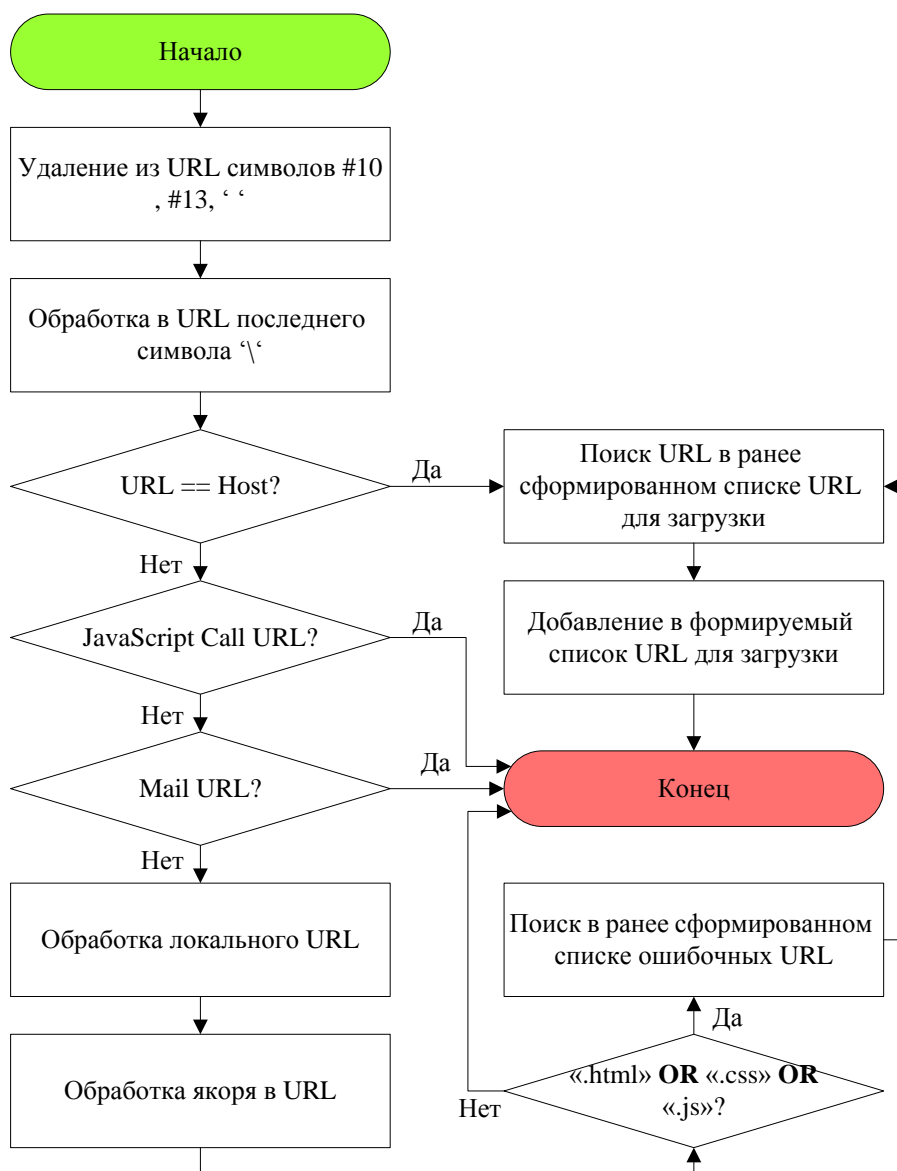


Рис. 15. Алгоритм обработки найденной на интернет-странице веб-ссылки

По результатам работы происходит формирование отчета, в котором содержится следующая информация:

- время запуска и окончания анализа, в том числе рассчитанное общее время работы и время бездействия анализатора¹⁴;
- актуальная версия анализатора;
- название анализируемого веб-ресурса;

¹⁴ Временем бездействия анализатора является суммарная продолжительность паузы при выявлении массовых ошибок загрузки веб-страниц

- результаты тестовой проверки соединения с анализируемым веб-ресурсом («пинг», предполагаемая скорость загрузки в Мбит/сек);
- путь корневой директории для сохранения структуры веб-ресурса;
- используемый сетевой компонент и выставленные параметры для проведения анализа;
- число загруженных веб-ссылок;
- общее число найденных ошибок;
- число файлов с ошибками, в том числе рассчитанный процент от общего числа загруженных веб-ссылок;
- общий объем загруженных ссылок (в МБайт);
- рассчитанная средняя скорость работы разработанного программного обеспечения (в Кбайт/сек);
- рассчитанная плотность ошибки на один файл и на один МБайт;
- статистика по найденным ошибкам, содержащая число ошибок по каждому известному коду ошибки, а также рассчитанный процент от общего числа ошибок;
- полный список загруженных веб-ссылок с указанием размера, типа ссылки и числа найденных на нем ошибок;
- краткий список загруженных веб-ссылок с найденными недочетами с указанием размера, типа ссылки, числа найденных на нем ошибок, а также позиции и кода найденной ошибки;
- число переадресованных веб-ссылок;
- полный список переадресованных веб-ссылок;
- число ошибочных веб-ссылок;
- полный список ошибочных веб-ссылок;
- рассчитанное число всего обработанных ссылок,
- рассчитанное отношение числа доступных ссылок ко всему числу обработанных ссылок;
- число найденных внешних ссылок с информацией о проверке их доступности;

- число ошибочных ссылок трансляции при работе анализаторов;
- список ошибочных ссылок трансляции с указанием возникших ошибок при проведении анализа.

В результате, **структура программных модулей разработанного веб-сканера** может быть представлена следующим образом (Рис. 16).

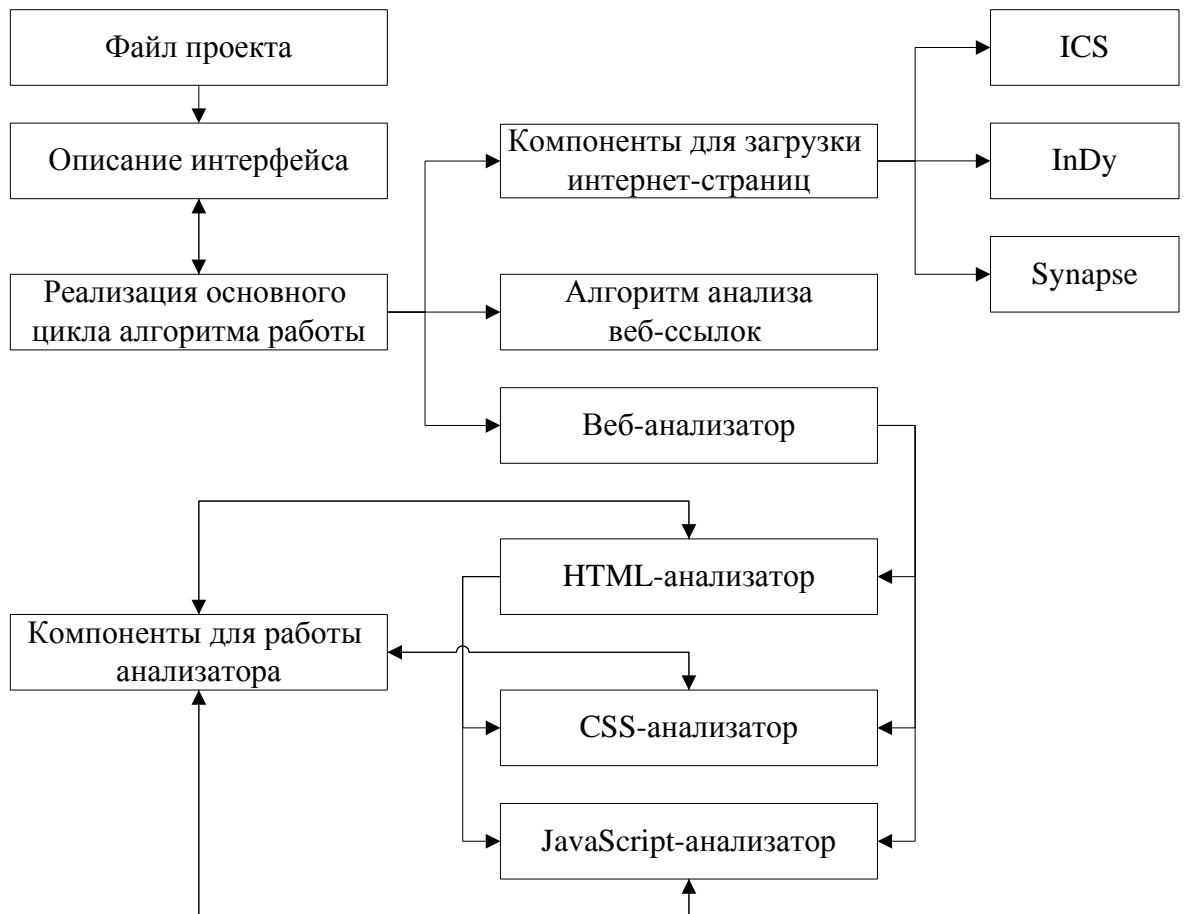


Рис. 16. Структура программных модулей

Отдельные программные модули необходимы для следующих ключевых особенностей веб-сканера:

- основной алгоритм/идея работы;
- каждый из рассмотренных этапов при проведении анализа (для разделения по принципу «одно следует за другим»);
- веб-анализатор исследуемого языка для разделения на логические части.

Рис. 17 представляет внешний вид основного окна «Анализатора информационных ресурсов «Акула».

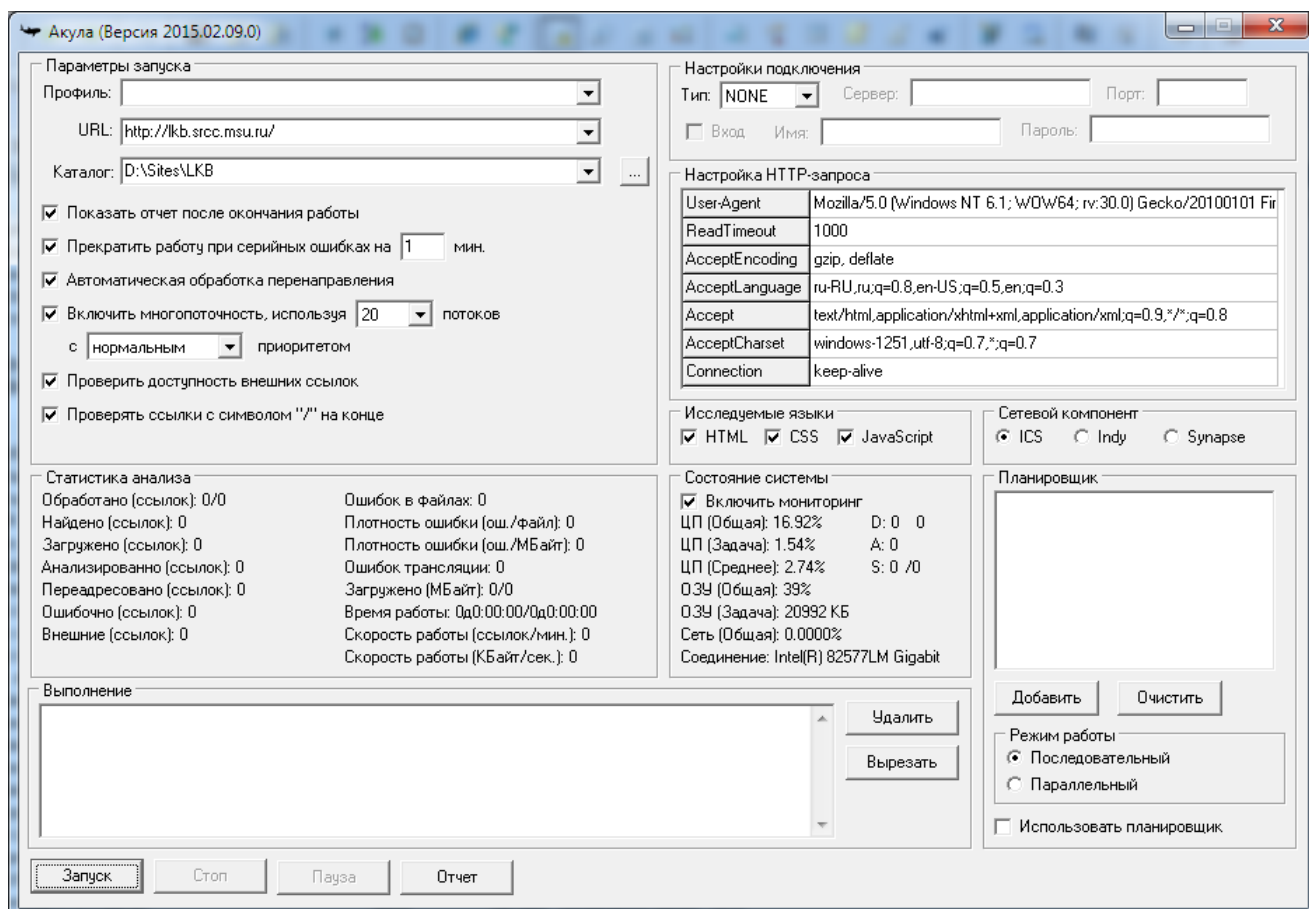


Рис. 17. Интерфейс «Анализатора информационных ресурсов «Акула»

3.3. Классификация выявляемых ошибок разработанным анализатором

В табл. 6 приведены ошибки, выявляемые HTML-анализатором при анализе исходных текстов интернет-страниц информационного ресурса.

Таблица 6. Диагностика ошибок HTML

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
000	Найден неизвестный тег	Тег не содержится в таблице известных тегов	Предупреждение

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
001	Некорректный закрывающий тег	Невозможность распознать закрывающий тег	Предупреждение
002	Некорректное окончание закрывающего тега	Отсутствие символа закрытия тега «>».	Предупреждение
003, 013 ¹⁵	Некорректное окончание тега	Искажение закрывающих символов тега (таких как «/>» и «?>»).	Предупреждение
004	Имя тега распознано некорректно	Ошибка в имени тега (возможно, был произведен перенос имени)	Предупреждение
005	Перекрестное закрытие тега	<a>	Предупреждение
006	Для закрывающего тега открытый тег не найден	Открытый тег отсутствует в структуре	Предупреждение
007	Недопустимое закрытие одиночного тега	Несоответствие стандарту. Тег используется в паре открытый - закрытый.	Предупреждение
008	Недопустимое повторение тега	Повторное использование обнаруженного тега не предусмотрено	Предупреждение
009	Некорректное окончание тега	Отсутствие символа закрытия тега «>» при распознавании атрибутов.	Предупреждение
010 ¹⁶ , 011	Некорректное распознавание значения у атрибута тега	Отсутствие повторяющей кавычки для окончания значения	Предупреждение
012	Неизвестный атрибут	Атрибут не содержится в списке допустимых атрибутов для данного тега	Предупреждение
014	Некорректное считывание атрибутов тега	Анализируемый текст закончился при считывании атрибутов.	Критическая

¹⁵ Выделенная ошибка для обозначения, что данная ошибка возникает именно после считывания атрибутов тега

¹⁶ Выделенная ошибка для тега style, который имеет ключевое значение при дальнейшем анализе CSS

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
015	Некорректное распознавание комментария	Анализируемый текст закончился при считывании комментария	Критическая
016, 017 ¹⁷	Не найден ни один параметр при чтении тега DOCTYPE	Несоответствие стандарту	Предупреждение
018 ¹⁸ , 019	Ошибка при чтении третьего параметра тега DOCTYPE	Несоответствие стандарту	Предупреждение
020 ¹⁹ , 021	Ошибка при чтении четвертого параметра тега DOCTYPE	Несоответствие стандарту	Предупреждение
022	Ошибка при чтении тега DOCTYPE	Символ конца тега не найден или анализируемый текст закончился	Предупреждение
023	В теге DOCTYPE описан другой вид диалекта	Неправильно настроенный диалект в параметрах для анализа	Предупреждение
024, 029 ²⁰	Закрывающий тег не найден	Анализируемый текст закончился и для открытого тега закрывающий тег не найден.	Критическая
025	Открывающий и закрывающий теги не совпадают	Несоответствие паре открытый – закрытый тег	Предупреждение
026	Файл не содержит полезной информации	Количество тегов в файле равно 0.	Критическая
027	Итоговая глубина тегов не совпадает	Какой-то из тегов был неправильно распознан и пара открытый - закрытый не совпала	Критическая
028	Исследуемый текст не соответствует диалекту	Анализируемый текст принадлежит другому диалекту	Предупреждение

¹⁷ Выделенная ошибка при чтении тега DOCTYPE для обозначения присутствия посторонних символов

¹⁸ Выделенная ошибка при чтении третьего параметра тега DOCTYPE для обозначения ошибочного чтения начального параметра. Аналогично, код ошибки 019 - для обозначения ошибочного чтения конечного параметра

¹⁹ Выделенная ошибка при чтении четвертого параметра тега DOCTYPE для обозначения ошибочного чтения начального параметра. Аналогично, код ошибки 021 - для обозначения ошибочного чтения конечного параметра

²⁰ Выделенная ошибка для тега style, который имеет ключевое значение при анализе CSS

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
030	В теге найдены посторонние символы	Внутри описания тега найдены символы, которые не относятся к ключевым словам	Предупреждение

Табл. 7 отражает ошибки, выявляемые CSS-анализатором при анализе исходных текстов интернет-страниц и «подключаемых» внешних «стилевых» файлов информационного ресурса.

Таблица 7. Диагностика ошибок CSS

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
100	Неизвестный селектор	Ошибка в имени селектора или селектор неизвестен	Предупреждение
101	Некорректное значение свойства	Недопустимое значение свойства	Предупреждение
102	Неизвестное свойство	Имя свойства распознано некорректно или не соответствует стандарту	Предупреждение
103	CSS правило описано, но не используется	CSS правило описано в теге style или прикрепляемом css-файле, но не используется в html-коде	Предупреждение

Табл. 8 отражает ошибки, выявляемые JavaScript-анализатором при анализе исходных текстов интернет-страниц и «подключаемых» внешних «сценарных» файлов информационного ресурса.

Таблица 8. Диагностика ошибок JavaScript

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
200	Ошибка при чтении тега script	Закрывающий тег script не найден	Критическая
201	Ошибка окончания тега script	Ошибочный закрывающий тег	Критическая
202	Объявляемый объект является предопределенным именем JavaScript	Изменение предопределенного имени JavaScript	Предупреждение
203	Переопределение объекта	Попытка определить ранее определенный объект	Предупреждение
204 ²¹ , 213 ²²	Недопустимый тип операнда	Невозможность выполнения операции	Предупреждение
205	Некорректный идентификатор	Имя идентификатора не может быть использовано при выполнении кода JavaScript	Предупреждение
206	Некорректное распознавание параметров функции	Невозможность обработки параметров функции	Предупреждение
207	Некорректный символ для обработки параметров функции	Начальный символ для параметров функции не найден	Предупреждение
208	Некорректный символ для обработки тела функции	Начальный символ для тела функции не найден	Предупреждение
209	Различное число параметров у вызываемой и описываемой функции	Несоответствие числа параметров	Предупреждение
210	Различные типы <i>N</i> -го параметра вызываемой и описываемой функции	Типы параметров не совпадают	Предупреждение
211	Различные типы <i>N</i> -го параметра вызываемого и описываемого объекта	Типы параметров не совпадают	Предупреждение
212	Неявное преобразование типов	Автоматическое преобразование типа	Предупреждение
214	Данный элемент массива может не существовать	Неявное обращение к элементу массива, которого может не быть	Предупреждение

²¹ Выделенная ошибка для второго операнда²² Выделенная ошибка для первого операнда

Код ошибки	Диагностика ошибки	Ситуация, когда возникает ошибка. Пример	Степень ошибки для анализатора
215	Неявная декларация переменной	Использование переменной без явной декларации	Предупреждение
216	Отсутствия знака препинания в конце конструкции	Возможная ситуация в конструкции var	Предупреждение
217	Некорректный возвращаемый тип функции	Несовпадение типов	Предупреждение

3.4. Принципы организации работы анализатора при исследовании сверх-больших информационных ресурсов

При организации работы веб-анализатора, существует необходимость в хранении следующих типов веб-ссылок:

- найденные веб-ссылки (совокупность отобранных ссылок с интернет-страниц);
- загруженные веб-ссылки;
- переадресованные веб-ссылки;
- ошибочные веб-ссылки;
- внешние веб-ссылки.

При проведении анализа существенными типами ссылок являются найденные, загруженные и ошибочные, т.к. при проведении этапа «анализа отобранных ссылок» необходимо найти отобранные ссылки среди ранее загруженных, ранее найденных и уже ошибочных. Таким образом, нужно провести поиск N отобранных веб-ссылок среди M загруженных, K найденных и L ошибочных.

Для совокупности веб-ссылок возможными вариантами организации структуры данных является:

- динамический массив строк;
- древовидная структура;
- база данных.

С одной стороны, веб-ссылка – фактически отдельный параметр, в то время как базу данных рационально использовать при хранении большого количества параметров. Таким образом, в текущей ситуации данная структура ничем не сможет помочь, несмотря на вполне удобный интерфейс и организацию работы. С другой стороны, веб-ссылка имеет ключевые части, в которой помимо используемого протокола и хоста содержится информация об n -ом уровне ссылки. В результате, если взять за корень название хоста, а листья и корневые вершины - все остальные подуровни ссылки, то массив веб-ссылок удобно ложится в древовидную структуру. Однако возможна следующая ситуация: все ссылки могут оказаться только на первом уровне, тогда древовидная структура теряет полностью смысл, ведь такое может возникнуть практически на любом интернет-ресурсе. Таким образом, остается самый простой вариант – динамический массив строк.

Для проведения оптимизации все веб-ссылки храним без названия хоста, т.к. эта информация будет совпадать у всех веб-ссылок. Из доступных вариантов поиска строки в массиве строк следует выделить следующие:

- прямой перебор строк в неотсортированном массиве;
- прямой перебор строк в отсортированном массиве.

Среднее время на поиск искомой строки для первого и второго варианта соответственно будет составлять

$$T_1 = p \frac{n}{2} + (1 - p)n,$$

$$T_2 = p \frac{n}{2},$$

где p - вероятность присутствия искомой строки в массиве строк, n - размерность массива строк. Второй вариант значительно быстрее первого, однако добавление строки в данный массив будет происходить более медленно. Тем не менее, оба варианта являются медленными при большом размере массива строк.

Таким образом, возможен следующий выход из данной ситуации: перед непосредственно посимвольным сравнением строк формировать хеш-таблицу со значениями. Посчитаем выигрыш от такого сравнения. Пусть

$$\alpha_1 = \frac{M}{D_1},$$

$$\alpha_2 = \frac{K}{D_2},$$

$$\alpha_3 = \frac{L}{D_3},$$

где M, K, L – ранее принятые обозначения, $D_1 \dots D_3$ – длины хеш-таблиц. Тогда среднее число сравнений для поиска искомой веб-ссылки без коллизий [151-152] для каждого из массивов веб-ссылок необходимо сделать

$$C_i = \frac{1}{\alpha_i} \ln \frac{1}{1 - \alpha_i} + O\left(\frac{1}{D_i}\right), i = \overline{1,3}.$$

В результате, общее число сравнений в таком случае будет

$$C = N \sum_{i=1}^3 C_i.$$

Стоит отметить, что среднее число сравнений для поиска искомой веб-ссылки с коллизиями будет больше. Тем не менее, в зависимости от выбранной длины хеш-таблицы, скорость поиска увеличивается в заданное количество раз, что является приемлемым для использования такой структуры как массив строк.

3.5. Алгоритм проведения эффективного анализа информационных ресурсов

В параграфе 3.2 отмечено, что главный цикл процесса анализа состоит из трех этапов: загрузка интернет-страницы, анализ интернет-страницы и анализ найденных интернет-ссылок. Как правило, запуск программы начинается с инициализации так называемого «главного потока», который координирует при выполнении все действия над выделенными ресурсами. Тогда все действия, которые должна произвести программа, согласно описанной функциональности с точки зрения многопоточного программирования, могут быть представлены на следующей схеме (Рис. 18).



Рис. 18. Однопоточная реализация программного комплекса «Акула»

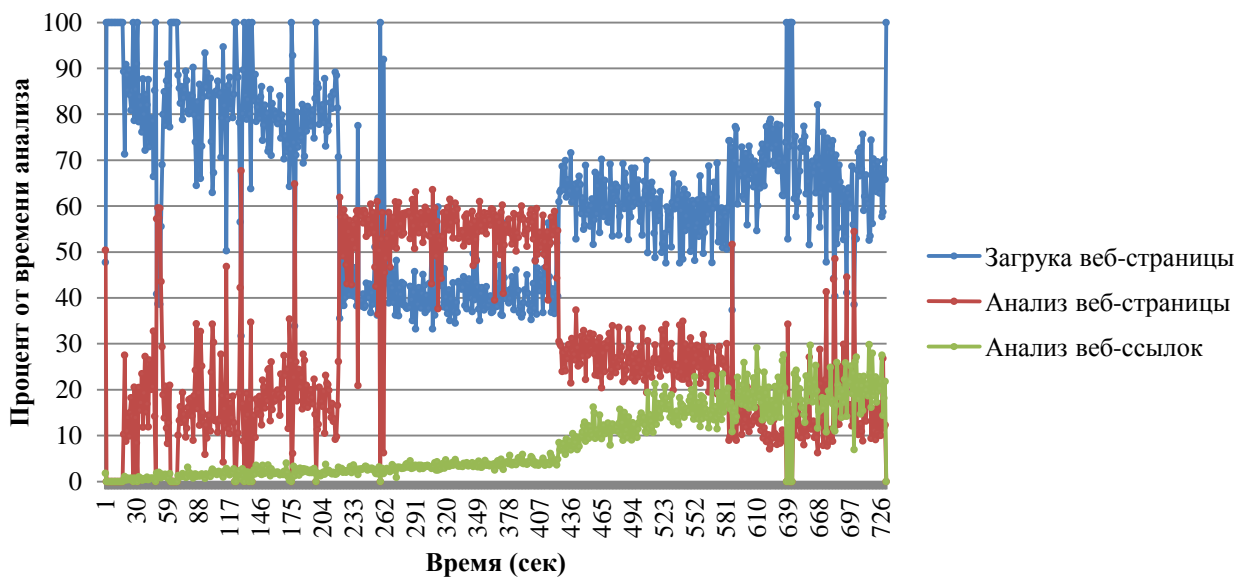


Рис. 19. Соотношение затраченного времени между основными этапами при проведении однопоточного анализа интернет-ресурса

На изображенной схеме (Рис. 18) представлен «главный поток», который имеет начало (инициализацию потока) и конец (завершение потока), а в «теле» потока - представлены его этапы.

Проведенный анализ интернет-ресурса (Рис. 19) показал, что большую часть времени, а именно 63%, занимает загрузка удаленной интернет-страницы на локальный компьютер. Это обусловлено несколькими факторами: во-первых, перед тем как загрузить веб-страницу, программа должна убедиться в ее существовании, таким образом, сначала необходимо отправить «эхо»-запрос, при ответе на который программа должна принять решение о загрузке, на которое уходит часть времени, во-вторых, при отправке «эхо»-запроса, ответ не всегда приходит быстро и программе приходится ждать этого ответа, в результате, србатывает так называемый «time-out», который сигнализирует о превышении лимита ожидания, в-третьих, это несовершенство работы сетевого компонента, т.к. для работы с сетевыми операциями можно либо напрямую использовать Socket API, либо сторонний компонент, который реализует все сетевые функции. Таким образом, возникает потребность в распараллеливании «главного потока» как минимум на два второстепенных потока, а именно загрузка и анализ.

Третий этап – анализ найденных веб-ссылок – заключается в поиске отобранных на веб-странице интернет-ссылок среди уже ранее найденных и загруженных интернет-ссылок веб-ресурса. В начале проведения анализа интернет-ресурса данный этап занимает всего лишь 1% от общего времени исследования веб-страницы и при дальнейшем анализе с увеличением числа найденных и загруженных интернет-ссылок увеличивается до 21% (при этом среднее составляет всего 8%). Поэтому возникает необходимость дополнительного деления второстепенного потока «анализ» на два вторичных потока. Таким образом, исходя из указанных выше особенностей, на каждый этап требуется выделять отдельный поток. Рис. 20 отображает результат представленных рассуждений.



Рис. 20. Предложенный вариант многопоточной реализации

Рис. 20 содержит представление о взаимодействии между главным и второстепенными потоками. Процесс анализа начинается с инициализации трех потоков, каждый из которых является функционально законченным этапом – загрузка исходного текста веб-страницы, анализ исходного текста веб-страницы и анализ отобранных веб-ссылок. Для начала, главный поток передает потоку загрузки исходного текста веб-страницы информацию о начале исследования указанного интернет-ресурса. Поток выполняет загрузку и передает сохраненную веб-страницу

потоку анализа исходного текста веб-страницы. При этом если в текущий момент у данного потока нет больше веб-ссылок для загрузки, то поток приостанавливает свою работу. Поток анализа исходного текста веб-страницы производит анализ, выявляет найденные ошибки и передает следующему потоку выборку из обнаруженных веб-ссылок. Данный поток приостанавливает свою работу, т.к. сохраненных веб-страниц не остается. Наконец, поток анализа веб-ссылок просматривает переданный список веб-ссылок в ранее найденных и загруженных веб-ссылках. Результатом работы данного потока является добавление отобранных веб-ссылок в список уже найденных веб-ссылок.

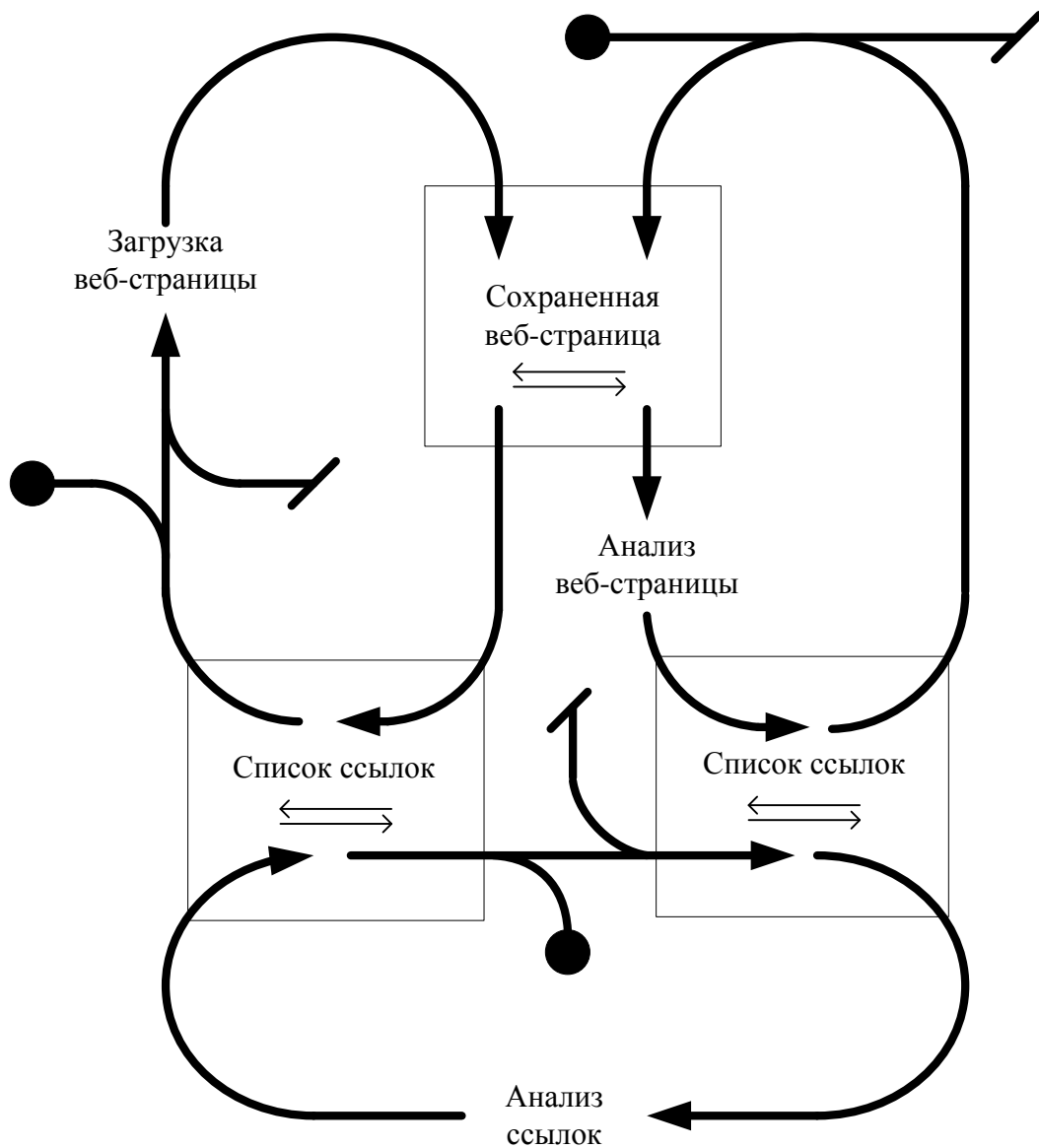


Рис. 21. Общая схема предложенного варианта многопоточной реализации

Рис. 21 отображает дальнейшую работу многопоточной реализации. Поток загрузки исходного текста интернет-страницы из списка найденных ссылок «забирает» ссылку для загрузки, которую загрузив, «передает» потоку для анализа исходного текста. Работа потока продолжается до тех пор, пока список найденных веб-ссылок не закончится. В это же время, поток анализа исходного текста интернет-страницы выполняет свою часть этапа, в результате которого передает список выбранных ссылок на веб-странице потоку анализа отобранных веб-ссылок. Поток проводит анализ до тех пор, пока не проанализированы все загруженные интернет-страницы. Аналогично, поток анализа отобранных веб-ссылок, «получив» очередные найденные ссылки от проведенного анализа над сохраненной веб-страницей проводит процесс сравнения и «передает» данные дальше.

Предложенная реализация помогает значительно повысить быстродействие процесса исследования информационного ресурса. Так, для сравнительно «небольших» веб-ресурсов, время анализа сокращается примерно в 2 раза. Однако такой подход имеет один заметный недостаток – нехватка предоставленных ресурсов для загрузки большого числа ссылок. Такая проблема возникает из-за того, что используемый сетевой компонент может «простаивать» при загрузке интернет-страниц, т.к. веб-страница может быть недоступна, но для определения данного факта необходимо дождаться превышения лимита ожидания. Таким образом, возникает задача оптимизации работы либо потока загрузки исходного кода интернет-страницы, либо оптимизации сетевого компонента.

Второй вариант решения проблемы предполагает, во-первых, быструю загрузку интернет-страницы, что невозможно без существования высокоскоростного доступа в сеть Интернет, а, во-вторых, сокращение времени на гарантированный отклик веб-страницы, что «грозит» ошибками во время загрузки интернет-страницы при плохом качестве канала связи. В результате, оптимизация потока загрузки исходного кода интернет-страницы возможна лишь при распараллеливании самого процесса.

Таким образом, в предложенной ранее общей схеме многопоточной реализации необходимо после получения списка новых отобранных веб-ссылок от по-

тока анализа отобранных ссылок последовательно динамически запускать потоки, каждый из которых будет содержать одну из веб-ссылок. Рис. 22 содержит представленную рекомендованную концепцию.

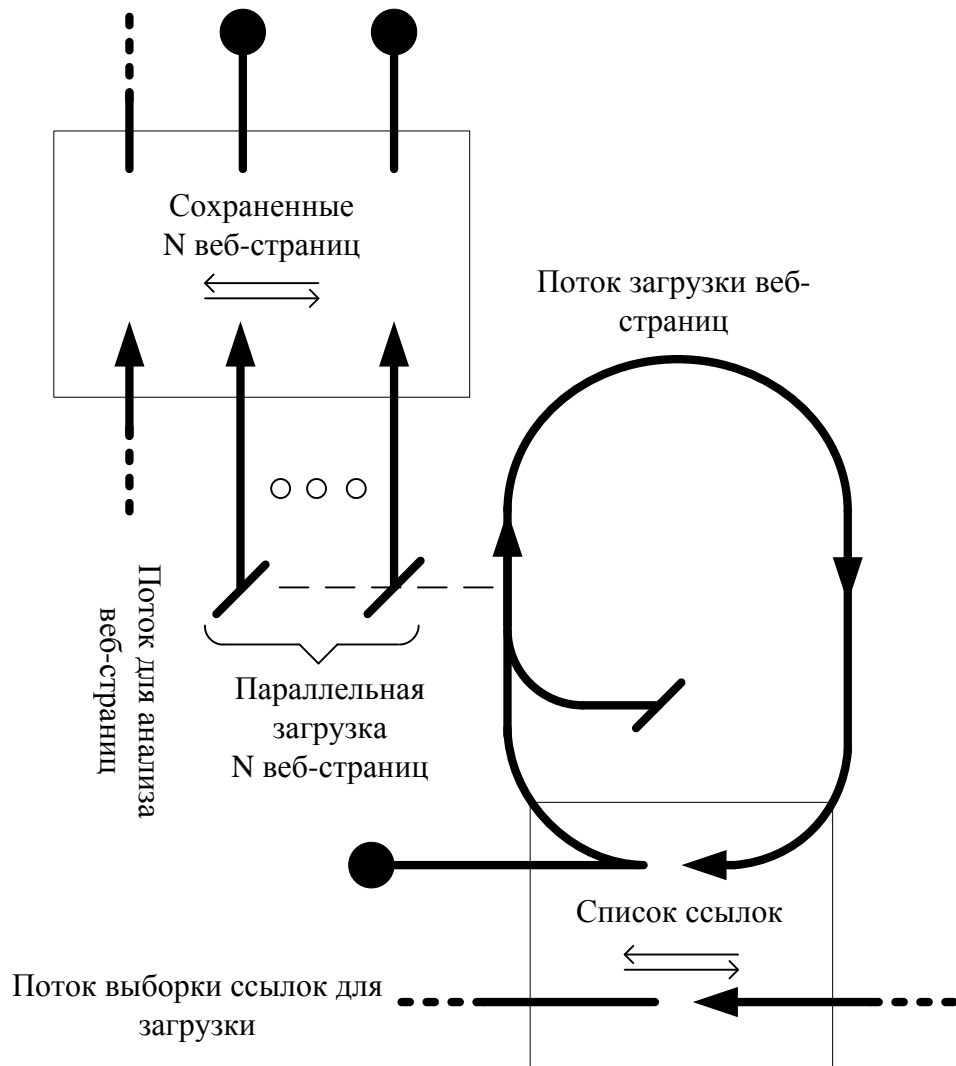


Рис. 22. Модификационный вариант многопоточной реализации

В результате, скорость работы на сравнительно «небольших» ресурсах возрастает дополнительно еще примерно в 1.5 раза. Такой результат удастся получить благодаря переносу нагрузки с «загрузочного блока» на «аналитический», где проводится этап анализа, а это значит, более эффективно используются имеющиеся компьютерные ресурсы.

Однако при проведении исследований информационного ресурса необходимо учитывать критическую нагрузку на него, т.к. при одновременно большом

количестве запросов от пользователя сервер может либо полностью, либо частично их проигнорировать (в таком случае «ответом» от сервера будет код состояния 503 Service Unavailable).

Таким образом, полученная многопоточная реализация из рассуждений выше представляет:

- параллельную загрузку нескольких веб-страниц;
- параллельный анализ веб-страницы;
- параллельный анализ найденных ссылок на веб-странице.

Такой вариант предполагает быструю загрузку всех веб-страниц интернет-ресурса и сравнительно долгий анализ структуры кода.

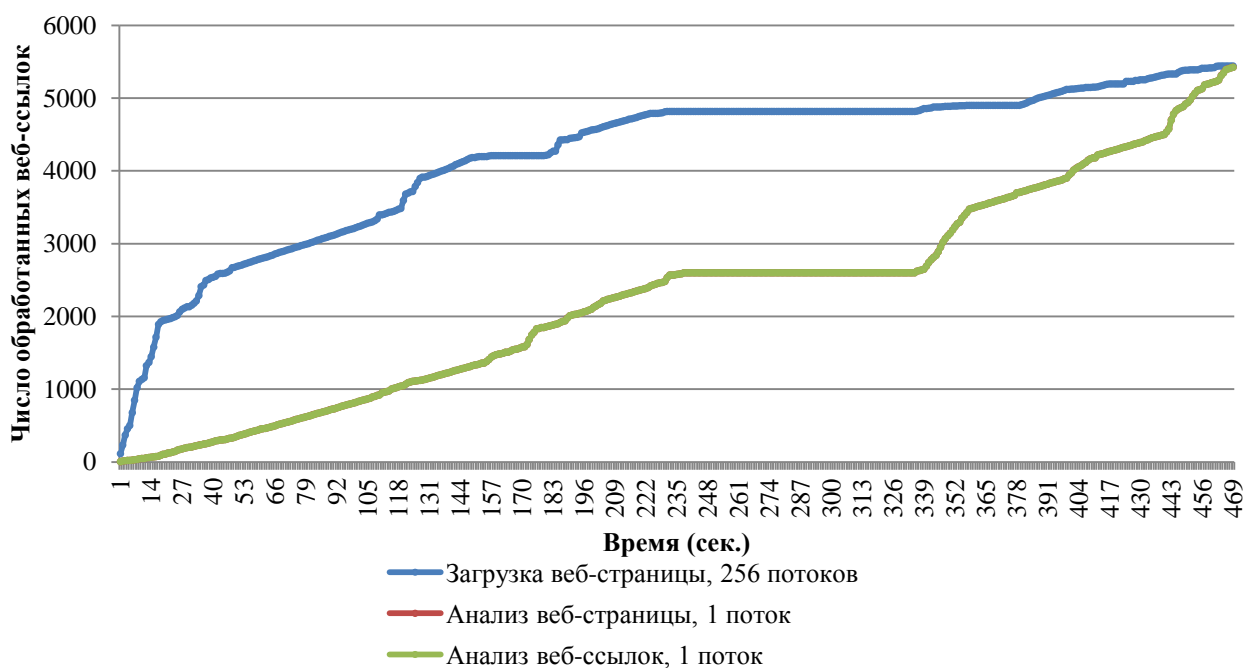


Рис. 23. Многопоточный анализ интернет-ресурса РОССВЯЗЬ²³ (конфигурация D256-A1-S1)²⁴

Так, Рис. 23 отображает почти двукратный отрыв загрузочного блока от аналитического, тем самым создавая очередь не только перед потоком анализа

²³ Тестовый стенд: Core i7-860 (Ядер: 4, Частота: 2.8 ГГц), 12 Gb RAM DDR3, Localhost <http://rossvyaz.ru> (Общий объем: 160.4 Мб, Число ссылок: 5446)

²⁴ D – поток загрузки веб-страниц, A – поток анализа веб-страниц, S – поток анализа веб-ссылок.

веб-страниц, но и перед потоком анализа веб-ссылок. Более того, учитывая тот факт, что потоку анализа веб-страниц необходимо некоторое время для проведения разбора структуры исходного кода (на участке анализа с 238 по 335 секунду происходит анализ сложных веб-страниц), поток анализа веб-ссылок фактически простаивает. Это связано как с синхронной работой с потоком анализа веб-страниц (ожидание очередной порции найденных веб-ссылок), так и со сравнительно меньшей сложностью потока. Таким образом, поток анализа веб-страниц и поток анализа веб-ссылок совпадают по производительности (Рис. 23).

В результате, для увеличения эффективности проведения анализа необходимо как минимум дополнительное распараллеливание потока анализа веб-страницы по аналогичной схеме для потока загрузки веб-страницы (Рис. 22).

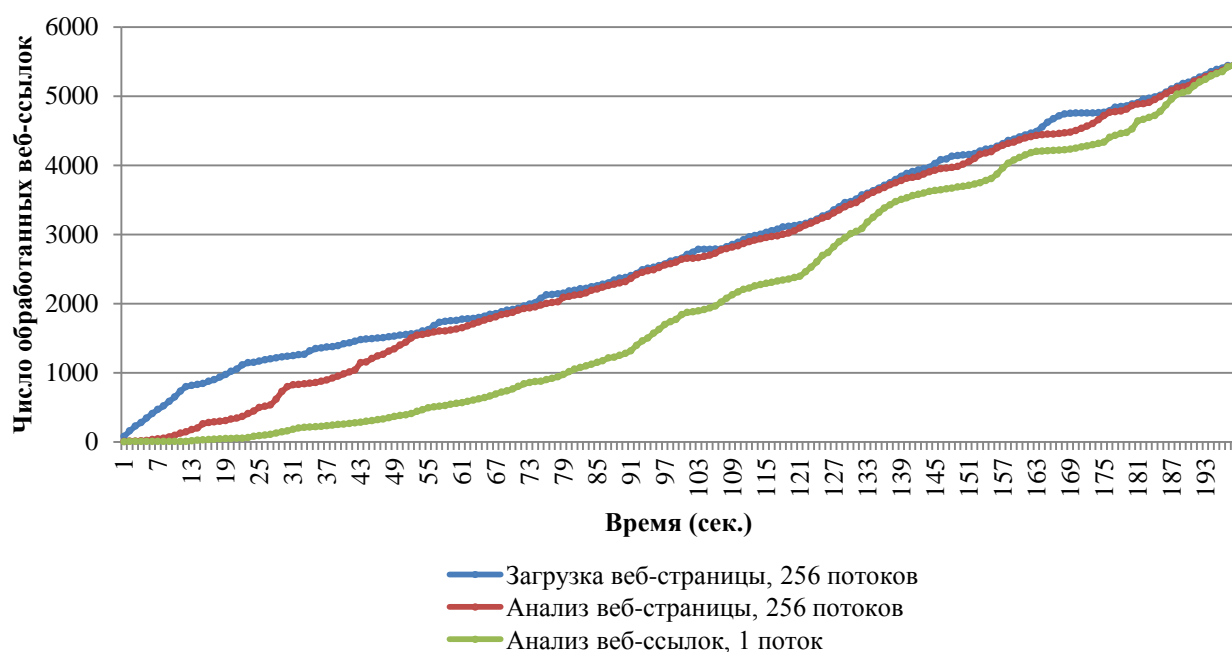


Рис. 24. Многопоточный анализ интернет-ресурса РОССВЯЗЬ (конфигурация D256-A256-S1)

Рис. 24 показывает, что скорость анализа ссылок значительно увеличивается и поток анализа веб-ссылок «практически не отстает» от потока загрузки веб-ссылок. Кроме этого, время проведения анализа сокращается более чем в 2 раза (с 7.8 минут до 3.5 минут). Однако, в таком случае, создается значительная очередь

для следующего потока - потока анализа веб-ссылок, тем самым замедляя работу потока загрузки веб-страниц. Таким образом, применим аналогичный принцип дополнительного распараллеливания потока.

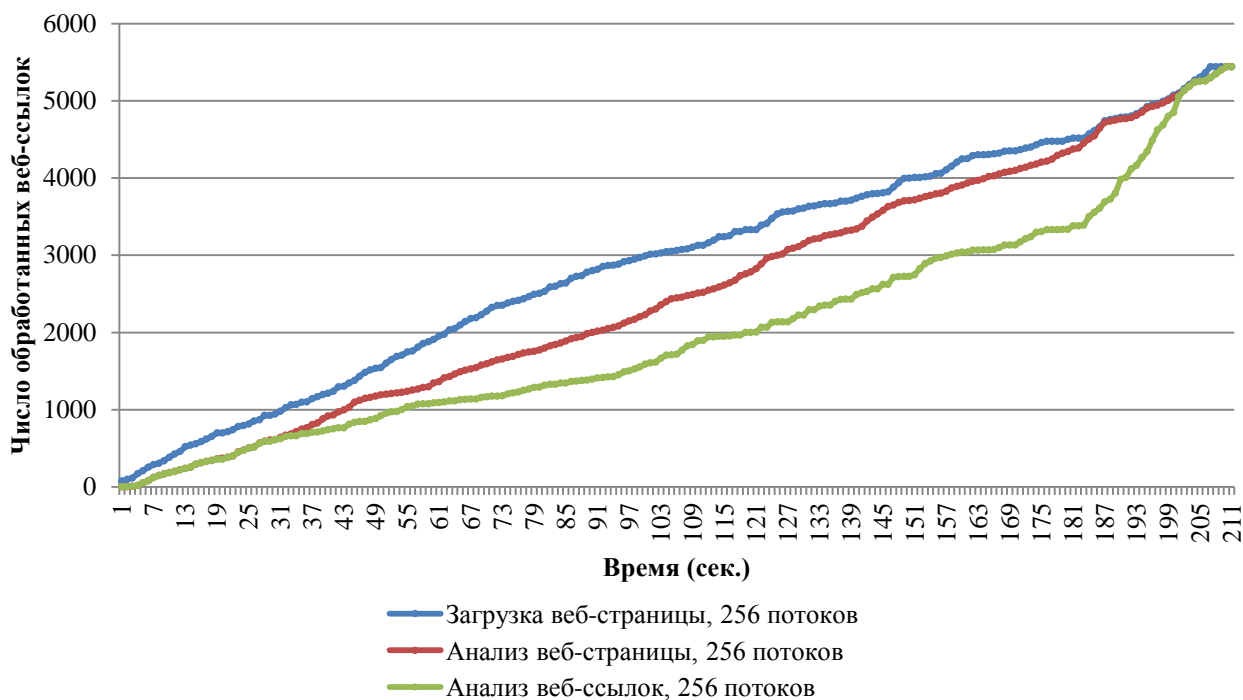


Рис. 25. Многопоточный анализ интернет-ресурса РОССВЯЗЬ (конфигурация D256-A256-S256)

Рис. 25 отображает тот факт, что производительность потока анализа веб-ссылок увеличивается (на участке анализа с 1 по 30 секунду скорость фактически совпадает с потоком анализа веб-страниц в отличие от предыдущего случая). Вместе с этим, происходит дополнительная нагрузка на поток загрузки веб-страниц, который довольно быстро выполняет свою работу, тем самым создавая очередь уже перед потоком анализа веб-страниц. На графике (Рис. 25) это видно из того, что скорости работ потоков довольно четко разделены и у каждого из потоков есть данные для обработки. Однако видно (Рис. 25), что с течением времени скорость потока анализа веб-ссылок замедляется. Учитывая тот факт, что задача данного потока состоит в поиске веб-ссылок среди ранее обнаруженных (обработанных) и еще не загруженных, масштабируемость данного потока заключается в

одновременном увеличении числа запросов к указанным данным и напрямую зависит от множественного быстрого доступа. Исходя из этого, время анализа практически остается неизменным по сравнению с предыдущим случаем. Такой факт говорит о том, что необходимы дополнительные исследования для увеличения производительности и выбор иных средств, обозначенных в параграфе 3.4, а выбранные средства автором в рамках данных исследований эффективны в конфигурации с одним потоком анализа веб-ссылок.

В таком случае, приведенные в параграфе предположения позволяют сделать вывод, что время исследования при проведении анализа интернет-ресурса удастся сократить на тестовом стенде почти до 3.5 раз (с 12 до 3.5 минут) (Рис. 26, Рис. 27, Табл. 9). Такой численный результат зависит не только от произведенных модификаций, но также от доступности компонентов информационного ресурса, вычислительной мощности оборудования и его программной реализации.

В результате проведенных выше исследований можно сделать вывод, что указанные в данном параграфе предложения для эффективного проведения анализа интернет-ресурсов помогают значительно увеличить скорость работы анализатора и сократить время проведения исследования интернет-ресурса.

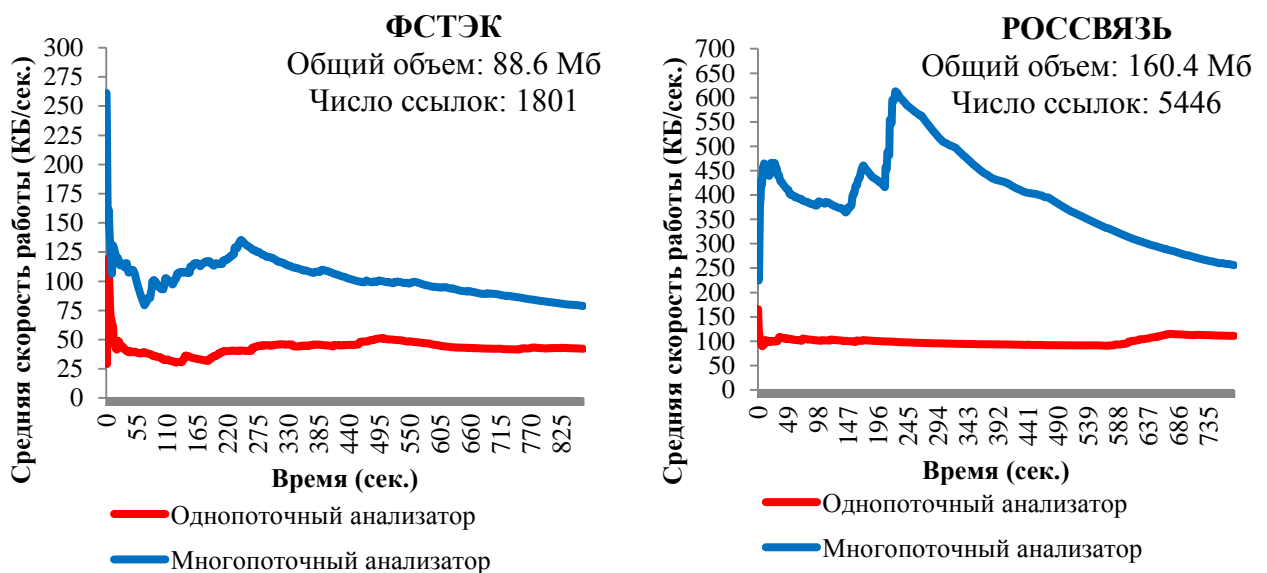


Рис. 26. Сравнение скоростей работы однопоточного и многопоточного анализаторов интернет-ресурса РОССВЯЗЬ и ФСТЭК

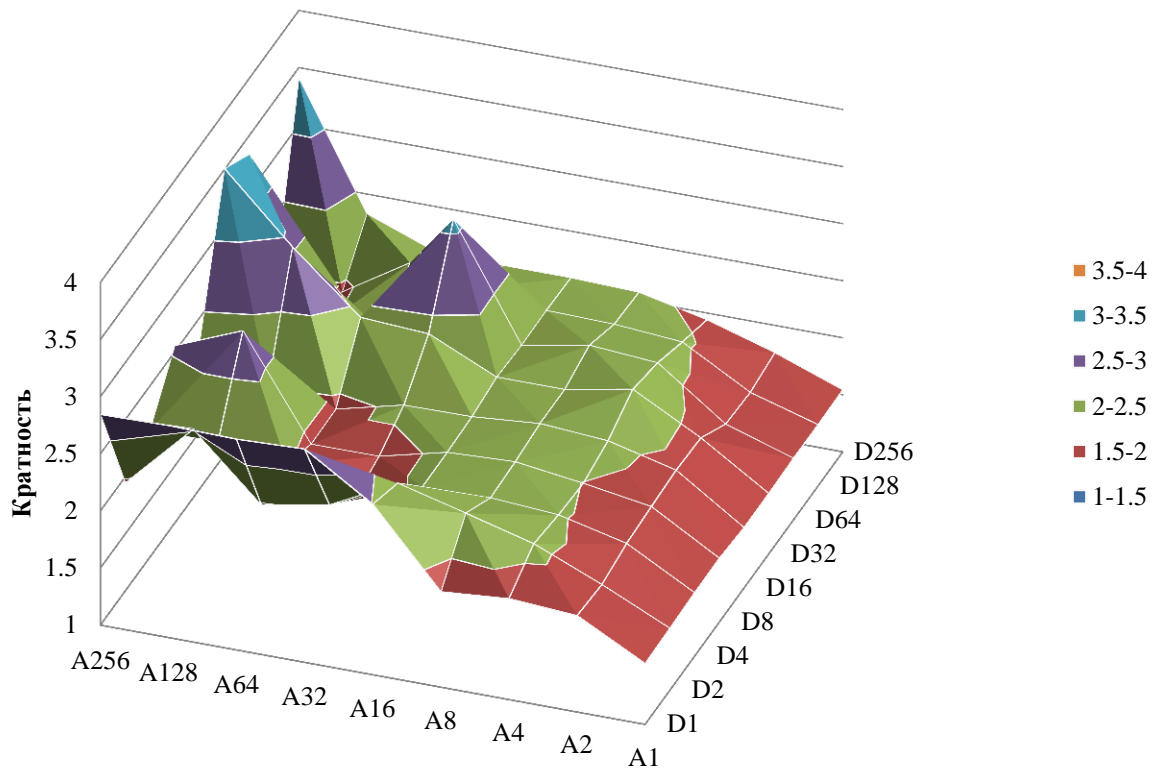


Рис. 27. Сравнение времени проведения анализа интернет-ресурса РОССВЯЗЬ многопоточной и однопоточной реализации (в конфигурации S=1)

Таблица 9. Значения сравнения времени проведения анализа интернет-ресурса РОССВЯЗЬ многопоточной и однопоточной реализации (в конфигурации S=1)

		Число потоков для загрузки веб-страниц (D)								
		256	128	64	32	16	8	4	2	1
Число потоков для анализа веб-старниц (A)	256	3.41	1.877	3.338	3.51	2.292	2.553	2.114	1.949	2.846
	128	2.325	1.938	1.908	2.93	2.001	2.792	2.12	2.528	2.858
	64	2.102	2.303	1.974	2.422	1.914	1.906	2.035	1.975	2.846
	32	2.09	2.198	3.09	2.386	2.058	1.908	1.826	2.084	2.869
	16	2.09	2.102	2.096	2.114	2.066	2.051	2.048	2.378	2.493
	8	2.072	2.09	2.204	2.114	2.114	2.078	2.06	2.225	1.835
	4	1.938	2.042	2.158	2.225	2.072	1.997	2.02	2.066	1.883
	2	1.759	1.709	1.73	1.908	1.923	1.903	1.878	1.817	1.844
	1	1.542	1.535	1.538	1.529	1.529	1.558	1.555	1.542	1.532

Выводы по главе 3

В данной главе:

1. Сформулированы требования к возможностям веб-анализатора.
2. Предложена общая структура веб-анализатора. Описаны и обоснованы ключевые особенности веб-анализатора согласно предъявляемым требованиям.
3. Предложена классификация выявляемых ошибок при анализе информационного ресурса.
4. Предложен и обоснован способ использования параллельных технологий при анализе информационных ресурсов. Предлагаемый подход с использованием параллельных технологий позволяет на пользовательском уровне более эффективно использовать возможности не только процессорной системы, но и существующего канала связи для проведения более быстрого анализа больших информационных ресурсов за обозримое время.
5. Предложен и обоснован способ анализа сверхбольших информационных ресурсов, который позволяет дополнительно увеличить скорость проведения анализа до 3.5 раз.

Глава 4. Прикладные исследования по оценке корректности функционирования информационных ресурсов

4.1. Общие положения при проведении прикладных исследований

Проведение анализа функциональной корректности информационных ресурсов предполагает использование:

1. Положений разработанного методического аппарата для проведения оценки корректности программного кода, описанных в параграфе 2.5;
2. Веб-анализатора для автоматизации прикладных исследований, описанного в главе 3;
3. Критерия оценки корректности функционирования информационного ресурса, предложенного в параграфе 1.7.

Целью проведения оценки корректности функционирования интернет-ресурсов является проверка исходного кода информационных ресурсов на наличие потенциально-опасных ошибок кода с целью выработки рекомендаций по повышению функциональной корректности и приведения их в соответствие с международными стандартами ISO 8879 [21], W3C [4, 52, 130] и ECMA 262 [53].

Значимость полученных результатов определяется предоставлением функционально-корректных электронных информационных услуг в государственных и коммерческих информационных ресурсах путем исключения неоднозначного трактования используемых конструкций.

Испытания информационных ресурсов согласно предложенной методике исследования информационных ресурсов проводились с помощью программно-аппаратного комплекса нового поколения «Анализатор исходных текстов информационных ресурсов «Акула» для оценки корректности функционирования размещенных в сети Интернет информационных ресурсов. Тестирование информационных ресурсов предполагает использование описанных и разработанных автором языковых модулей HTML, CSS и JavaScript в рамках диссертационных иссле-

дований. Некоторые из используемых в тестировании программных модулей находятся в настоящее время на стадии разработки, поэтому полученные результаты работы носят предварительный характер.

В ходе практических исследований проведен анализ следующих параметров, на основании которых можно сделать вывод о функциональной корректности информационного ресурса:

- процент доступных интернет-ссылок информационного ресурса;
- тип, число и плотность найденных ошибок исследованного информационного ресурса

Учитывая указанные выше параметры практических исследований, стоит отдельно отметить, что при проведении анализа информационного ресурса возникает необходимость в активном исследовании каждой найденной на веб-странице ссылки (URL). Причем, при обработке таких ссылок необходимо учитывать следующие возможные ситуации:

- обработка длинных веб-ссылок (определение максимально допустимой ссылки для перехода и занесение ее в список некорректных);
- выявление и обработка веб-ссылок, указывающих на одну и ту же интернет-страницу (например, такая ситуация возможна при обнаружении веб-ссылок с символом «\» в конце).

Первая указанная особенность проявляется при непосредственном исследовании веб-ссылки. Согласно RFC 3986 «Uniform Resource Identifier (URI): Generic Syntax» [153], спецификация не накладывает никаких ограничений на URL, за исключением ограничения в 255 символов при использовании названия хоста (в силу ограничений DNS). Дополнительно, согласно RFC 2616 «Hypertext Transfer Protocol - HTTP/1.1» [154, п.п. 3.2.1], описанная спецификация для URI также не накладывает особых ограничений на длину URL, кроме как о предупреждении обработки URI длиннее 255 символов старыми клиентами и прокси-серверами. Таким образом, теоретически максимальная длина веб-ссылки неограниченна. Однако память компьютера не является бесконечной и каждый конечный пользователь имеет ограниченные ресурсы для обработки информации. Так,

по результатам исследования [63], проведенного в 2006 году оказалось, что при использовании веб-ссылки длиннее, чем 2000 символов возникают проблемы при ее обработке. А в популярном веб-обозревателе Internet Explorer длина ссылки не может быть больше 2083 символов [58]. Иные веб-браузеры не имеют явно документированного ограничения. Для Internet Information Server (IIS) длина веб-ссылки по умолчанию определена атрибутом maxURL со значением 4096 [59]. Кроме этого, согласно проведенным в 2010 году исследованиям, популярная поисковая «машина» Google имеет трудности при обработке ссылок больше чем 1855 символов [60].

Поэтому, хотя спецификации и не ограничивают длину веб-ссылки, на практике очень длинная веб-ссылка является аномалией. Таким образом, интернет-страница, имеющая адрес в сети Интернет больше чем 2000 символом, является, скорее всего, ошибочной. Данная ситуация может возникнуть в следующих случаях:

- система управления содержимым автоматически при работе генерирует веб-ссылку большой длины (ошибка проектирования);
- разработчиком интернет-ресурса была специально внедрена данная аномалия, где интернет-страница имеет веб-ссылку недопустимой длины (ошибка при разработке).

Кроме этого, при выполнении загрузки интернет-ресурса на локальный носитель, длина пути, позволяющего однозначно определить месторасположение файла в системе, должна быть не более 255 символов (ограничение операционной системы).

Таким образом, при разработке веб-анализатора необходимо учитывать не только фактор ограниченности URL, но и фактор ограниченности месторасположения локального файла.

Следующей особенностью является определение однозначного месторасположения интернет-страницы в сети Интернет. Такая проблема возникла ввиду того, что спецификация RFC 1738 Uniform Resource Locators (URL) [61] формально допускает вариант записи URL без конечного символа «/» и URL с конечным

символом «/». Причем обе записи в большинстве случаев подразумевают одну и ту же интернет-страницу (при условии, что интернет-ресурс не настроен специально на предоставление разного контента по этим ссылкам). Такое положение дел не может полностью устраивать SEO-разработчиков или разработчиков поисковых «машин», ведь для поисковой оптимизации желателен определенный вид URL. Однако, учитывая подход спецификации, запрет на использование того или иного варианта не является обоснованным требованием.

Более того, спецификация RFC 1738 содержит следующий ряд фактов:

- URL является абстрактным обозначением месторасположения ресурса [61, п. 2];

- схемы URL можно считать иерархическими, причем элементы иерархии должны быть разделены символом «/» [61, п.п. 2.3];

- некоторые части схемы URL, такие как логин, пароль, порт и даже локальная ссылка внутри ресурса могут быть исключены при определении ссылки на домен (хост), причем символ «/» обязателен только при использовании локальной ссылки ресурса после имени домена [61, п.п. 3.1];

- символ «/» является разделителем для обозначения иерархической структуры URL [61, п.п. 3.2.4];

- в элементах <path> и <searchpart> URL схемы HTTP символы «/», «;» и «?» являются зарезервированными, причем символ «/» может использоваться для определения иерархической структуры [61, п.п. 3.3];

- формальная запись конкретных схем URL подразумевает наличие непустых имен при делении локальной ссылки [61, п. 5].

Последнее требование фактически признает, что запись URL с конечным символом «/» является не совсем логичной, потому как содержит неопределенность для построения иерархии. Однако т.к. следующий уровень является фактически пустым, адресация неявно указывает на предшествующий уровень. В результате, обе ссылки являются аналогичными.

Таким образом, явных причин для отказа обработки обоих видов ссылок не выявлено, а значит, поисковые машины должны уметь предусматривать такой ва-

риант событий. На сегодняшний день, для контроля ошибок индексирования популярными «поисковиками», такими как Google [124], Яндекс [125] и др., создан сервис проверки списка индексируемых страниц и при необходимости их корректировка.

Принимая во внимание данную особенность, при разработке веб-анализатора необходимо учитывать факт вариативной проверки, а именно:

- проводить поиск и обработку ссылок со значащим символом «/» в конце URL (обе ссылки содержат различные веб-страницы);
- проводить поиск и обработку ссылок без символа «/» в конце URL (обе ссылки содержат одну и ту же веб-страницу).

Такое разделение обуславливается тем, что, во втором случае анализ таких ссылок удобно проводить без конечного символа «/». В противном случае необходимо проводить поиск как одной, так и другой ссылки, что увеличивает количество проверок в 2 раза.

Сопутствующей особенностью, косвенно вытекающей из вышеобозначенных некорректностей, является однозначное определение веб-страницы на информационном ресурсе при различных запросах к ней. Такая особенность является аномалией ввиду того, что к одной и той же интернет-странице можно сконфигурировать различные параметры запроса (<searchpart> URL схемы HTTP [61, п.п. 3.3]). Однако при создании интернет-ресурса разработчиками иногда не учитывается тот факт, что совершая «обход» всего информационного ресурса одной из ссылок, сгенерированной CMS, может быть ссылка на ту же интернет-страницу с новым добавленным параметром. Таким образом, провести анализ доступности всех ссылок интернет-ресурса, имея несколько таких аномалий, каждая из которых указывает с новым добавленным параметром друг на друга, фактически не представляется возможным. Так как каждый раз генерируемая ссылка будет становиться все длиннее, пользователь в определенный момент будет пытаться получить доступ к интернет-ресурсу по ссылке, длина которой является недопустимой. В таком случае это создает лишнюю нагрузку на сервер и браузер пользователя и результатом таких действий будет один из возможных сценариев:

1. сервер отправит пользователю ответ 414 Request-URL Too Long;
2. сервер, имея возможность обрабатывать сверхдлинные ссылки, обрабатывает и отправит пользователю запрашиваемую страницу;
3. веб-браузер ошибочно обработает ссылку и не отправит запрос;

Из этого следует, что дополнительно анализатор должен учитывать возможные заикливания на информационном ресурсе.

Помимо этого, необходимо также учитывать обработку и хранение больших объемов данных в памяти. Это связано с тем, что при анализе сверхбольших интернет-ресурсов образуется большой список обрабатываемых веб-ссылок. Таким образом, разрабатываемый веб-анализатор должен уметь правильно организовывать работу с используемой памятью. Для этих целей в компиляторе FPC предусмотрены встроенные утилиты для поиска утечек памяти [56], которые необходимо использовать при разработке.

4.2. Применение разработанного веб-анализатора при эксплуатации информационных ресурсов

Созданный в рамках диссертационной работы Анализатор исходных текстов информационного ресурса «Акула» был зарегистрирован Роспатентом (свидетельство о государственной регистрации программы для ЭВМ № 2015616442) и использован для сопровождения информационных ресурсов, перечисленных в табл. 10, а также внедрен в практическую деятельность управления безопасности администрации г. Фрязино и центра защиты информации ООО «НПЦ «СОТИС», что подтверждается соответствующими актами внедрения. Это позволило выявить и отредактировать у исследуемых информационных ресурсов функционально-некорректные конструкции, и тем самым улучшить их работоспособность.

Таблица 10. Список информационных ресурсов, исследованных разработанным веб-анализатором

Адрес ресурса в сети Интернет	Описание информационного ресурса
http://lkb.srcc.msu.ru/	Официальный информационный ресурс лаборатории Компьютерной безопасности НИВЦ МГУ имени М.В. Ломоносова
http://soyuz1812.ru/	Союз потомков участников бородинской битвы. Историко-патриотическое объединение «Багратион»
http://num-meth.srcc.msu.ru/	Научный журнал «Вычислительные методы и программирование» НИВЦ МГУ имени М.В. Ломоносова
http://srcc.msu.ru/	Официальный информационный ресурс НИВЦ МГУ имени М.В. Ломоносова
http://vmk78.ru/	Информационный ресурс выпускников факультета ВМК МГУ им. М.В. Ломоносова 1978 года выпуска
http://www.fryazino.org/	Официальный информационный ресурс наукограда Фрязино
http://sotis-ltd.com/	Официальный информационный ресурс ООО «Научно-производственный центр «СОТИС»
http://nalog.ru	Официальный информационный ресурс Федеральной налоговой службы
http://www.ssec.ru/	Официальный информационный ресурс Группы компаний «Центр Специальной Системотехники»

Кроме этого, полученные в диссертационной работе результаты были интегрированы в многоуровневую информационно-аналитическую Систему регистрации, анализа и мониторинга событий информационной безопасности (т.н. Система РАМС ИБ), разработанную группой компаний «Центр Специальной Системотехники»

техники» для обеспечения защищенности автоматизированных информационных систем как небольших организаций, так и распределенных систем федерального масштаба [155]. Анализатор исходных текстов «Акула» был использован как один из датчиков-сенсоров Системы РАМС ИБ с целью предоставления информации о состоянии объекта защиты при проведении аудита интернет-ресурса. Помимо покрытия требований нормативных документов и рекомендаций для безопасного функционирования информационных ресурсов, совместно с другими датчиками, анализатор исходных текстов «Акула» предоставляет дополнительные данные для использования в алгоритмах мониторинга состояния защищенности ресурса. Взаимодействие с системой РАМС ИБ осуществляется с помощью журнала аппаратных и программных событий Microsoft Windows. Данное решение может применяться как на тестовом сегменте для оценки функциональной корректности вносимых изменений в его структуру или данных веб-ресурса перед применением этих изменений на общедоступном сегменте, так и на общедоступном сегменте интернет-ресурса для контроля действующей версии сайта, а также использовано для сопровождения проектов построения систем защиты информационных ресурсов веб-порталов. Описанное внедрение подтверждено актом ООО «ЦСС-Сервис».

Созданное совместное проектное решение нашло свое применение в Федеральной Налоговой Службы России (ФНС России) в рамках работ по обнаружению и предотвращению компьютерных атак на информационные ресурсы официального сайта, что позволило при проведении расследований инцидентов информационной безопасности существенно расширить функционал ведомственного центра и комплексно подойти к контролю сервисов портала. Анализатор исходных текстов «Акула» был использован в качестве «Агента мониторинга доступности и качества сервисов» в составе Системы РАМС ИБ (рис. 28). В настоящее время уже завершён этап опытной эксплуатации и готовится проектное решение по внедрению, что подтверждается актом ФНС России.

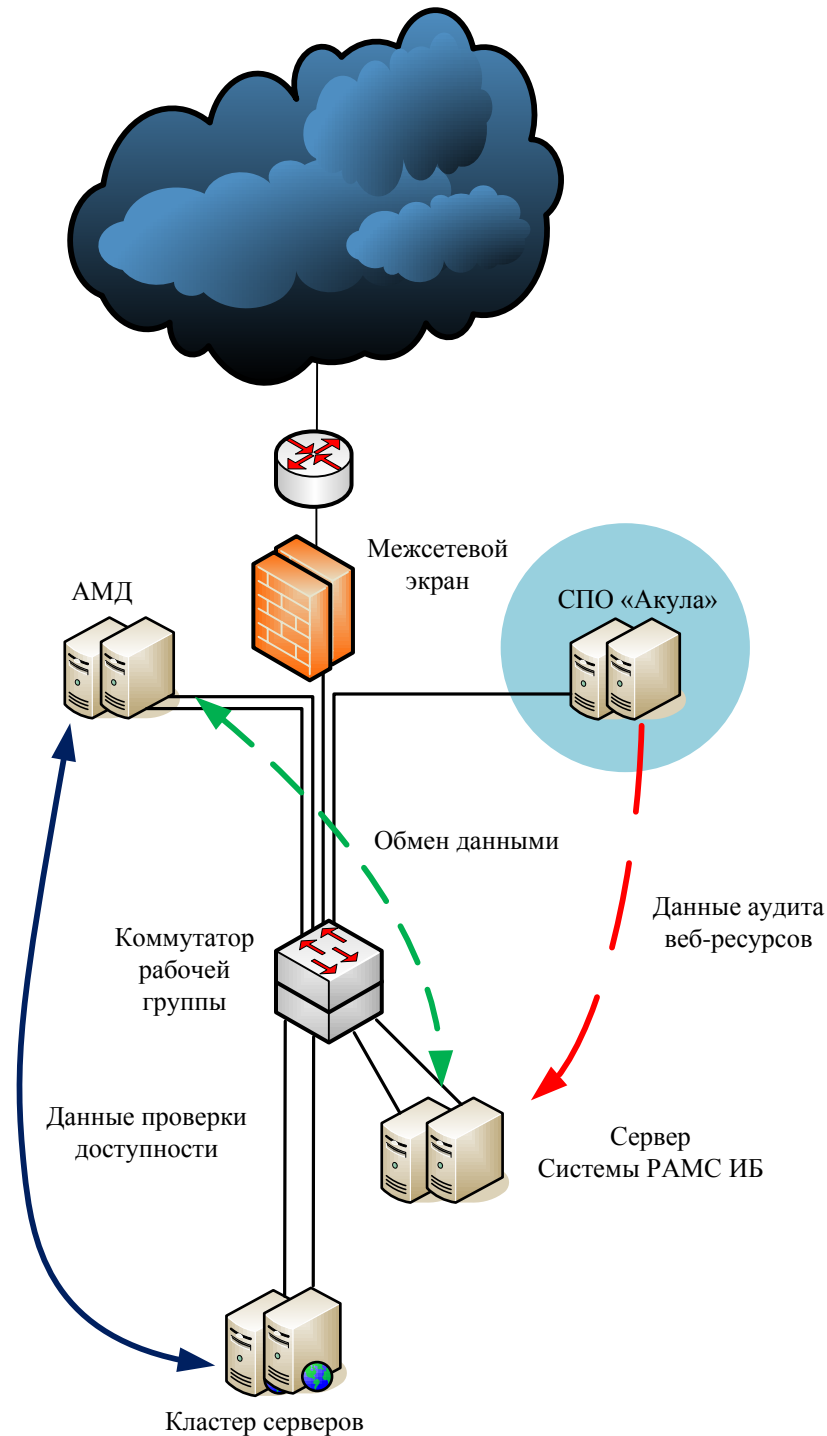


Рис. 28. Схема взаимодействия Анализатор исходных текстов «Акула» в составе Системы РАМС ИБ при проведении этапа опытной эксплуатации

Более того, по результатам диссертационной работы и результатам проведенного анализа нормативной документации и функционирования АИС «Мониторинг государственных ресурсов» подготовленного проектное решение по внедрению Анализатор исходных текстов «Акула» для расширения возможностей

определения технического рейтинга государственных информационных ресурсов как одного из компонентов получения информации об интернет-ресурсе (рис. 29).

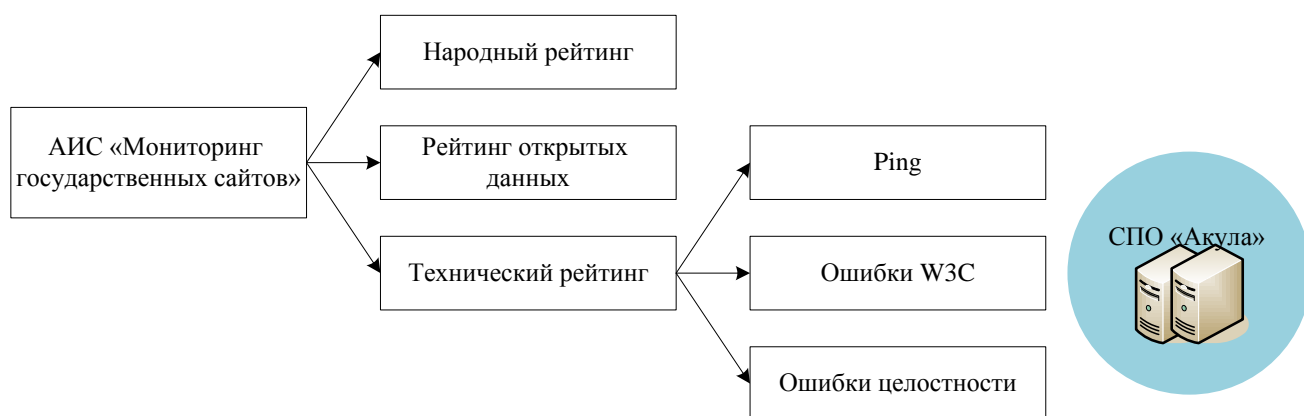


Рис. 29. Проектное решение по внедрению в АИС «Мониторинг государственных сайтов»

Проведенные прикладные исследования позволили сформулировать **рекомендации для повышения функциональной корректности информационных ресурсов:**

1. Составляющие информационного ресурса, соответствующие требованиям, предъявляемые ISO 8879, W3C и ECMA 262 на отсутствие недеklarированных конструкций, межтеговых программных вставок и некорректно заданных значений атрибутов и свойств позволит получить кроссбраузерный код.

2. Составляющие информационного ресурса, имеющие правильно-построенную структуру HTML документа HTML DOM, позволяет обеспечить функционально-корректное восприятие кода браузером.

3. Информационный ресурс, не содержащий ссылки на один и тот же материал под разными именами, позволит строго структурировать материал, что даст возможность различным поисковым системам точно определить запрашиваемый материал.

4. Информационный ресурс, не содержащий сверхдлинные ссылки, позволит исключить некорректную работу веб-обозревателя (обеспечить гарантированное предоставление информации пользователю).

Выводы по главе 4

В данной главе:

1. Предложена и обоснована методика проведения прикладных исследований по корректности функционирования информационных ресурсов.
2. Определены и детально рассмотрены ключевые особенности анализа веб-ресурсов, возникающие при проведении исследований по корректности функционирования информационных ресурсов.
3. Сформулированы предложения для повышения корректности функционирования информационных ресурсов.

Заключение

В диссертационной работе поставлена и решена актуальная научная задача по разработке методики оценки корректности функционирования информационных ресурсов для проведения анализа и выявления критических ошибок исходного кода составных частей информационного ресурса.

В процессе решения данной задачи получены следующие новые наиболее значимые научные результаты:

1. На основании анализа особенностей построения интернет-ресурсов и различных исследований, активно проводимых мировым сообществом в области обеспечения корректности их функционирования (с учетом факторов, оказывающих наибольшее влияние на информационные ресурсы, принятых ограничений и допущений) позволило усовершенствовать автоматизированное выявление ошибок в работе информационных ресурсов и сформировать: критерий для оценки корректности функционирования информационных ресурсов; сформулировать смысловую и формализованную постановку задачи исследования и предложить методическую схему её решения.

2. На основе принципов построения интернет-страниц и исследования функциональных связей разработаны новые алгоритмы проведения анализа исходных текстов информационных ресурсов, что позволяет, с учетом рекомендаций международных интернет-стандартов, строго формализовать синтаксис интернет-страниц и выявить функционально-некорректные конструкции.

3. Разработана новая методика оценки корректности функционирования информационного ресурса. Предложенный методический подход, в отличие от существующих, позволяет получить достоверную оценку корректности функционирования всего информационного ресурса, т.к. предполагает проведение анализа не только всей его структуры и выявление особенностей взаимодействия составляющих его элементов между собой, но и предоставляет возможность прогнозировать динамику изменений корректности функционирования информационного ресурса во времени.

4. Для автоматизированного проведения этапов прикладных исследований исходных текстов информационного ресурса на основании разработанной новой методики и алгоритмов анализа реализован программный комплекс «Анализатор исходных текстов информационного ресурса «Акула», который позволяет проводить оценку корректности функционирования сверхбольших информационных ресурсов за приемлемое для анализа время.

5. Полученные экспериментальные результаты по оценке корректности функционирования информационных ресурсов подтвердили объективность выбранного критерия оценки и определили высокую практическую значимость проведенной работы. Это позволяет использовать разработанную методику для широкого применения при разработке и эксплуатации как государственных, так и коммерческих информационных ресурсов для повышения их функциональности и выяснения причин их некорректной работы. Разработанный программный комплекс «Анализатор исходных текстов информационного ресурса «Акула» даст возможность не только существенно уменьшить потенциальный вред пользователю информационного ресурса, но и выйти исследованиям в этой области на качественно новый уровень развития и определить перспективные направления исследований в таких областях как:

- решение актуальных фундаментальных задач в области кибербезопасности, участие в формировании стратегии национальной кибербезопасности Российской Федерации;

- моделирование на базе суперкомпьютеров МГУ компьютерных инцидентов для исследования критических ситуаций с целью выявления аномалий, формирование методов расследования, выдача рекомендаций по блокированию киберугроз;

- формирование методов оперативного обнаружения и эффективного отражения компьютерных атак, обеспечения безопасности критически важных информационных систем, разработка методов координации и управления деятельности субъектов критической информационной инфраструктуры (система поддержки принятия решения);

- комплексное обучение специалистов по обнаружению, предупреждению и ликвидации последствий компьютерных атак на критические объекты информационной инфраструктуры;
- поддержка государственной системы обнаружения, предупреждения и ликвидации последствий компьютерных атак на информационные ресурсы Российской Федерации;
- разработка методов защиты национальных протоколов каналов управления критических объектов с использованием индустриального и промышленного интернета;
- разработка перспективных средств защиты информации от компьютерных атак;
- формирование методов блокирования сбора неструктурированных публичных данных, промышленной и экономической компьютерной разведки информационных ресурсов Российской Федерации;
- разработка методов визуализации представления данных в области информационной безопасности;
- создание отечественных анализаторов исходных текстов нового поколения для проведения сертификации средств защиты информации по требованиям безопасности информации.

Литература

1. The WorldWideWeb browser [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/People/Berners-Lee/WorldWideWeb.html>, свободный.
2. Методика мониторинга официальных сайтов органов государственной власти и местного самоуправления [Электронный ресурс] // Автоматизированная информационная система «Мониторинг государственных сайтов» [Официальный сайт]. Режим доступа: <http://gosmonitor.ru/sites/default/files/pages/65/227.doc>, свободный.
3. АИС «Мониторинг государственных сайтов» [Электронный ресурс] // МЭР РФ [Официальный сайт]. Режим доступа: <https://gosmonitor.ru/>, свободный.
4. Standard W3C: HTML5 A vocabulary and associated APIs for HTML and XHTML. [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/html5/>, свободный.
5. Биячуев, Т. А. Модель и методы мониторинга и оценки защищенности веб-сайтов сети Интернет: дис. канд. тех. наук: 05.13.19 / Т. А. Биячуев. — Санкт-Петербург, 2005. — 120 с.
6. Проценко, Е. А. Модель и метод анализа эффективности систем защиты информации сайтов органов власти Российской Федерации: дис. канд. тех. наук: 05.13.19 / Е. А. Проценко. — Санкт-Петербург, 2008. — 150 с.
7. Павлютенков, А. А. Модель и метод логического контроля использования стандартов информационной безопасности в критически важных системах информационно-телекоммуникационной инфраструктуры: дис. канд. тех. наук: 05.13.19 / А. А. Павлютенков. — Санкт-Петербург, 2009. — 107 с.
8. Политов, М. С. Экспериментально-аналитический метод оценки и прогнозирования уровня защищенности информационных систем на основе модели временных рядов: дис. канд. тех. наук: 05.13.19 / М. С. Политов. —

- Уфа, 2010. — 143 с.
9. Шаньгин, В. Ф. Защита информации в компьютерных системах и сетях / В. Ф. Шаньгин. — М: ДМК Пресс, 2012. — 592 с.
 10. Марков А. С. Методы оценки несоответствия средств защиты информации / А. С. Марков, В. Л. Цирлов, А. В. Барабанов. — М: Радио и связь, 2012. — 192 с.
 11. Мельников, Д. А. Организация и обеспечение безопасности информационно-технологических сетей и систем / Д. А. Мельников — IDO PRESS, 2012. — 597 с.
 12. Шелухин, О. И. Обнаружение вторжений в компьютерные сети (сетевые аномалии) / О. И. Шелухин, Д. Ж. Сакалема, А. С. Филинова. — Горячая линия - Телеком, 2013. — 220 с.
 13. Ерохин, В. В. Безопасность информационных систем: учебное пособие / В. В. Ерохин, Д. А. Погоньшева, И. Г. Степченко. — ФЛИНТА: Наука, 2015. — 183 с.
 14. Stuttard, D., Pinto, M. The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws, Second Edition ed. / Dafydd Stuttard, Marcus Pinto. — Wiley, 2011. — 878 p.
 15. Shema, M. Hacking Web Apps: Detecting and Preventing Web Application Security Problems / M. Shema. — Elsevier, 2012. — 296 p.
 16. Canavan, T. CMS Security Handbook: The Comprehensive Guide for WordPress, Joomla!, Drupal, and Plone / T. Canavan. — Wiley, 2011. — 432 p.
 17. Purewal, S. Learning Web App Development / S. Purewal. — O'Reilly, 2014. — 286 p.
 18. Eilers, Carsten. HTML5 Security / Carsten Eilers. — Entwickler.press, 2012. — 97 p.
 19. Vacca, John. Network and System Security / John Vacca. — Elsevier, 2010. — 405 p.

20. Davidoff, S., Ham, J. Network Forensics - Tracking Hackers through Cyberspace / S. Davidoff, J. Ham. — Prentice Hall, 2012. — 574 p.
21. International Organization for Standardization. Information Processing: Text and Office Systems: Standard Generalized Markup Language (SGML). – ISO, 1986.
22. Cygwin [Официальный сайт]. Режим доступа: <https://www.cygwin.com/>, свободный.
23. CrossOver [Электронный ресурс] // Компания CodeWeavers [Официальный сайт]. Режим доступа: <https://www.codeweavers.com/products/>, свободный.
24. WineHQ [Официальный сайт]. Режим доступа: <https://www.winehq.org/>, свободный.
25. ФЗ РФ от 27.07.2006 №149 (ред. от 21.07.2014) «Об информации, информационных технологиях и о защите информации».
26. IBM Security AppScan [Электронный ресурс] // Компания IBM [Официальный сайт]. Режим доступа: <http://www-03.ibm.com/software/products/en/appscan>, свободный.
27. Acunetix Web Vulnerability Scanner. Brochure. [Электронный ресурс] // Компания Acunetix [Официальный сайт]. Режим доступа: <http://www.acunetix.com/resources/wvsbrochure.pdf>, свободный.
28. NTOSpider. Data sheet [Электронный ресурс] // Компания NT Objectives [Официальный сайт]. Режим доступа: <http://www.ntobjectives.com/files/data-sheets/NTOSpider-Data-Sheet.pdf>, свободный.
29. Netsparker. Data sheet [Электронный ресурс] // Компания Netsparker [Официальный сайт]. Режим доступа: <https://www.netsparker.com/s/Netsparker-ProductBrochure.pdf>, свободный.
30. HP WebInspect. Data sheet [Электронный ресурс] // Компания HP [Официальный сайт]. Режим доступа: <http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA1-5363ENW&cc=us&lc=en>, свободный.
31. PT Application Inspector. Описание продукта [Электронный ресурс] // Исследовательский центр Positive Research [Официальный сайт]. Режим доступа: http://www.ptsecurity.ru/appsecurity/application-inspector/PT_Application_

Inspector_rus.pdf, свободный.

32. SkipFish [Электронный ресурс] // Google Project Hosting [Офиц. сайт]. Режим доступа: <http://code.google.com/p/skipfish/>, свободный.
33. W3C Validator Suite (shareware) [Электронный ресурс] // W3C [Офиц. сайт]. Режим доступа: <https://validator-suite.w3.org/>, свободный.
34. Быстрицкий, Н. Д., Макаров-Землянский, Н. В. Анализ web-приложений / Н. Д. Быстрицкий, Н. В. Макаров-Землянский // Естественные и технические науки. — 2013. — №5(67). — С. 294-295.
35. Быстрицкий, Н. Д., Макаров-Землянский, Н. В. Функционирование анализатора web-ресурсов / Н. Д. Быстрицкий, Н. В. Макаров-Землянский // Естественные и технические науки. — 2013. — №6(68). — С. 295-296.
36. Быстрицкий, Н. Д. Актуальные проблемы исследования веб-ресурсов / Н. Д. Быстрицкий // Современное состояние естественных и технических наук: материалы XV международной научно-практической конференции. — 2014. — С. 44-47.
37. Быстрицкий, Н. Д. Проблемы безопасного использования пользователем веб-приложений / Н. Д. Быстрицкий // Актуальные проблемы современной науки. — 2014. — №5(78). — С. 142-145.
38. Быстрицкий, Н. Д., Макаров-Землянский, Н. В. Необходимые требования для обеспечения безопасности функционирования интернет-ресурса / Н. Д. Быстрицкий, Н. В. Макаров-Землянский // Актуальные проблемы современной науки. — 2014. — №6(79). — С. 239-242.
39. Быстрицкий, Н. Д., Мартьянов, Е. А. Получение оценки защищенности веб-ресурсов / Н. Д. Быстрицкий, Е. А. Мартьянов // Аспирант и соискатель. — 2014. — № 6(84). — С. 81-83.
40. Быстрицкий, Н. Д. Исследование защищенности региональных и муниципальных информационных ресурсов / Н. Д. Быстрицкий // Техника и технология: новые перспективы развития: материалы XVI

- международной научно-практической конференции. — 2015. — С. 55-64.
41. Быстрицкий, Н. Д. Алгоритм анализа интернет-страниц информационного ресурса / Н. Д. Быстрицкий // *Фундаментальные исследования*. — 2015. — №6-3. — С. 443-446.
 42. Быстрицкий, Н. Д. Проведение анализа информационных ресурсов с использованием параллельных технологий / Н. Д. Быстрицкий // *Успехи современной науки*. — 2016. — Т. 8. — № 12. — С. 182-187.
 43. Firefox. Узнайте об основах: с чего начать [Электронный ресурс] // Поддержка Mozilla [Официальный сайт]. Режим доступа: <https://support.mozilla.org/ru/products/firefox/get-started>, свободный.
 44. Firefox. Появляется сообщение об ошибке «Это соединение является недоверенным» - Что делать [Электронный ресурс] // Поддержка Mozilla [Официальный сайт]. Режим доступа: <https://support.mozilla.org/ru/kb/poyavlyaetsya-soobshenie-ob-oshibke-eto-soedinenie>, свободный.
 45. Firefox. Как работают встроенная Защита от Фишинга и Вредоносных программ? [Электронный ресурс] // Поддержка Mozilla [Официальный сайт]. Режим доступа: <https://support.mozilla.org/ru/kb/kak-rabotayut-vstroennye-fishing-i-zashita-ot-vred>, свободный.
 46. Брандмауэр Windows [Электронный ресурс] // Семейство операционных систем Microsoft Windows [Официальный сайт]. Режим доступа: <http://windows.microsoft.com/ru-ru/windows7/products/features/windows-firewall>, свободный.
 47. Информационный лист продуктовой линейки Outpost 9 [Электронный ресурс] // Компания Agnitum [Официальный сайт]. Режим доступа: http://dl2.agnitum.com/docs/Features_Outpost9_RU.pdf, свободный.
 48. Outpost Firewall Pro. Руководство пользователя [Электронный ресурс] // Компания Agnitum [Официальный сайт]. Режим доступа: http://dl2.agnitum.com/docs/firewall/OFP_User_Guide_RU.pdf, свободный.
 49. Документация семейства антивирусов Dr.Web [Электронный ресурс] //

- Компания «Доктор Веб» [Официальный сайт]. Режим доступа: <http://download.drweb.com/doc>, свободный.
50. Руководящий документ 45.129-2000. Телематические службы.
 51. Постановление Правительства РФ от 18.02.2005 №87 «Об утверждении перечня наименований услуг связи, вносимых в лицензии, и перечней лицензионных условий».
 52. Standard W3C: Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/CSS21/>, свободный.
 53. Standard ECMA-262: ECMAScript Language Specification. Edition 5.1 [Электронный ресурс] // Ассоциация Ecma International [Официальный сайт]. Режим доступа: <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>, свободный.
 54. Top 20 countries with the highest number of internet users [Электронный ресурс] // Internet world stats [Официальный сайт]. Режим доступа: <http://www.internetworldstats.com/top20.htm>, свободный.
 55. March 2017 Web Server Survey [Электронный ресурс] // Netcraft Ltd. [Официальный сайт]. Режим доступа: <https://news.netcraft.com/archives/2017/03/24/march-2017-web-server-survey.html>, свободный.
 56. Профилирование [Электронный ресурс] // Lazarus and Free Pascal wiki [Официальный сайт]. Режим доступа: <http://wiki.lazarus.freepascal.org/profiling/ru>, свободный.
 57. ФЗ РФ от 25.07.2011 №261 «О внесении изменений в федеральный закон «О персональных данных».
 58. Maximum URL length is 2,083 characters in Internet Explorer [Электронный ресурс] // Microsoft Support [Официальный сайт]. Режим доступа: <http://support2.microsoft.com/default.aspx?scid=KB;en-us;q208427>, свободный.

59. Request Limits [Электронный ресурс] // IIS [Официальный сайт]. Режим доступа: <http://www.iis.net/configreference/system.webserver/security/requestfiltering/requestlimits>, свободный.
60. What is the Maximum Character Length for a URL that Google Will Index? [Электронный ресурс] // SEOMOFO [Официальный сайт]. Режим доступа: <http://www.seomof.com/experiments/>, свободный.
61. RFC 1738 Uniform Resource Locators (URL) [Электронный ресурс] // IETF [Официальный сайт]. Режим доступа: <http://tools.ietf.org/html/rfc1738>, свободный.
62. W3C Validator [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://validator.w3.org>, <http://validator.w3.org/nu/>, свободный.
63. WWW FAQs: What is the maximum length of a URL? [Электронный ресурс] // Boutell.Com [Официальный сайт]. Режим доступа: <http://www.boutell.com/newfaq/misc/urllength.html>, свободный.
64. Grandison, T. Security and privacy in web 2.0 / T. Grandison // IEEE Internet Computing. — 2014. — Vol. 18. — Iss. 6. — PP. 41-42. — Article number 6939612.
65. Manoharan, A., Carrizales, T. J. Recent trends in e-government: states' and local governments' utilisation of websites / Aroon Manoharan, Tony J. Carrizales // Int. J. of Electronic Governance. — 2011. — Vol. 4. — No. 4. — PP. 283- 303.
66. Choi, J.-H. On cyberattack mechanisms / Jae-Hyuk Choi, Ok-Ran Jeong, Woo-Jin Han, Chulyun Kim, Won Kim // Int. J. of Web and Grid Services. — 2013. — Vol. 9. — No. 4. — PP. 351-368.
67. Hofmann, A., Ramaj H. Interdependent risk networks: the threat of cyber attack / Annette Hofmann, Hidajet Ramaj // Int. J. of Management and Decision Making. — 2011. — Vol.11, No.5/6. — PP.312–323.
68. Остапенко, Г. А. Оценка защищенности информационно-телекоммуникационных систем, подвергающихся DDoS-атакам / Г.А. Остапенко, М.В. Бурса, Н.И. Баранников, И.Л. Батаронов // Информация и безо-

- пасность. — 2013. — Т. 16. — № 4. — С. 496-497.
69. Бурса, М. В., Пастернак, Ю. Г. DDoS-атаки на информационно-телекоммуникационные системы: управление рисками / М. В. Бурса, Ю. Г. Пастернак // Информация и безопасность. — 2013. — Т. 16. — № 2. — С. 255-256.
70. Борисов, В. И. Вероятностные аналитические модели сетевой атаки с внедрением вредоносного программного обеспечения / В.И. Борисов, Н.М. Радько, А. А. Голозубов, И. Л. Батаронов, Е. В. Ермилов // Информация и безопасность. — 2013. — Т. 16. — № 1. — С. 5-30.
71. Al-Qudah, Z. DDoS protection as a service: hiding behind the giants / Zakaria Al-Qudah, Basheer Al-Duwairi, Osama Al-Khaleel // Int. J. of Computational Science and Engineering. — 2014. — Vol.9. — No.4. — PP. 292-300.
72. Nam, S.Y., Djuraev, S. Defending HTTP web servers against DDoS attacks through busy period-based attack flow detection / S.Y. Nam, S. Djuraev// KSII Transactions on Internet and Information Systems. — 2014. — Vol. 8. — Iss. 7. — PP. 2512-2531.
73. Doyal, A., Zhan, J. Triple DoS: DDoS defence and traceback / Alex Doyal, Justin Zhan // Int. J. of Information Privacy, Security and Integrity. — 2013. — Vol.1. — No.4. — PP.299-311.
74. Hidayanto, A. N. How secure your applications are? Analysis of web developers awareness to application security / Achmad Nizar Hidayanto, Rinaldi, Putu Wuri Handayani, Samuel Louvan // Int. J. of Innovation and Learning. — 2013. — Vol.14. — No.1. — PP.53-78.
75. Куликов, С. С., Белоножкин В. И. Исследование характеристик уязвимостей информационно-телекоммуникационных систем / С. С. Куликов, Бело-ножкин В.И. // Информация и безопасность. — 2013. — Т. 16, № 2. — С. 257-258.
76. Статистика уязвимостей веб-приложений 2012 [Электронный ресурс] //

Исследовательский центр Positive Research [Официальный сайт]. Режим доступа: http://ptsecurity.ru/download/analitika_web.pdf, свободный.

77. Dharmendra, C. Developing secure web applications / Dharmendra Choukse, Dimitris N. Kanellopoulos, Umesh Kumar Singh // *Int. J. of Internet Technology and Secured Transactions*. — 2012. — Vol. 4. — No. 2/3. — PP.221-236.
78. Shteiman, B. Why CMS platforms are breeding security vulnerabilities / B. Shteiman // *Network Security*. — 2014. — Vol. 2014. — Iss. 1. — PP. 7-9.
79. Choi, M., Kim, N. Design and implementation of electronic authentication for web contents / M. Choi, N. Kim // *Contemporary Engineering Sciences*. — 2014. — Vol. 7. — Iss. 13-16. — PP. 691-697.
80. Al-Sakib Khan, P., Diallo Abdoulaye, K. Lethality of SQL injection against current and future internet technologies / Pathan Al-Sakib Khan, Kindy Diallo Abdoulaye // *Int. J. of Computational Science and Engineering*. — 2014. — Vol. 9. — No. 4. — PP.386-394.
81. Cecchini, S., Gan D. SQL injection attacks with the AMPA suite / Simone Cecchini, Diane Gan. // *Int. J. of Electronic Security and Digital Forensics*. — 2013. — Vol.5. — No.2. — PP.139-160.
82. Jang, Y.-S., Choi, J.-Y. Detecting SQL injection attacks using query result size / Y.-S. Jang, J.-Y. Choi // *Computers and Security*. — 2014. — Vol. 44. — PP. 104-118.
83. Shanmuganeethi, V., Praveen, Ra. Yagna, Swamynathan, S. CIVD: detection of command injection vulnerabilities in web services through aspect-oriented programming / V. Shanmuganeethi, Ra. Yagna Praveen, S. Swamynathan // *Int. J. of Computer Applications in Technology*. — 2012. — Vol.44. — No.4. — PP. 312 – 320.
84. Salas, M.I.P., Martins, E. Security testing methodology for vulnerabilities detection of XSS in web services and WS-security / M.I.P. Salas, E. Martins // *Electronic Notes in Theoretical Computer Science*. — 2014. — Vol. 302. — PP.

- 133-154.
85. Kern, C. Preventing script injection vulnerabilities through software design / C. Kern // *Communications of the ACM*. — 2014. — Vol. 57. — Iss. 9. — PP. 38-47.
 86. Liu, Q., Wen, T., Wen, G. Detection of XSS vulnerabilities in online Flash / Q. Liu, T. Wen, G. Wen // *Jisuanji Yanjiu yu Fazhan/Computer Research and Development*. — 2014.— Vol. 51. — Iss. 7. — PP. 1624-1632.
 87. Fonseca, J. Analysis of field data on web security vulnerabilities / J. Fonseca, N. Seixas, M. Vieira, H. Madeira // *IEEE Transactions on Dependable and Secure Computing*. — 2014. — Vol 11. — Iss. 2. — PP. 89-100. — Article number 6589556.
 88. Shahriar, H. Server-side code injection attack detection based on Kullback-Leibler distance / Hossain Shahriar, Sarah M. North, YoonJi Lee, Roger Hu // *Int. J. of Internet Technology and Secured Transactions*. — 2014. — Vol.5, No.3. — PP. 240-261.
 89. SriNithi, D. Improving web application security using penetration testing / D. SriNithi, G. Elavarasi, T.F. Michael Raj, P. Sivaprakasam // *Research Journal of Applied Sciences, Engineering and Technology*. — 2014. — Vol. 8. — Iss. 5. — PP. 658-663.
 90. Rautila, M., Suomalainen, J. Secure inspection of web transactions / Mika Rautila, Jani Suomalainen // *Int. J. of Internet Technology and Secured Transactions*. — 2012. — Vol.4. — No.4. — PP. 253-271.
 91. Хейн, А.А., Щукин, Б.А. Обеспечение информационной безопасности при взаимодействии с веб-сервисом / А.А. Хейн, Б.А. Щукин // *Безопасность информационных технологий*. — 2012. — № 1. — С. 124-127.
 92. Любченков, А.В., Юрасов, В.Г. Особенности взаимодействия владельцев информационных ресурсов при передаче конфиденциальной информации / А.В. Любченков, В.Г. Юрасов // *Информация и безопасность*. — 2013. — Т.

16. — № 2. — С. 185-190.
93. De Groef, W., Devriese, D., Nikiforakis, N., Piessens, F. Secure multi-execution of web scripts: Theory and practice / W. De Groef, D. Devriese, N. Nikiforakis, F. Piessens // Journal of Computer Security. — 2014. — Vol. 22. — Iss. 4. — PP. 469-509.
94. Shahriar, H. Effective detection of vulnerable and malicious browser extensions / H. Shahriar, K. Weldemariam, M. Zulkernine, T. Lutellier// Computers and Security. — 2014. — Vol. 47. — PP. 66-84.
95. Razzaq, A., Anwar, Z., Ahmad, H.F., Latif, K., Munir, F. Ontology for attack detection: An intelligent approach to web application security / A. Razzaq, Z. Anwar, H.F. Ahmad, K. Latif, F. Munir // Computers and Security. — 2014. — Vol. 45. — PP. 124-146.
96. Губарева О.Ю., Пугин В.В. Современные методики, применяемые для оценки угроз и уязвимостей информационных систем / О.Ю. Губарева, В.В. Пугин // Инфокоммуникационные технологии. — 2013. — Т. 11. — № 1. — С. 100-105.
97. ГОСТ Р ИСО/МЭК 15408 «Информационная технология. Методы и средства обеспечения безопасности. Критерий оценки безопасности информационных технологий».
98. Руководящий документ «Безопасность информационных технологий. Положение по разработке профилей защиты и заданий по безопасности». Гостехкомиссия России, 2003 г.
99. Zalewski M. The Tangled Web: A Guide to Securing Modern Web Applications / Michal Zalewski. — William Pollock, 2012. — 302 p.
100. ГОСТ Р ИСО/МЭК 18045 «Информационная технология. Методы и средства обеспечения безопасности. Методология оценки безопасности информационных технологий».
101. ГОСТ Р ИСО/МЭК 15408-3 «Информационная технология. Методы и

средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Компоненты доверия к безопасности».

102. Аналитика. Статьи специалистов исследовательского центра Positive Research [Электронный ресурс] // Исследовательский центр Positive Research [Официальный сайт]. Режим доступа: <http://ptsecurity.ru/lab/analytics/>, свободный.
103. Acunetix: Web Application Security Blog [Электронный ресурс] // Компания Acunetix [Официальный сайт]. Режим доступа: <http://www.acunetix.com/blog/>, свободный.
104. Web Application Security Blog by Netsparker Web Security Experts [Электронный ресурс] // Компания Netsparker [Официальный сайт]. Режим доступа: <https://www.netsparker.com/blog/>, свободный.
105. Краткая брошюра о XSSpider 7.8 [Электронный ресурс] // Исследовательский центр Positive Research [Официальный сайт]. Режим доступа: http://ptsecurity.ru/files/XSpider_7.8.pdf, свободный.
106. WAVSEP 2013/2014 Score Chart: The Web Application Vulnerability Scanners Benchmark [Электронный ресурс] // Security Tools Benchmarking [Официальный сайт]. Режим доступа: <http://sectooladdict.blogspot.ro/2014/02/wavsep-web-application-scanner.html>, свободный.
107. Price and Feature Comparison of Web Application Scanners (Unified List of Commercial, Free and Open Source Products) [Электронный ресурс]. Режим доступа: <http://sectoolmarket.com/price-and-feature-comparison-of-web-application-scanners-unified-list.html>, свободный.
108. WIVET [Электронный ресурс] // Google Project Hosting [Официальный сайт]. Режим доступа: <http://code.google.com/p/wivet/>, свободный.
109. WAVSEP [Электронный ресурс] // Google Project Hosting [Официальный сайт]. Режим доступа: <http://code.google.com/p/wavsep/>, свободный.

110. W3AF [Официальный сайт]. Режим доступа: <http://w3af.org/>, свободный.
111. Arachnid [Официальный сайт]. Режим доступа: <http://www.arachni-scanner.com/>, свободный.
112. Wapiti [Официальный сайт]. Режим доступа: <http://wapiti.sourceforge.net/>, свободный.
113. Sqlmap [Официальный сайт]. Режим доступа: <http://sqlmap.org/>, свободный.
114. Zed Attack Proxy Project [Электронный ресурс] // The Open Web Application Security Project (OWASP) [Официальный сайт]. Режим доступа: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project, свободный.
115. Burp Suite [Электронный ресурс] // Компания PortSwigger [Официальный сайт]. Режим доступа: <http://www.portswigger.net/burp/>, свободный.
116. Burp Suite Help [Электронный ресурс] // Команда Hack4Sec [Официальный сайт]. Режим доступа: <http://h4s-team.ru/bs/index/index.html>, свободный.
117. Paros Proxy [Официальный сайт]. Режим доступа: <http://www.parosproxy.org/index.shtml>, свободный.
118. IronWASP [Официальный сайт]. Режим доступа: <https://ironwasp.org/>, свободный.
119. Blue Coat. Описание продукции [Электронный ресурс] // Компания WebControl [Официальный сайт]. Режим доступа: http://web-control.ru/d/271962/d/bc_product_description_v2.pdf, свободный.
120. W3C Link Checker [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://validator.w3.org/checklink/>, свободный.
121. Firebug [Официальный сайт]. Режим доступа: <https://getfirebug.com/>, свободный.
122. Использование средств разработчика F12 [Электронный ресурс] // Центр разработки Internet Explorer [Официальный сайт]. Режим доступа: <http://msdn.microsoft.com/ru-RU/library/ie/bg182326%28v=vs.85%29>, свободный.

123. Static Analysis Technologies Evaluation Criteria [Электронный ресурс] // The Web Application Security Consortium [Офиц. сайт]. Режим доступа: <http://projects.webappsec.org/w/page/66094278/Static%20Analysis%20Technologies%20Evaluation%20Criteria>, свободный.
124. Инструменты для вебмастера [Электронный ресурс] // Google [Офиц. сайт]. Режим доступа: <http://www.google.ru/webmasters/>, свободный.
125. Яндекс.Вебмастер [Электронный ресурс] // Яндекс [Офиц. сайт]. Режим доступа: <http://help.yandex.ru/webmaster/service/what-is-webmaster.xml>, свободный.
126. Безкорвайный, М. М., Костогрызов, А. И., Львов, В. М. Инструментально-моделирующий комплекс для оценки качества функционирования информационных систем «КОК»: Руководство системного аналитика. — М.: Вооружение. Политика. Конверсия. 2002. — с. 305., 2-е издание.
127. Neufelder, Ann Marie. Current defect destiny statistics. Режим доступа: <http://www.softrel.com/Current%20defect%20density%20statistics.pdf>, свободный.
128. Standard W3C: HTML 4.01 Specification [Электронный ресурс] // W3C [Офиц. сайт]. Режим доступа: <http://www.w3.org/TR/html401/>, свободный.
129. Standard W3C: XHTML 1.0 The Extensible HyperText Markup Language (Second Edition). A Reformulation of HTML 4 in XML 1.0 [Электронный ресурс] // W3C [Офиц. сайт]. Режим доступа: <http://www.w3.org/TR/xhtml1/>, свободный.
130. Standard W3C: XHTML 1.1 - Module-based XHTML - Second Edition. [Электронный ресурс] // W3C [Офиц. сайт]. Режим доступа: <http://www.w3.org/TR/xhtml11/>, свободный.
131. Справочник по HTML и CSS [Электронный ресурс]. Режим доступа: <http://htmlbook.ru>, свободный.
132. Standard W3C: Extensible Markup Language (XML) 1.1 (Second Edition).

- [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/2006/REC-xml11-20060816/>, свободный.
133. The Extensible Stylesheet Language Family (XSL) [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/Style/XSL/>, свободный.
134. Standard W3C: XSL Transformations (XSLT) Version 2.0 [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/xslt20/>, свободный.
135. Standard W3C: Extensible Stylesheet Language (XSL) Version 1.1 [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/xsl/>, свободный.
136. Standard W3C: XML Path Language (XPath) 3.0 [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/xpath-30/>, свободный.
137. Standard W3C: XQuery 1.0: An XML Query Language (Second Edition) [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/xquery/>, свободный.
138. XPath JS Library [Официальный сайт]. Режим доступа: <http://js-xpath.sourceforge.net/>, свободный.
139. Майэр, Э. CSS-каскадные таблицы стилей. Подробное руководство., 3е изд. / Майэр, Э. — СПб: Символ-Плюс, 2008. — 576 с.
140. Standard W3C: Selectors Level 3. W3C Recommendation 29 September 2011 [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/TR/css3-selectors/>, свободный.
141. Флэнаган, Д. JavaScript. Подробное руководство / Д. Флэнаган. — СПб: Символ-Плюс, 2008. — 960 с.
142. Крокфорд, Д. JavaScript: Сильные стороны / Д. Крокфорд — СПб: Питер, 2012. — 176 с.

143. JavaScript [Электронный ресурс] // Mozilla Developer Network [Официальный сайт]. Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>, свободный.
144. Microsoft JScript [Электронный ресурс] // Microsoft Developer Network [Официальный сайт]. Режим доступа: <http://msdn.microsoft.com/ru-ru/library/vstudio/72bd815a%28v=vs.100%29.aspx>, свободный.
145. What does your user agent claim to support? [Электронный ресурс] // W3C [Официальный сайт]. Режим доступа: <http://www.w3.org/2003/02/06-dom-support.html>, свободный.
146. Lazarus [Официальный сайт]. Режим доступа: <http://lazarus.freepascal.org/>, свободный.
147. Free Pascal [Официальный сайт]. Режим доступа: <http://freepascal.org/>, свободный.
148. Indy Project [Официальный сайт]. Режим доступа: <http://www.indyproject.org/>, свободный.
149. ICS [Электронный ресурс] // OverByte [Официальный сайт]. Режим доступа: http://www.overbyte.be/frame_index.html?redirTo=/products/ics.html, свободный.
150. Synapse [Официальный сайт]. Режим доступа: <http://synapse.ararat.cz/doku.php>, свободный.
151. Лукин, В. Н., Чернышов, Л. Н. Вычислительные процессы: теория трансляций, управление данными и сети Петри / В. Н. Лукин, Л. Н. Чернышов. — М.: Вузовская книга, 2015. — с. 180.
152. Кнут, Д. Искусство программирования. Том 2: Получисленные алгоритмы / Д. Кнут — Вильямс, 2001. — 788 с.
153. RFC 3986. Uniform Resource Identifier (URI): Generic Syntax [Электронный ресурс] // IETF [Официальный сайт]. Режим доступа: <http://www.ietf.org/rfc/rfc3986.txt>, свободный.
154. RFC 2616. Hypertext Transfer Protocol -- HTTP/1.1 [Электронный ресурс] //

IETF [Офиц. сайт]. Режим доступа: <https://www.ietf.org/rfc/rfc2616.txt>, свободный.

155. РАМС ИБ [Электронный ресурс] // Центр Специальной Системотехники [Офиц. сайт]. Режим доступа: <http://www.ssec.ru/2014-rams-ib.htm>, свободный.

