Team Amadeus:
MAD Assembly Builder
Design Review 2

Members:
Wyatt Evans, Kyle Krueger,
Melody Pressley, Evan Russell
Mentor:
Austin Sanders
Sponsors:
Dr. Hélène Coullon & Dr. Frédéric Loulergue

Team Introductions

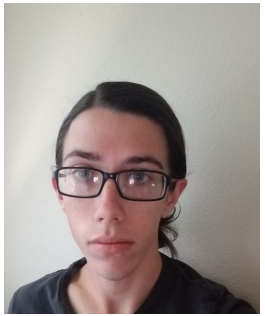
Wyatt Evans

Team Leader

Kyle Krueger

Release Manager

Melody Pressley



Document Architect

Evan Russell

Documenter

Software Deployment

- Deployment of software across multiple devices
- Many interrelated, interconnected activities
- All software is unique
  - Different dependencies, characteristics, specifications
  - Deployment process must be unique

Fig. 1: Software Deployment Example

Our Clients

NORTHERN ARIZONA UNIVERSITY

Dr. Frédéric Loulergue

Professor @ School of Informatics Computing and Cyber Systems

Inria inventors for the digital world

Dr. Hélène Coullon

Assistant Professor at IMT Atlantique, Inria researcher

Madeus / MAD

- Madeus
  - Theoretical Model for Software Deployment
  - Explicitly Defined Steps and Dependencies
- MAD
  - Madeus Application Deployer
  - Formal Implementation
  - Python

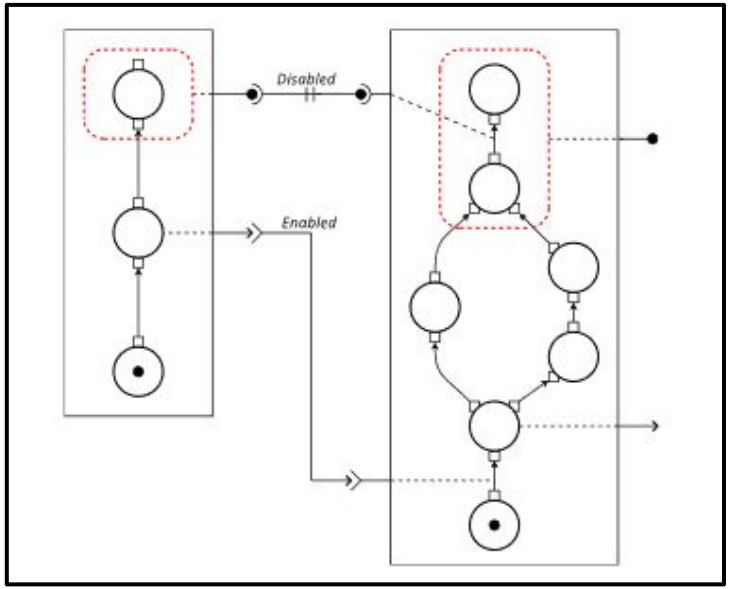


Fig. 2: Basic Madeus Assembly

The Problem

- Current process is slow
- Designing an assembly in code is tedious
- Complex to edit
- Easier to visualize and modify with diagrams

Our Solution: Develop a GUI

- Visualization
- Simulation
- Easier for users to edit
- Decrease turnaround time on MAD Assembly development

Key Requirements

- Visualize the creation of Madeus assemblies
- Extensible framework that allows for future additions
- Generate MAD code that represents the user’s diagram
- Simulate deployment of an assembly

Architecture Overview

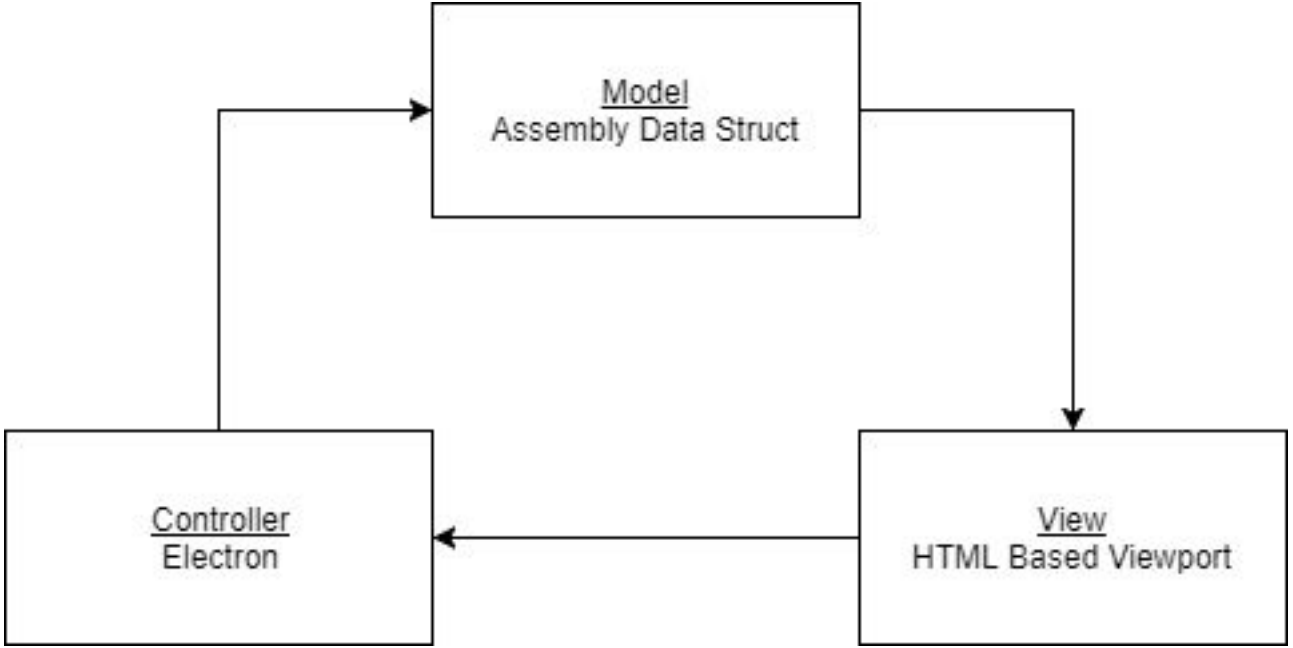


Fig. 3: MVC Architecture

Implementation Overview

- [Model] Global Data Structures
  - Assembly Component List
    - Contains all user created components in one centralized location for [Controller] use as well as any provided plugins.
  - Connection List
    - Contains all dependency connections between components.

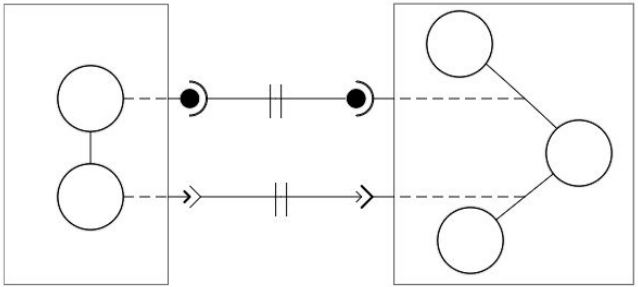


Fig. 4: Component/Connection Example

Object Breakdown

- Component
  - Parameters: Name
  - Contains lists: place, transition, & dependency
  - Contains Konva object component group

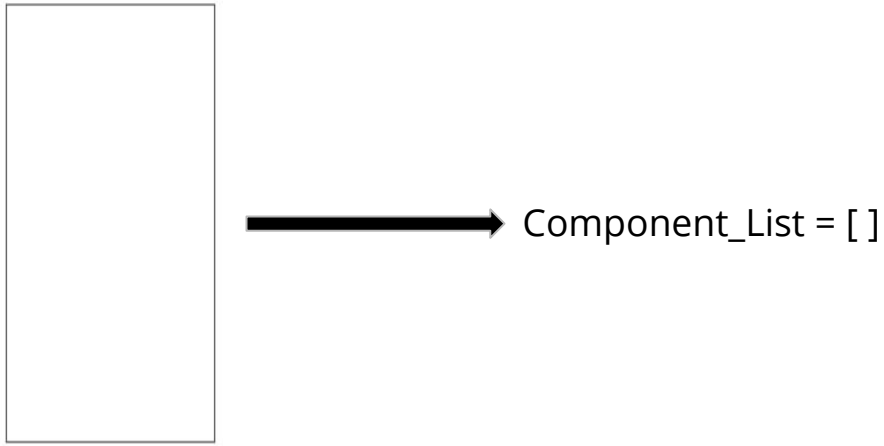
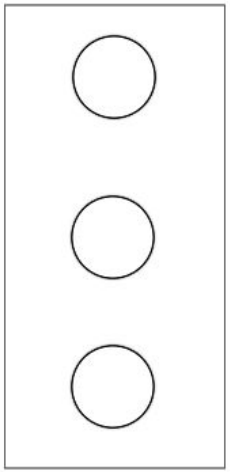


Fig. 5: Component

Object Breakdown Cont.

- Place
  - Parameters: name, index, transition_count, dependency_count/type

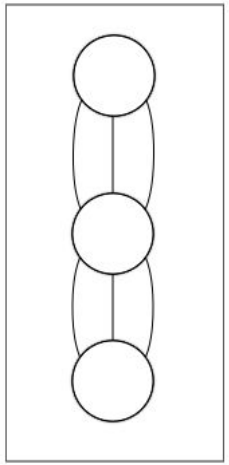


Component_List = [component.place_list = [ ]]

Fig. 6: Places

Object Breakdown Cont.

- Transition
  - Parameters: name, source, destination, function

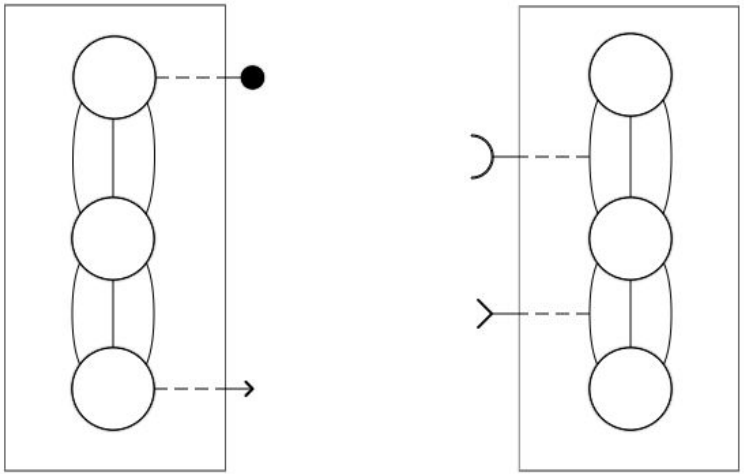


Component_List = [component.transition_list = [ ]]

Fig. 7: Transitions

Object Breakdown Cont.

- Dependency
  - Parameters: name, type, source_obj, connection_obj



Component_List = [component.dependency_list = [ ]]

Fig. 8: Dependencies

Object Breakdown Cont.

- Connection
  - Parameters: provide_port, use_port, status

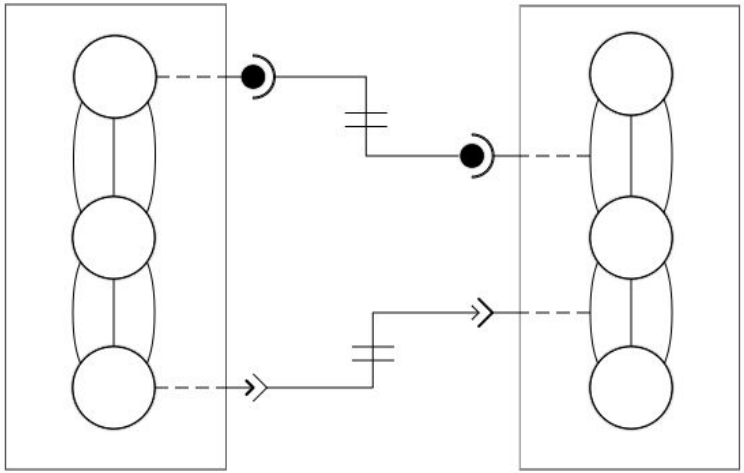


Fig. 9: Connections

Connection_List = [connection]

Challenges and Resolutions

- Limitations with Kivy Python framework
  - Switching over to Electron (Node.js and Chromium)
  - Electron framework behind Atom, Visual Studio Code, Slack, and Discord
- Saving and Loading of User Created Assemblies
  - Amended our Data-structure to serialize and store the Konva objects/groups
  - Saving will capture all objects and their attributes (size, position)
  - Loading will build an assembly from the data-structure
  - User created assembly and data-structure generated assembly
- Deployment Simulation through Konva Animation
  - Simulation mode creates a layer on top of workspace
  - Prohibits editing while in Simulation mode
  - In Simulation mode the user able to play, stop the animation

Schedule

Gantt Chart / Development Schedule

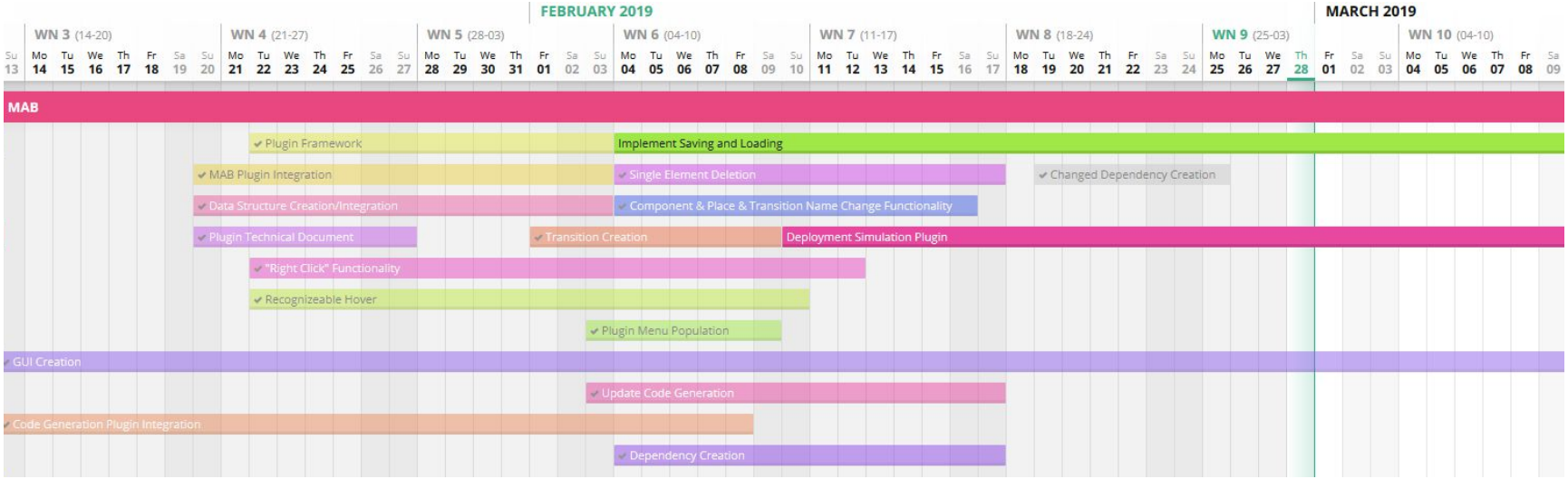


Fig. 10: Gantt Chart

Conclusion

- The Problem
  - MAD software results in good deployment performance but is tedious and complicated to implement
  - Need a way to help visualize software deployments
- Our Solution
  - Develop a Graphical User Interface
    i. Help Visualize an Assembly of components with dependencies
    ii. Accurately Simulate Software Deployment via animation
    iii. Automate the Generation of Madeus Application Deployer Code
    iv. Allow for Saving and Loading of a user created Assembly
- Our Plan Moving Forward
  - Deployment simulation and Saving and Loading

Thank you!

Any questions?