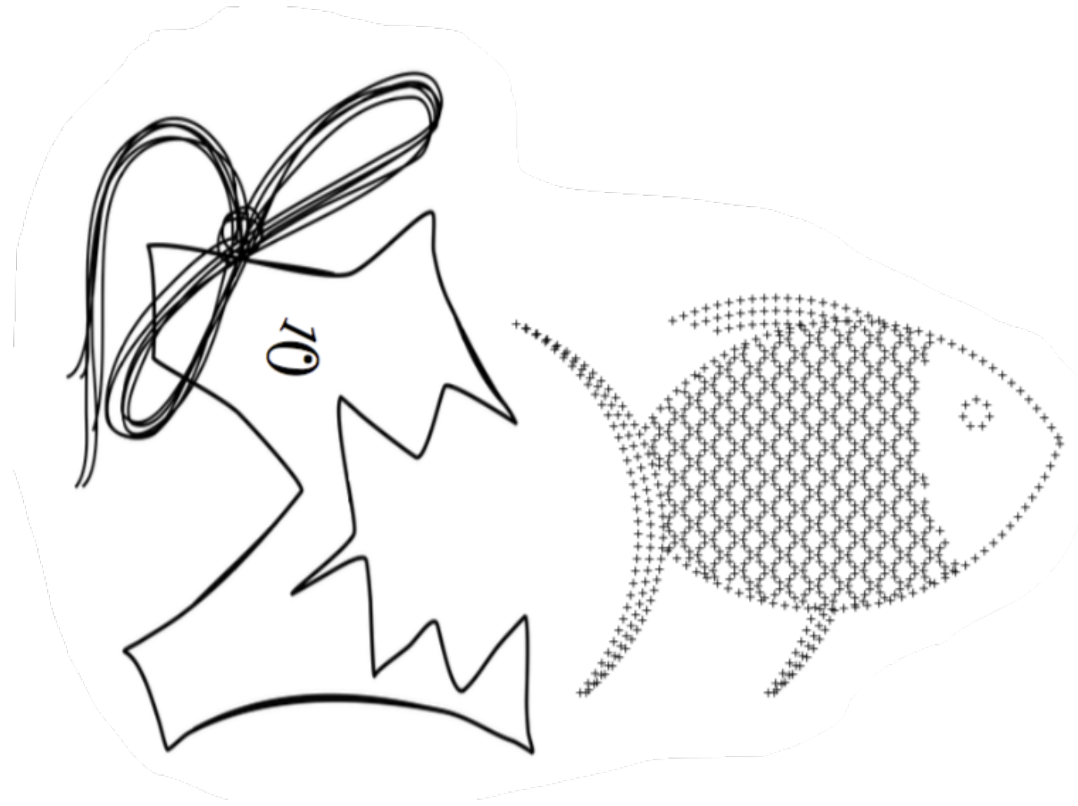
Petri App Land: A Model-Driven Framework for Functional Client-Server Applications

Christopher W. Schankula[†], Ankit Kapoor[†], Lucas Dutton[†], Nhan Thai[†], Senan Gohar[†], Christopher Anand[†]
{schankuc, kapooa13, duttonl, thain1, gohars1, anandc}@mcmaster.ca

[†]Department of Computing and Software, McMaster University
1280 Main St. W, Hamilton, Ontario, Canada L8S 4L8

July 30, 2019



Introduction

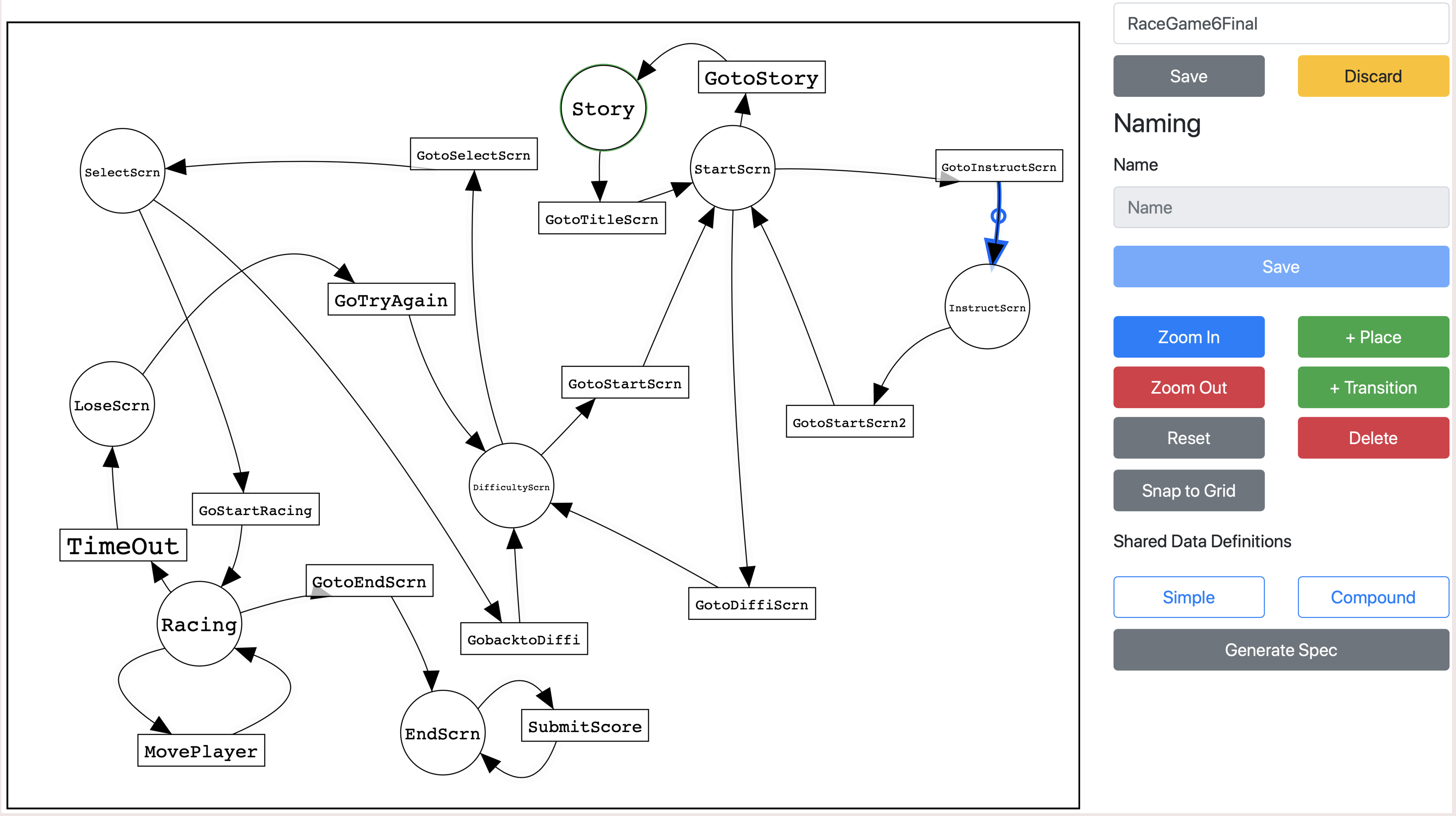
In Model-Driven Development (MDD), developers draw the structure of an app, and a tool generates a template for them to fill in, saving time and avoiding errors. Petri App Land (PAL) is a novel MDD tool for creating multi-user web apps.

PAL Framework

Petri App Land (PAL) is a coloured Petri net, ie, it contains places and transitions moving tokens from place to place. Typically used for analysis, *our* Petri nets are executable, with messages attached to arcs, and state partitioned into places. Our framework generates code for data types and communication, leaving stubs for program logic to be filled in using Elm (client) or Haskell (server).

PALDraw

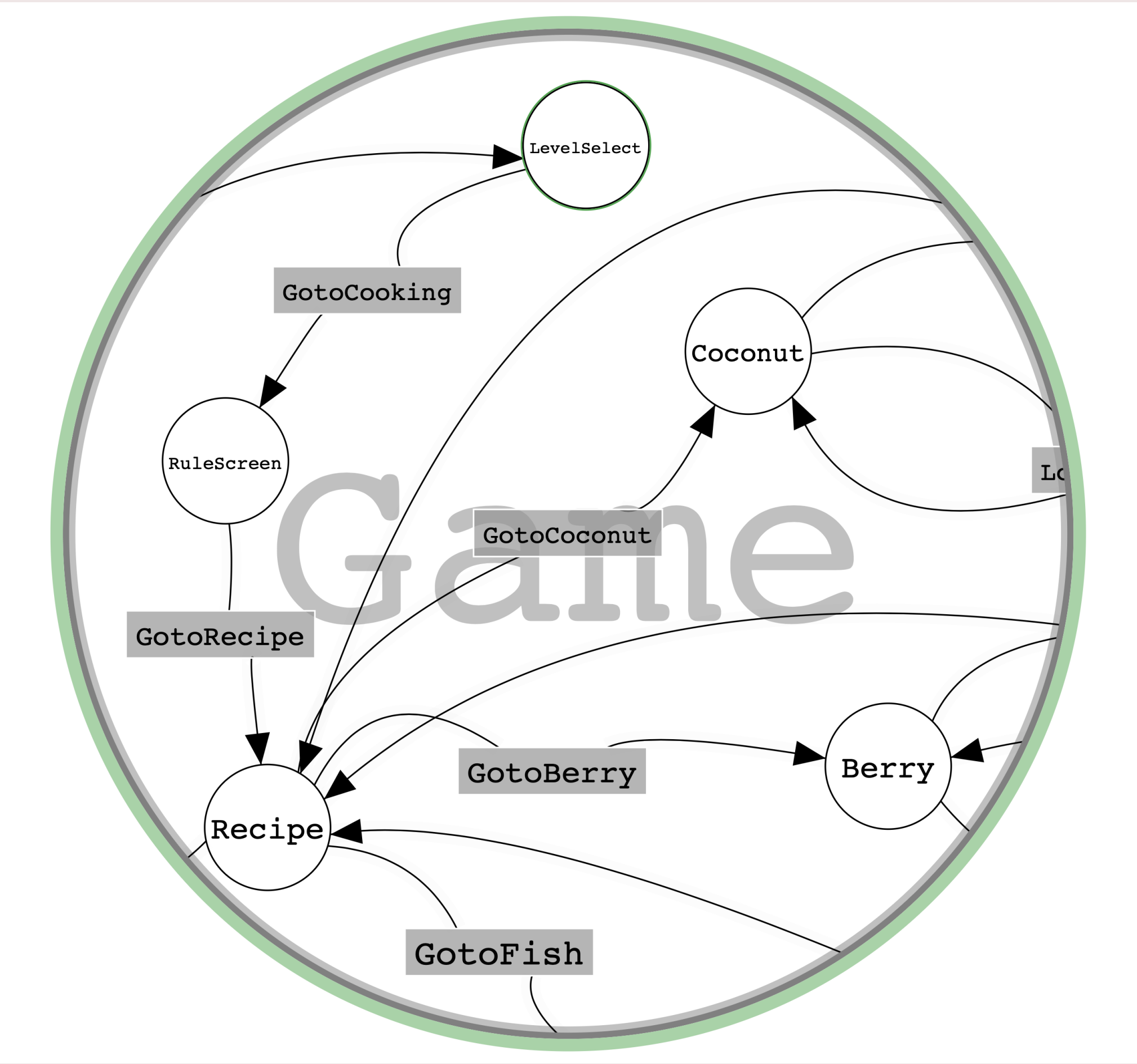
PALDraw is a visual design tool for creating PALs. This screenshot shows a game designed by grade 8s. Places are circles, and transitions are rectangles. Players can only move along arcs.



RaceGame6Final

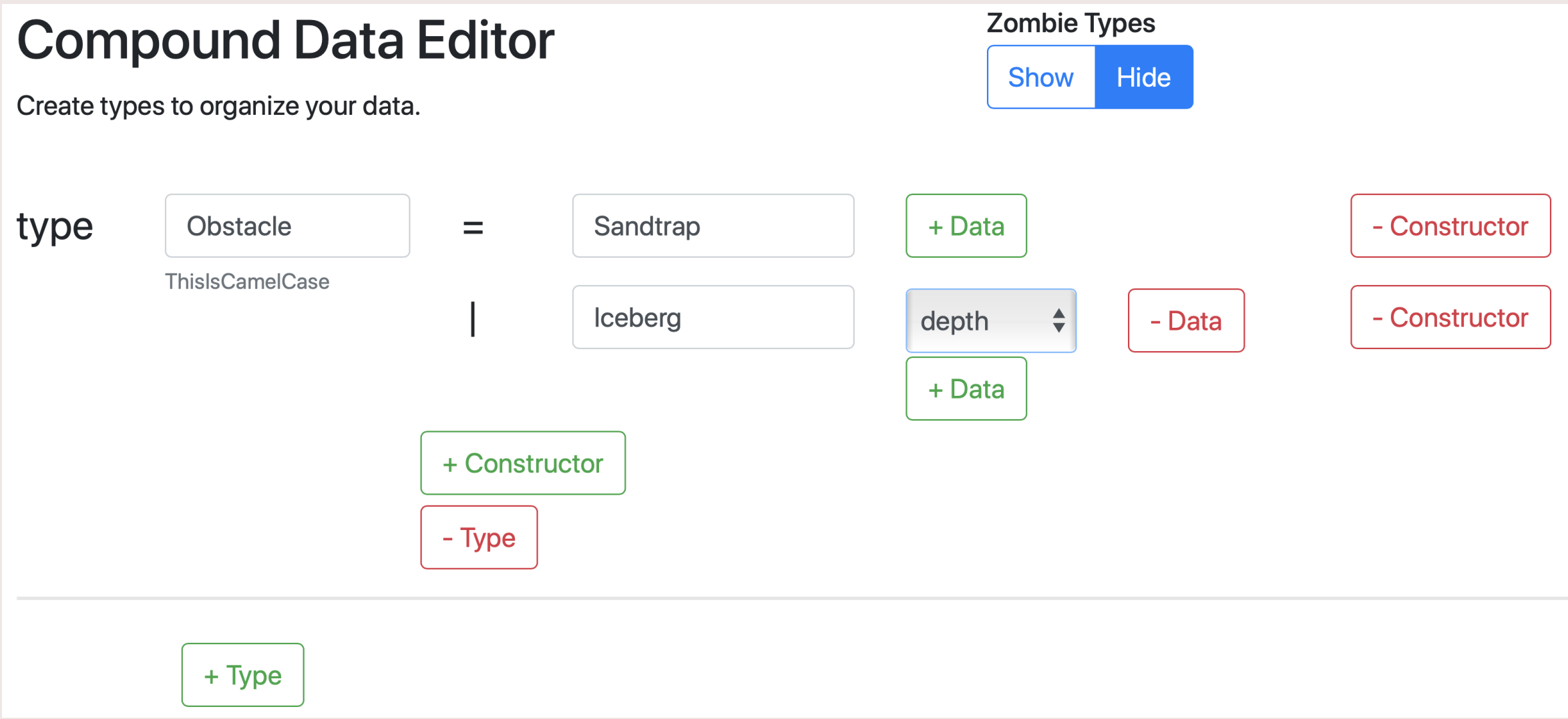
State Diagrams

Novice developers begin by designing State Diagrams for client-only interaction, including for stand-alone apps. Client-only interaction in PAL apps reduces communication overhead, and increases snappiness. PALDraw generates Elm code for each place with an embedded State Diagram, which can be compiled in MacOutreach.Rocks game slots during development and included in the final PAL app without further editing.



Data Modelling

Users model data in states and transition messages using simple and compound shared data definitions.



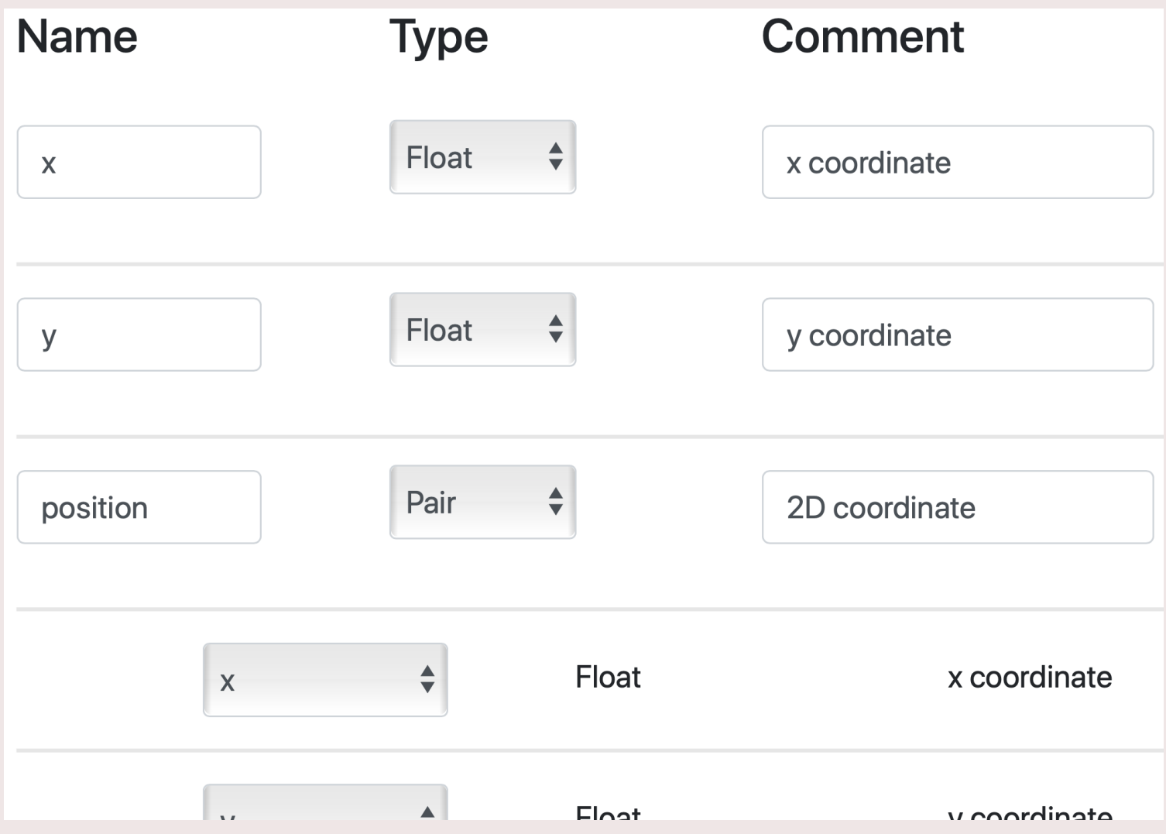
Compound Data Editor

Create types to organize your data.

type Obstacle = Sandtrap | Iceberg depth

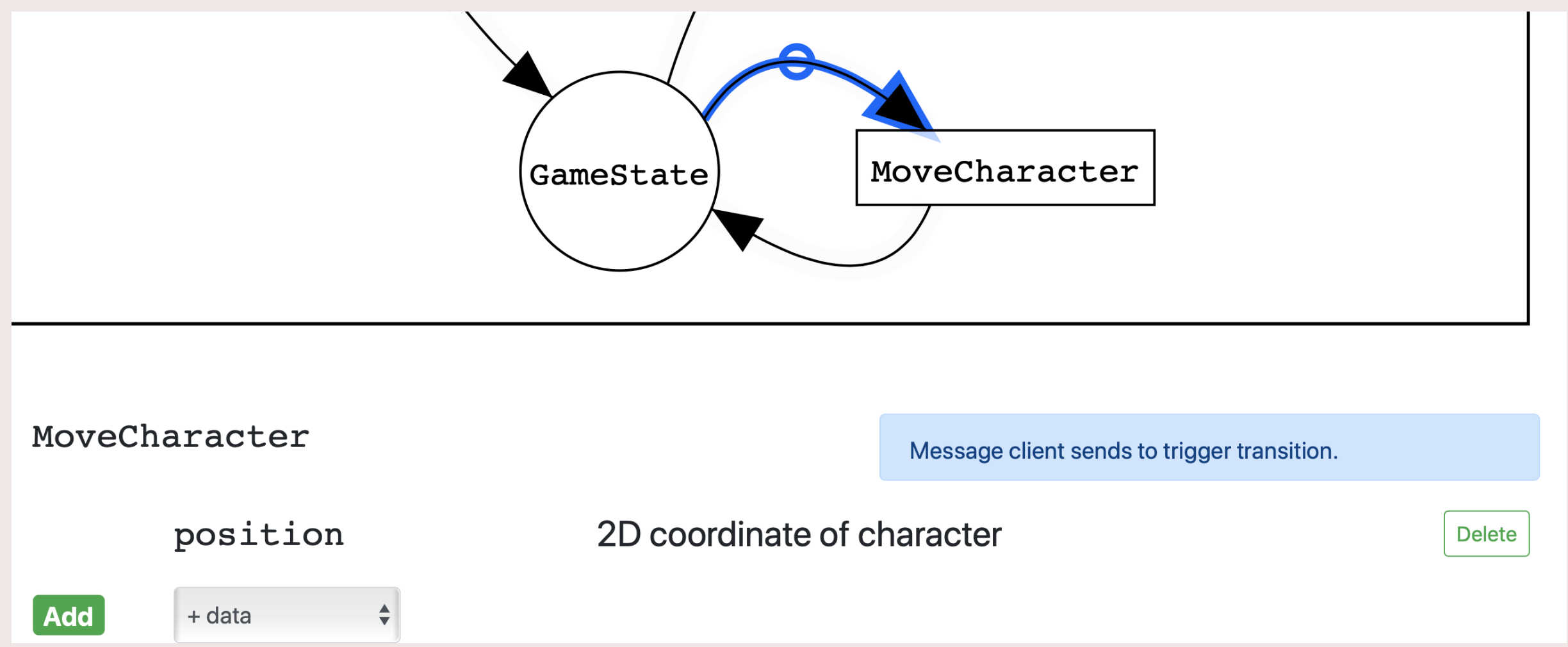
ThisIsCamelCase

The compound data editor creates algebraic data types including sums (e.g., enums) and products (e.g., structs).



Name	Type	Comment
x	Float	x coordinate
y	Float	y coordinate
position	Pair	2D coordinate
x	Float	x coordinate
y	Float	y coordinate

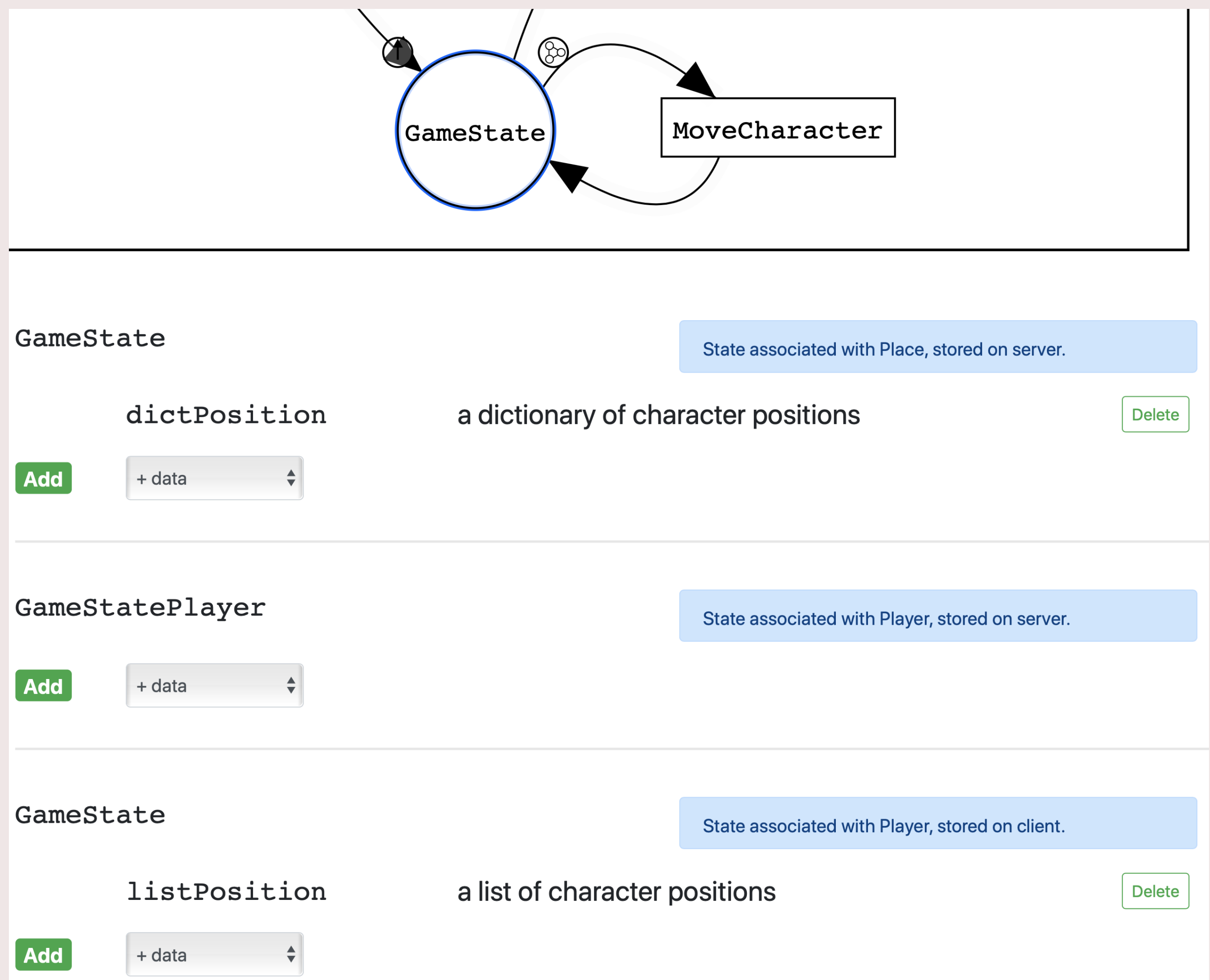
The simple data editor creates named and commented data, equivalent to documented type aliases.



MoveCharacter — Message client sends to trigger transition.

position — 2D coordinate of character

When an arc is selected, users can edit the data associated with the message.



GameState — State associated with Place, stored on server.

dictPosition — a dictionary of character positions

GameStatePlayer — State associated with Player, stored on server.

GameState — State associated with Player, stored on client.

listPosition — a list of character positions

When a place is selected, users can edit the data associated with the state decomposition.

Discussion & Future Work

PALDraw puts all of the architecture information needed by implementers in one place, the next step is to integrate code editing with PALDraw to save time searching and allow building on the server.

Acknowledgements

McMaster Outreach thanks the Dean of Engineering and NSERC PromoScience for funding. We appreciate all the volunteers and K-12 students that make this program possible.