Elevator 51

Demonstration of Elevator Scheduling Algorithm

R. Cotrina, K. Day, J. Del Prete, M. Frystacky

Plan

Project Goal

- Create a elevator algorithm that is:
  - Time-efficient
  - Energy-efficient
  - Scalable

Tools Used

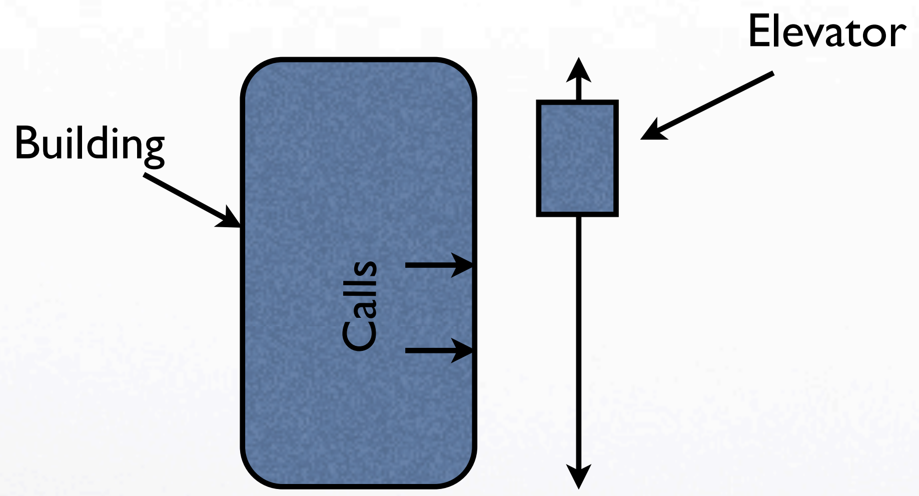
- Eclipse IDE
  - Development
- Git Hub
  - Sharing code, version control
- Asana App
  - Communicating, scheduling, setting goals

Plan

- Create a "basic" algorithm
  - Elevator only changes direction at the top and bottom floors
- Refine to make an "intelligent" algorithm
  - Elevator changes direction when needed
- Compare the "basic" and "intelligent" solutions with different building configurations

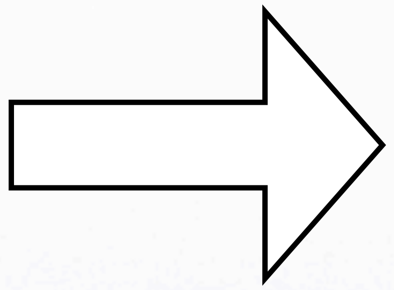
Example

“basic”

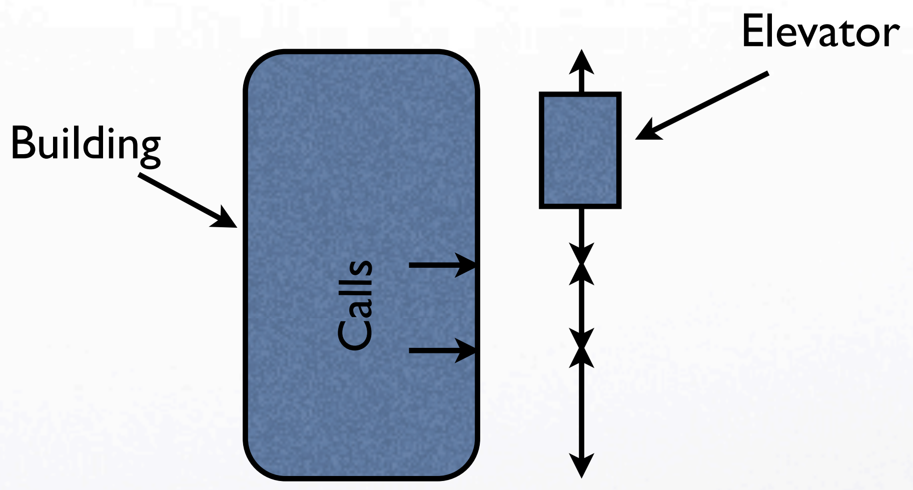


Elevator goes all the way up, all the way down

- Slower
- Inefficient



“intelligent”



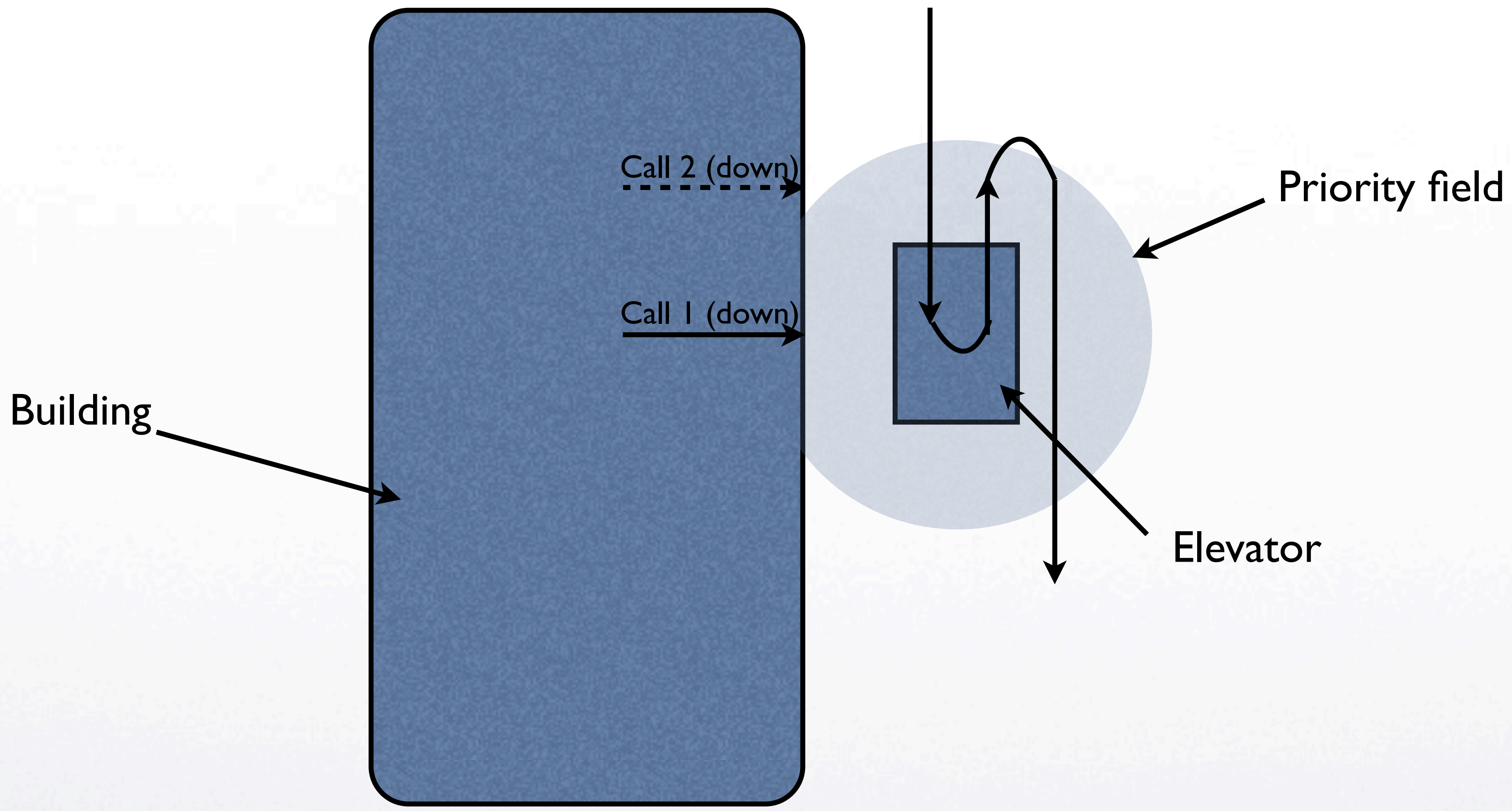
Elevator changes direction intelligently

- Faster
- More energy-efficient
- Description on following slides

“Intelligent” Design

- Elevators are surrounded by a “field”
  - Calls within the field going in the same direction take top priority
  - Prevents wasting people’s time if they miss an elevator by a couple minutes

Example



Because Call 2 is within the elevator's priority field and in the same direction as the elevator, the elevator will go back for it.

Test Cases

- Generate intelligent test cases
  - Flood of arrivals in the morning
  - Most people leaving in the evening
  - Resembles a typical workday

GUI Plan

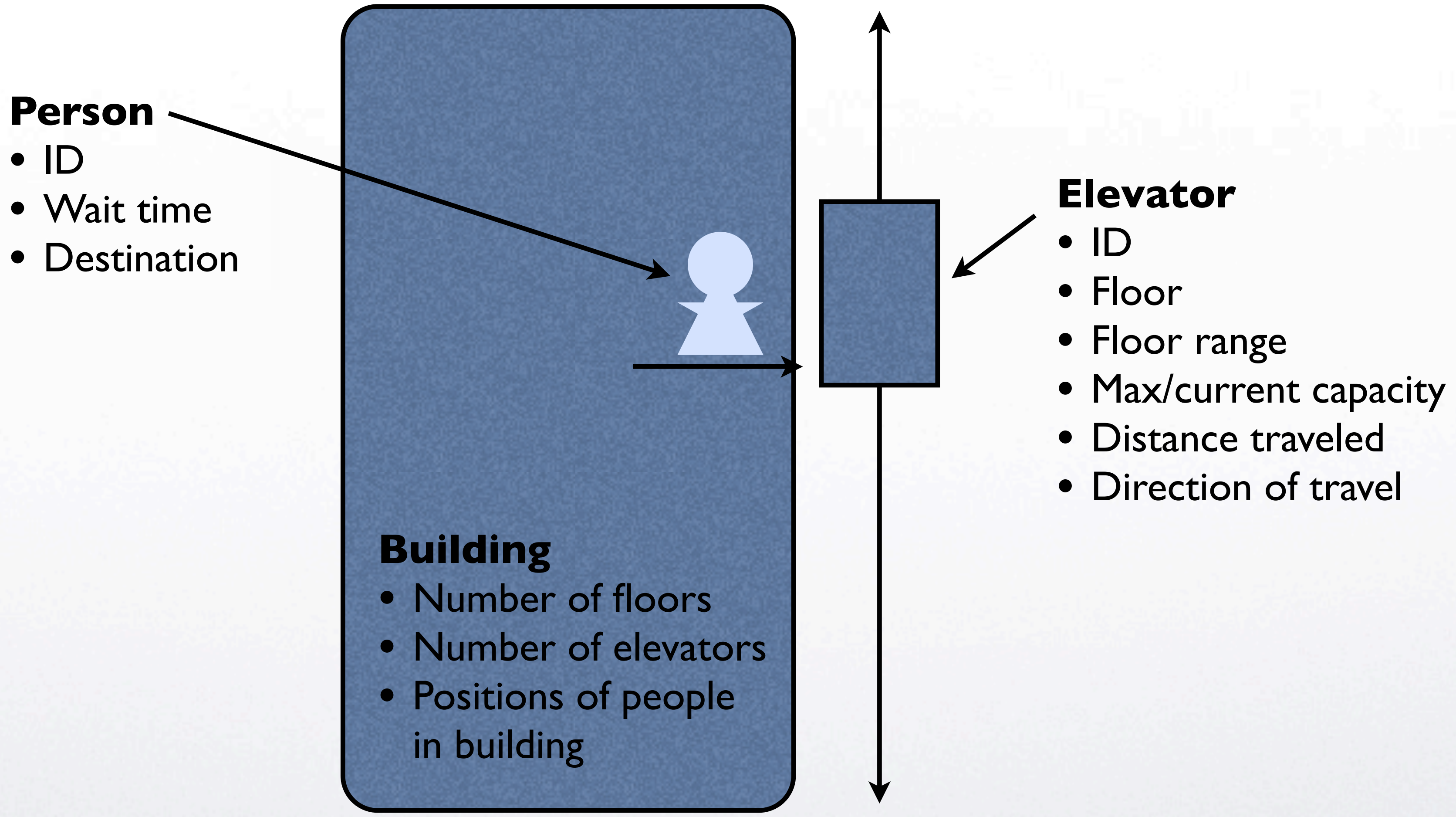
- Initial plan for GUIs:
  - “User experience GUI” - what a person using the elevator will see
  - “Building view GUI” - positions of all elevators in the building
- “User experience” GUI cancelled due to time issues

Solution

Assumptions

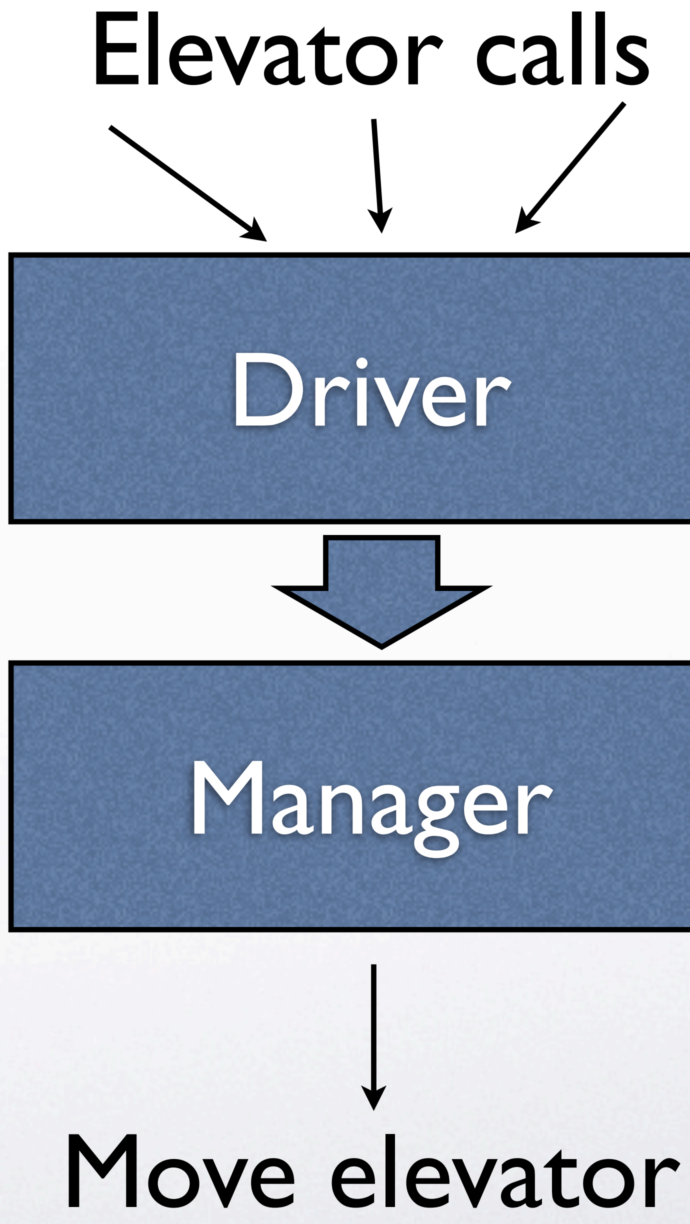
- All elevators move at the same speed
    - 1 floor per time unit
- All floor indexes start at 0

Classes: Objects



Classes: Main

- Driver - Interprets the elevator calls and sends the requests to the manager
- Elevator Manager - Evaluates the situation and moves the elevators according to our algorithm



Classes: Other

- CustomQueue - Custom comparators for comparing elevators and floors

Classes - GUI

- BuildingSwing - Creates a frame to represent a building with *n* floors and *m* elevators
- ElevatorSlider - Component of the BuildingSwing frame that represents one elevator.

Our Implementation

Conclusion

Findings

Issues

- (fixed) People still inside elevators at the end of the day
- (fixed) Elevators spontaneously teleporting between floors