Monitoring is an important concern for software development. This concern tracks various metrics to help the developer find problems and bottlenecks, allowing him to quickly fix and optimize his code. To do so, the concern has to present the relevant information clearly by generating reports with provided logs and alerting when something goes wrong. The following features are part of our monitoring concern.

Monitor is the main controller. The monitor is essentially keeping the data (info frames) between our features synced up. As soon as a new frame is created containing logs, the monitor keeps track of it. The report or alert feature then takes use of these frames to do their function.

Resource monitoring is our first mandatory feature. It has its own separate timer that invokes the collection of all information. Data from memory, networks, CPU, and disk are all being collected within info frames. This data can then be used to identify memory leaks, compile statistics and monitor the overall performance of the application.

Report is our second mandatory feature. Report has a simple, but important functionality. It's main role is to gather all data stored within different info frames and generate a report that displays the data in a convenient way.

Function Call is our first optional feature. Monitoring Function Call enables us to track a few important metrics regarding function calls. This feature tracks the input, output and execution time among other things. This allows us to find bottlenecks, and bugs much easier. Data is stored within info frames which monitor then keep track of them.

Alert is our second optional feature. Alert uses a set of defined rules to evaluate the frames provided by both resource and function call features. If something unusual is detected, alert sends a notification containing the frame that triggered it.

Engine is the third optional feature, while being part of the mandatory Resource feature. This feature main responsibility is to log all relevant information regarding the engine. Data like stack and heap are being stored.

Finally, Garbage Collection is our final optional feature. Its role extends from that of the engine. This feature tracks garbage collection data done by the engine.

TODO: IMPACT AND NUMBERS
The impacts of this concern are the following:

Complexity:
Having a proxy for monitoring function calls add to the complexity of the application.

Information:
This is the main focus. The more you gather information the more you have to work with to

diagnose the problem in your app.

Disk Space:
The more frequently you monitor the whole system the more data you have to store somewhere.

CPU:
This is minor but monitoring will draw a bit on the performance of the app. The more part of the application is monitor the more the performance might take a hit.

This is a part of every system because it provides developers multiple levels of monitoring information. This information is critical to address performance related issue and ensure that if a problem is found, they are noticed quickly. Without monitoring, developer is debugging blindly when trying to optimise a system. Furthermore, it helps the developer a chance to adapt. They can release a product quickly and optimise only the relevant and critical part of the system according to the monitoring system data.