Health Informatics and Data Collaboration (HIDC)

Submitted to:

Prof. Vittorio Cortellessa & Prof. Daniele Di Pompeo

Department of Computer Science and Engineering, and Mathematics

University of L'Aquila (Italy)

Submitted by:

Adam Bouafia

Matricula : 293137

E-mail: adam.bouafia@student.univaq.it

Table of contents

1. Introduction

This project aligns with the ambitious goals of the SoBigData initiative by constructing a system dedicated to bolstering collaborative efforts in the field of health informatics and data analytics. Targeted at a dual user base of researchers and administrators, the system's design ensures streamlined access and management of critical health data, fostering an ecosystem where innovation and research synergy can thrive.

The functionality for **Health Data Analyst** includes:

- Browsing health datasets to gain insights and identify trends.
- Downloading clinical data for in-depth analysis.
- Publishing research findings, contingent upon administrator approval.
- Commenting on published research to stimulate academic discourse, subject to administrator approval.
- Engaging in real-time dialogue with fellow researchers to exchange knowledge and collaborate on projects.

The **Health Data Manager** are vested with the authority to:

- Approve the publication of resources to maintain data integrity and quality.
- Sanction comments on published resources to ensure constructive and academic discussion standards are met.

The UML models from Homework 1 are believed to be comprehensive and remain unmodified for this phase. They encapsulate the system's functional essence and serve as the bedrock for the algorithmic implementations addressed in this homework. The execution graphs, performance requirements, and queueing network models have been meticulously analyzed and optimized to address the demands and challenges inherent in managing large-scale health datasets and user interactions within the SoBigData platform. This process has revealed critical insights into system bottlenecks and performance thresholds, informing our approach to system enhancement and ensuring the platform remains responsive and efficient under various load conditions.

Our commitment to optimizing the system's performance is unwavering. This document unfolds the in-depth analysis undertaken to meet performance metrics, offering a narrative that chronicles the iterative process of refinement and optimization that embodies the core ethos of the SoBigData initiative—efficiency, collaboration, and innovation.

2. Sequence Diagrams Overview & Chosen Functionalities

In our project, we aimed to address a diverse set of functionalities derived from our sequence diagrams. To ensure comprehensive coverage, we selected one key functionality from each category of sequence diagrams. These functionalities serve as focal points for our implementation efforts in Homework 2.

As a result, the chosen functionalities are as follows:

1. **Browse Health Dataset**: This functionality enables users to explore and access health-related datasets within the SoBigData++ platform, facilitating research and analysis in the field of health informatics.
2. **Communicate with Health Professionals**: Enabling real-time communication with health professionals, this feature fosters collaboration and knowledge exchange, empowering users to seek expert guidance and insights.
3. **Contribute Resource Findings**: Users are empowered to contribute their research findings and insights to the platform, enriching the collective knowledge base and facilitating collaboration among researchers.
4. **Provide Feedback on Studies**: This functionality allows users to provide feedback and comments on studies and research findings, fostering a culture of peer review and constructive criticism within the platform.

By focusing on these functionalities, we aim to showcase the platform's capabilities in supporting diverse interactions and workflows, as depicted in our sequence diagrams.

2.1 Chosen Functionalities

In this homework, our attention was directed towards the performance requirements, which vary depending on the specific use case:

1. **Browse Health Dataset**:
    - **Response Time Requirement**: The system should respond within 5 seconds under a maximum workload of 20 requests/second.
2. **Communicate with Health Professionals**:
    - **Response Time Requirement**: Sending and receiving a message in the chat should have a response time of 4 seconds.

- **Throughput Requirement**: The chat system should support a minimum throughput of 10 requests/second under a maximum workload of 20 requests/second.

3. **Contribute Resource Finding**:
    - **Throughput Requirement:** The system should support a minimum of 5 request/second (meaning uploads of resources per second) by researchers under a maximum workload of 10 requests/second.
    - **Utilization Requirement:** The total utilization of CPU, Disk, and Network should be maintained below 80% during the resource publication of the researcher.
4. **Provide Feedback on Studies**:
    - **Response Time Requirement:** Providing feedback on studies should have a response time of 5 seconds.
    - **Throughput Requirement:** The system should support a minimum throughput of 5 request / 4 second

3. Execution Graphs :

In our project, we encountered challenges when translating demands from execution graphs into service times within the JMT (Java Modelling Tools) environment. Initially, we used a scale of 0 to 100 to describe the demands of each instruction node in our execution graphs. However, during our iterative process, we found it more convenient and accurate to map these values into a smaller scale of 0 to 20. This adjustment facilitated easier and more precise calculations when transferring the numbers into service times inside JMT.

Additionally, we discovered that JMT had difficulties handling values of 0, particularly when no Network or Disk usage was necessary. To address this issue, we adopted a workaround by assigning a very small value of 1% demand for these cases, which is negligible in practical terms but allows JMT to process the data effectively. However, it's important to note that while we adjusted these values within JMT, we retained the original 0 values in our execution graphs for clarity and consistency.

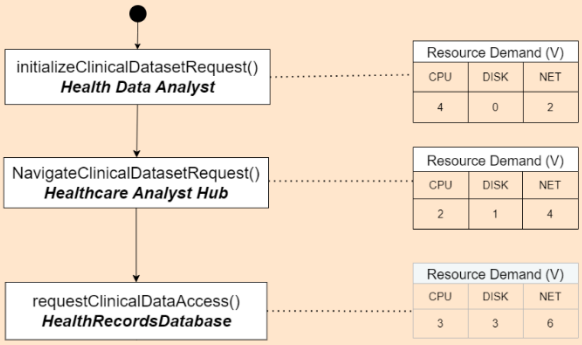
As for determining the demand values for each request, we considered the specific requirements and functionalities associated with each operation.

3.1 **Browse Health Dataset Graph**:

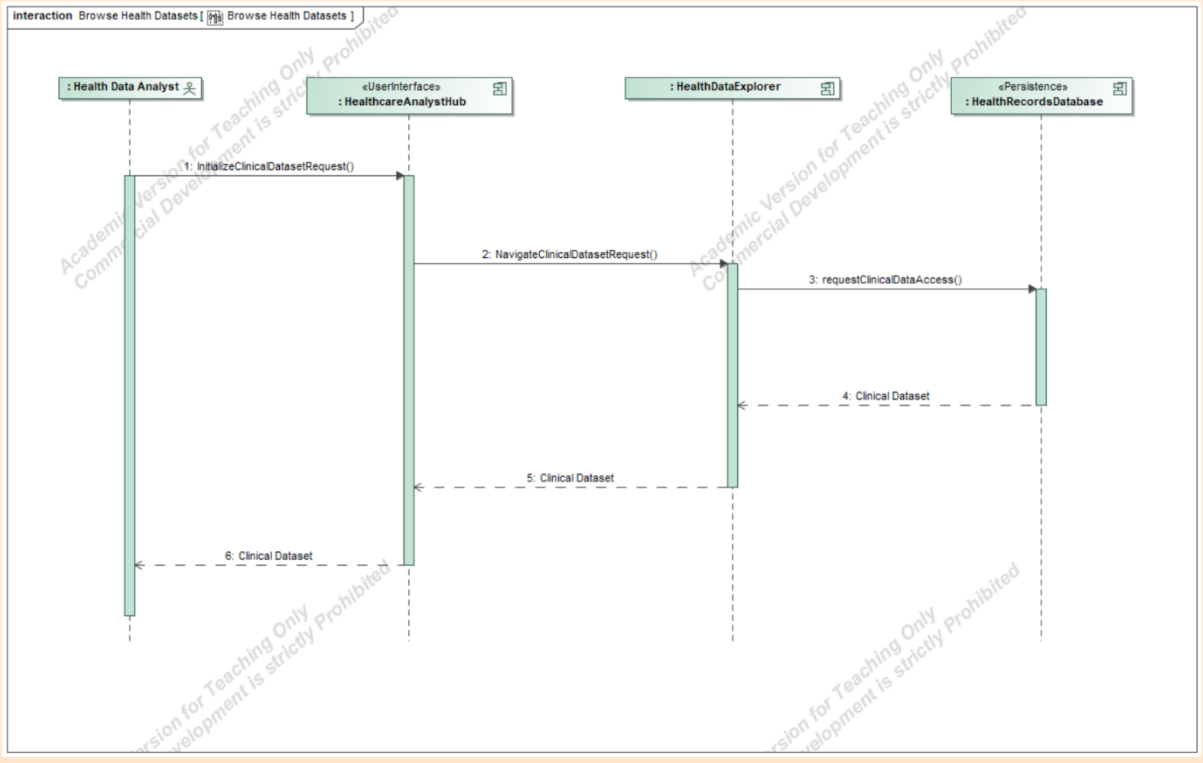
Given the nature of the operations involved in browsing a health dataset, the utilization of disk resources primarily occurs during the **requestClinicalDataAccess** phase.
This is because accessing clinical data often involves reading from disk storage where the dataset is stored. During the initialization and navigation phases (**initializeClinicalDatasetRequest** and **NavigateClinicalDatasetRequest**), disk utilization is minimal as these operations primarily involve setting up the environment and navigating metadata, which can typically be performed in memory or require minimal disk interaction."

1. initializeClinicalDatasetRequest:
    - CPU: 4
    - Disk: 0
    - Network: 2
2. NavigateClinicalDatasetRequest:
    - CPU: 2
    - Disk: 1
    - Network: 4
3. requestClinicalDataAccess:
    - CPU: 3
    - Disk: 3
    - Network: 6

Browse Health Dataset



Total Resource Demand (V)		
CPU	DISK	NET
9	4	12



3.2 **Communicate with Health Professionals Graph**:

In this sequence diagram for "**Communicate with Health Professionals**" the graph depicts the concurrent execution of storing messages and verifying the online status of the receiver. Through the utilization of a parallel node, these two essential operations are performed simultaneously.

It is crucial to note that both tasks must be completed before advancing to the subsequent part of the graph, represented by the case node.

Within the case node, the diagram illustrates a probability-based decision-making process. With a 25% probability, the "**isOnline**" expanded node is executed, indicating that the receiver is online.

Conversely, there is a 75% probability that the "**isNotOnline**" expanded node is executed, signaling that the receiver is not online.

These two expanded nodes are further elaborated in their respective subgraphs, providing detailed insights into their functionalities and interactions within the system.

1. InitialeSecureHealhMessageCreation:
    - CPU: 10
    - Disk: 5
    - Net: 8
    - Explanation: Creating a secure health message likely involves intensive CPU processing to encrypt the message securely. Disk usage might be moderate for temporarily storing the message before transmission, and network usage would be relatively high due to the transmission of encrypted data.
2. InitiateSecureCommunication:
    - CPU: 7
    - Disk: 3
    - Net: 7
    - Explanation: Initiating secure communication would require moderate CPU processing to establish the secure connection. Disk usage might be minimal, and network usage would be high as it involves establishing a secure communication channel over the network.
3. SaveSecureMessage:
    - CPU: 6
    - Disk: 6

- Net: 3
- Explanation: Saving the secure message involves moderate CPU processing for encryption and disk usage for storing the encrypted message. Network usage might be minimal as it doesn't involve network transmission.

4. isOnline (Online Status Check):
    - CPU: 3
    - Disk: 1
    - Net: 2
    - Explanation: Checking if the recipient is online would require minimal CPU processing for the status check, minimal disk usage, and moderate network usage for sending a status request and receiving a response.
5. isOnline (Expanded Node - Online):
    - CPU: 2
    - Disk: 1
    - Net: 1
    - Explanation: If the recipient is online, minimal CPU processing, disk usage, and network usage are required for decision-making and potential message transmission.
6. isNotOnline (Expanded Node - Offline):
    - CPU: 2
    - Disk: 1
    - Net: 1
    - Explanation: If the recipient is not online, similar to the online scenario, minimal CPU processing, disk usage, and network usage are required for decision-making without message transmission.

For “isOnline” :

1. transmitSecureMessage:
    - CPU: 5
    - Disk: 2
    - Net: 6
    - Explanation: Transmitting a secure message would require moderate CPU processing for encryption and decryption, minimal disk usage for temporary storage, and high network usage for transmitting encrypted data.
2. relaySecureHealthMessages:
    - CPU: 3
    - Disk: 1
    - Net: 4

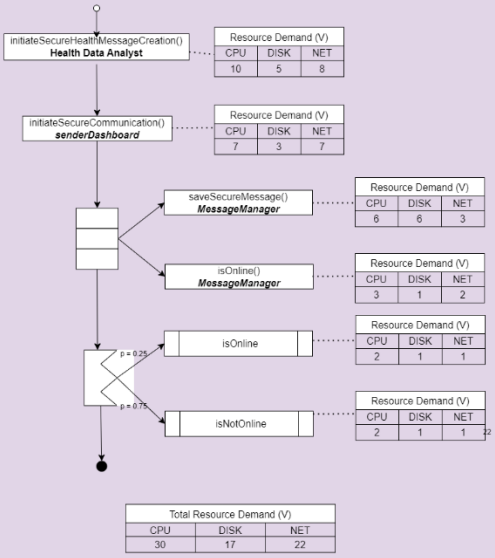
- Explanation: Relaying secure health messages would involve minimal CPU processing, minimal disk usage, and moderate network usage for transmitting messages between nodes.

3. message (Expanded Node - Online):
    - CPU: 2
    - Disk: 1
    - Net: 1
    - Explanation: If the recipient is online, minimal CPU processing, disk usage, and network usage are required for decision-making without message transmission.

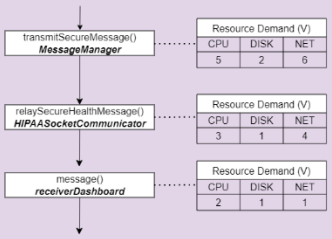
For “isNotOnline”:

1. retrieveUnseenHealthMessages:
    - CPU: 4
    - Disk: 2
    - Net: 3
    - Explanation: Retrieving unseen health messages would involve moderate CPU processing for data retrieval, moderate disk usage for storing retrieved messages, and moderate network usage for retrieving messages from the server.
2. receiveSecureMessage:
    - CPU: 3
    - Disk: 1
    - Net: 4
    - Explanation: Receiving a secure message would require minimal CPU processing, minimal disk usage, and high network usage for receiving encrypted data.
3. getMessage (Expanded Node - Offline):
    - CPU: 2
    - Disk: 1
    - Net: 1
    - Explanation: If the recipient is not online, minimal CPU processing, disk usage, and network usage are required for decision-making without message retrieval.

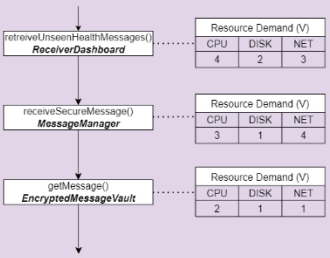
Communicate with Health Professionals

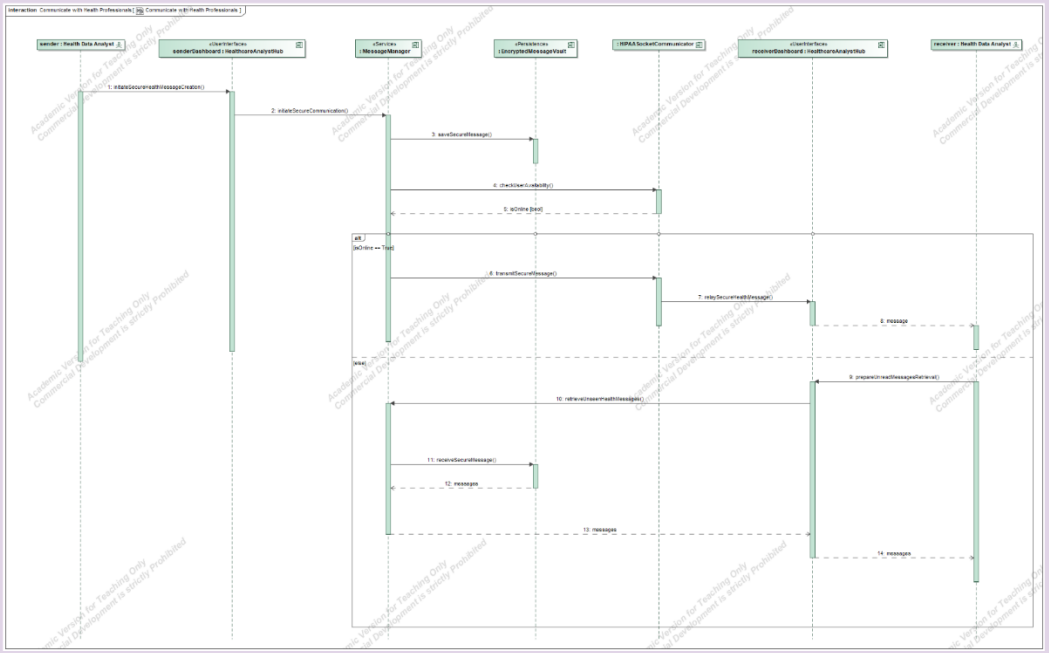


isOnline



isNotOnline





3.3 **Contribute Resource Findings**:

For the sequence diagram "Contribute Resource Findings," the graph structure and operations are as follows:

1. **submitHealthDataUploadInquiry**:
    - This step initiates the process of submitting a health data upload inquiry to the system.
    - The graph flow for this step is linear, indicating a sequential execution of operations.
2. **resourcePublishRequest**:
    - Sends a request to publish the health resource finding to the system.
    - Similar to the previous step, this operation follows a linear flow.
3. **Parallel Node (Insert resource and Notify)**:
    - This step executes two operations in parallel: inserting the resource into the system and notifying users or stakeholders about the new resource.
    - The use of a parallel node allows for concurrent execution of these operations, improving efficiency.
4. **approveHealthResearchSubmissions**:
    - Approves the submission of health research findings to the system.
    - This step, like the others, follows a linear flow in the graph.

In this sequence diagram, we used a combination of linear flows and parallel nodes to efficiently handle the process of contributing resource findings to the system.

The use of parallel execution for insert resource and notify operations ensures timely processing and notification to stakeholders. Finally, the approval step signifies the completion of the contribution process, marking the successful submission of health research findings to the system.

here's a brief explanation for Resource Demand for each step:

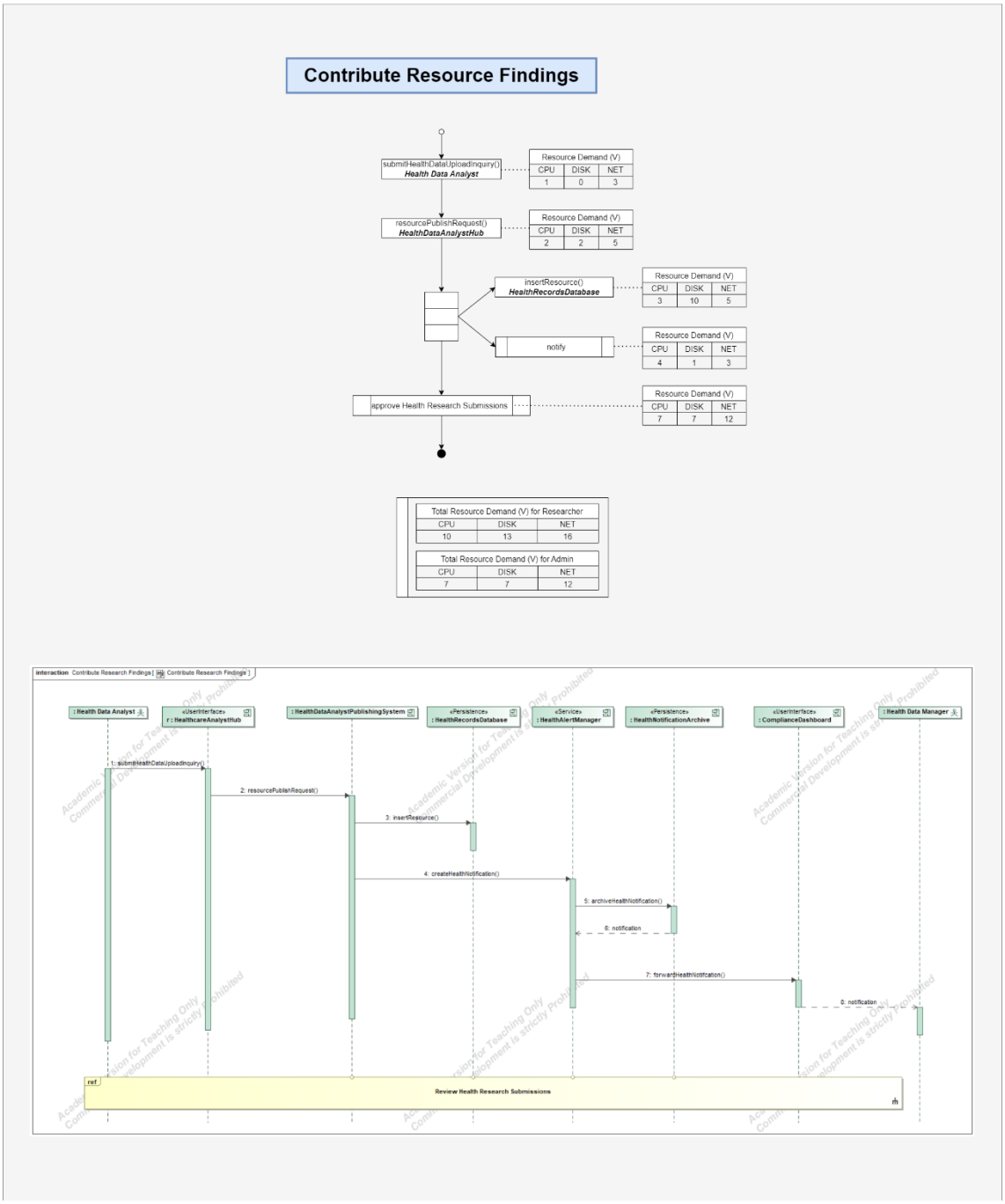
1. **submitHealthDataUploadInquiry**:
    - Explanation: Initiates the process of submitting a health data upload inquiry to the system.
    - Resource Demand:
        - CPU: 1
        - Disk: 0
        - Net: 3

- Explanation: Moderate CPU processing for inquiry submission, moderate disk usage for storing inquiry data temporarily, and moderate network usage for sending the inquiry to the server.

2. **resourcePublishRequest**:
    - Explanation: Sends a request to publish the health resource finding to the system.
    - Resource Demand:
        - CPU: 2
        - Disk: 2
        - Net: 5
        - Explanation: Moderate to high CPU processing for request processing, moderate disk usage for storing request data temporarily, and moderate to high network usage for sending the request to the server.

3. **Parallel Node (Insert resource and Notify)**:
    - Explanation: Executes two operations in parallel: inserting the resource into the system and notifying users or stakeholders about the new resource.
    - Resource Demand:
        - Insert resource:
            - CPU: 3
            - Disk: 10
            - Net: 5
            - Explanation: Moderate to high CPU processing for resource insertion, moderate to high disk usage for storing the resource data, and moderate to high network usage for transmitting the resource data.
        - Notify:
            - CPU: 4
            - Disk: 1
            - Net: 3
            - Explanation: Moderate CPU processing for notification processing, moderate disk usage for storing notification data temporarily, and moderate network usage for sending notifications to users.
4. **approveHealthResearchSubmissions**:
    - Explanation: Approves the submission of health research findings to the system.

- Resource Demand:
    - CPU: 7
    - Disk: 7
    - Net: 12
    - Explanation: Moderate to high CPU processing for approval processing, moderate disk usage for storing approval data temporarily, and moderate to high network usage for sending the approval to the server.



3.4 **Provide Feedback on Studies**

Approve Feedback on Study Subgraph:

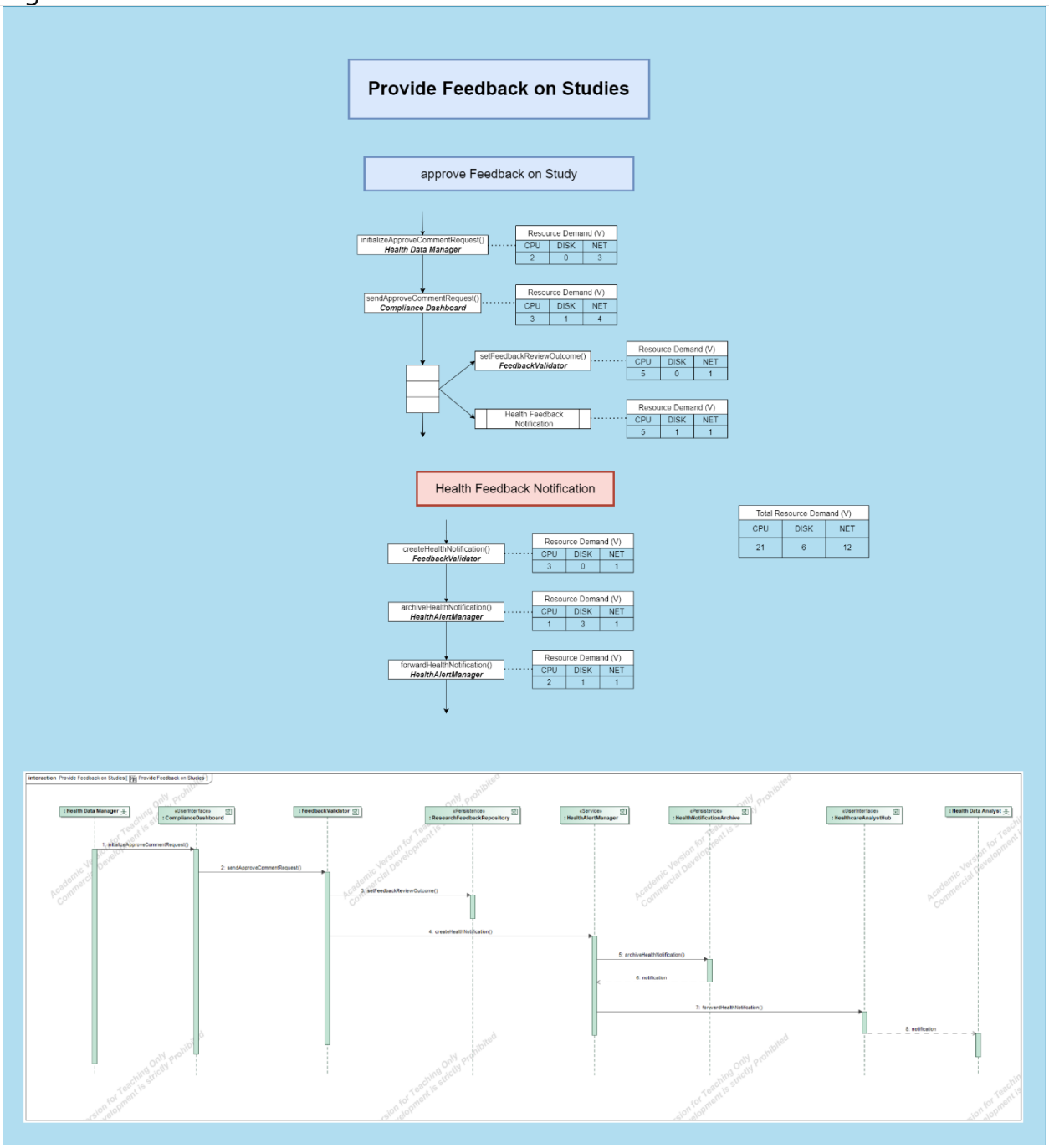
1. **InitializeApproveCommentRequest**:
    - CPU: 2 (Moderate CPU usage due to initial processing of the request)
    - DISK: 0 (No disk operations if the request is in memory)
    - NET: 3 (Higher network usage to communicate with other services)
2. **SendApproveCommentRequest:**
    - **CPU:** 3 (Higher CPU for processing and validating the comment data)
    - DISK: 1 (Minimal disk activity for logging or temporary storage)
    - NET: 4 (Significant network activity to send data to the next service)
3. **Parallel Node**:
    - **setFeedbackReviewOutcome:**
        - CPU: 5 (Intensive CPU usage for analytical processes and decision logic)
        - DISK: 0 (Moderate disk usage for writing outcomes to a database)
        - NET: 1 (Minimal network use; action is internal to the service)
    - **Health Feedback Notification** (cumulative for parallel processes):
        - CPU: 5 (Same as above; assumes similar processing requirements)
        - DISK: 1 (Disk usage for logging and queuing notifications)
        - NET: 1 (Moderate network use to queue notification messages)

Health Feedback Notification Subgraph:

1. **createHealthNotification**:
    - CPU: 3 (Creation of notifications involves data processing and template rendering)
    - DISK: 0 (Assuming notification is created in-memory without immediate persistence)
    - NET: 1 (Minimal network use for internal service communication)
2. **archiveHealthNotification**:
    - CPU: 1 (Low CPU since the task is primarily writing to storage)
    - DISK: 3 (Higher disk usage as notifications are being archived to a database)
    - NET: 1 (Minimal network activity as archiving may happen on the same server)
3. **forwardHealthNotification**:
    - CPU: 2 (Some processing to determine the correct recipient and route the message)
    - DISK: 1 (Minimal disk activity unless the notification is logged)
    - NET: 1 (Network activity to send the notification to an external service or user)

These values reflect a typical pattern where initialization and sending operations are more network-intensive, while review and notification creation require more CPU for processing data. Archiving involves more disk usage since it's primarily a storage

operation. Forwarding notifications is a balance between CPU and network usage depending on the system's implementation details and the complexity of the routing logic for notifications.



4. Worst-Case Analysis :

When conducting the worst-case standalone analysis, we chose the following maximum service times (in seconds):

CPU	DISK	NET
0.01	0.05	0.0125

4.1 Browse Health Dataset :

The Total Resource Demanded in the "Browse Health Dataset" as it displayed in the Execution Graph is:

CPU	DISK	NET
9	4	12

- ✓ **Response Time Requirement**: The system should respond within 5 seconds under a maximum workload of 10 requests/second.

Worst-Case Analysis:

- CPU: 9 x 0.01 = 0.9 sec
- DISK: 4 x 0.05 = 0.2 sec
- NET: 12 x 0.0125 = 0.35 sec

- ✓ TOTAL (sum): 0.44 sec
- ✓ Number of seconds needed to complete 10 requests: 0.44 x 10 = 4.4 sec

- ❖ In the Worst-Case scenario, the system meets the response time requirement

4.2 Communicate with Health Professionals :

The Total Resource Demanded in the "Communicate with Health Professionals" as it displayed in the Execution Graph is:

CPU	DISK	NET
30	17	22

✓ **Response Time Requirement**: The system should respond within 2 seconds

Worst-Case Analysis:

- CPU: 30 x 0.01 = 0.3 sec
- DISK: 17 x 0.05 = 0.85 sec
- NET: 22 x 0.0125 = 0.275 sec

✓ TOTAL (sum): 1.425 sec

❖ In the Worst-Case scenario, the system meets the response time requirement.

4.3 Contribute Resource Finding :

The Total Resource Demanded in the "Contribute Resource Finding" as it displayed in the Execution Graph is:

CPU	DISK	NET
10	13	16

✓ **Response Time Requirement**: The total utilization of CPU, Disk, and Network should be maintained below 80% during the resource publication of the researcher.

Worst-Case Analysis:

- CPU: 10 x 0.01 = 0.1 sec
- DISK: 13 x 0.05 = 0.65 sec

- NET: 16 x 0.0125 = 0.2 sec

✓ TOTAL (sum): 0.95 sec

❖ In the Worst-Case scenario: (10+13+16) * 100 / 60 = 65%

➢ the system meets the response time utilization requirement.

4.4 Provide Feedback on Studies:

The Total Resource Demanded in the "Provide Feedback on Studies" as it displayed in the Execution Graph is:

CPU	DISK	NET
21	6	12

✓ **Response Time Requirement**: The system should respond within 2 seconds

Worst-Case Analysis:

- CPU: 21 x 0.01 = 0.21 sec
- DISK: 6 x 0.05 = 0.45 sec
- NET: 12 x 0.0125 = 0.275 sec

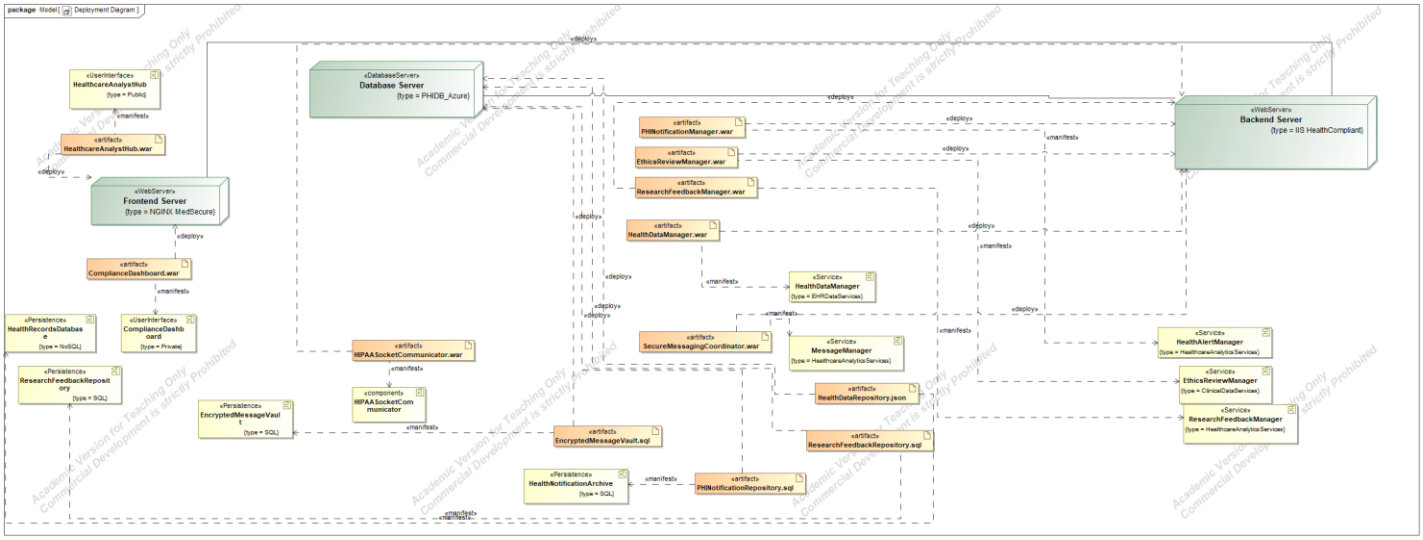
✓ TOTAL (sum): 0.785 sec

❖ In the Worst-Case scenario: 0.785 * 5 = 3.925, the system meets the response time requirement.

5. Queueing Network Model:

5.1 Explanation:

Starting with our Deployment Diagram, our Queueing Network was modeled, which depicts three nodes as shown in the picture: Frontend Server, Backend Server, and DataBase Server.



We segmented these nodes into corresponding resources ”CPU, DISK, and NET” as we had previously done in the execution graphs.

The flow was then identified using the sequence diagrams for the functionalities:

Browse Health Dataset, Communicate with Health Professionals, Contribute Resource Finding, and Provide Feedback on Studies.

We categorized the jobs into classes based on these functionalities and their corresponding “Done” classes, plus a temporary class that aids in proper flow routing.

Define customer classes

Classes Characteristics

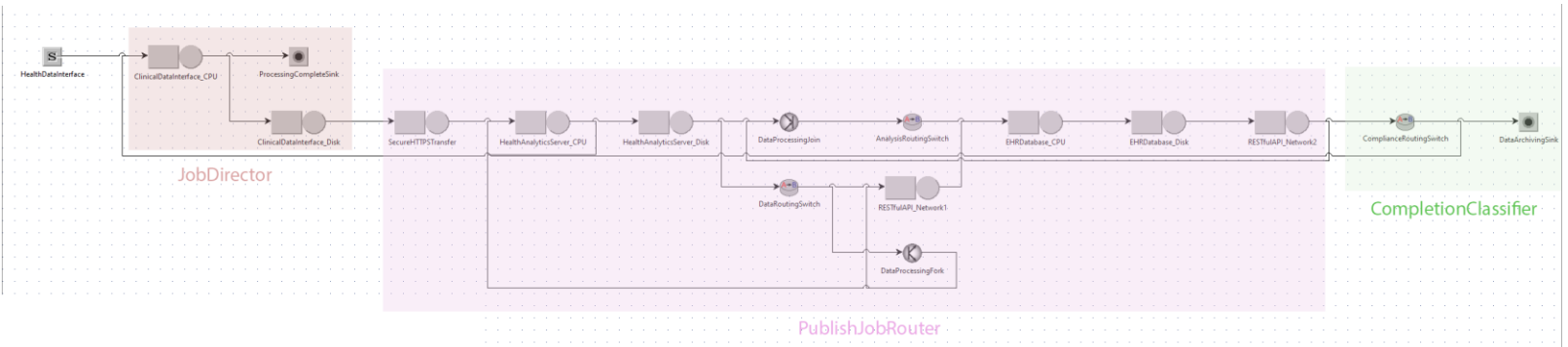
Define type (Open or Closed), name and parameters for each customer class.
Closed Classes: If a **ClassSwitch** is in the model, then **all** the closed classes must have the **same** reference station.
Open Classes: An open class that has **Fork**, **ClassSwitch**, **Scaler** or **Transition** as the reference station is **not** generated by **any** Source.
Priorities: A larger value implies a higher priority.

Add Class

Classes: 8

Color	Name	Type	Priority	Population	Interarrival Time Distribution		Soft Deadline	Reference Station	
	BrowseHealthRecords	Open	0		exp(10)	Edit	0.0000	HealthDataInterface	
	CompletedHealthRecordsBrowsing	Open	0				0.0000	ClassSwitch	
	InitiateCommunication	Open	0		exp(5)	Edit	0.0000	HealthDataInterface	
	CompletedCommunication	Open	0				0.0000	ClassSwitch	
	ContributeResourceFinding	Open	0		exp(2)	Edit	0.0000	HealthDataInterface	
	CompletedContributeResourceFinding	Open	0				0.0000	ClassSwitch	
	ProvideFeedbackStudies	Open	0				0.0000	ClassSwitch	
	CompletedProvidingFeedback	Open	0				0.0000	ClassSwitch	

Done



JobDirector :

This component manages the initial job distribution within the system. It directs jobs to either the ClinicalDataInterface_CPU for processing or to the ClinicalDataInterface_Disk if they are completed. The SecureHTTPTransfer ensures that data is moved securely to the next phase, leading to the HealthAnalyticsServer_CPU, which processes the analytical tasks.

PublishJobRouter :

After processing at the HealthAnalyticsServer_Disk, the PublishJobRouter takes over. It serves to reroute the Publish job by changing it into a PublishTemp job. This temporary job bifurcates into two parallel paths via the DataProcessingFork:

- The first path sends the job through the EHRDatabase_CPU, indicating the data storage step, then via the RESTfulAPI_Network2, it finally converges at AnalysisRoutingSwitch.
- The second path redirects the job back towards HealthAnalyticsServer_CPU, signifying the admin notification process, and merges at the same AnalysisRoutingSwitch.

CompletionClassifier :

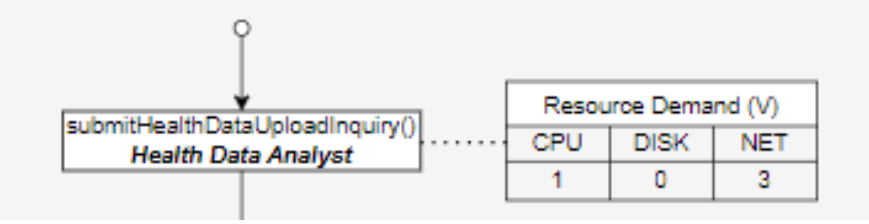
This is a critical component that determines the endpoint for each job based on its completion status. The CompletionClassifier changes the job class into its "done" state. For instance, DonePublish jobs exit the system at the DataArchivingSink, while other completed job types like DoneBrowseResource might be routed to the ClinicalDataInterface_CPU for any final computations before they exit the system.

6.1 Parametrize:

For the Queueing Network parameterization, we relied on the performance requirements that stipulate the arrival rate for job classes.

The "mean" parameter of the exponential distribution, representing our queues' service time distribution, was informed by the CPU, DISK, and NET service times and the resource demands (calculated as ServiceTime x ResourceDemand).

Subsequent to parameterization, the network is ready for resolution.



Taking the submitHealthDataUploadInquiry() function, for example, which operates on the Frontend Server, the impact is observed on the corresponding FrontendServer nodes: HealthAnalytics_CPU, HealthAnalyticsServer_Disk, and the SecureHTTPSTransfer, which reflects the network demands of the Frontend.

We then assess the default hardware component values that have been set previously:

CPU DISK NET : 0.01 0.05 0.0125

By applying the execution graphs' resource demands to the hardware component's default values, we derive the ensuing results:

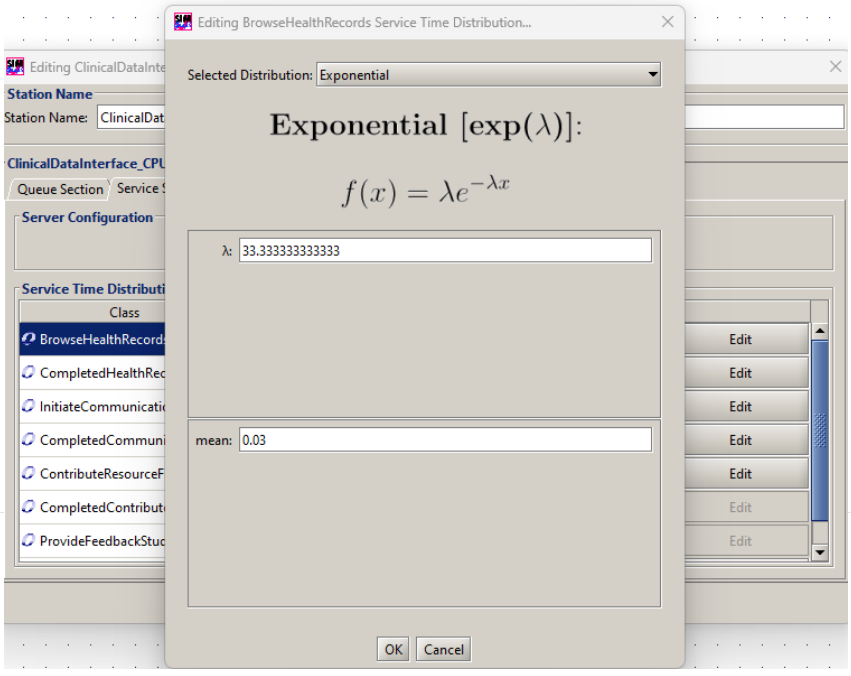
📌 Calculation:

HealthAnalytics_CPU ← 1 * *0.01 = 0.01*

*HealthAnalyticsServer_Disk ← 0 * 0.05* = 0

SecureHTTPSTransfer ← 3 * 0.0125 = 0.0375

The JMT tool then utilizes these values within the mean parameter for the respective class and station.

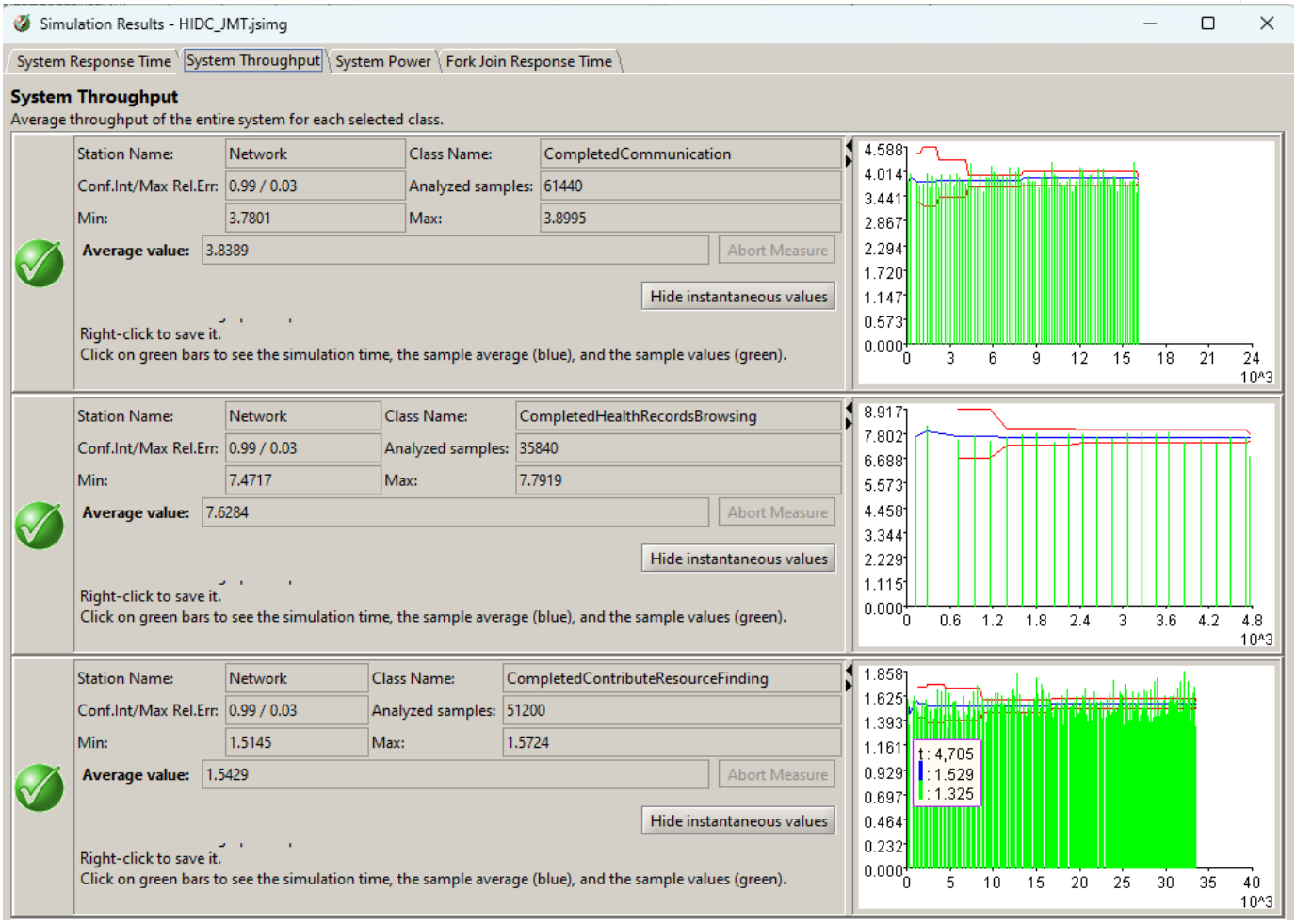


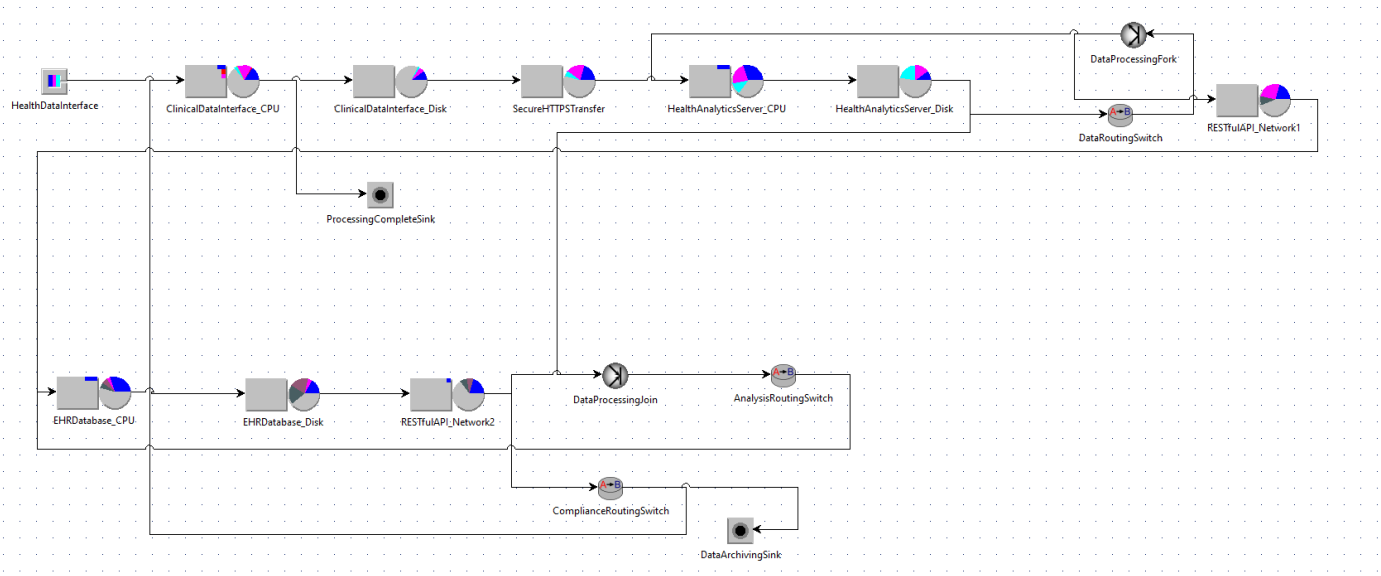
6. Bottleneck Analysis and Resolution Post-Simulation:

The execution of the simulation has led to the identification of three primary bottlenecks within the system. They are as follows:

1. **Contribute Resource Finding Bottleneck:**
   The operation "insertResource" presented a significant delay, manifesting as the first bottleneck. This was attributed to an elevated Disk demand value of 13, which when computed using our service time equation, resulted in a prolonged processing time of 0.65 seconds for each individual request. The optimal strategy employed to mitigate this issue was to increase the number of servers at the HealthAnalyticsServer_Disk node to three, which effectively alleviated the congestion.
2. **Provide Feedback on Studies Bottleneck:**
   Similarly, the "archiveHealthNotification" operation emerged as a second bottleneck, stemming from an identical complication as the first, with high disk demand causing delays. Implementing the same resolution as for the first bottleneck, which involved the enhancement of server capacity at the respective node, proved successful in resolving this bottleneck.
3. **Communicate with Health Professionals Bottleneck:**
   The third bottleneck was observed within the "Communicate with Health Professionals" functionality. However, upon a detailed review, it was decided to categorize this as a non-critical issue and thus, it was not addressed in the same manner as the others. The rationale behind this decision is rooted in the system design, particularly the functionality for offline users. Considering that only 25% of users are expected to be online at any given time, it was determined that the notification delivery to offline Health Professionals—which only occurs upon their return online—need not be factored into the immediate demand. This led to a strategic decision to prioritize system resources and focus exclusively on the online user interactions. Consequently, this targeted approach resolved the bottleneck without necessitating changes to the system's infrastructure for the offline component.

The interventions detailed above are based on a careful examination of system performance under simulated conditions. These modifications are aimed at optimizing the throughput and ensuring a more efficient flow of operations within the network.


Simulation Results - HIDC_JMT.jsimg
System Response Time
System Throughput
System Power
Fork Join Response Time
System Throughput
Average throughput of the entire system for each selected class.
Station Name: Network
Class Name: CompletedCommunication
Conf.Int/Max Rel.Err: 0.99 / 0.03
Analyzed samples: 61440
Min: 3.7801
Max: 3.8995
Average value: 3.8389
Abort Measure
Hide instantaneous values
Right-click to save it.
Click on green bars to see the simulation time, the sample average (blue), and the sample values (green).
4.588
4.014
3.441
2.867
2.294
1.720
1.147
0.573
0.000
0 3 6 9 12 15 18 21 24
10^3
Station Name: Network
Class Name: CompletedHealthRecordsBrowsing
Conf.Int/Max Rel.Err: 0.99 / 0.03
Analyzed samples: 35840
Min: 7.4717
Max: 7.7919
Average value: 7.6284
Abort Measure
Hide instantaneous values
Right-click to save it.
Click on green bars to see the simulation time, the sample average (blue), and the sample values (green).
8.917
7.802
6.688
5.573
4.458
3.344
2.229
1.115
0.000
0 0.6 1.2 1.8 2.4 3 3.6 4.2 4.8
10^3
Station Name: Network
Class Name: CompletedContributeResourceFinding
Conf.Int/Max Rel.Err: 0.99 / 0.03
Analyzed samples: 51200
Min: 1.5145
Max: 1.5724
Average value: 1.5429
Abort Measure
Hide instantaneous values
Right-click to save it.
Click on green bars to see the simulation time, the sample average (blue), and the sample values (green).
1.858
1.625
1.393
1.161
0.929
0.697
0.464
0.232
0.000
t: 4,705
: 1.529
: 1.325
0 5 10 15 20 25 30 35 40
10^3


HealthDataInterface
ClinicalDataInterface_CPU
ClinicalDataInterface_Disk
SecureHTTPSTransfer
HealthAnalyticsServer_CPU
HealthAnalyticsServer_Disk
DataProcessingFork
DataRoutingSwitch
RESTfulAPI_Network1
ProcessingCompleteSink
EHRDatabase_CPU
EHRDatabase_Disk
RESTfulAPI_Network2
DataProcessingJoin
AnalysisRoutingSwitch
ComplianceRoutingSwitch
DataArchivingSink

7. What-If Analysis:

1. **What if the system experiences a sudden surge in user activity?**
    - *JMT Approach*: We doubled the arrival rate of jobs to represent the increase in simultaneous user access.
    - Set Up the Scenario: We create a copy of your current model in JMT.
    - Adjust Parameters: Double the arrival rates of jobs to the Browse Health Dataset and exactly “SearchHealthRecords” Class functionality.

✓ **Solution**:

After running simulations to assess the effect of doubling the arrival rate for the `SearchHealthRecords` functionality, we observed the following key points which informed our resolution strategy:

- **System Power Trend**: The system's power initially increases, suggesting that more computational resources are being engaged as the arrival rate rises. However, the system power starts to decline after 15 arrivals per second, indicating a point of diminishing returns, possibly due to resource saturation.
- **Response Time Increase**: The average system response time for `CompletedHealthRecordsSearch` remained stable up to approximately 16 arrivals per second, beyond which it sharply increased. This sharp increase indicates that the system’s capacity to handle additional load without degrading performance is being exceeded.
- **Throughput Linearity**: The system's throughput increased linearly with the arrival rate, which suggests that while the system can handle more work initially, the quality of service might be compromised as indicated by the rise in response times.

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

System Power

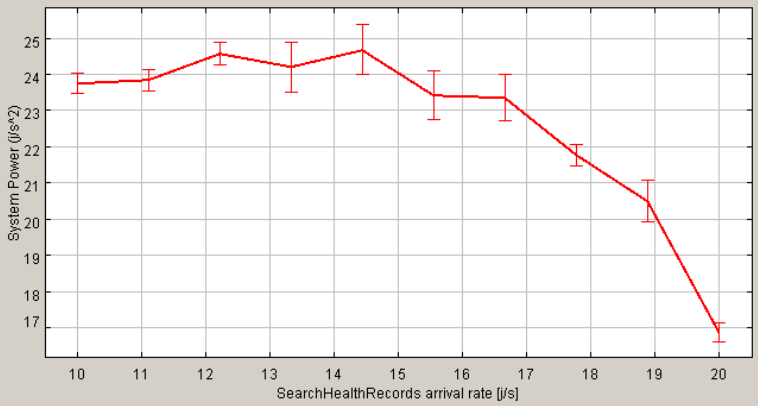
Average power of the entire system for each selected class.

Description

Station Name:	Network
Class Name:	All classes
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	10
X min:	10
X max:	20
Y min:	16.604
Y max:	25.386

Plot

Show confidence interval range



Simulation Results

*	10.000 j/s	11.111 j/s	12.222 j/s	13.333 j/s	14.444 j/s	15.556 j/s	16.667 j/s	17.778 j/s	18.889 j/s	20.000 j/s
Mean Value (j/s^2)	23.7498	23.8282	24.5710	24.1860	24.6673	23.4055	23.3516	21.7614	20.4927	16.8626
Max (j/s^2) (Conf.Int.)	24.0403	24.1271	24.8839	24.9025	25.3862	24.1113	24.0038	22.0474	21.0824	17.1299
Min (j/s^2) (Conf.Int.)	23.4662	23.5366	24.2659	23.5095	23.9879	22.7399	22.7339	21.4827	19.9350	16.6036

The values in red were not computed with the requested precision

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

System Throughput

Average throughput of the entire system for each selected class.

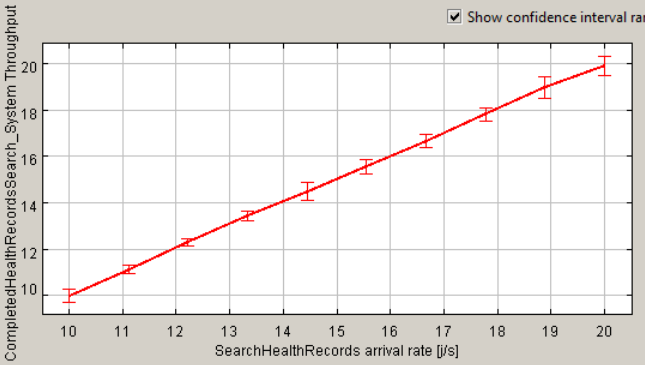
The values in red were not computed with the requested precision

Description

Station Name:	Network
Class Name:	CompletedHealthRecordsSearch
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	10
X min:	10
X max:	20
Y min:	9.692
Y max:	20.317

Plot

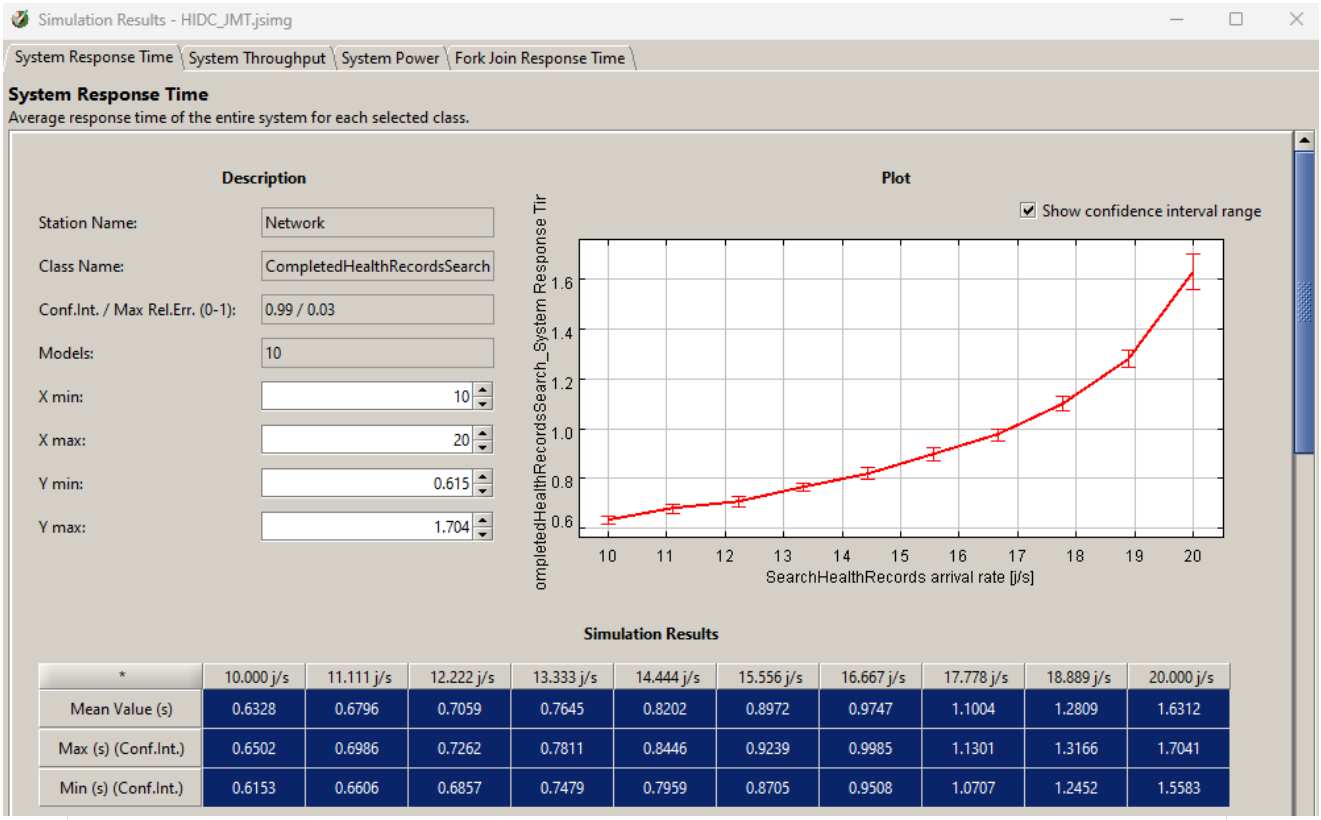
Show confidence interval range



Simulation Results

*	10.000 j/s	11.111 j/s	12.222 j/s	13.333 j/s	14.444 j/s	15.556 j/s	16.667 j/s	17.778 j/s	18.889 j/s	20.000 j/s
Mean Value (j/s)	9.9689	11.1087	12.2925	13.4312	14.4641	15.5365	16.6553	17.7997	18.9516	19.8853
Max (j/s) (Conf.Int.)	10.2620	11.2970	12.4449	13.6437	14.8557	15.8404	16.9461	18.0851	19.4060	20.3175
Min (j/s) (Conf.Int.)	9.6922	10.9266	12.1439	13.2253	14.0927	15.2440	16.3742	17.5233	18.5179	19.4712

The values in red were not computed with the requested precision



2. **What if network latency increases?**
    - *JMT Approach*: Simulate higher network latency by adjusting the service times at communication nodes. We Analyze the impact on real-time communication features like Chat with Other Researchers.

✓ **Solution**:

- The system's response time increases as the service time at RESTfulAPI_Network1 increases, which is expected behavior. Beyond a certain point, the rise in response time becomes more pronounced, indicating a threshold beyond which the user experience may be significantly impacted.
- The system power, which can be thought of as a measure of resource utilization weighted by time, decreases as the service time increases. This decrease might suggest that the system becomes less efficient as it spends more time waiting on network-related tasks.

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

System Response Time

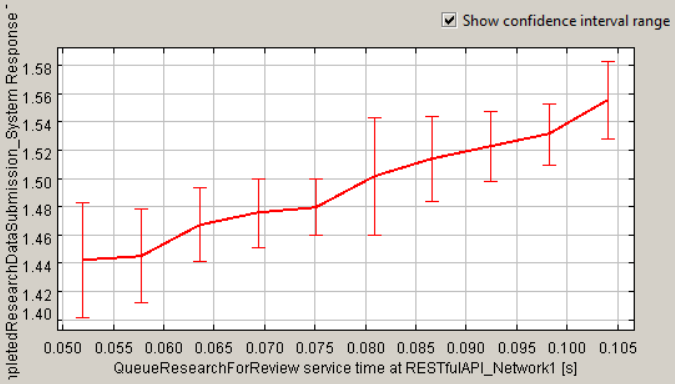
Average response time of the entire system for each selected class.

The values in red were not computed with the requested precision

Description

Station Name:	Network
Class Name:	CompletedResearchDataSubmission
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	10
X min:	0.052
X max:	0.104
Y min:	1.401
Y max:	1.583

Plot



Simulation Results

*	0.052 s	0.058 s	0.064 s	0.069 s	0.075 s	0.081 s	0.087 s	0.092 s	0.098 s	0.104 s
Mean Value (s)	1.4421	1.4450	1.4671	1.4753	1.4797	1.5011	1.5137	1.5229	1.5313	1.5557
Max (s) (Conf.Int.)	1.4830	1.4782	1.4934	1.4994	1.4995	1.5426	1.5435	1.5477	1.5530	1.5830
Min (s) (Conf.Int.)	1.4012	1.4117	1.4409	1.4513	1.4600	1.4595	1.4839	1.4980	1.5095	1.5284

The values in red were not computed with the requested precision

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

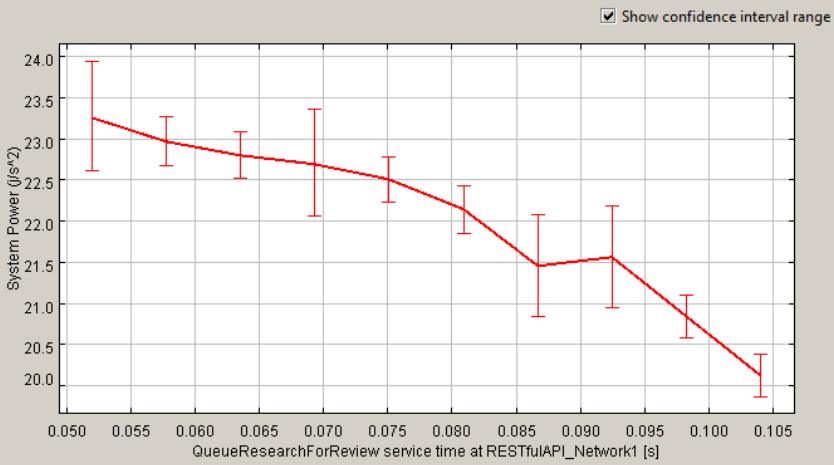
System Power

Average power of the entire system for each selected class.

Description

Station Name:	Network
Class Name:	All classes
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	10
X min:	0.052
X max:	0.104
Y min:	19.864
Y max:	23.944

Plot



Simulation Results

*	0.052 s	0.058 s	0.064 s	0.069 s	0.075 s	0.081 s	0.087 s	0.092 s	0.098 s	0.104 s
Mean Value (j/s^2)	23.2621	22.9693	22.8033	22.6961	22.5031	22.1353	21.4465	21.5558	20.8401	20.1219
Max (j/s^2) (Conf.Int.)	23.9435	23.2644	23.0943	23.3646	22.7840	22.4288	22.0840	22.1891	21.1036	20.3860
Min (j/s^2) (Conf.Int.)	22.6184	22.6816	22.5196	22.0648	22.2291	21.8493	20.8447	20.9576	20.5831	19.8645

The values in red were not computed with the requested precision

3. **What if we quadruple the server count?**

   If the current server capacity is increased fourfold, what impact would this have on the system's overall throughput and response time, particularly during peak usage periods?

✓ **Solution**:

Based on the JMT simulation results, it appears that increasing the number of EHRDatabase_CPU servers beyond two does not significantly improve the system's response time.

Therefore, for the current workload and system configuration, two servers are sufficient to handle the tasks efficiently. Adding more servers beyond this does not yield a proportional benefit.

For System Response Time: There's a slight decrease in the response time when the number of servers increases from one to two. Beyond two servers, the response time does not significantly improve, which indicates that two servers are currently optimal for our system in terms of response time.

For System Power: The system's power usage increases as more servers are added, suggesting that having more than two servers doesn't contribute to efficiency and may lead to unnecessary energy consumption without tangible performance benefits.

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

System Power

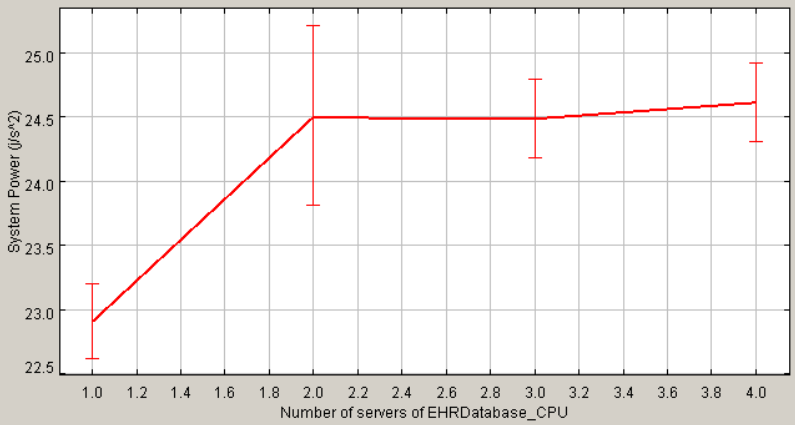
Average power of the entire system for each selected class.

Description

Station Name:	Network
Class Name:	All classes
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	4
X min:	1
X max:	4
Y min:	22.616
Y max:	25.212

Plot

Show confidence interval range



Simulation Results

*	1.000	2.000	3.000	4.000
Mean Value (j/s^2)	22.9039	24.4919	24.4819	24.6093
Max (j/s^2) (Conf.Int.)	23.1989	25.2119	24.7913	24.9226
Min (j/s^2) (Conf.Int.)	22.6162	23.8118	24.1801	24.3038

The values in red were not computed with the requested precision

Simulation Results - HIDC_JMT.jsimg

System Response Time | System Throughput | System Power | Fork Join Response Time

System Response Time

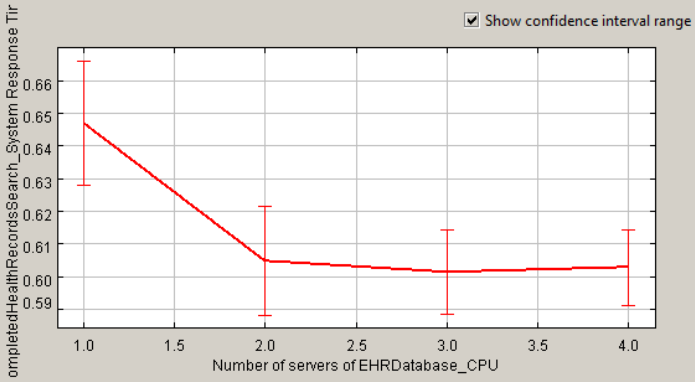
Average response time of the entire system for each selected class.

Description

Station Name:	Network
Class Name:	CompletedHealthRecordsSearch
Conf.Int. / Max Rel.Err. (0-1):	0.99 / 0.03
Models:	4
X min:	1
X max:	4
Y min:	0.588
Y max:	0.666

Plot

Show confidence interval range



Simulation Results

*	1.000	2.000	3.000	4.000
Mean Value (s)	0.6470	0.6048	0.6013	0.6028
Max (s) (Conf.Int.)	0.6660	0.6216	0.6142	0.6144
Min (s) (Conf.Int.)	0.6281	0.5880	0.5885	0.5913

The values in red were not computed with the requested precision

Description

Plot

Show confidence interval range