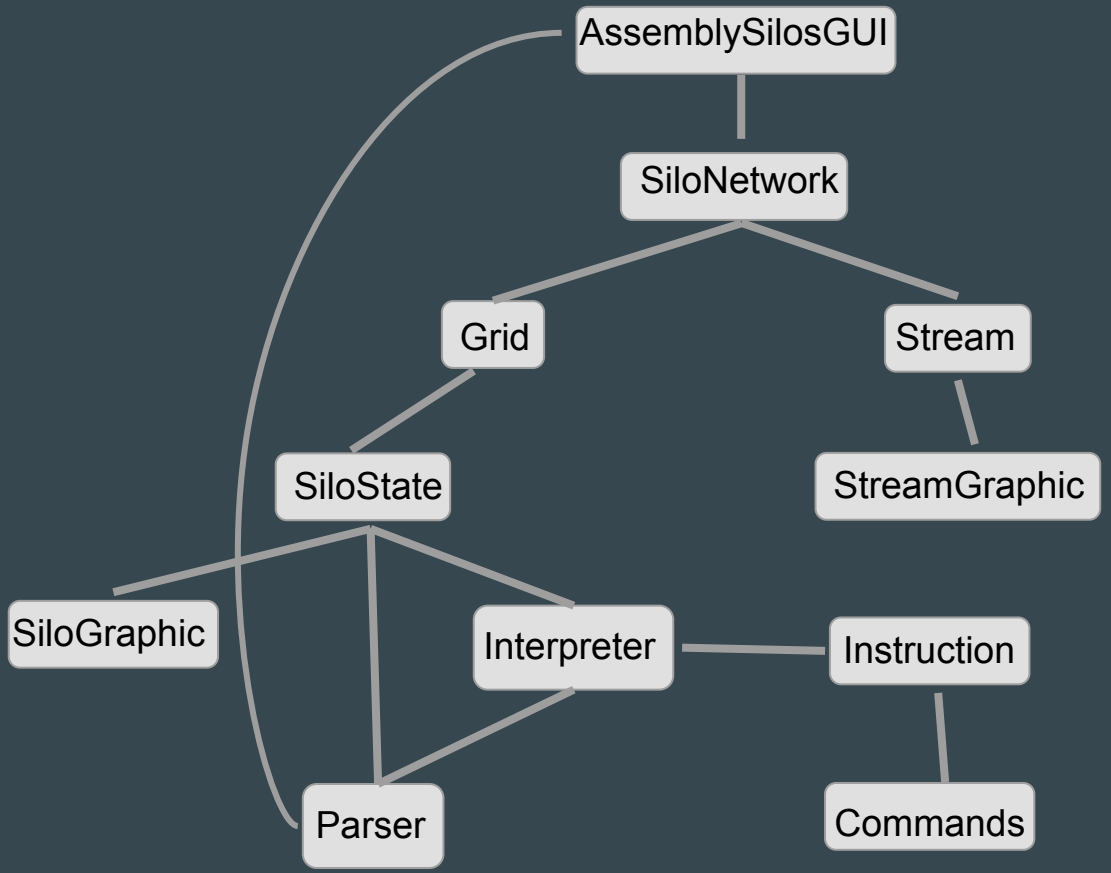
Assembly Silos

CS 351

Luke McDougal, Jack Vanlyssel, Spoorthi Menta

Project structure



Project overview

The Assembly Silos project aims to create a grid of virtual machines called silos, each running a simple assembly language. Each silo runs on its own thread and communicates with other silos through a transfer region. They must remain in sync, executing one instruction at a time and waiting for others to finish their current instruction.

Architecture overview

Each silo contains a program written in a custom assembly language. Silos can communicate with each other through ports (UP, DOWN, LEFT, RIGHT) and store data in registers (ACC, BAK, NIL).

The Assembly language supports various instructions such as NOOP, MOVE, SWAP, SAVE, ADD, SUB, NEGATE, JUMP, JEZ, JNZ, JGZ, JLZ, and JRO.

GUI classes

- AssemblySilosGUI - Implements a graphical user interface for the assembly silo simulation featuring a grid of silos on the right (created with SiloGraphic), input and output streams on the left and buttons under those streams.
- SiloGraphic - Each silo in the grid features an ACC, BAK, MODE, and LAST fields as well as an editable text box in the silo itself containing the instructions that are to be carried out in the simulation.
- StreamGraphic - Creates visual representation of the input and output that silos receive. The stream shown as a number accompanied by an arrow.

Network classes

- SiloNetwork - Implements a network containing a grid of SiloState objects, a list of input streams, a list of output streams, and a Phaser to help synchronize the operation of silos. Also provides methods to start, pause, step, and stop the simulation.
- SiloState - Represents the state of an individual silo in the SiloNetwork. It contains an ACC, BAK, instructionIndex, row, column, and a reference to the SiloNetwork. It also has a phaser to aid in synchronization and a SiloGraphic for the GUI representation.
- Grid - Contains a 2D array of silos (represented by SiloState) and a 2D array of maps that aid in the communication between silos.

Network classes continued

- Parser - Parses an input file into a list of instruction objects, grid dimensions, silo instructions, input streams, and output streams.
- Interpreter - Executes the instructions associated with a specific SiloState. Implements a runnable interface
- Stream - Implements input and output streams and provides methods for controlling the streams.

Command classes

- Instruction - An interface implementing instructions that can be executed by a silo.
- Add - The value of [SRC] is added to the value in the ACC register
- Jez - Jumps control of the program to the instruction following the given [LABEL] if the value in the register ACC is equal to zero
- Jgz - Jumps control of the program to the instruction following the given [LABEL] if the value in the register ACC is greater than zero
- Jlz - Jumps control of the program to the instruction following the given [LABEL] if the value in the register ACC is less than zero
- Jnz - Jumps control of the program to the instruction following the given [LABEL] if the value in the register ACC is not equal to zero

Command classes continued

- Jro - Jumps control of the program to the instruction specified by the offset which is the value contained within [SRC].
- Jump - Jumps control of the program to the instruction following the given [LABEL]
- Label - Labels are used to mark a line of code to jump to. When jumped to the instruction following the label is executed.
- Move - Read [SRC] and write the result to [DST]
- Negate - The value of the register ACC is negated, zero remains zero
- Noop - NOOP is simply an instruction which does nothing
- Save - Write the value from the ACC register onto the BAK register

Command classes continued

- Sub - The value of [SRC] is subtracted for the value in the ACC register
- Swap - Switch the value of the ACC register with the value of the BAK register