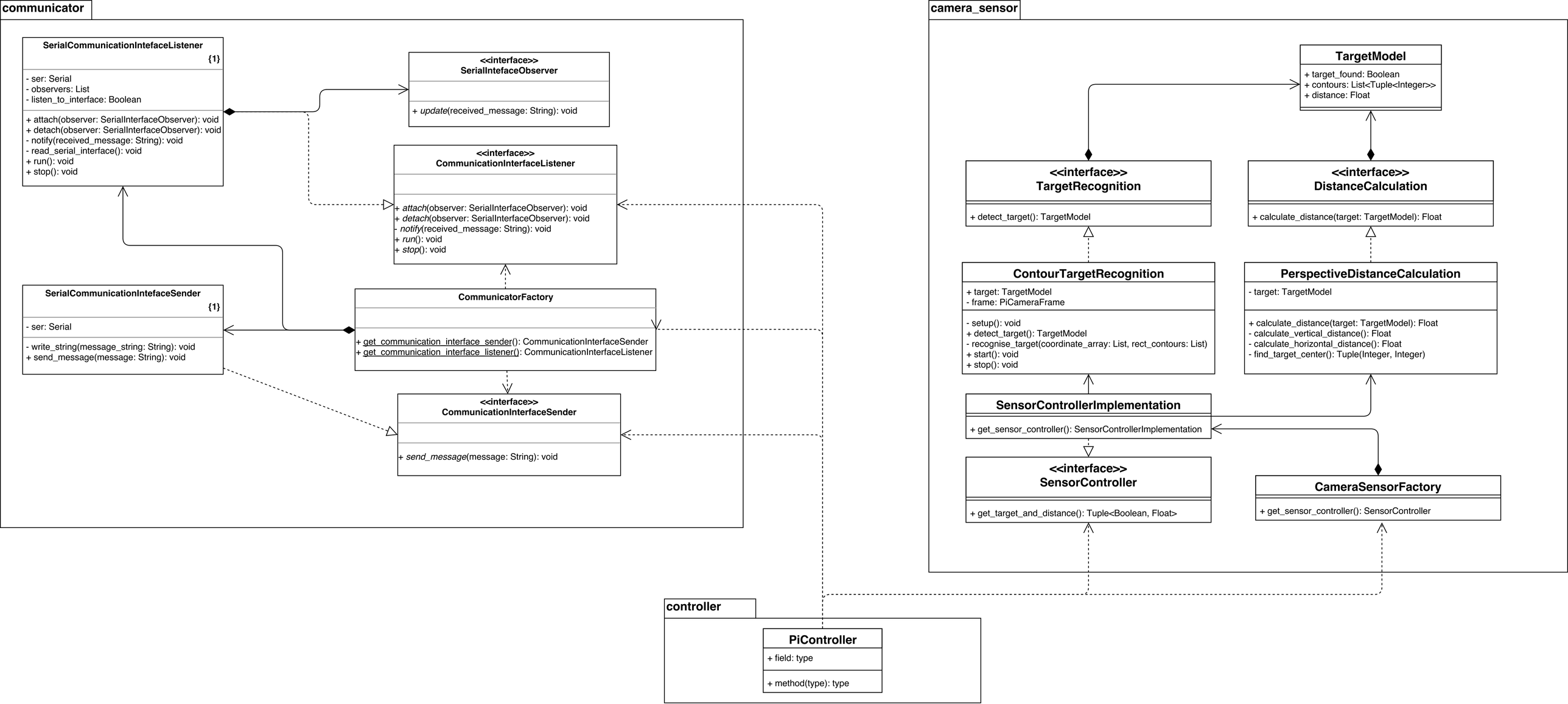
communicator
SerialCommunicationIntefaceListener
{1}
- ser: Serial
- observers: List
- listen_to_interface: Boolean
+ attach(observer: SerialInterfaceObserver): void
+ detach(observer: SerialInterfaceObserver): void
- notify(received_message: String): void
- read_serial_interface(): void
+ run(): void
+ stop(): void
<<interface>>
SerialIntefaceObserver
+ update(received_message: String): void
<<interface>>
CommunicationInterfaceListener
+ attach(observer: SerialInterfaceObserver): void
+ detach(observer: SerialInterfaceObserver): void
- notify(received_message: String): void
+ run(): void
+ stop(): void
SerialCommunicationIntefaceSender
{1}
- ser: Serial
- write_string(message_string: String): void
+ send_message(message: String): void
CommunicatorFactory
+ get_communication_interface_sender(): CommunicationInterfaceSender
+ get_communication_interface_listener(): CommunicationInterfaceListener
<<interface>>
CommunicationInterfaceSender
+ send_message(message: String): void
camera_sensor
TargetModel
+ target_found: Boolean
+ contours: List<Tuple<Integer>>
+ distance: Float
<<interface>>
TargetRecognition
+ detect_target(): TargetModel
<<interface>>
DistanceCalculation
+ calculate_distance(target: TargetModel): Float
ContourTargetRecognition
+ target: TargetModel
- frame: PiCameraFrame
- setup(): void
+ detect_target(): TargetModel
- recognise_target(coordinate_array: List, rect_contours: List)
+ start(): void
+ stop(): void
PerspectiveDistanceCalculation
- target: TargetModel
+ calculate_distance(target: TargetModel): Float
- calculate_vertical_distance(): Float
- calculate_horizontal_distance(): Float
- find_target_center(): Tuple(Integer, Integer)
SensorControllerImplementation
+ get_sensor_controller(): SensorControllerImplementation
<<interface>>
SensorController
+ get_target_and_distance(): Tuple<Boolean, Float>
CameraSensorFactory
+ get_sensor_controller(): SensorController
controller
PiController
+ field: type
+ method(type): type