Boad Visualizer
Multiagent and Agent System

Arun Prabhu
Md Zahiduzzaman
Dharmin Bakaraniya

January 12, 2019

1 UI

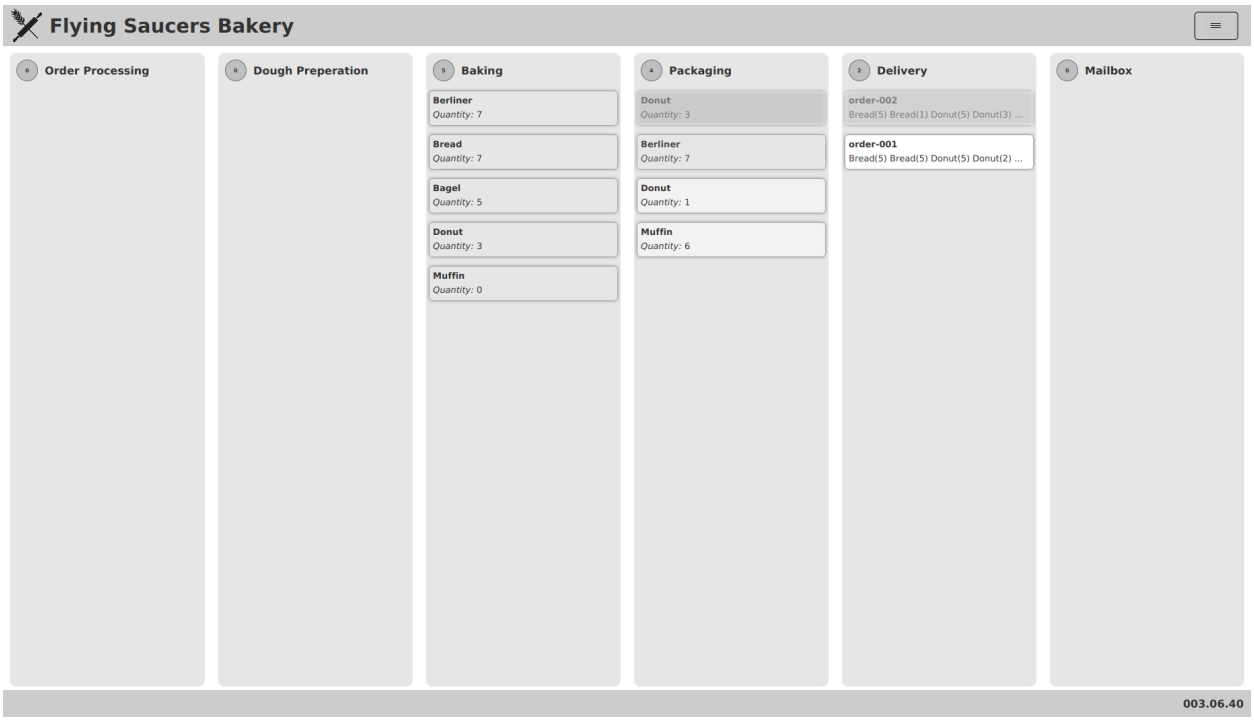

Figure 1: Board visualization in progress

The cards in **Baking** and **Packaging** stage shows unbaked and baked products respectively along with quantity. The **Delivery** stage shows packed products ready for delivery. New cards are added at the top and represented by darker background color compared to older cards in the stage. The circle on left of the label of the stage shows number of cards currently visible in the corresponding stage. The label at bottom right shows the current window. Upon completion, the cards in baking and packaging stage gets cleared and packed orders appears in delivery stage.

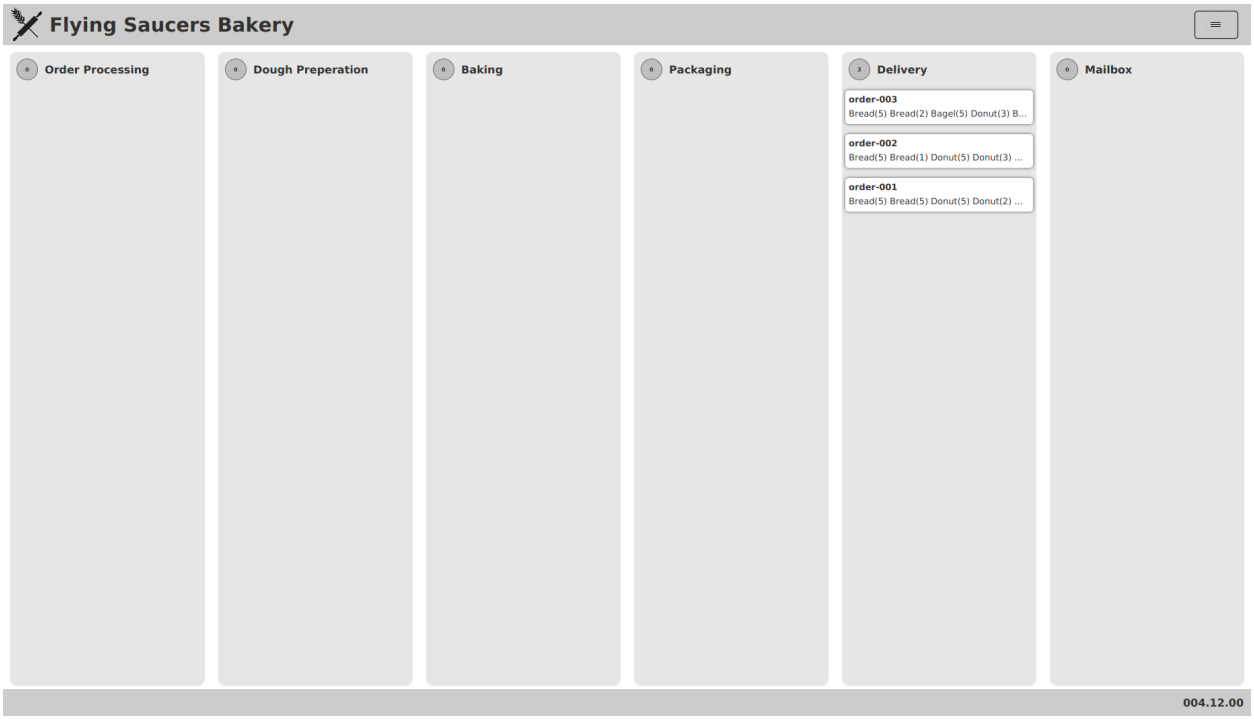

Figure 2: Board visualization complete

We have added a **Replay** menu item at top right to replay the visualization as the visualization at real time is very fast and difficult to trace. The replay shows the visualization again with a small delay.

The visualizer also communicates with **TimeKeeper** and signals the completion of last time step when the UI is closed, so that the TimeKeeper does not shut down the platform and close the UI automatically.

2 Architecture

2.1 VisualizationAgent

VisualizationAgent receives the output messages of the interface agents. It is a JADE agent extended from **BaseAgent**. It creates an instance of JavaFX application instance **Visualizer** during initialization and forwards all interface agent messages to the Visualizer instance.

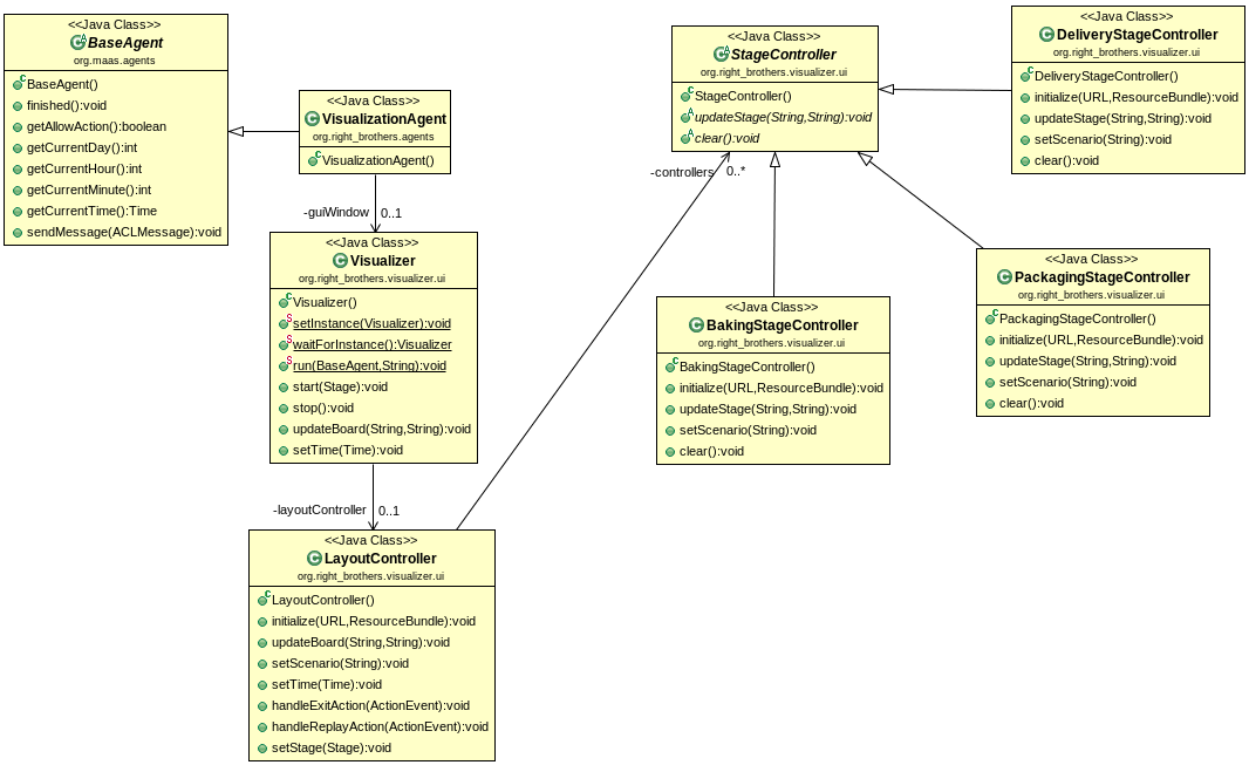

Figure 3: Class diagram of the major classes of the visualizer

2.2 Visualizer

Visualizer is a JavaFX application extended from *javafx.application.Application*. It is the root of all the UI components. It creates the layout of the visualizer UI.

2.3 LayoutController

LayoutController along with the corresponding **FXML** renders the layout of the UI including the logo and label at top left, menu at top right and label to show current time at bottom right.

It updates the the current time based on current time received from **TimeKeeper** and stops TimeKeeper from shutting down the platform until the Visualizer is closed.

It receives the visualization messages from interface agents via Visualizer and forwards to **StageController** instances for visualization. The messages are forwarded to all the StageController instances along with conversation id. It is responsibility of the specific stage controller to decide whether it should process or ignore a message based on the conversation id. For example, **BakingStageController** would process **Proofer** and **Cooling Rack** messages to add or remove unbaked products in baking stage but it will ignore **Loading Bay** messages.

The LayoutController also keeps track of messages forwarded to the StageController instances along with the **Time**. These records are used for replay (top right menu) of the simulation. During replay the LayoutController relays the previously recorded messages after with a small delay for each message to make tracing messages easier.

2.4 StageController

StageController is the **abstract** base class for **BakingStageController**, **PackagingStageController** and **DeliveryStageController**. We have implemented visualization of baking, packaging and input for delivery stage only. Order Processing, Dough Preparation and Mailbox are dummy placeholders and do not visualize the corresponding stages.

Each stage has a label at the top containing the name of stage and circle at it's left showing number of items/cards in that stage at a given time. The sages also have scroll pane which is activated when the hight of stage UI is not large enough for all the cards currently visible in that stage.

2.5 BakingStageController

BakingStageController shows cards representing unbaked products with quantity which are going through baking process. Cards are added for output message of **Proofer** and removed based on output message of **Cooling Rack**.

2.6 PackagingStageController

PackagingStageController shows baked products with quantity which are going through packaging process. Cards are added for output message of **Cooling Rack** agent and removed based on output of **Loading Bay** message.

2.7 DeliveryStageController

DeliveryStageController shows packed products which are ready for delivery. It adds cards for output message of the **Loading Bay** agent. It shows order id along with boxes packed for the corresponding products. The quantity in each box depends on **breadsPerBox** in scenario file.

3 Usage

The visualizer can be run using **visualization** flag. To visualize using **baking** and **packaging** stage agents, following command should be used.

```
gradle run --args="-baking -packaging -visualization"
```