COGSI - Configuração e Gestão de Sistemas

Mestrado em Engenharia Informática, Ramo Sistemas Computacionais

Class Assignment

P2 - Application Monitoring

Alexandre Bragança atb@isep.ipp.pt

Dep. de Engenharia Informática – ISEP

2022/2023

P2: Overview

- **Start Date:** 15, March
- **End Date:** 29, March
- **Development Repository:** Your individual repository
  - Create Issue(s) for your sprint
  - Expected several commits (at least 1 for each lab class!)
  - You should commit to the repository only files that you created or edited (e.g., do not commit the nagios directory!)
  - Documentation should be provided **only in the readme.md file related to the assignment!**
- **Assignment Presentation and Review:**
  - Lectures: March 29
- **If you are selected for presentation (sprint review)** then you should **pull request** the final version of your work into the class shared repository (`https://bitbucket.org/mei-isep/cogsi-22-23-class-rep`) before the deadline.

P2: Topic

- The topic of this sprint is **Application Monitoring**
- For the application monitoring it is **mandatory to use JMX**
- The system monitoring tool must be **Nagios**
- For alternative tools you may use other application monitoring APIs, such as **WMI** (Windows Management Instrumentation) or specific instrumentation APIs for a programming language (e.g., Prometheus). You may use other tool/api but you should check before with the teacher.
- The overall objective is to extend the scenario developed for P1 with application monitoring using JMX.
  - You should, as much as possible, start with the complete scenario from P1
  - You should add remote monitoring for the TODD sample application (the server). See sample application in `https://bitbucket.org/mei-isep/todd`
  - You should explore Tomcat JMX services and add remote monitoring for Tomcat using JMX.
  - **Note:** You may choose the virtualization software to use (VirtualBox, etc.) but use Vagrant to manage the virtual machines.

P2: Specific Concerns

Integration with Nagios

- You should integrate the system monitoring tools (e.g., Nagios) with JMX monitoring
- The system monitoring server (e.g., Nagios) should be able to remotely monitor the TODD server.
- It should be able to:
  - Check (polling) if TODD server is running. If TODD server is not running you should try to automatically start it.
  - Check (polling) if the "AvailableSessions" attribute is below 20% of the "Size" attribute. If so, you should consider that the service is critical and notify someone.

JMX Notifications

- You should develop a **JMX Agent Application** that is able to use JMX notifications to react to "events" in a JMX monitored resource.
- By using JMX notifications, this application should be able to monitor the TODD server and increase the size of the "SessionPool" by using the method "grow" when the "AvailableSessions" attribute is below 20% of the "Size" attribute. You should try to **integrate with the system monitoring tools using passive checks**.
- You should also use a similar approach to handle possible JVM memory limitations regarding the Tomcat server. You should define thresholds for this concern and try to free memory or even restart tomcat if necessary (with different settings for the memory usage of the JVM).

P2: Alternative

The alternative for this assignment is to use other application monitoring framework or API (to "replace" JMX).

You should research about alternatives. For instance, possible examples:

- In Windows applications one can use Windows Management Instrumentation (see `https://docs.microsoft.com/en-us/dotnet/framework/wcf/diagnostics/wmi/`)
- API for code instrumentation in Prometheus (See `https://prometheus.io/docs/practices/instrumentation/` and `https://www.robustperception.io/instrumenting-java-with-prometheus`)

You should present and compare the alternative with JMX and describe how to implement some particular aspect of the requirements with the alternative monitoring tool.

P2: Technical Report

You should produce a technical report documenting the assignment.

- The technical report **must be produced** in the **readme.md** file located in the repository folder related to P2 (e.g., 1133224-maria-ferreira/p2/)
- The report should include:
  - The Analysis of the Problem
  - The Design of your Solution
    - Present an overview of the tools (e.g., software used, major concepts, major processes, architecture of the tools)
    - Present an overview of the solution (e.g., the architecture and major configurations required)
  - The Steps required to Reproduce your Solution (it should include references/links to configuration files, scripts or code included in the same folder of the repository)

You may also include:

- Justification of Design Options
- Analysis of the Alternative
- The Steps required to Reproduce the Alternative (i.e., implement the alternative)

P2: How to Submit to the Class Shared Repository

If you have been selected to make a presentation for this component you must share your work with the class using the shared repository.

- The shared repository is located in `https://bitbucket.org/mei-isep/cogsi-22-23-class-rep`.
- You should make a fork of this repository.
- You should then clone the forked repository into your local computer.
- Copy to this repository only the folder where you developed P2 (e..g, 1133224-maria-ferreira/p2/).
- Commit and push the changes to the forked repository.
- In Bitbucket do a pull request against the original shared class repository in `https://bitbucket.org/mei-isep/cogsi-23-23-class-rep`.
- The teacher will review your pull request and, once accepted, it will become available to all other students.