# VTRemote

## An Android Application for the VirtuTrace 3D Simulator

## Group May14-21

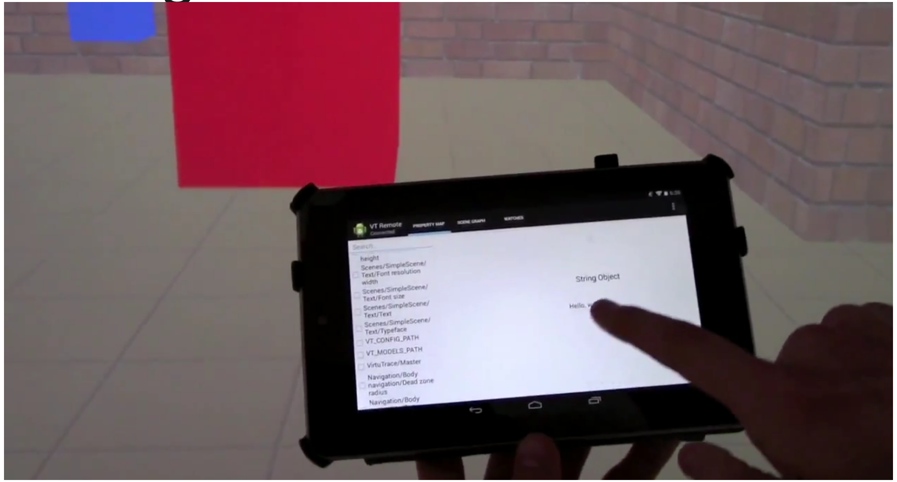
Tanner Borglum | Kollin Burns | Lukas Herrmann | Alexander Maxwell | Sheil Patel

# Project Overview

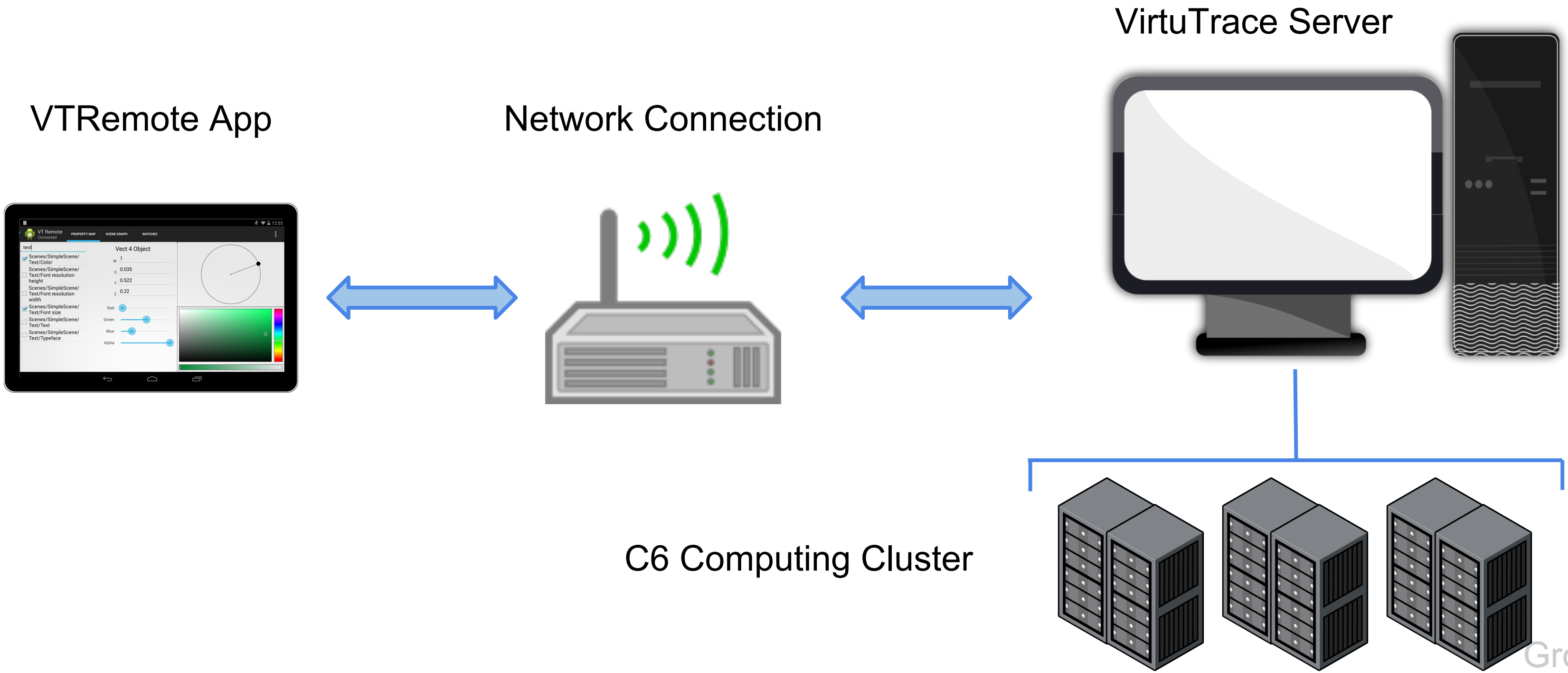
- VirtuTrace (VT)
  - Simulation engine used in the C6 virtual reality cave to create 3D virtual environments

# Project Overview

- VTRemote
  - Android app that remotely controls VT and enables manual changes to the simulations in real time


VT Remote
Connected
PROPERTY MAP
SCENE GRAPH
WATCHES
Search
height
Scenes/SimpleScene/Text/Font resolution
width
Scenes/SimpleScene/Text/Font size
Scenes/SimpleScene/Text/Text
Scenes/SimpleScene/Text/Typeface
VT_CONFIG_PATH
VT_MODELS_PATH
VirtuTrace/Master
Navigation/Body
navigation/Dead zone radius
Navigation/Body
String Object
Hello,

# Conceptual Model



VTRemote App

Network Connection

VirtuTrace Server

C6 Computing Cluster

text

☑ Scenes/SimpleScene/Text/Color

□ Scenes/SimpleScene/Text/Font resolution height

□ Scenes/SimpleScene/Text/Font resolution width

☑ Scenes/SimpleScene/Text/Font size

□ Scenes/SimpleScene/Text/Text

□ Scenes/SimpleScene/Text/Typeface

Vect 4 Object

W 1

X 0.035

Y 0.522

Z 0.22

# Problem Statement

- Deployment and startup of VirtuTrace simulations within C6 is costly (~ $50 per reset)
- Each configuration change is also time intensive
    - Physically exit C6
    - Shut down VirtuTrace simulation
    - Change source code or config file
    - Re-initialize simulation
    - Re-enter C6

# Solution

- VTRemote allows real-time changes
  - Greatly reduces the number of resets required
  - Assists with the debugging process while developing new VT simulation programs

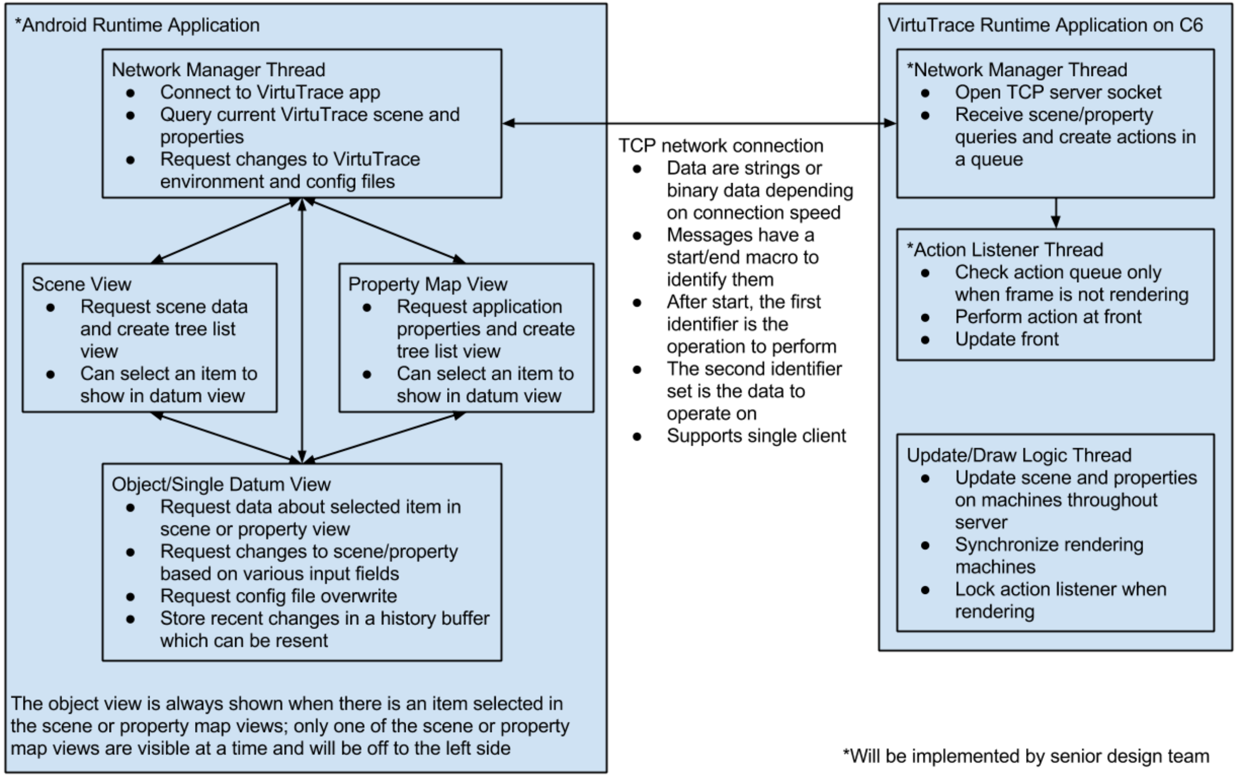
# Market Research

- The C6 virtual reality environment at Iowa State is one of two of its kind in the world
  - VT was built specifically for the C6
- An iOS application was previously created
  - Functionality of this app was very limited
- Dynamic real-time changes to VT were not previously possible

# Functional Decomposition

- **VTRemote App**
  - Network management thread
  - Tabs for each mode (Property Map, Scene Graph, Watches)
  - Each has a dynamic list of properties or nodes
- **VirtuTrace Server**
  - Network management thread
  - Work order queue for processing changes in the Scene Graph
  - Update map used to process changes in the Property Map
  - Observer to send live updates for the property being viewed in-app

# Block Diagram



*Android Runtime Application

Network Manager Thread
- Connect to VirtuTrace app
- Query current VirtuTrace scene and properties
- Request changes to VirtuTrace environment and config files

Scene View
- Request scene data and create tree list view
- Can select an item to show in datum view

Property Map View
- Request application properties and create tree list view
- Can select an item to show in datum view

Object/Single Datum View
- Request data about selected item in scene or property view
- Request changes to scene/property based on various input fields
- Request config file overwrite
- Store recent changes in a history buffer which can be resent

The object view is always shown when there is an item selected in the scene or property map views; only one of the scene or property map views are visible at a time and will be off to the left side

TCP network connection
- Data are strings or binary data depending on connection speed
- Messages have a start/end macro to identify them
- After start, the first identifier is the operation to perform
- The second identifier set is the data to operate on
- Supports single client

VirtuTrace Runtime Application on C6

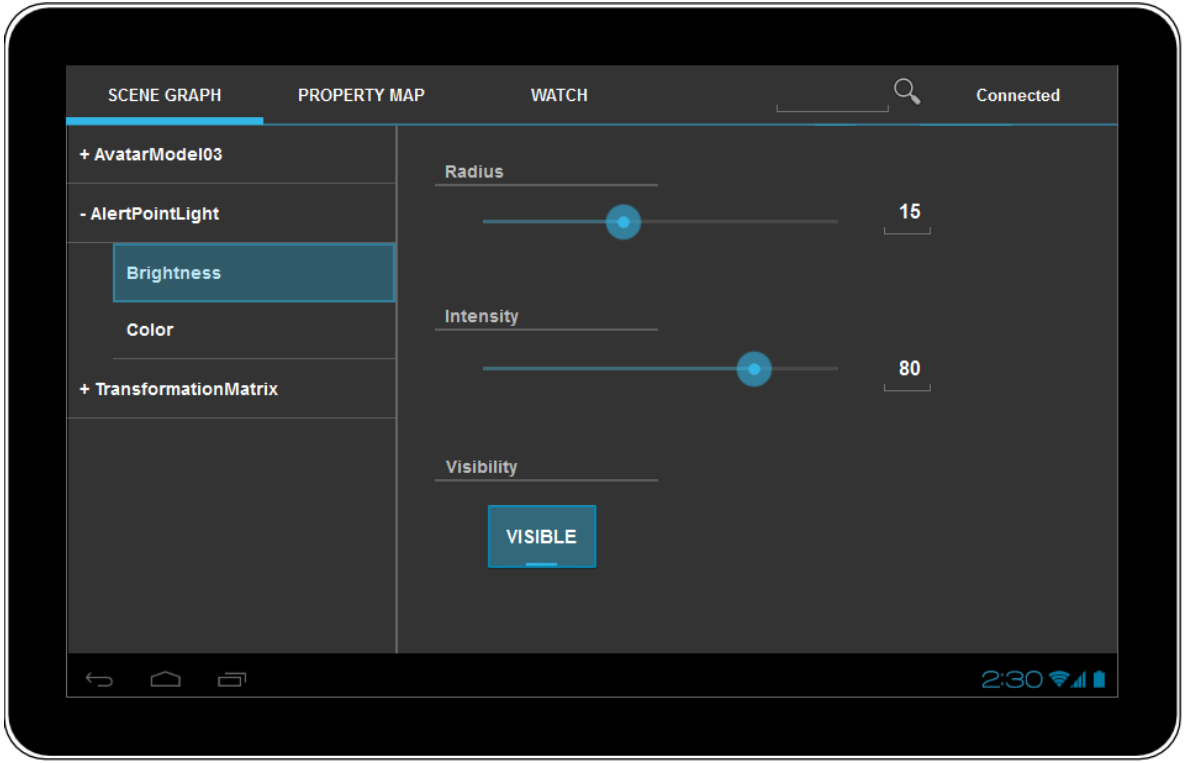
*Network Manager Thread
- Open TCP server socket
- Receive scene/property queries and create actions in a queue

*Action Listener Thread
- Check action queue only when frame is not rendering
- Perform action at front
- Update front

Update/Draw Logic Thread
- Update scene and properties on machines throughout server
- Synchronize rendering machines
- Lock action listener when rendering

*Will be implemented by senior design team

# Initial UI Mockup

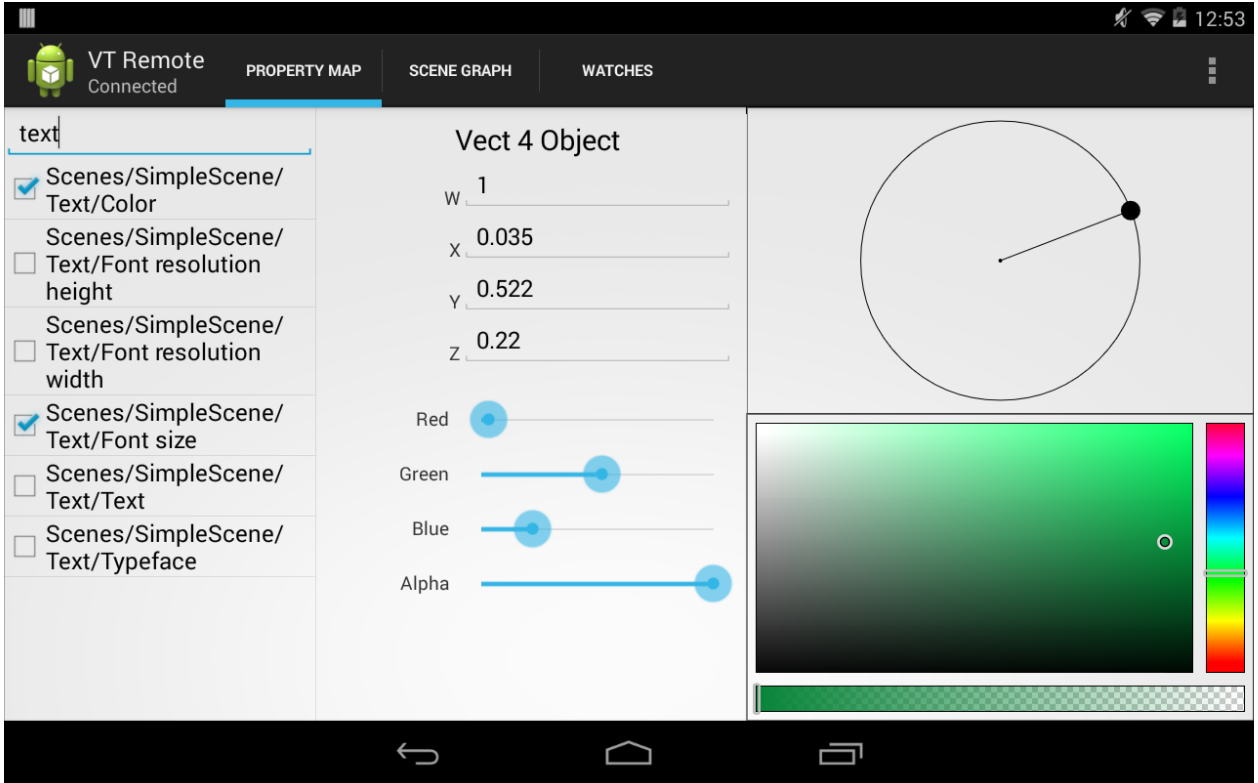


Radius 15

Intensity 80

Visibility VISIBLE

# Current UI - Property Map



text

- ☑ Scenes/SimpleScene/Text/Color
- □ Scenes/SimpleScene/Text/Font resolution height
- □ Scenes/SimpleScene/Text/Font resolution width
- ☑ Scenes/SimpleScene/Text/Font size
- □ Scenes/SimpleScene/Text/Text
- □ Scenes/SimpleScene/Text/Typeface

Vect 4 Object

W 1

X 0.035

Y 0.522

Z 0.22

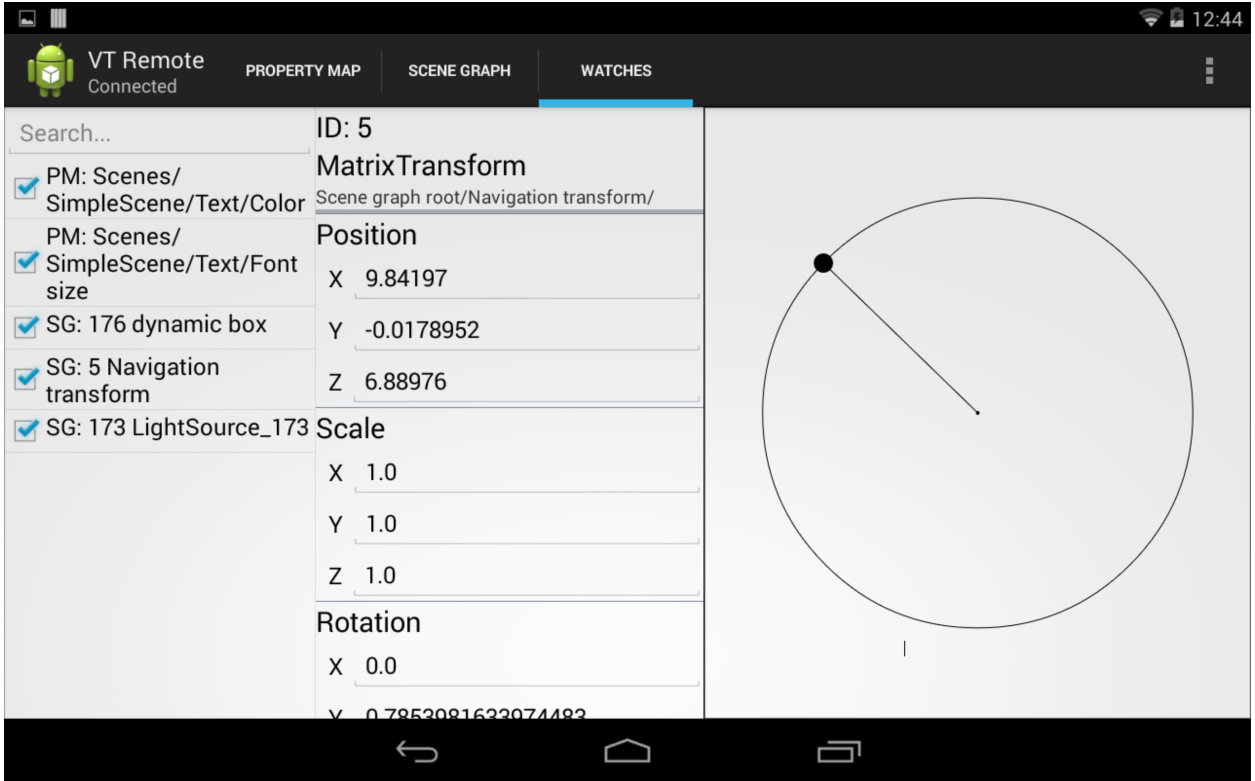
Red

Green

Blue

Alpha

# Current UI - Watches



☑ PM: Scenes/ SimpleScene/Text/Color

☑ PM: Scenes/ SimpleScene/Text/Font size

☑ SG: 176 dynamic box

☑ SG: 5 Navigation transform

☑ SG: 173 LightSource_173

ID: 5

MatrixTransform

Scene graph root/Navigation transform/

Position

X 9.84197

Y -0.0178952

Z 6.88976

Scale

X 1.0

Y 1.0

Z 1.0

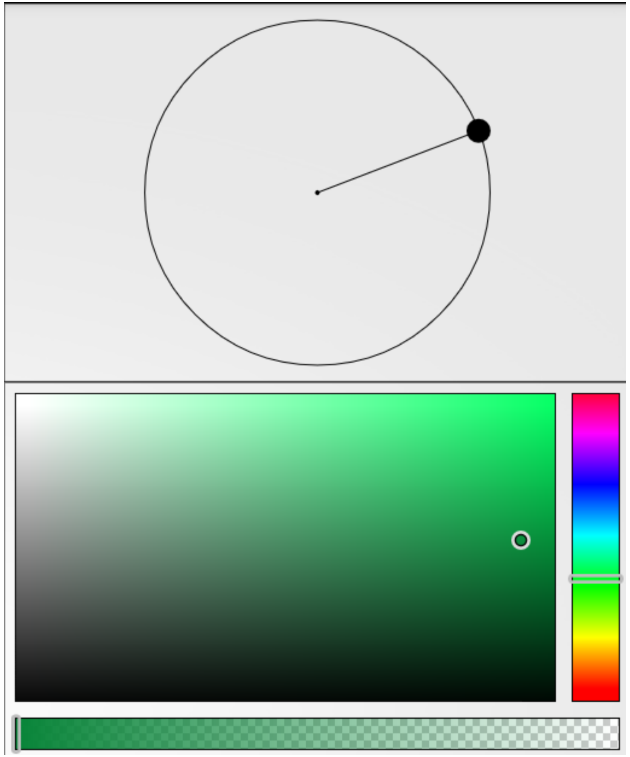
Rotation

X 0.0

Y 0.7853981633974483

# Custom Widgets

- Circular Endless Slider
  - handles values with unknown bounds
  - dynamic precision
- Color Chooser
  - Originally developed by Daniel Nilsson
  - Adapted for our project



# Design Rationale

- **Overall UI**
  - **Tabs vs. Navigation Drawer**
- Target Platform
  - HTML5 vs. Android
- Network Communication
  - TCP vs. UDP
- Network Messaging Interface

# Navigation Drawer vs Tabs

## Navigation Drawer

### Pros

- supports more tabs
- allows non-lateral navigation

### Cons

- more complex to implement and use

## Tabs

### Pros

- one-click switching
- screen consistency

### Cons

- screen size limits number of tabs

# Design Rationale

- Overall UI
  - Tabs vs. Navigation Drawer
- **Target Platform**
  - **HTML5 vs. Android**
- Network Communication
  - TCP vs. UDP
- Network Messaging Interface

# Target Platform

## HTML5

- Pros
  - OS independent
- Cons
  - Persistent threading not fully supported

## Android

- Pros
  - Group experience
  - Built-in UI tools
- Cons
  - Requires Android OS

    - limits systems that may be used

# Design Rationale

- Overall UI
  - Tabs vs. Navigation Drawer
- Target Platform
  - HTML5 vs. Android
- **Network Communication**
  - **TCP vs. UDP**
- Network Messaging Interface

# TCP vs UDP

## TCP

- Pros
  - Error detection/ resolution
  - Congestion/flow control
- Cons
  - Much slower

## UDP

- Pros
  - Much faster due to low overhead
- Cons
  - No error detection or congestion control

# Design Rationale

- Overall UI
  - Tabs vs. Navigation Drawer
- Target Platform
  - HTML5 vs. Android
- Network Communication
  - TCP vs. UDP
- **Network Messaging Interface**

# Messaging Interface

| | |
|---|---|
| **Start Keyword** | A static keyword indicating the start of a message. |
| **Mode** | The mode parameter indicating whether the message is related to Scene Graph or Property Map. |
| **Operation** | The operation to be performed. |
| **Operands** | The operation’s parameters. |
| **Stop Keyword** | A static keyword indicating the end of a message. |

# Design Changes

- Activities → Fragments
  - Dynamic layout composition
  - Less overhead
- TreeView → ListView
  - Simpler to implement and use
  - Less overhead
- Watched Tab
  - Original idea deemed not feasible
  - Moved to a bookmarks system

# Challenges

- **Threading in Android**
- Synchronization across VT cluster nodes
- Consistency across systems
- Architecting UI for dynamic views

# Threading in Android

- Android’s Activity Lifecycle
  - Background tasks can be killed
  - Managing connection through lifecycle changes
- Interacting with the UI
  - Main UI thread cannot be blocked
  - Only main thread can alter UI elements

# Challenges

- Threading in Android
- **Synchronization across VT cluster nodes**
- Consistency across systems
- Architecting UI for dynamic views

# VT Cluster Synchronization

- Serializable data structures
- Compliance with existing API
  - VR Juggler
- Avoiding duplicate work orders
  - less processing and network data

# Challenges

- Threading in Android
- Synchronization across VT cluster nodes
- **Consistency across systems**
- Architecting UI for dynamic views

# System Consistency

## VirtuTrace

- C++
- Libraries
  - OpenSceneGraph
  - Boost
  - VR Juggler

## VTRemote

- Java
- Libraries
  - Android 4.x
  - Support library for older versions of Android OS

# Challenges

- Threading in Android
- Synchronization across VT cluster nodes
- Consistency across systems
- **Architecting UI for dynamic views**

# Dynamic UI

- Fragments
  - Everything is a fragment
  - Fragments are swapped in and out of containers
- Supporting real-time data updates
- Keeping UI elements synced

# Test Plan

- VTRemote
  - JUnit test cases
  - Manual testing
- VirtuTrace
  - Unit tests using Google Test framework
- Network Communication
  - Round-trip Serialization/Deserialization tests
  - Stress tests

# Demo

# Thank You

Questions?