**Design for COLT**

**Introduction:**

A website that automatically adjusts bandwidth and measures and characterizes any application on the links that make up the system of networks. The goal is to watch the network traffic collected from probes placed against applications and make guesses about how much a particular AR/VR application uses in terms of mean, variance, standard deviation.

QoS: Quality of service - the performance of the service

SLA: Service-level agreement - commitment between a service provider and a client

VR: virtual reality

AR: augmented reality

**Scenarios:**

* if a specific street in London tends to be extremely busy at 5 pm on weekdays and most people use devices or applications that require significant amounts of bandwidth (anything video-related), then Colt will make sure to increase their bandwidth in that area during that time each week.

**Background:**

* Colt transforms the way the world works through the power of connectivity and provides a world-class network and voice connectivity to businesses in Europe, Asia, and the US. It has been leading the industry for over two decades.
* Colt IQ Network connects 900+ data centers across Europe, Asia, and North America’s largest business hubs, with over

**Scope of the job:**

* SLA characterization: Can see the different characterization of data (mean SD, variance...)
* Network Traffic Data
* Users can change the beta, bandwidth values.
* Create graphs, such as line graphs, pie charts, etc.
* Create a website/ app

**System design/ Design**

**Data:**

1-2 columns of data will be given which are time and bandwidth

What’s needed:

Using anvil, streamlit, python, java to create app/ webpage

Buttons/ keys on the side:

* Link for textbox
* VR/ AR are radio
* QoS level is shifter/ shader slider
* Current bandwidth displays on the sidebar
* Checkbox for showing raw data

**Part 0: Two different forms**

Form 1 and 2

* + Form 1 and Form 2 are running locally in the browser by converting your code through Java, while the server module is connected to the database/server.
  + Form 1 in our case would be the application settings.
  + Form 2 in our case would be the algorithm settings.

**Part 1: Choosing Link**

Use python, NumPy, streamlit, pandas, scipy

Needs a checkbox to allow the user to choose the Link

st.checkbox(Link)

User input feature is selected

This would predetermine the bandwidth

From

**Part 1: Choosing APP(VR/ AR)**

Use python, NumPy, streamlit, pandas, scipy

Needs a checkbox to allow the user to choose VR or AR

st.checkbox(VR, AR)

From

Network traffic data collected from probes placed against AR/VR applications

Log event data

**Part 2: Pick QOS Level**

Use python, NumPy, streamlit, pandas, scipy

Picking a Quality of service among different beta values

st.slider(1, 5)

Beta values are 1- 5

Find the QoS of interest and understand how it’s serving the QoS

**Part 3: Decide on a limit**

Use python, NumPy, streamlit, pandas, scipy

Decide on whether to increase or decrease the limit

Limit int = (increasing or decreasing) probably use the checkbox

**Part 4: Create a new bandwidth**

Use python, NumPy, streamlit, pandas, scipy

Select and adjust the bandwidth you want for that specific application from the link

Display bandwidth in the user interface

In the end, add all the individual bandwidths to set the total bandwidth (sum)

Bandwidth.sum()

total (optimized) bandwidth=sum of what current bandwidth is set to and all adjustments made (increase or decrease)=output as int

**Part 5: Showing Raw data(Characterization)**

Use python, NumPy, streamlit, pandas, scipy

Apache Zeppelin

Data includes mean, median, sd, variance, min, max, variance, packet loss of the given data in real-time

st.checkbox(data)

From bandwidth get packet loss for the characterization

Take data value get mean, median, sd, variance, min, max, variance, packet loss, etc. from the .descibe method or separately, and put it into a table.

characterization: map the application traffic data to QoS level (return as part of the output) from the beta value

Show packet loss on the mean median, the variance of the data

Option to show data’s trait

**Part 6: Algorithm**

Use python, NumPy, streamlit, pandas, scipy

Apache Zeppelin

algorithm: adjust the bandwidth and it fills and again " until the application network settings on edge of link reach sufficient bandwidth for QoS measure (max out until it is set high enough so traffic never sees boundary)

Algorithmic prototyping and mathematical computations

**Part 7: Test**

Use python, NumPy, streamlit, pandas, scipy

Apache Zeppelin

define test: has to be something pulling the bandwidth down so we need to keep trying to drop it with each adjustment we make (check for optimal bandwidth setting)

**Part 8: Output**

Use python, NumPy, streamlit, pandas, scipy

Use of Apache Zeppelin

output: line plot of mean network traffic over one-minute intervals with two horizontal lines representing current and optimized bandwidth, currently suggested bandwidth and optimal (adjusted) bandwidth, characterizations (descriptive statistics), and QoS measure (assigned to some beta value)

The pie chart on user Happiness

Bell graph for background traffic data

Line plot: Optimizing bandwidth in on-demand data, Packet loss Quality of service application

Charts: On Bandwidth, mean bandwidth, and the new bandwidth.

Line chart: On Bandwidth, mean bandwidth, and the new bandwidth.

**Algorithm Prototyping:**

Optimized bandwidth

Dynamic SLA Mathematical Computations

Stats including mean, variance, etc.

Statistical Multiplexing (see application)

**Storage:**

Short: Apache Zeppelin (Python/R)

Long: .csv files

Save it as a CSV. file

New settings (Bandwidth)

Graph of network and measures of current Quality of Service (QoS)

Network delays experienced on link

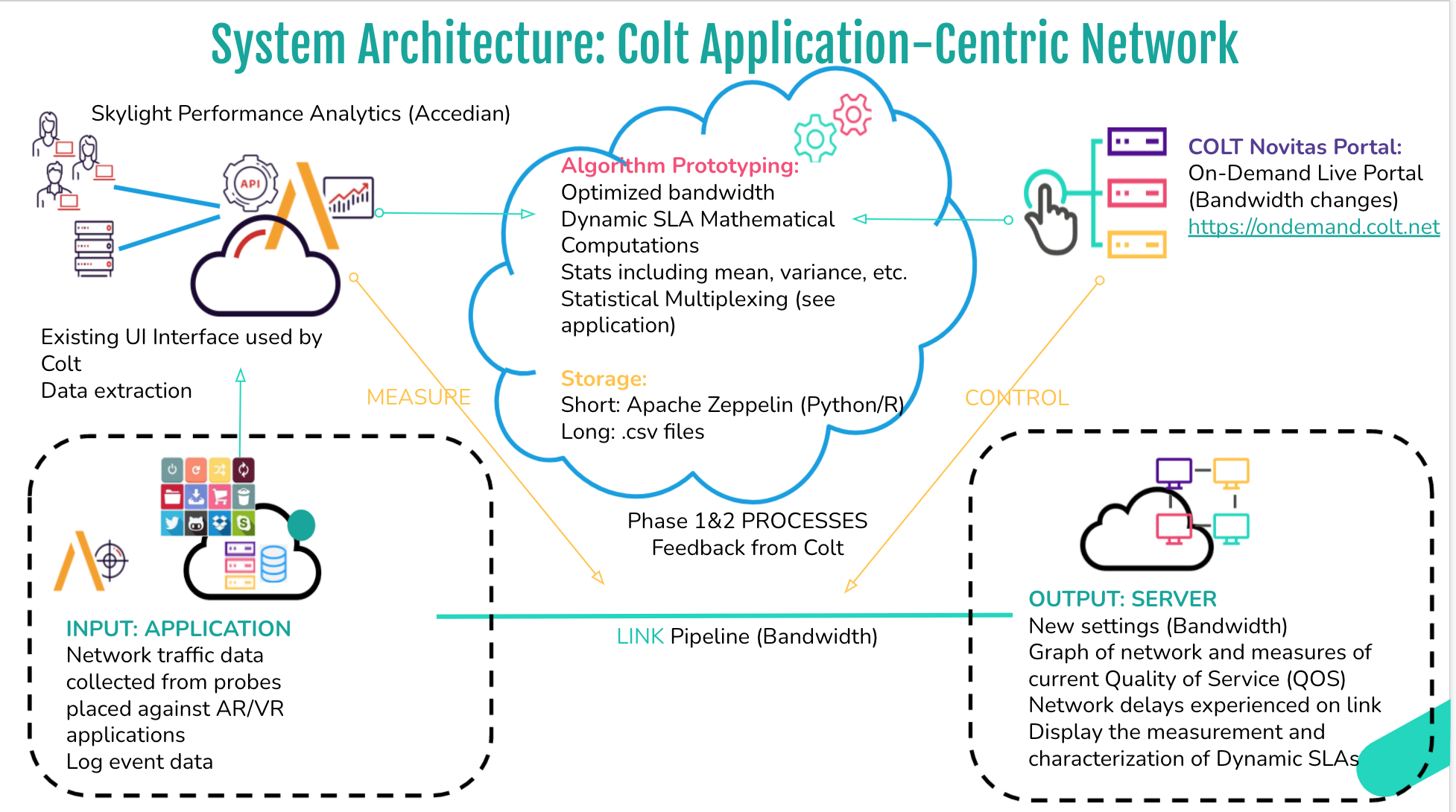
Display the measurement and characterization of Dynamic SLAs

**User interface:**

* Choosing Link
* Choice of AR or VR
* Beta values
* Picking of QoS levels
* Choose of Bandwidth
* Show of raw data

**Part 9: Copy 3xtimes**

Repeat part 1 to part 8 but with different Links for another copy during 3 times



**Future Contingencies:**

**Non-Functional Requirements:**

* Response bandwidth

**Testing Uasge:**

Look at data

Help from mentors

## **Deliverables:**

Notion, slack, google drive

**Risks:**

Don’t have data yet to do the project

**Timeline:**

Ahead of other teams

1. Brainstorm/Project Story with Colt Mentor (Week 1)
2. Low Tech Demo v1 (Week 2)
3. **Low Tech Demo v2 (Week 3)**
4. Technology Strategy (Week 4)
5. Low Tech Demo complete with tech strategy v3 (Week 5)
6. Execution in 4 Weekly Sprints (Week 6-10)
7. V1 Demo, Presentation, and Scope of next project (Week 11)
8. Final Demo, Presentation, and Scope of next project (Week 12)

**Validation information:** “We (Colt) need intelligence from data that tells us about degradation of services.” - Francesca Serravalle, Emerging Technology Director