УДК 004.4

СИСТЕМНЫЕ ПОДХОДЫ К УПРАВЛЕНИЮ ПРОИЗВОДИТЕЛЬНОСТЬЮ ВЕБ-ПРИЛОЖЕНИЙ

Сытник А. А., *аспирант*
Черкасский государственный технологический университет,
бул. Шевченко, 460, Черкассы, Украина,
e-mail: sytnik.anton@gmail.com

Аннотация. *В данной статье рассмотрены основные типы мониторинга веб-приложений. Описаны популярные программные продукты управления производительностью веб-приложений с их преимуществами и недостатками. Выделены задачи, которые актуальны с научной точки зрения с целью повышения эффективности и качества управления производительностью веб-приложений.*

Ключевые слова: *мониторинг, управление производительностью веб-приложений.*

SYSTEM APPROACHES TO WEB APPLICATIONS PERFORMANCE MANAGEMENT

Sytnyk A. O., *postgraduate*
Cherkasy State Technological University,
Shevchenko blvd, 460, Cherkasy, Ukraine,
e-mail: sytnik.anton@gmail.com

Abstract. *In the article the main types of web application monitoring are described. Popular software products for web applications performance management with their advantages and disadvantages are considered. Also the main tasks which are actual from scientific point of view to improve the efficiency and quality of web applications performance management have been allocated.*

Keywords: *monitoring, web applications performance management (APM).*

Введение. Приложения, работающие в сети Интернет в соответствии со стандартами и соглашениями всемирной паутины, получили общее название веб-приложений (ВП), в которых клиентом выступает браузер, а сервером – веб-сервер. Логика приложения сосредотачивается на сервере, а браузер лишь отображает информацию, загруженную по сети с сервера. Для ВП главной мерой производительности является время отклика приложения на запрос пользователя. Но в процессе своей работы на ВП могут возникать разного рода перегрузки, ошибки, исключительные ситуации, которые замедляют работу приложения, а в худшем случае делают ее невозможной. Это негативно сказывается на пользователе, который заинтересован в решении своей задачи с помощью ВП. Такие ситуации нужно отслеживать, а в случае возникновения – быстро решать. Поэтому требуются механизмы для непрерывного процесса наблюдения за работоспособностью и доступностью ВП, а также способы управления производительностью подобных систем. Существует множество методов для управления производительностью ВП, разработанных за рубежом [1, 2], которые также применяются в Украине.

Цель работы – выявление актуальных проблем управления производительностью ВП и системный анализ существующих подходов к управлению производительностью веб-приложений, а также решений и механизмов APM для ВП (их области применения и основных характеристик использования).

Задача управления производительностью ВП тесно связана с мониторингом. В широком смысле под мониторингом ВП подразумевается процесс проверки работоспособности и его доступности в сети Интернет. Также мониторинг применяют на всех уровнях работы ВП от аппаратной части (центрального процессора (ЦП), оперативной памяти (ОП)) до программного обеспечения (ПО) и состояния приложения. Основные функции мониторинга ВП:

- уведомления и оповещения об ошибках;
- отслеживание производительности и нагрузки на веб-сервер;
- сбор показателей отклика, скорости работы приложения, частота отказов;
- мониторинг ошибок в работе приложения и функциональности его компонентов;
- мониторинг запросов в базу данных (БД) и время отклика.

Мониторинг организовывают для систематического сбора и обработки информации, которую используют для улучшения процесса принятия решения, в случае возникновения проблем в работе ВП. При обнаружении какой-либо неисправности сервис мониторинга может посылать разработчику оповещение, благодаря чему специалист в кратчайший срок сможет приступить к восстановлению работоспособности ВП. Но перед этим специалисту важно знать, какого рода проблема возникла. Базовое представление об этом дает анализ источника сообщения о проблеме. Ниже рассмотрены типы мониторинга ВП.

Типы мониторинга ВП. Мониторинг осуществляют на многих уровнях (аппаратном, БД, бизнес-логики, отображения данных) веб-приложения. На этих уровнях организовывают разные типы мониторинга [3]:

- системный мониторинг (System Monitoring);
- серверный мониторинг (Server Monitoring);
- мониторинг доступности (Availability Monitoring);
- мониторинг производительности (Performance Monitoring);
- мониторинг транзакций (Application / Transaction Monitoring);
- мониторинг действий конечного пользователя (End-User Monitoring).

Рассмотрим подробнее каждый тип мониторинга.

Системный мониторинг отображает информацию о производительности системы на «низком» уровне. Основная цель состоит в получении представления о текущем состоянии и поведении аппаратных составляющих системы. Метрики, которые применяются к данному типу мониторинга, – это: загрузка ЦП, пропускная способность сети, потребление памяти.

Серверный мониторинг, в свою очередь, поддерживает наблюдение за веб-серверами, такими как *Jetty* [4], *Tomcat* [5]. Эти веб-серверы обычно используют свои внутренние метрики и посылают результаты на уровень представления.

Мониторинг доступности применяется для проверки работоспособности ВП в целом. Это базовая форма мониторинга, основная задача которой – убедиться в доступности рабочей инфраструктуры ВП. Примером служит проверка доступности сервера в сети путем посылки запроса и получения ответа.

Мониторинг производительности используется для наблюдения за производительностью системы. Цель мониторинга – в получении информации о том, как долго выполняются задачи и сколько времени нужно для их завершения. Общепринятой мерой измерения является установление временного предела, за который должна выполниться конкретная задача. Примером для этого служит время «ответа» ВП на запрос пользователя, которое обычно не превышает 500 ms.

Мониторинг транзакций применяется для сбора информации о запросах (транзакциях), пока они выполняются. В этом случае формируются метрики, которые могут отличаться в зависимости от приложения, для описания текущего статуса транзакций. По статусу можно определить транзакции, которые выполнились успешно, еще выполняются или выполнились неудачно.

Мониторинг действий конечного пользователя используется для проверки доступности всех сервисов, которыми может воспользоваться конечный пользователь при использовании ВП. Для этого программируются сценарии, которые симулируют поведение пользователя.

В основном многие из приведенных типов мониторинга являются составляющими общей системы управления производительностью приложения. Ключевые особенности и функции подобных систем описаны ниже.

Управление производительностью приложения (APM – Application Performance Management). Управление производительностью приложения – это комплекс средств для визуального контроля за работой приложения (в том числе ВП).

Основными принципами APM систем являются: обнаружение, диагностика и решение проблем производительности приложения. Модули APM обрабатывают данные, полученные от разных типов мониторинга, и представляют результаты в виде графиков в

режиме реального времени. Системы АРМ имеют гибкие теоретическую и программную реализации, которые стремятся выполнять следующие функции:

- комплексный анализ производительности и доступности приложения;
- изучение работы конечных пользователей с помощью сценариев, выполняемых в разных местах приложения;
- отслеживание реальных операций пользователей для ускоренного решения проблем;
- получение аналитической картины приложения для быстрого выявления и решения аномалий;
- обмен сценариями между тестовым и операционным этапом для повышения качества приложения;
- анализ и обработка отчетов.

Американской исследовательской компанией Gartner были сформулированы и описаны пять направлений для АРМ, которые представлены в APM Conceptual Framework [6], а именно:

- опыт конечного пользователя (End user experience);
- динамическое исследование архитектуры приложения и моделирование (Application runtime architecture discovery and modeling);
- управление бизнес-транзакциями (Business transaction management);
- мониторинг компонентов приложения (Application component monitoring);
- анализ данных приложения, формирование отчетов (Reporting & Application data analytics).

Каждое направление фокусируется на сборе и анализе данных с целью улучшения производительности приложения.

Аналитический обзор существующих решений АРМ для ВП. Набор соответствующих решений и инструментов для АРМ достаточно велик – от крупных коммерческих приложений, научных проектов до малых программных библиотек, которые используются для сбора и агрегации информации. Краткое описание основных инструментов, а также их преимущества и недостатки описаны ниже.

1. Kieker [7] – научный проект университета Christian-Albrechts-University Kiel. Данная система АРМ имеет графическую визуализацию результатов и программную реализацию в виде Java-фреймворка с открытым исходным кодом.

Данное решение реализует средства для управления производительностью ВП, а также механизмы динамического анализа поведения приложения. Разработчики Kieker фокусируются на двух сторонах исследования производительности:

- методы управления и анализа производительности ВП, обнаружения аномалий в его работе;
- исследование архитектуры ВП, а именно извлечение информации об архитектуре приложения из данных, собранных путем использования механизмов мониторинга в существующей системе ПО.

Гибкая архитектура этого фреймворка позволяет заменять и добавлять модули инфраструктуры, включая мониторинг (с использованием зондов или аспектов), анализ данных метрик, протоколирования и т.д. Kieker – научный проект, поэтому для теоретического анализа используется академический подход, а именно: построение UML диаграмм последовательностей, графов зависимостей, цепей Маркова.

Преимущества:

- решение представлено в виде Java фреймворка с открытым исходным кодом, что позволяет гибко внедрить систему мониторинга в существующее ВП;
- реализован механизм сбора и анализа данных для распределенных приложений (больше одного сервера);
- функции исследования архитектуры ВП предоставляют возможность анализа поведения приложения и диагностирование проблемных мест;
- возможность добавления новых модулей, хорошая расширяемость фреймворка.

Недостатки:

- необходимость хранения больших объемов данных результатов протоколирования для последующего анализа;
- необходимость изменения исходного кода приложения для добавления аннотаций мониторинга.

Таким образом, Kieker фреймворк – это перспективный проект с эффективными функциями сбора и анализа данных производительности ВП. В качестве главного достоинства данного решения можно выделить

применение математических моделей к исследованию проблем производительности приложения.

2. New Relic [8] является коммерческим проектом, который реализует модель мониторинга SaaS – Software as a Service, в которой вся необходимая инфраструктура предоставляется поставщиком (New Relic). В этом случае пользователь имеет доступ к сервису в любой точке мира, используя веб-интерфейс. Преимущество этой модели состоит в том, что все обслуживание и обновления выполняются производителем самостоятельно.

Целью New Relic является обеспечение ВП на разных уровнях и устройствах широким спектром мониторинга, а также удобным пользовательским интерфейсом. Данная APM предоставляет пользователю возможность мониторинга приложений, реализованных на разных платформах (Java, Ruby, .NET). Информация отображается пользователю в виде разнообразных графиков. Особенность New Relic состоит в том, что пользователь имеет гибкий контроль над метриками мониторинга, которые он может комбинировать для удобного отображения результатов.

Преимущества:

- обслуживание, обновления сервисов выполняются производителем самостоятельно;
- большой набор настраиваемых функций мониторинга, которые дают возможность пользователю получить уникальную и необходимую для него информацию;
- удобный пользовательский интерфейс, а также гибкость в управлении метриками;
- доступность и простота в использовании для конечного пользователя через веб-интерфейс;
- возможность мониторинга приложений, реализованных на разных платформах;
- не нужно дополнительных изменений в исходном коде приложения;
- возможность создавать правила для метрик, по которым пользователь будет получать уведомления о производительности системы.

Недостатки:

- слабая поддержка мониторинга при использовании защищенных протоколов передачи данных, к примеру, по HTTPS;
- полная версия данной системы APM – платная, но существует и бесплатная версия с ограниченной функциональностью.

Программный продукт APM New Relic – мощная и гибкая система APM для ВП. Возможность комбинирования метрик, отсутствие необходимости изменять исходный код, богатый выбор функций, а также удобный пользовательский интерфейс являются достоинствами данного решения. Это хороший выбор для ВП малого и среднего бизнеса с использованием небольшого количества серверов.

3. Glassbox [9] – агент поиска и устранения неисправностей для приложений Java, который автоматически диагностирует типичные проблемы. Модуль инспектора Glassbox контролирует работу виртуальной машины (JVM) с помощью AspectJ. Основная цель – мониторинг производительности ВП, а также удобное описание текущих проблем и ошибок в производительности ВП.

Преимущества:

- проводит мониторинг на уровне JVM, нет необходимости изменять исходный код приложения;
- удобный графический интерфейс, который доступен через веб-браузер;
- адаптируется к приложению и выявляет проблемные места в его работе;
- проект разработан с использованием аспектно-ориентированного программирования, что дает возможность качественно расширять существующую функциональность;
- распространяется по свободной лицензии LGPLv2, что дает возможность разрабатывать и интегрировать в приложение собственные модули;
- прост в развертывании на веб-сервере.

Недостатки:

- отсутствие мониторинга для распределенных приложений, которые используют больше одного сервера для своей работы;
- слабое взаимодействие с Java Management Extensions (JMX).

Программная библиотека Glassbox предлагает качественный мониторинг производительности для небольших приложений. За счет открытого исходного кода существует возможность разрабатывать и внедрять собственные модули. Данное решение является простой, эффективной и продуктивной версией APM, но с ограниченной функциональностью для сложных распределенных систем.

4. AppDynamics [10] – это коммерческая система APM, которая дает широкие возможности для мониторинга, диагностики и устра-

нения неполадок в сложных распределенных приложениях. Также реализует модель мониторинга SaaS и обладает похожими преимуществами, что и NewRelic. Как особенность данного решения можно выделить мощные средства для анализа производительности ВП и гибкую систему идентификации «узких мест».

Преимущества:

- реализован механизм автоматического обнаружения архитектуры приложения;
- простота в использовании, интуитивно понятный интерфейс;
- возможность идентификации «узких мест» в работе ВП путем просмотра методов приложения через веб-интерфейс;
- наличие полезных расширений (плагинов) для AppDynamics Pro;
- не нужно дополнительных изменений в исходном коде приложения.

Недостатки:

- расширенная версия данной системы APM – платная, но существует и бесплатная версия с ограниченной базовой функциональностью;

Таким образом, AppDynamics предоставляет широкий и мощный спектр услуг для управления производительностью сложных распределенных приложений. Хороший выбор для коммерческих высоконагруженных приложений при использовании платной версии.

Выводы. Системный анализ существующих решений и механизмов APM для ВП показывает, что данная область информационных технологий активно развивается и реализовывается в практических приложениях, а также исследуется с применением математического моделирования. Причиной этого служит увеличение количества сложных ВП и необходимость быстрой идентификации, диагностики и устранения проблем, связанных с их работоспособностью. С научной точки зрения, актуальными являются такие задачи:

1) динамический анализ архитектуры приложения, его поведения для диагностики проблемных мест;

2) исследование и разработка методов математического анализа производительности компонентов ВП. Основываясь на этих данных, можно решать задачи оптимизации модулей ВП и повышать их производительность с максимальной эффективностью;

3) прогнозирование производительности ВП на основе анализа имеющихся данных протоколирования или его поведения.

References

1. Menasce, Daniel A. & Almeida, Virgilio A. F. (1998) Capacity planning for Web performance: metrics, models, and methods, Upper Saddle River, NJ: Prentice Hall.
2. Killelea, Patrick (1998) Web performance tuning. 1st ed. Beijing; Sebastopol, CA: O'Reilly.
3. Reitbauer, A. & Novakovic, Andm (2009) Effizientes Performance management, ***Java Magazin,*** (11).
4. Jetty – Servlet Engine and Http Server. N.p., n.d. Web. 14 Oct. 2013, available at: http://www.eclipse.org/jetty/
5. Vivek Chopra, Sing Li, & Jeff Genender (2007) Professional Apache Tomcat 6; Wiley Publishing, Inc. Moscow, 672 p.
6. Gartner Research "Keep the Five Functional Dimensions of APM Distinct". 16 Sept. 2010.
7. Van Hoorn, A., Rohr, M., Hasselbring, W. et al. Continuous Monitoring of Software Services: Design and Application of the Kieker Framework, available at: http://eprints.uni-kiel.de/14459/, 2009.
8. Application Performance Monitoring. New Relic. N.p., n.d. Web. 14 Oct. 2013, available at: http://www.newrelic.com/.
9. Glassbox. N.p., n.d. Web. 14 Oct. 2013, available at: http://glassbox.sourceforge.net/ glassbox/Home.html
10. AppDynamics. Application Performance Monitoring and Management. N.p., n.d. Web. 14 Oct. 2013, available at: http://www. appdynamics.com/