Take-home Exercise 3

Quynh Trang Do

2025-06-06

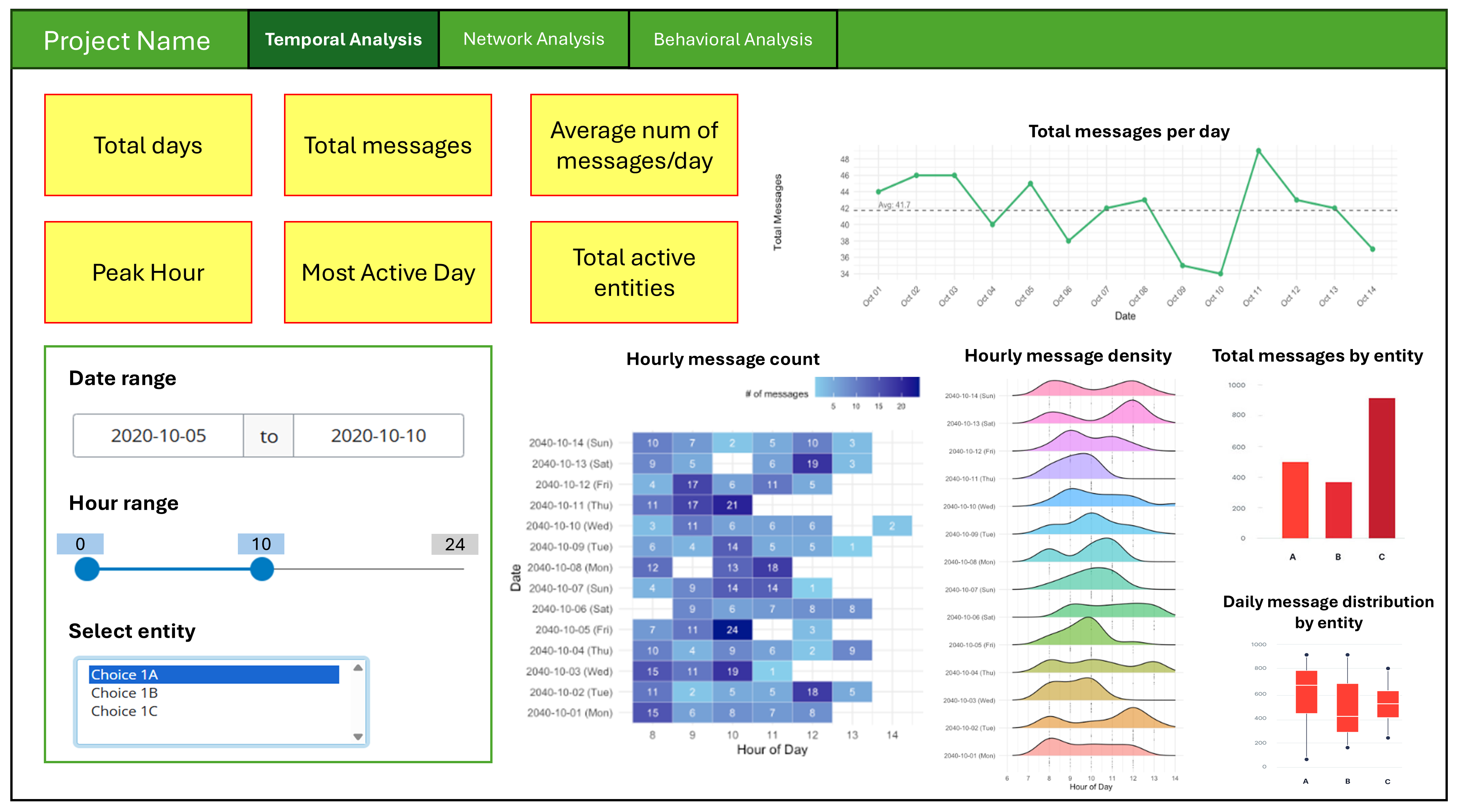
# Overview

This exercise presents the storyboard for my group’s Shiny application, with a focus on UI design, layout, and interactive components. Our project addresses three questions (1, 2 and 4) from [Mini-Challenge 3](https://vast-challenge.github.io/2025/MC3.html) of the [VAST Challenge 2025](https://vast-challenge.github.io/2025/index.html). Accordingly, the Shiny application is structured into three main sections:

1. Temporal Analysis – to explore communication patterns over the period.
2. Network Analysis – to visualize relationships and communication interactions among entities.
3. Behavioral Analysis – to examine the ego network and activity timeline of a selected entity based on user input, uncovering signs of illegal involvement.

# Section 1: Temporal Analysis

## UI Design



## Summary

This section is designed to help users explore and understand communication patterns over the time period. At the top, six value boxes display summary statistics, including total days, total messages, average messages per day, peak hour, most active day, and total active entities. On the left panel, users can interact with a date range picker, an hour range slider, and an entity selector to filter the data. The main display area on the right presents various charts: a line chart showing daily message volume, a heatmap of hourly message counts, density plots illustrating hourly activity patterns, a bar chart comparing message volume by entity, and a boxplot showing daily message distribution across entities. This layout allows users to gain both a high-level overview and detailed insights into temporal communication behavior.

## Details

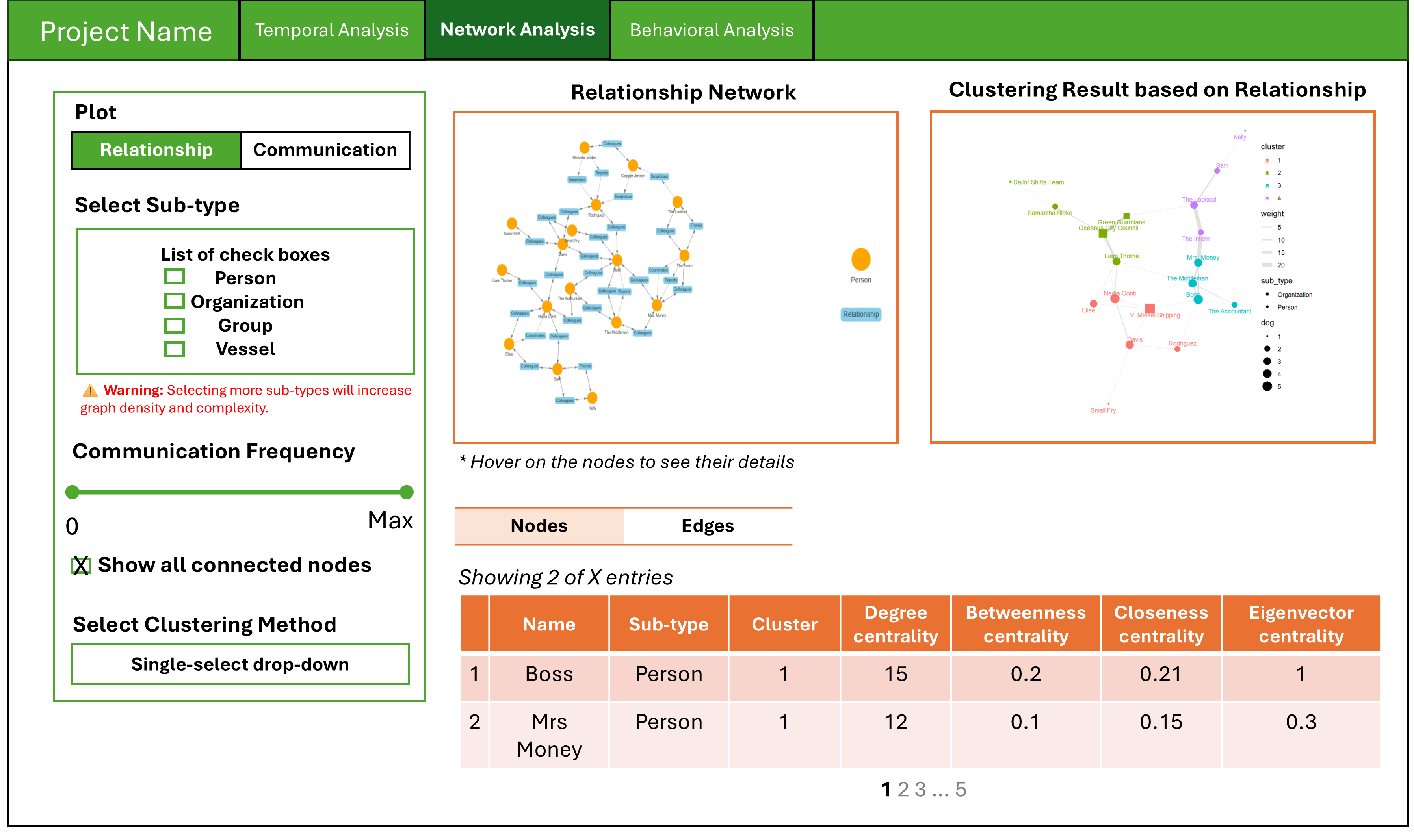
| Component | Type | Description |
| --- | --- | --- |
| Total days | Value box | Displays the number of unique days in the selected date range. |
| Total messages | Value box | Shows the total number of messages sent during the selected period. |
| Average number of messages/day | Value box | Calculates the average daily message count over the selected time window. |
| Peak hour | Value box | Identifies the hour of the day with the highest message volume. |
| Most active day | Value box | Highlights the single day within the selected period that has the greatest number of messages. |
| Total active entities | Value box | Indicates the number of unique entities that sent or received messages within the selected period. |
| Date range picker | User input | Allows users to define the time window for analysis. |
| Hour range slider | User input | Filters the analysis by specific hours of the day (e.g., 0–24). |
| Entity selector | User input | Enables selection of one or more entities to focus the analysis on specific participants. |
| Total messages per day | Line chart | Visualizes daily message volume over time, with a dashed line for the average. |
| Hourly message count | Heatmap | Shows message frequency by hour and day, helping detect temporal peaks. |
| Hourly message density | Density plots | Illustrates the distribution of messages across hours for each day using ridge or density plots. |
| Total messages by entity | Bar chart | Compares the total number of messages sent by each entity. |
| Daily message distribution by entity | Box plot | Displays the variation in daily message counts for each entity, identifying patterns or outliers. |

# Section 2: Network Analysis

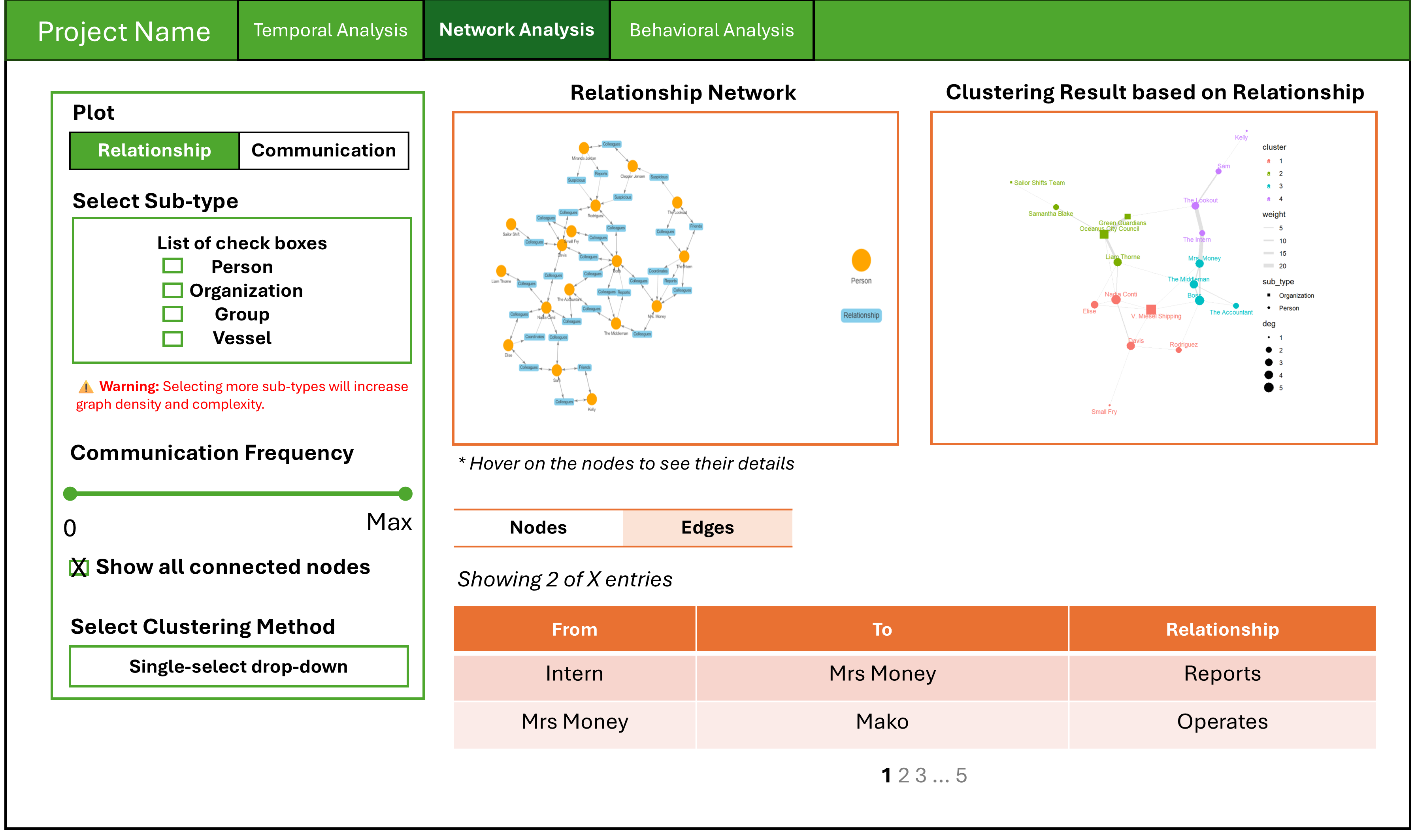
This section offers an interactive environment to explore entity relationships and communication structures within the dataset. Users can toggle between two network types, **Relationship** and **Communication**, via tab selection.

## Relationship Network

### Node-centric view



### Edge-centric view



### Summary

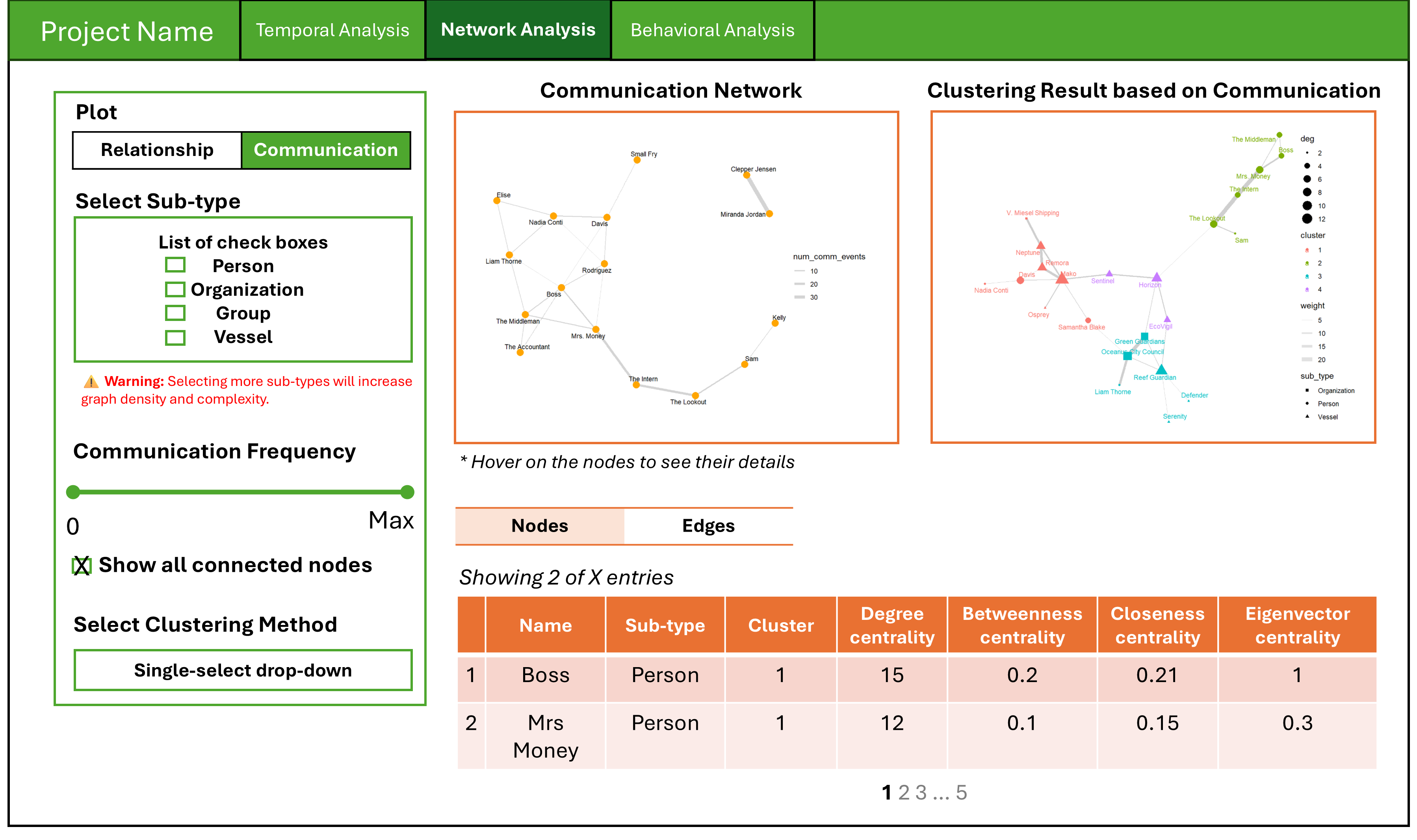
The Relationship Network view enables users to explore how entities such as people, organizations, groups, and vessels are connected through defined relationship types (e.g., “Reports”, “Operates”). Users can filter the network by selecting sub-types of entities and control graph complexity by adjusting the communication frequency slider or toggling the option to show all connected nodes. The main visualization displays a network graph where nodes represent entities and edges represent relationships. Hovering over a node reveals its identity and connections. On the right, a clustering result graph groups related entities into clusters based on network topology, with cluster colors and node size reflecting attributes like sub-type, cluster ID, and degree. Below, a dynamic node table presents centrality metrics (degree, betweenness, closeness, eigenvector) for selected entities, while the edge table displays direct relationship ties (e.g., who reports to whom), providing both structural and contextual insights into the network.

### Details

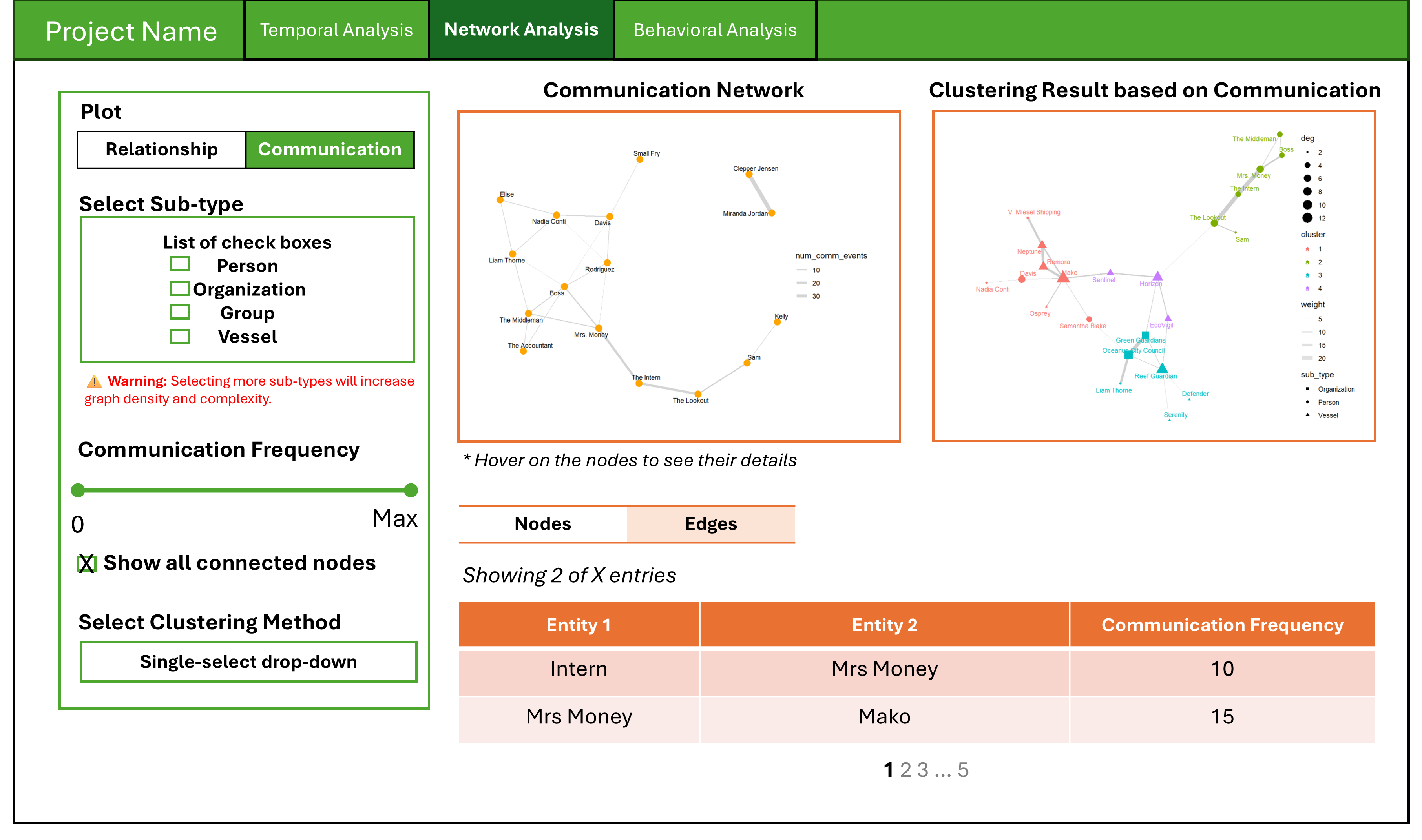
| **Component** | **Type** | **Description** |
| --- | --- | --- |
| Entity Type Selector | Checkbox Input | Allows users to select one or more entity sub-types (e.g., Person, Organization, Group, Vessel) to include in the relationship graph. |
| Connection Frequency Filter | Slider Input | Filters out edges (relationships) based on a selected threshold of frequency. |
| Clustering Method Selector | Dropdown Input | Allows users to select a specific clustering algorithm to group structurally similar entities. |
| Show All Connected Nodes | Checkbox Input | Option to include all nodes regardless of the communication frequency. |
| Relationship Network Graph | Network Visualization | Interactive visualization showing entities as nodes and relationships (e.g., Reports, Operates) as edges. Hover to reveal node details. |
| Clustering Result Graph | Network Visualization | Color-coded graph showing communities of entities based on their structural proximity. Node size represents centrality (e.g., degree). |
| Node Table | Data Table | Displays centrality measures (degree, betweenness, closeness, eigenvector) for selected entities, helping identify key players in the network. |
| Edge Table | Data Table | Lists direct relationships between entities, including relationship type and participating nodes for interpretability and traceability. |

## Communication Network

### Node-centric view



### Edge-centric view



### Summary

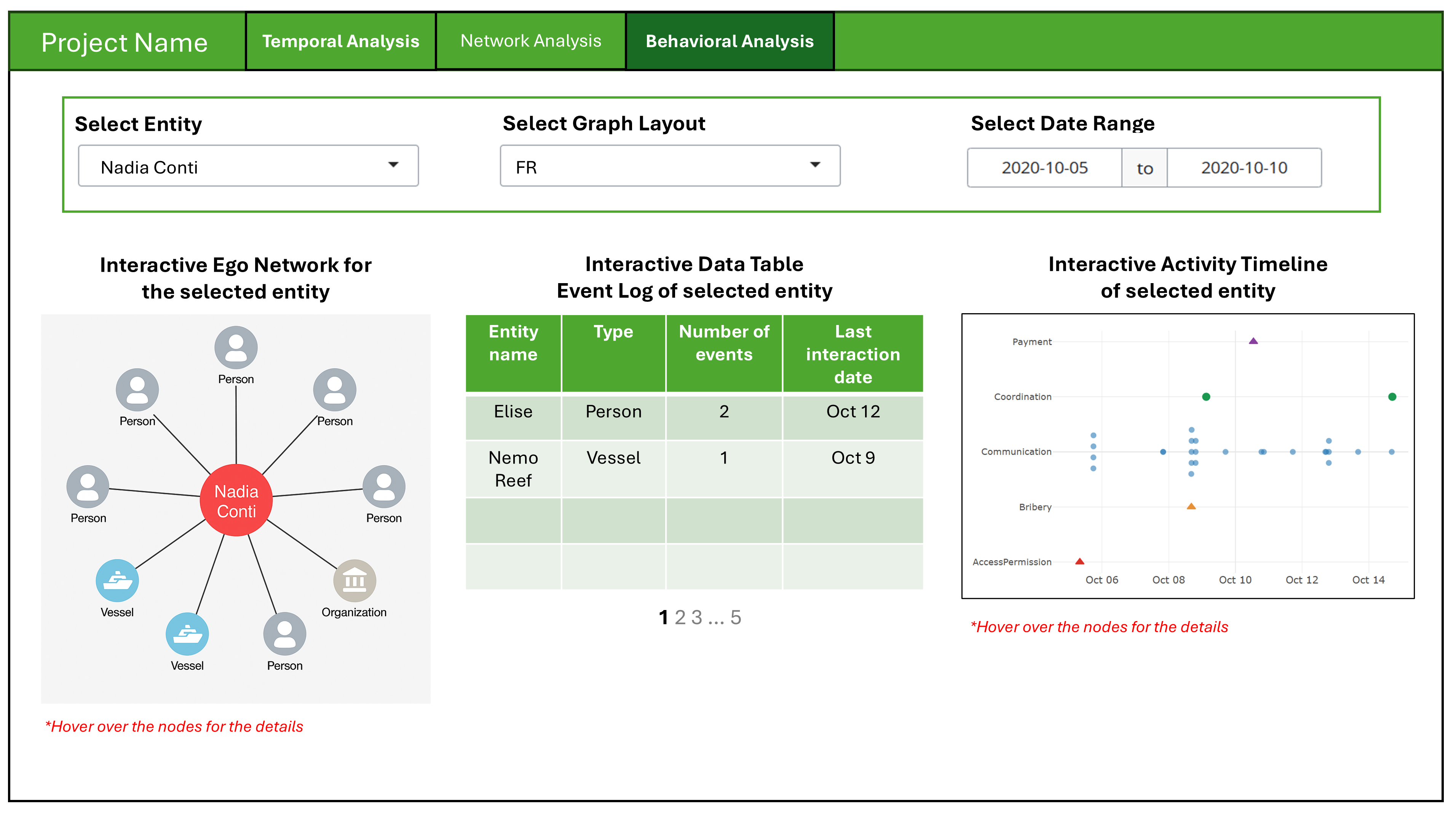
The Communication Network mode focuses on actual message exchanges between entities. Similar to the relationship view, users can filter by entity sub-type, set thresholds for communication frequency, and choose a clustering method. The communication graph in the center illustrates how frequently different entities interact, with edge thickness reflecting the number of communication events. The adjacent clustering result graph shows communities formed based on communication behavior, helping to identify tightly connected groups. Entities are colored by cluster, sized by degree, and marked by sub-type for easy interpretation. At the bottom, the node table lists each entity’s centrality measures to identify influential communicators, while the edge table summarizes entity-to-entity communication frequencies, offering granular insight into who is talking to whom and how often.

### Details

| **Component** | **Type** | **Description** |
| --- | --- | --- |
| Entity Type Selector | Checkbox Input | Allows users to select one or more entity sub-types (e.g., Person, Organization, Group, Vessel) to include in the relationship graph. |
| Connection Frequency Filter | Slider Input | Filters out edges (relationships) based on a selected threshold of frequency. |
| Clustering Method Selector | Dropdown Input | Allows users to select a specific clustering algorithm to group structurally similar entities. |
| Show All Connected Nodes | Checkbox Input | Option to include all nodes regardless of the communication frequency. |
| Communication Network Graph | Network Visualization | Interactive graph where nodes represent entities and edge thickness reflects communication frequency. Hover reveals node information. |
| Clustering Result Graph | Network Visualization | Visual grouping of entities into clusters based on communication behavior. Nodes are sized by degree and colored by cluster. |
| Node Table | Data Table | Lists centrality metrics (degree, betweenness, closeness, eigenvector) for each entity to identify key communicators. |
| Edge Table | Data Table | Displays pairwise communication frequencies between entities, showing who interacts with whom and how often. |

# Section 3: Behavioral Analysis

## UI Design



## Summary

This section enables in-depth investigation of a specific entity’s behavior, interactions, and activities over time. Users begin by selecting an entity of interest, a preferred graph layout, and a date range. The main interface then updates to show three coordinated views:

1. An **interactive ego network** showing direct relationships between the selected entity and others, categorized by entity type (e.g., person, vessel, organization).
2. A **dynamic event log table** listing all associated entities, their types, number of interactions, and the last interaction date.
3. An **interactive activity timeline** that visualizes events (e.g., communication, bribery, coordination) involving the selected entity across the chosen date range.

This section supports behavioral profiling, surfacing anomalies, and potentially flagging suspicious or illegal interactions through temporal and structural context.

## Details

| **Component** | **Type** | **Description** |
| --- | --- | --- |
| Select Entity | Dropdown Input | Allows users to choose a specific entity (e.g., person, vessel) for detailed behavioral exploration. |
| Select Graph Layout | Dropdown Input | Provides layout options (e.g., Force-directed, Circular) to change the arrangement of nodes in the ego network. |
| Select Date Range | Date Range Picker | Filters all visualizations and tables based on the specified activity window. |
| Ego Network Visualization | Network Graph | Shows a focused view of the selected entity’s direct relationships with other entities, color-coded by type. |
| Event Log Table | Data Table | Lists interacting entities, their types, number of shared events, and the last interaction date. |
| Activity Timeline | Scatter Plot / Timeline | Plots event types involving the selected entity across time, helping identify suspicious patterns or irregular activities. |

# Shiny App Pseudocode

The following pseudocode outlines the architecture of our Shiny application, which is designed to support visual analytics across three sections mentioned above.

* The **UI** is structured as a multi-tab layout, with each tab corresponding to a specific analysis module.
* The **server logic** handles data filtering, network computations, centrality calculations, and dynamic rendering of visualizations and tables.

## UI Layout

ui <- navbarPage("Project Name",  
  
 # 1. Temporal Analysis  
 tabPanel("Temporal Analysis",  
 sidebarLayout(  
 sidebarPanel(  
 dateRangeInput("date\_range", "Date range"),  
 sliderInput("hour\_range", "Hour range", min=0, max=24, value=c(0, 24)),  
 selectInput("entity\_select", "Select entity", choices = entity\_list, multiple=TRUE)  
 ),  
 mainPanel(  
 fluidRow( # Top metrics  
 valueBoxOutput("total\_days"),  
 valueBoxOutput("total\_messages"),  
 valueBoxOutput("avg\_msg\_day"),  
 valueBoxOutput("peak\_hour"),  
 valueBoxOutput("most\_active\_day"),  
 valueBoxOutput("total\_entities")  
 ),  
 fluidRow( # Charts  
 plotOutput("line\_daily\_msg"),  
 plotOutput("heat\_hourly\_msg"),  
 plotOutput("density\_hourly\_msg"),  
 plotOutput("bar\_total\_entity"),  
 plotOutput("boxplot\_entity\_daily")  
 )  
 )  
 )  
 ),  
  
 # 2. Network Analysis  
 tabPanel("Network Analysis",  
 sidebarLayout(  
 sidebarPanel(  
 radioButtons("network\_type", "Plot", choices = c("Relationship", "Communication")),  
 checkboxGroupInput("entity\_type", "Select Sub-type", choices = c("Person", "Organization", "Group", "Vessel")),  
 sliderInput("comm\_freq", "Communication Frequency", min=0, max=100, value=0),  
 checkboxInput("show\_connected", "Show all connected nodes", TRUE),  
 selectInput("clustering\_method", "Select Clustering Method", choices = c("Louvain", "Walktrap", "Label Propagation"))  
 ),  
 mainPanel(  
 fluidRow(  
 visNetworkOutput("main\_network"),  
 visNetworkOutput("cluster\_network")  
 ),  
 tabsetPanel(  
 tabPanel("Nodes", DTOutput("node\_table")),  
 tabPanel("Edges", DTOutput("edge\_table"))  
 )  
 )  
 )  
 ),  
  
 # Behavioral Analysis  
 tabPanel("Behavioral Analysis",  
 fluidPage(  
 fluidRow(  
 selectInput("selected\_entity", "Select Entity", choices = entity\_list),  
 selectInput("layout\_type", "Select Graph Layout", choices = c("FR", "Circle", "Grid")),  
 dateRangeInput("behavior\_date", "Select Date Range")  
 ),  
 fluidRow(  
 column(4, visNetworkOutput("ego\_network")),  
 column(4, DTOutput("event\_log")),  
 column(4, plotOutput("activity\_timeline"))  
 )  
 )  
 )  
)

## Server Logic

server <- function(input, output, session) {  
  
 ## 1. Temporal Analysis  
 filtered\_data <- reactive({  
 filter\_messages\_by\_date\_hour\_entity(input$date\_range, input$hour\_range, input$entity\_select)  
 })  
  
 output$total\_days <- renderValueBox({ count\_unique\_days(filtered\_data()) })  
 output$total\_messages <- renderValueBox({ count\_messages(filtered\_data()) })  
 output$avg\_msg\_day <- renderValueBox({ avg\_messages\_per\_day(filtered\_data()) })  
 output$peak\_hour <- renderValueBox({ find\_peak\_hour(filtered\_data()) })  
 output$most\_active\_day <- renderValueBox({ find\_most\_active\_day(filtered\_data()) })  
 output$total\_entities <- renderValueBox({ count\_active\_entities(filtered\_data()) })  
  
 output$line\_daily\_msg <- renderPlot({ plot\_messages\_per\_day(filtered\_data()) })  
 output$heat\_hourly\_msg <- renderPlot({ plot\_hourly\_heatmap(filtered\_data()) })  
 output$density\_hourly\_msg <- renderPlot({ plot\_hourly\_density(filtered\_data()) })  
 output$bar\_total\_entity <- renderPlot({ plot\_total\_messages\_by\_entity(filtered\_data()) })  
 output$boxplot\_entity\_daily <- renderPlot({ plot\_boxplot\_entity\_daily(filtered\_data()) })  
  
 ## 2. Network Analysis  
 filtered\_graph <- reactive({  
 filter\_network\_by\_type\_and\_threshold(input$network\_type, input$entity\_type, input$comm\_freq)  
 })  
  
 output$main\_network <- renderVisNetwork({  
 plot\_network\_graph(filtered\_graph(), show\_connected = input$show\_connected)  
 })  
  
 output$cluster\_network <- renderVisNetwork({  
 cluster\_graph(filtered\_graph(), method = input$clustering\_method)  
 })  
  
 output$node\_table <- renderDT({  
 compute\_node\_centralities(filtered\_graph())  
 })  
  
 output$edge\_table <- renderDT({  
 list\_entity\_relationships(filtered\_graph())  
 })  
  
 ## 3. Behavioral Analysis  
 ego\_graph <- reactive({  
 get\_ego\_network(input$selected\_entity, input$behavior\_date)  
 })  
  
 output$ego\_network <- renderVisNetwork({  
 plot\_ego\_graph(ego\_graph(), layout = input$layout\_type)  
 })  
  
 output$event\_log <- renderDT({  
 get\_event\_log\_for\_entity(input$selected\_entity, input$behavior\_date)  
 })  
  
 output$activity\_timeline <- renderPlot({  
 plot\_entity\_activity\_timeline(input$selected\_entity, input$behavior\_date)  
 })  
}