1. Необходимо выбрать место для хранения протоколов работы системы, им может быть файловая система или база данных.
2. Далее следует эти данные обрабатывать, сортировать по разным параметрам, находить веб-сайты в верхних 100\300\1000 строчках рейтинга, определять долю целевых пользователей в общем трафике. Далее будет использована парадигма MapReduce и искусственные алгоритмы вида:

- фильтрации;
- оптимизации: соединения «map-join» с использованием распределенного кэша;
- оптимизации: применение «combiner», как одно из требований реализации «reducer»;
- соединения «reduce-join», к примеру, для поиска рейтинговых сайтов;

3. Хотя данное заключение верно не только для Big Data, но также и для большого объёма данных становится крайне критично, для этого необходимо построить некий процесс предобработки данных, с помощью машинного обучения.

Машинное обучение

Первый этап работы – информационная очистка и предобработка данных:

- работа с аномальнымизначениями;
- удалить без информативные данные, такие как: технические посещения и т.д;
- нормализоватьURL-адреса;
- работа с пропущенными, при трассировке, значениями.

Второй этап – превращение данных, в понятные для модели векторы.

На этом этапе существует множество подходов, но нет однозначной методики их применения. Было выделено несколько общепринятых подходов для решения этой задач:

- проанализировать домен и сгруппировать веб-сайты по нему;
- скачать описательную часть страницы и проанализировать ее;
- скачать весь код страницы и постараться выявить тематику страницы;
- связать посещения пользователем сайтов в цепочку и определить логику в последовательности переходов.

Последний этап – применение машинного обучения.

Для применения машинного обучения нету определенного подхода, это наиболее творческий вариант, требующий применения здравого смысла и умения итерационно улучшать выбранное решение.

Список использованных источников:
Real-WorldMachineLearning. Henrik Brink, Joseph W. Richards.

КОНТРОЛЬ КАЧЕСТВА РАЗРАБОТКИ ПРОГРАММНЫХ СРЕДСТВ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Новицкий А.А.
Бахтизин В.В. – профессор каф. ПОИТ, к.т.н., доцент

Работы по контролю качества разработки востребованы в течение всего жизненного цикла программных средств (ПС). Выбор оптимальных для проекта работ по контролю качества (КК) отвечает целям повышения надежности итогового продукта, уменьшения издержек, повышения рентабельности процесса разработки и является актуальной темой. Повсеместное внедрение программных продуктов и рост сложности используемых технологий и ПС становится причиной роста цены ошибки в продукте. Поэтому не теряет актуальности вопрос дальнейшей оптимизации процессов контроля качества (КК) разработки программных средств.

В данном докладе рассматривается классификация тестирования в зависимости от необходимости выполнения программного кода. Это позволяет четко выделить как минимум одну пригодную для анализа группу работ. По данному критерию методы тестирования делятся на статические, не требующие запуска программного кода, и динамические, непосредственно взаимодействующие с функционирующим приложением или его моделью.

Статические виды тестирования включают работы по верификации требований и анализу программного кода, вследствие чего могут обеспечивать наиболее раннее выявление различных проблем, потому обязательны к включению в план разработки [1].

Динамические методы тестирования в основном обеспечивают обнаружение дефектов функциональности в ходе проверки тестовых сценариев или же отвечают задачам проверки устойчивости системы к превышению пределов нормального функционирования, закрытости от влияния извне. Подходы к динамическому тестированию наиболее разнообразны, что создает дополнительные трудности в выборе работ для полного неизбыточного покрытия функций разрабатываемого ПС.

Независимо от наличия автоматизации процесса динамического тестирования общим вариантом критерия оценки покрытия является покрытие, основанное на спецификации или требованиях. Главное требование состоит в покрытии некоторого утвержденного минимума – набора требований. Разница в механизме доступа обуславливает специфику проведения тестирования, предпочтительного для каждого из методов тестирования. Таким образом, рассматривать следует критерии эффективности ручного и

автоматизированного тестирования.

Механизм планирования и оценки покрытия разрабатываемого ПС тестированием может быть выбран в соответствии с потребностями системы и имеющейся документации. С учетом этого выбор применяемых методов и техник тестирования строго не регламентируется. План, методы и техники тестирования могут определяться командой на этапе планирования, исходя из субъективных мнений, предпочтений, доступных ресурсов [3]. Таким образом, задача формирования полного неизбыточного плана работ по КК разработки, пригодного для внедрения в проекты, процессы которых выстроены с использованием популярных моделей и методологий разработки на базе некоторых систем планирования процессов, может быть рассмотрена с аналитической точки зрения.

Отдельно при разработке в каждый момент времени стоят следующие задачи:

- оценка количества ошибок, оставшихся в проекте, и их критичность;
- оценка количества ошибок, оставшихся в выбранном компоненте, и их критичность;
- оценка времени, за которое текущая версия станет стабильной, т.е. количество ошибок и их критичность в данной версии будут меньше заданного порога.

Для решения данных задач можно использовать два подхода – изучение исходного кода программного продукта и применение моделей оценки надежности (МОН) ПС. Использование МОН позволяет посредством построения вероятностной модели случайных процессов и использования различных статистических методов получить оценки различных метрик (ожидаемое число ошибок, вероятность ошибки).

В докладе рассматривается возможность использования техник вероятностного (статистического) тестирования с целью создания репрезентативного сценария тестирования. В качестве некоторого критерия оценки эффективности использования данных техник тестирования можно принять достижение надежностью некоторого приемлемого для конкретного проекта значения [4].

На основе полученного сценария и располагая некоторыми данными для анализа с места планируемого внедрения плана работ, предлагается более предметно рассчитывать рентабельность автоматизации работ в процессе тестирования с учётом планируемой к использованию модели/методологии разработки с точки зрения затрат финансов и времени [2].

Списокиспользованныхисточников:

1. Royce, W. W. Managing the Development of Large Software Systems / Winston W. Royce // Article / Proceedings of IEEE WESCON 26 : Article / The Institute of Electrical and Electronics Engineers, Inc. – 1970.
2. Extreme Programming Explained: Embrace Change, 2nd Edition: Book / Addison-Wesley. – NY, 2004.
3. Липаев, В. В. Человеческие факторы в программной инженерии: рекомендации и требования к профессиональной квалификации специалистов : Учебник / В. В. Липаев. – М.: Синтег, 2009. - 328 с.
4. Sayre, K. Improved Techniques for Software Testing Based on Markov Chain Usage Models: Dissertation / K. Sayre. – Knoxville: The University of Tennessee, 1999. –128c.

ОПТИМИЗАЦИЯ АВТОМАТИЗИРОВАННОГО ОБНАРУЖЕНИЯ УЯЗВИМОСТЕЙ К SQL-ИНЪЕКЦИЯМ В WEB-ПРИЛОЖЕНИЯХ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Оношко Д.Е.

Бахтизин В.В. – к.т.н., доцент

Широкое распространение web-приложений как способа реализации программных средств, в сочетании с их доступностью неограниченному кругу пользователей предъявляет повышенные требования к качеству web-приложений. По данным OWASP [1], по состоянию на 2017 г. среди уязвимостей web-приложений наиболее распространёнными остаются SQL-инъекции. Обнаружение таких уязвимостей вручную является трудоёмким процессом, что создаёт необходимость в разработке методов их автоматизированного обнаружения.

Предложенная в [2] модель обнаружения уязвимостей предполагает выделение в web-приложении процедур, параметры каждой из которых в дальнейшем подвергаются оценке. Однако, несмотря на относительную простоту модели, ввиду значительного объёма исходных кодов, присущего современным web-приложениям, при реализации метода обнаружения уязвимостей на основе этой модели возникает необходимость оптимизации процесса назначения оценок.

Очевидным способом оптимизации является исключение из рассмотрения тех частей web-приложения, которые не оказывают влияния на результаты обнаружения уязвимостей. В основу такой оптимизации предлагается положить разделение модели на 2 уровня: внутрипроцедурный и межпроцедурный.

Межпроцедурный уровень модели предлагается представить в виде графа зависимостей, отражающего характер взаимосвязей между отдельными процедурами web-приложения. Упрощённый пример такого графа представлен на рисунке 1.