УДК 004

ЖУРНАЛИРОВАНИЕ КАК ИНСТРУМЕНТ ДЛЯ ПОДДЕРЖКИ И СОПРОВОЖДЕНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Бобышев Петр Петрович, *студент 2 курса магистратуры, факультет экономики и финансов, РГАУ-МСХЛ имени К. А. Тимирязева, 141963141963@mail.ru*

Щедрина Елена Владимировна, *доцент кафедры вычислительной техники и прикладной математики, ФГБОУ ВО РГАУ-МСХА имени К.А. Тимирязева, к.п.н., доцент, shchedrina@rgau-msha.ru*

Аннотация: *Разработано расширение для среды разработки Eclipse для работы с промышленными файлами журналирования*

Ключевые слова: *Лог, журнал, поддержка ПО.*

В настоящее время все более актуальной проблемой является возможность разработчиками программных приложений отслеживать события, происшедшие во время работы системы или при проектировании приложения. Возможность наблюдения за поведением приложения в тех или иных ситуациях является очень ценным инструментом для поддержки и сопровождения программы, так как позволяет разработчикам выявлять нестандартные ситуации, в которых работа программы может быть не полностью корректной, плохо прогнозируемой или вовсе не соответствующей изначально предполагаемой бизнес логике.

Одним из самых естественных способов отслеживания состояния контролируемых ресурсов является просмотр и анализ файлов логов. Журналы, производимые приложением или системой, представляют собой ценный источник информации для мониторинга, поэтом}' логирование играет важную

роль в процессе поддержки приложения. Однако, форматы журналов могут сильно различаться среди разработчиков. Большинство записей происходящих в системе событий производится в виде естественного языка и не содержат какой-либо предварительно определенной структуры, что делает их очень труднообрабатываемыми, не говоря уже об извлечении из них полезной информации. Эта проблема может быть решена путем стандартизации формата логов и продвижения структурированных логов на замену традиционным подходам.

По словам Антона Чувакина [1], идеальная запись журнала приложения должна содержать, по крайней мере, информацию о том, что случилось, когда это произошло, где это произошло, кто был вовлечен и откуда пришло само сообщение. Кроме того, было бы полезно знать, где можно получить более подробную информацию, как убедиться в том, что сообщение о событии правильно, и что еще могло послужить причиной этого сообщения.

Существуют специальные библиотеки для удобного и правильного логирования событий системы. Обычно они предоставляют возможности для гибкого форматирования выходных записей и их разделения по уровням логирования. Так как разные сообщения в журнале могут содержать информацию разной критичности, их стоит разделять по разным уровням логирования. Состав и значения уровней логирования могут иметь отличия в зависимости от конкретной библиотеки, но негласные договоренности для разделения по уровням такие [2-4]:

1. Уровень INFO - наиболее часто используемый уровень, на котором может быть записана любая информация о событиях системы или бизнес логики.
2. Уровень DEBUG — это уровень, используемый при отладке и тестировании приложения.
3. Уровень WARN - это уровень, который содержит сообщения об ошибках или не штатных и потенциально опасных ситуациях.
4. Уровень ERROR - уровень, для сообщений об ошибках, не прерывающих ход работы программы в целом.
5. Уровень FATAL - на этот уровень попадают сообщения об ошибках, после которых нормальная работа приложения становится невозможной. Обычно после такого сообщения полностью завершается работа программы.

Для того, чтобы лучше понимать информацию от анализируемого программного обеспечения, собранную в виде записей журнала, используются различные инструменты и методы анализа. Из-за большой изменчивости в форматах журнала, не существует никакого общего способа для анализа и обработки логов; подход зависит от синтаксиса журнала и формы. Еще одна сложность обусловлена субъективной природой журнала сообщений; например, при фильтрации записей журнала по степени тяжести, нет никакой гарантии, что реальная важность сообщения, которое производитель решил пометить определенным типом тяжести, соответствует заявленном)' типу тяжести. Кроме того, там могут быть записи журнала, которые сами по себе являются бессмысленными и имеют значение только в определенном контексте.

В целом, анализ журнала не простая задача, но она должна быть выполнена по разным причинам, например, для поддержания ситуационной осведомленности • об управляемых ресурсах, обеспечения безопасности и обнаружения атак или диагностики неисправностей. Ручной анализ журналов, как правило, не представляется возможным, не только из-за количества записей журнала, с которыми придется работать, но и из-за того, что такой анализ вряд ли обеспечит полную картину произошедших событий [1].

Для упрощения работы с логами и повышения эффективности их анализа существует обширный список специального программного обеспечения. Такие программы призваны как можно сильнее облегчить процесс анализа и обладают соответствующим инструментарием, помогающим автоматизировать и тем самым упростить и ускорить действия, не связанные непосредственно с анализом, но необходимые для его успешного проведения, такие как фильтрация, изъятие из анализируемых логов явно лишней (не представляющей ценности для данного анализа) информации, приведение к удобочитаемому виду и так далее.

Библиографический список

1. Chuvakin A., Schmidt K., Phillips C. Logging. The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management / Chuvakin A., Schmidt K., Phillips C. Logging / Syngress, 2012.

2. Электронный ресурс: Chan C. Effective logging practices ease enterprise development URL: https://geekdetected.files.wordpress.com/2013/12/ effective -logging-practices- ease-enterprise-development.pdf.

3. Электронный ресурс: Apache Log4j Documentation. URL: https://logging.apache.Org/log4j/l.2/apidocs/org/apache/log4j/Level.html.

4. Электронный ресурс: Eclipse marketplace. Java logging viewer plugin page. URL: https://marketplace.eclipse.org/content/jlv.