**Проектирование ОС и компонентов**

Управление памятью. Создание библиотек разных типов и их размещение.  
Загрузка программ и библиотек, анализ дампа

**Теоретическая часть**

Профилирование — сбор характеристик работы программы, таких как время выполнения отдельных фрагментов (обычно подпрограмм), число верно предсказанных условных переходов, число кэш-промахов и т. д. Инструмент, используемый для анализа работы, называют профилировщиком или профайлером. Обычно выполняется совместно с оптимизацией программы.

Характеристики могут быть аппаратными (время) или вызванные программным обеспечением (функциональный запрос). Инструментальные средства анализа программы чрезвычайно важны для того, чтобы понять поведение программы. Проектировщики ПО нуждаются в таких инструментальных средствах, чтобы оценить, как хорошо выполнена работа. Программисты нуждаются в инструментальных средствах, чтобы проанализировать их программы и идентифицировать критические участки программы.

Это часто используется, чтобы определить, как долго выполняются определенные части программы, как часто они выполняются, или генерировать граф вызовов (Call Graph). Обычно эта информация используется, чтобы идентифицировать те участки программы, которые работают больше всего. Эти трудоёмкие участки могут быть оптимизированы, чтобы выполняться быстрее.

Также выделяют анализ покрытия (Code Coverage) — процесс выявления неиспользуемых участков кода при помощи, например, многократного запуска программы.

JVM Monitor - это профилировщик Java, интегрированный с Eclipse для мониторинга использования процессоров, потоков и памяти приложений Java.

Монитор JVM был бы полезен для быстрой проверки приложений Java без предварительной подготовки любой конфигурации запуска. JVM Monitor автоматически находит запущенные JVM на локальном хосте, и вы можете легко начать их мониторинг. Он также поддерживается для мониторинга приложений Java на удаленном хосте, указывая имя хоста и номер порта.

Приложения Enterprise Java - это N многоуровневые и сложные. Виртуальная машина Java (JVM) повышает производительность разработчиков Java, скрывая детали обработки JVM. Однако это создает новые проблемы в мониторинге JVM в производстве, поскольку неправильная очистка памяти JVM может привести к замедлению работы приложения. Поэтому каждый администратор приложения должен обеспечить оптимальную конфигурацию JVM и контролировать JVM-память и контролировать производительность JVM.

Плохо настроенное приложение Java может привести к медленной деградации в производительности JVM, что, в свою очередь, может привести к тому, что приложение станет медленным для конечных пользователей. Сбор мусора JVM - это механизм, с помощью которого виртуальная машина (виртуальная машина) освобождает память для приложений Java.

**Мониторинг памяти JVM**

ManageEngine Applications Manager может помочь контролировать память JVM: память кучи и без кучи. Кроме того, вы можете контролировать, как виртуальная машина выделяет объекты в памяти для различных пространств памяти, таких как Eden Space, Survivor space, Tenured Generation и т. Д. Каждое место памяти имеет свое значение и контролирует их в процессе производства, а также настраивает параметры конфигурации при запуске JVM , Приложение Java может быть настроено хорошо.

**Eclipse Memory Analyzer**

Анализатор памяти предоставляет универсальный инструментарий для анализа дампов кучи Java. Помимо кучного хождения и быстрого вычисления сохраненных размеров, инструмент Eclipse сообщает о подозрениях на утечку и потреблении памяти. Основной областью применения являются ошибки из памяти и высокое потребление памяти.

**JVM Monitor:**

Профилирование делается с помощью JVM Monitor.

Существует несколько вариантов установки.

Вариант 1) откройте мастер Eclipse Marketplace (Справка> Eclipse Marketplace ...), выполните поиск с текстом «Монитор JVM» и нажмите кнопку «Установить».

Вариант 2) Если меню Eclipse Marketplace не найдено, откройте мастер установки (Справка> Установить новое программное обеспечение ...> Добавить ...) и введите следующий сайт обновления.

|  |
| --- |
| http://www.jvmmonitor.org/updates/3.8 |

Вариант 3) вручную установить, загрузив zip-файл и распакуя папку Eclipse.

Когда запускается мониторинг JVM можно открывать страницу Timeline в представлении «Свойства», по умолчанию можно увидеть тренд используемой памяти кучи, загруженного класса, количества потоков и использования ЦП на диаграммах.

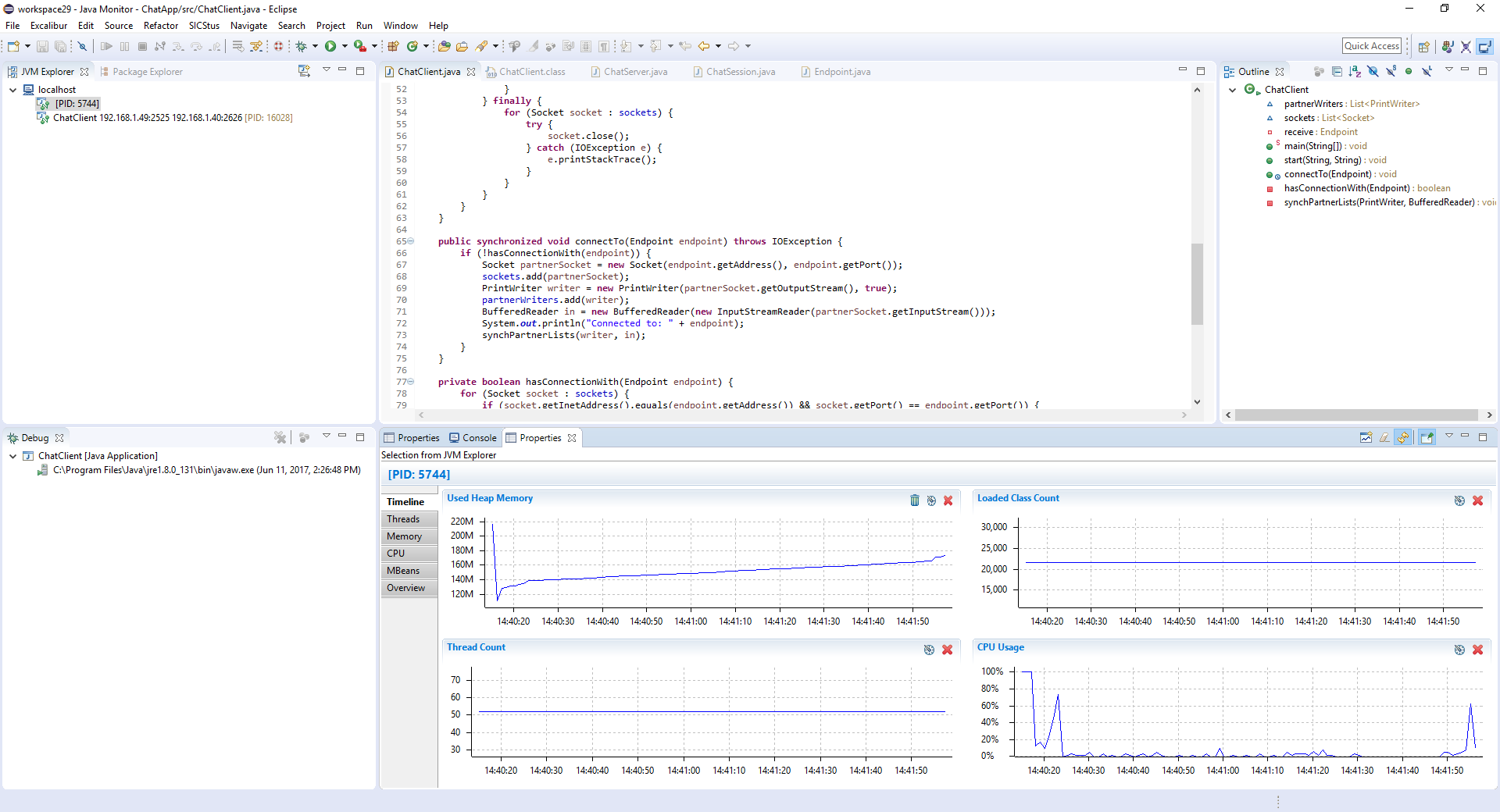
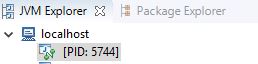


Рисунок общий вид

По умолчанию можно запускать PID: 5744 (localhost)

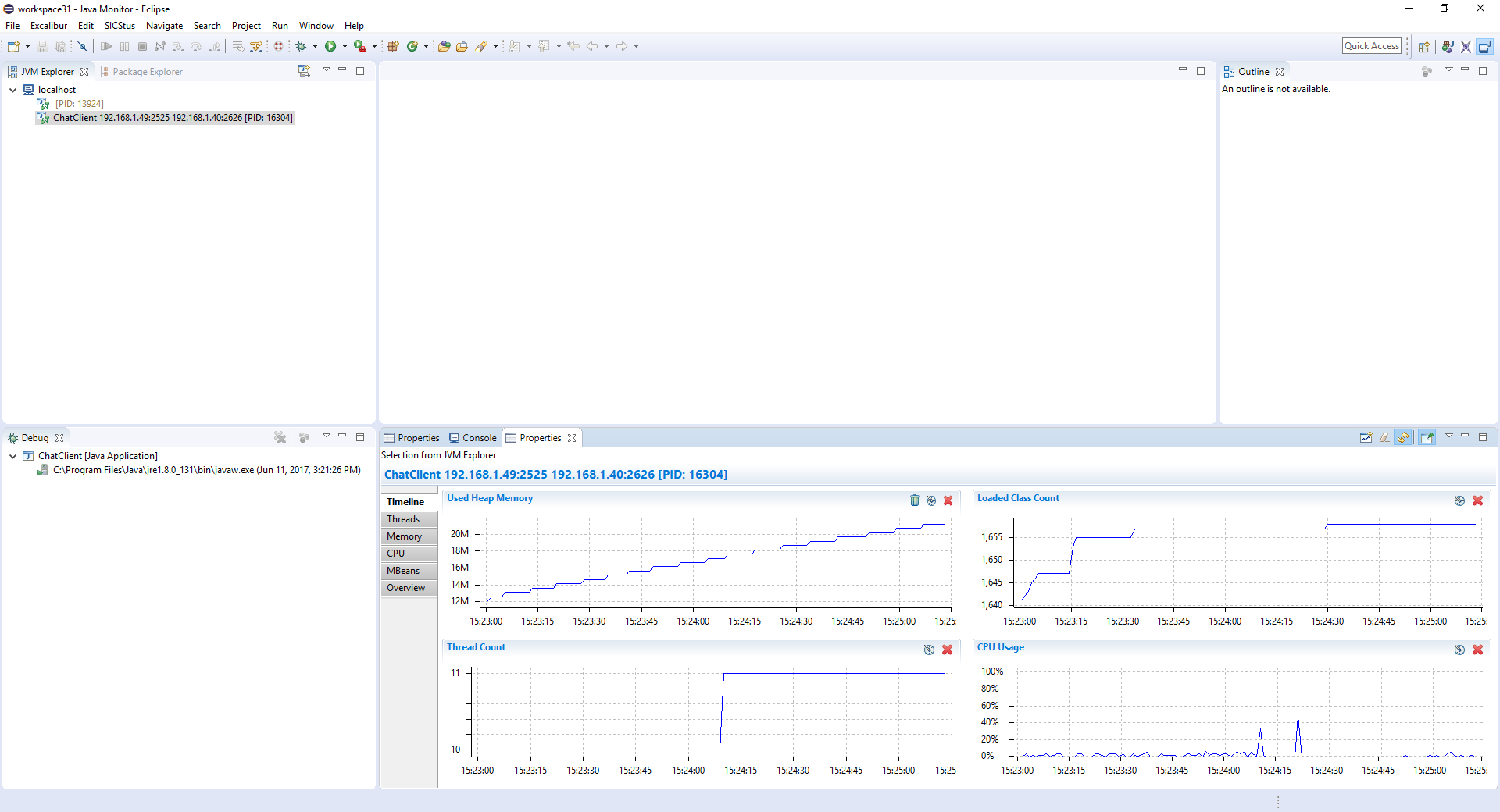


Рисунок

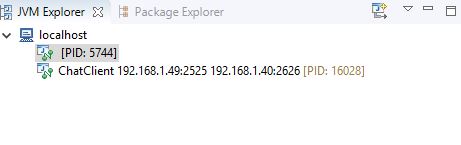
После запуска программа начинает работать.



Начнем запуска нашей программы.



Рисунок



Рисунок

**Мониторинг использования памяти** (ChatClient 192.168.1.49:2525 192.168.1.40:2626 (PID: 16304))

Когда вы запускаете мониторинг JVM и открываете страницу памяти в окне «Свойства», вы можете увидеть гистограмму кучи, отображающую размер объекта и количество объектов для каждого класса.

Идентификация объектов, содержащих кучную память

Когда вы замечаете, что память кучи неожиданно потребляется (например, при мониторинге используемой памяти кучи на странице временной шкалы), вы можете захотеть определить, какие объекты фактически потребляют кучную память.

Чтобы идентифицировать объекты, потребляющие кучу памяти:

Надо убедится что гистограмма кучи отсортирована по столбцу «Размер» и просмотрит размер памяти кучной памяти для каждого класса.

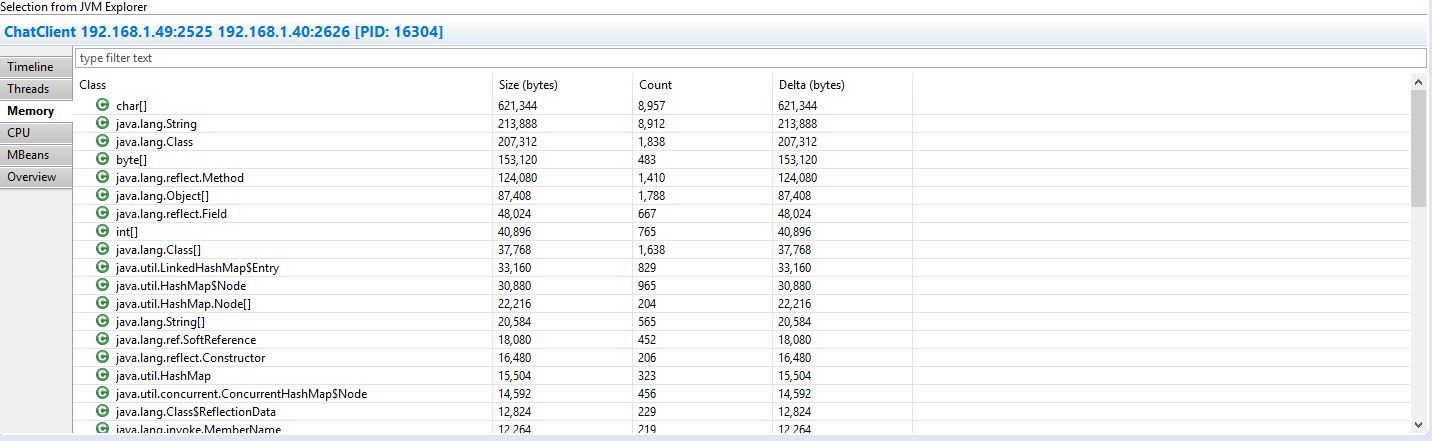
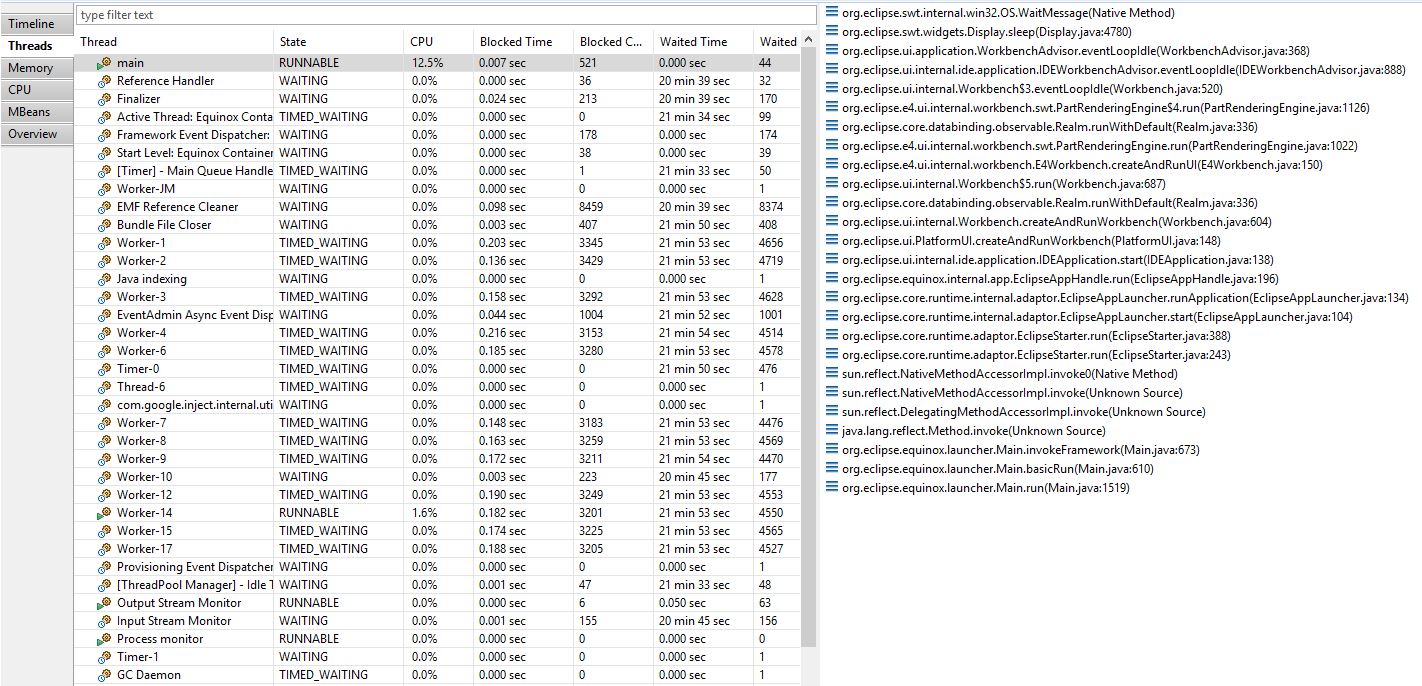


Рисунок 5

Идентификация ресурса ЦП, потребляющего кода

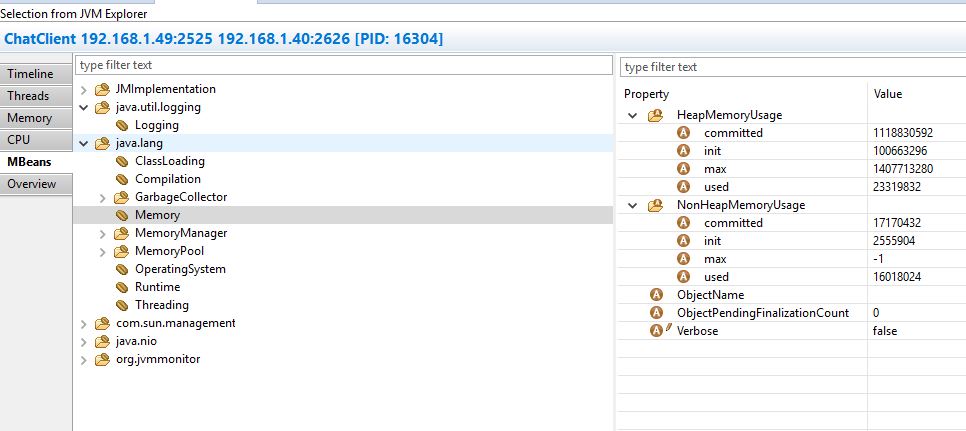
Процессор может неожиданно перегружатся (например, при мониторинге использования ЦП на странице Timeline), может потребоваться определить, какая часть кода фактически потребляет ресурсы ЦП.



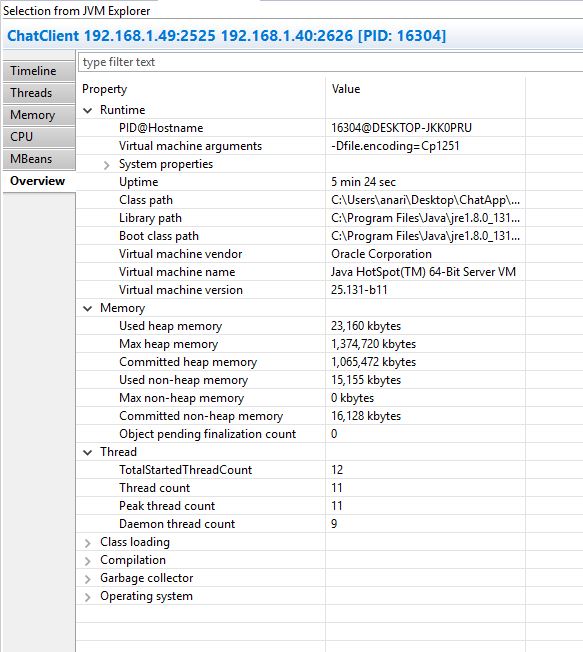
Рисунок

**Мониторинг атрибутов MBeans**

Когда начинается контролировать JVM и надо открывать страницу MBeans в окне «Свойства», можно увидеть MBeans.



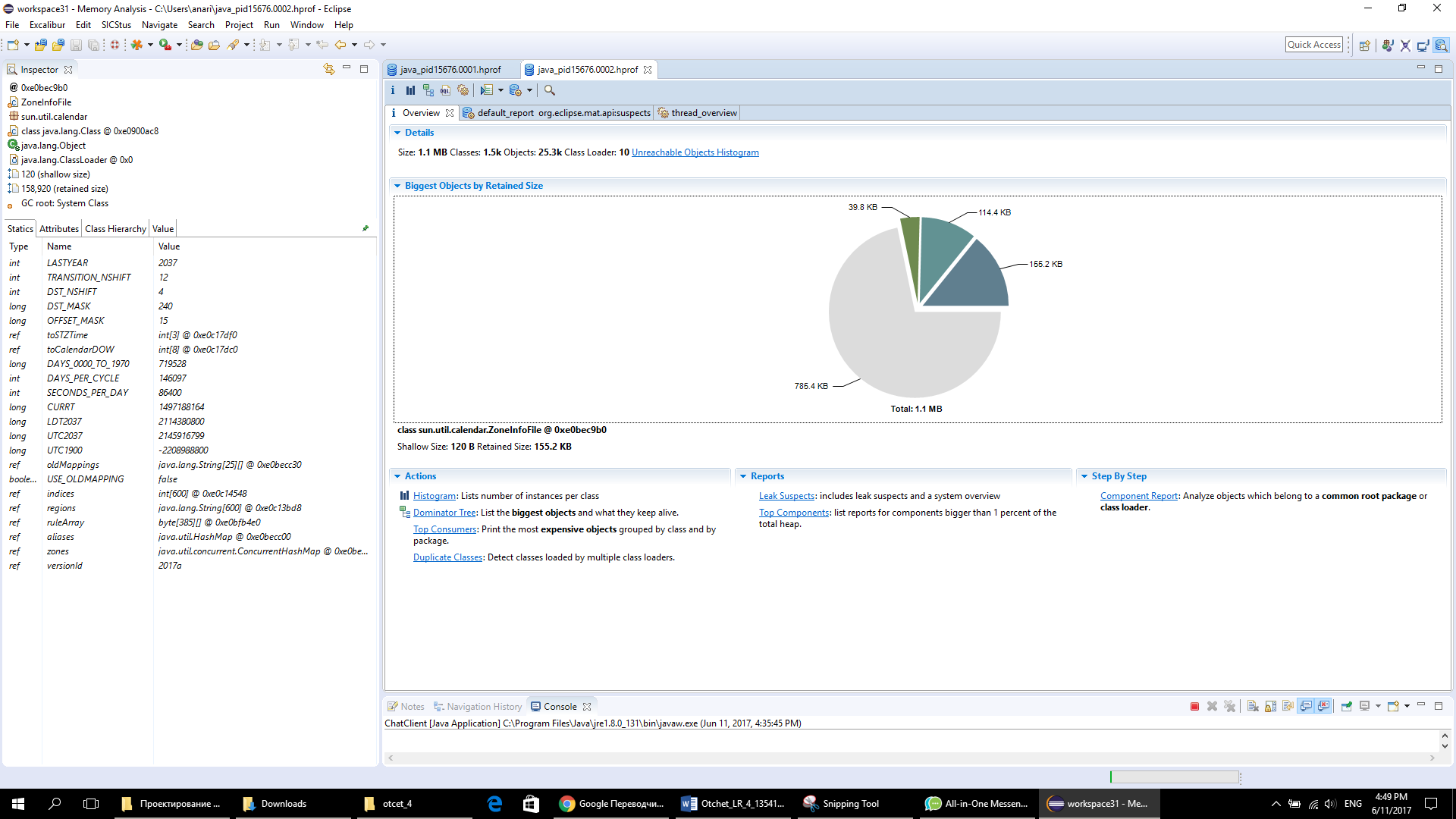
**Информация**



**Eclipse Memory Analyzer**

Анализатор памяти Eclipse - это быстрый и многофункциональный анализатор кучи Java, который помогает вам обнаруживать утечки памяти и уменьшать потребление памяти.

Анализатор памяти используется для того чтобы анализировать продуктивные свалки кучи с сотнями миллионов объектов, быстро вычислять сохраняемые размеры объектов, видеть, кто мешает сборщику мусора собирать объекты, запускать отчет, чтобы автоматически извлекать подозреваемые в утечке.



Рисунок

**Анализ потоков**

Memory Analyzer предоставляет несколько запросов для проверки потоков в момент, когда был сделан снимок.

Запрос предоставляет некоторые свойства, такие как Name, Object, Context Classloader и т. Д. Для каждого из потоков.

Некоторые форматы дампа кучи (например, дампы HPROF от недавних виртуальных машин Java 6 и системные дампы IBM) содержат информацию о стеках вызовов потоков и локальных объектах Java на стек стека.

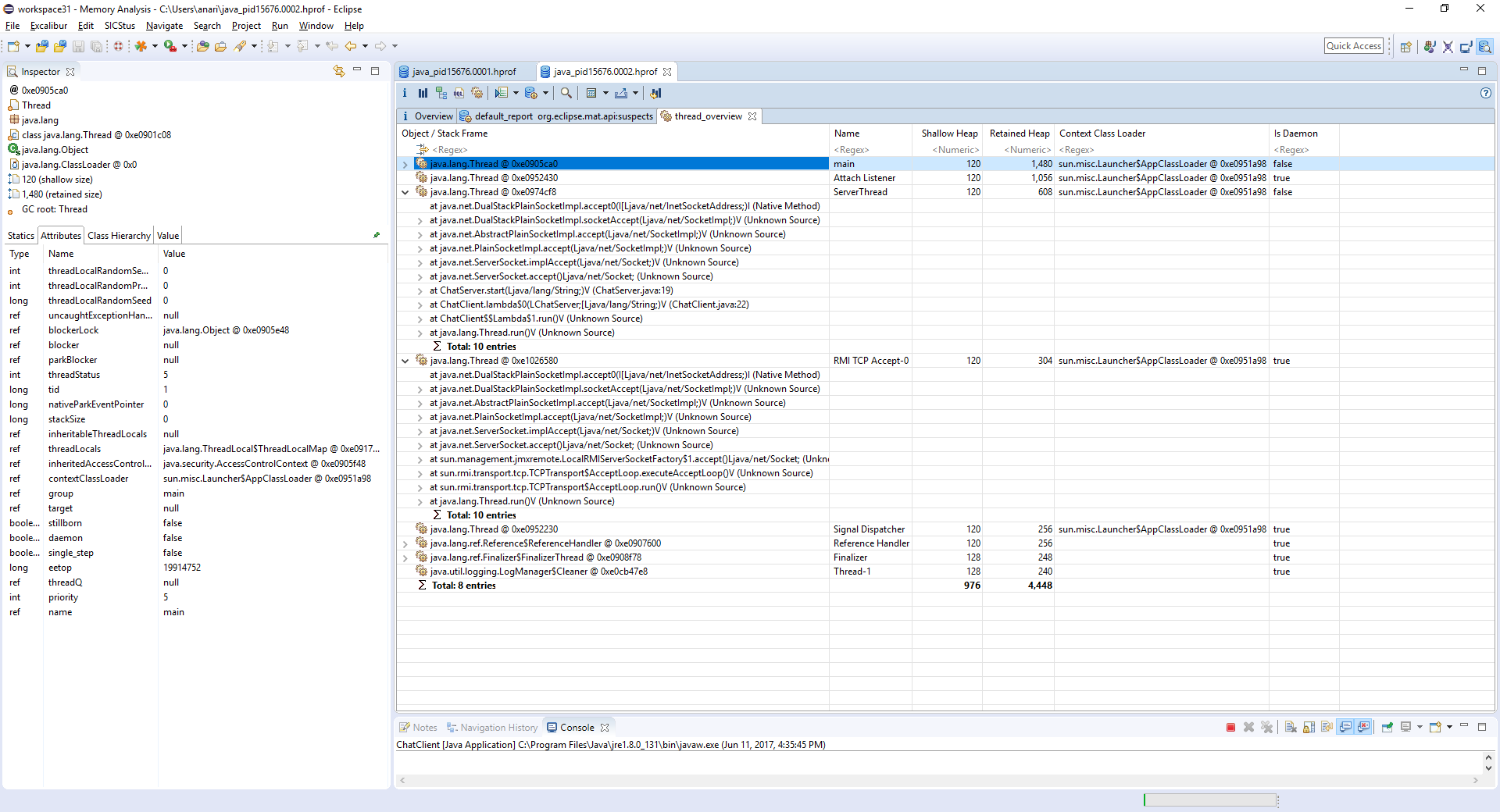


Рисунок 8

**Выводы:**

JVM Monitor отлично справился со своей работой: по каждой ошибке дана детальная инструкция.  
JVM Monitor и Eclipse Memory Analyzer дали результаты по профилированию. Сделаны анализ памяти, дампа. JVM Monitor очень хороший инструмент для анализа своей программы.

**Список используемых источников:**

1. <http://jvmmonitor.org/download/index.html>