LOUNGE
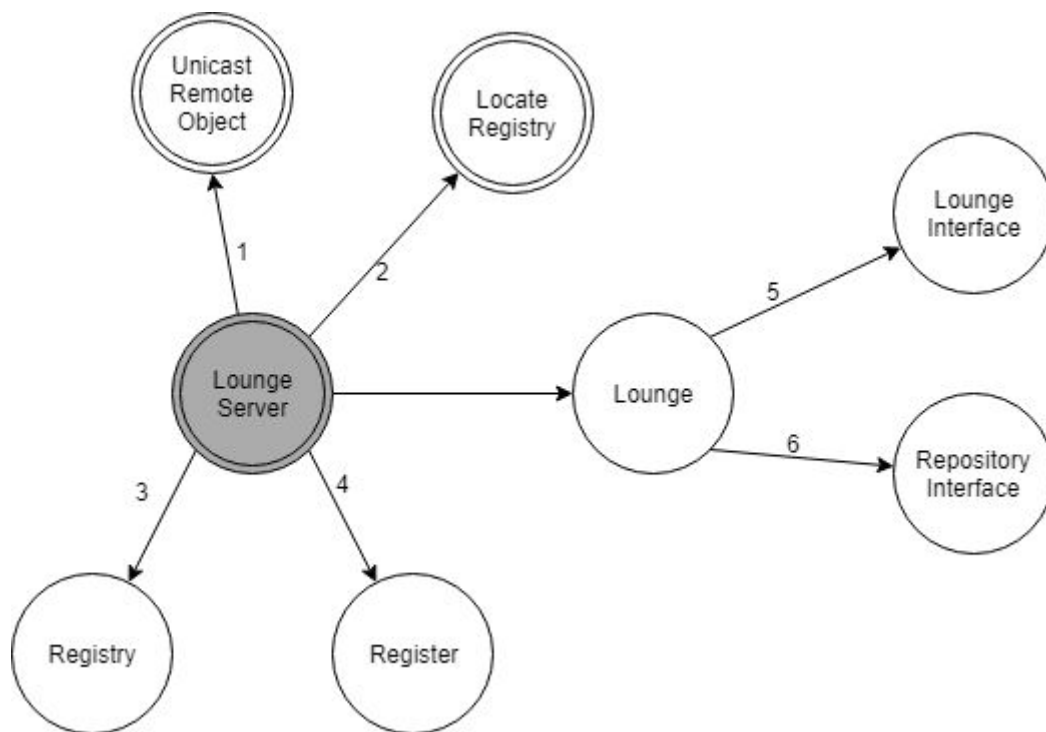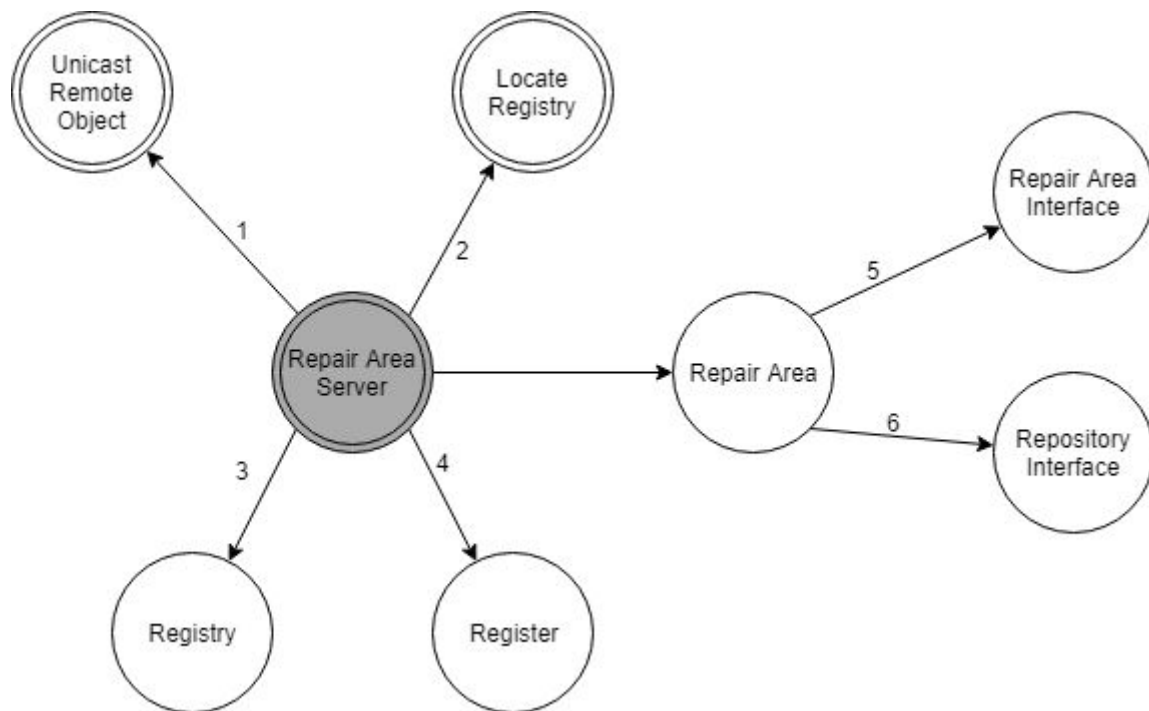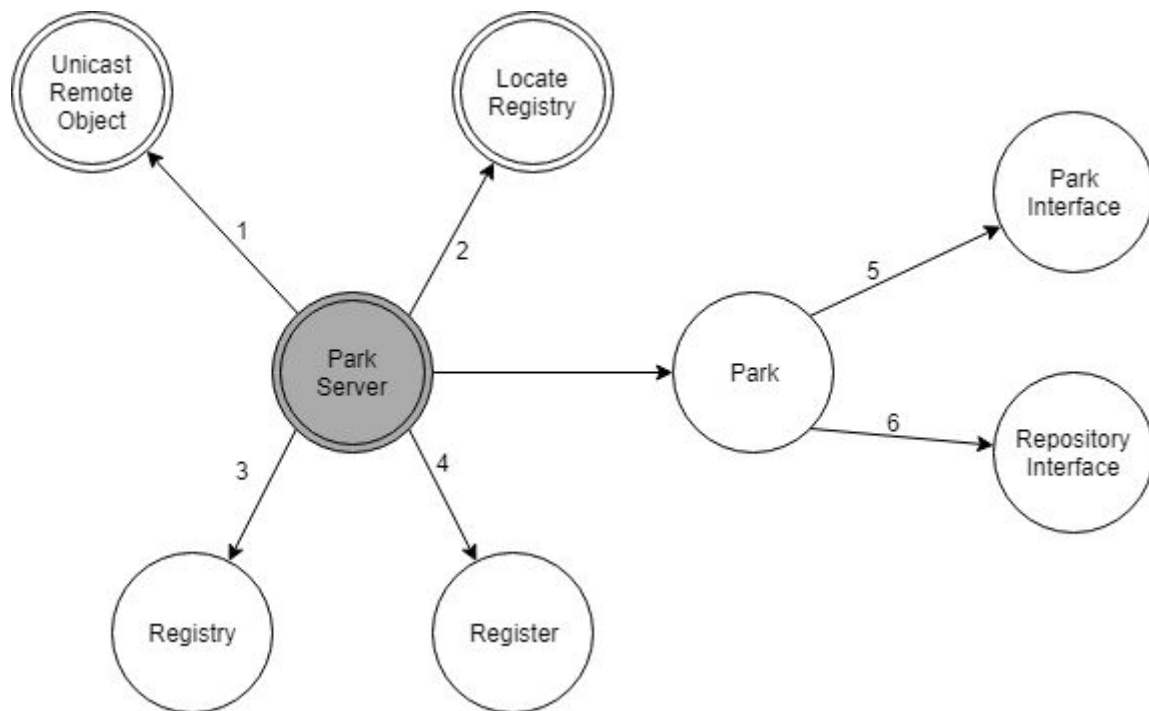

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) queueIn(int id), talkWithManager(boolean carRepaired, boolean requiresCar, int idCustomer), talkWithCustomer(), handCarKey(int replacementCarId, int idCustomer), getCarReplacementId(int id), payForTheService(), receivePayment(), currentCustomer(), collectKey(int id), getNextTask(), checkWhatToDo(), getIdToCall(), appraiseSit(), alertManager(Piece piece, int customerId, int idMechanic), getPieceToReStock(), goReplenishStock(), alertCustomer(int id), addToReplacementQueue(int cust_id), signalShutdown()
6) setManagerState(), setCustomerState(), setMechanicState(), setCustomersQueue(), setReplacementQueue(), setNumCarsRepaired()
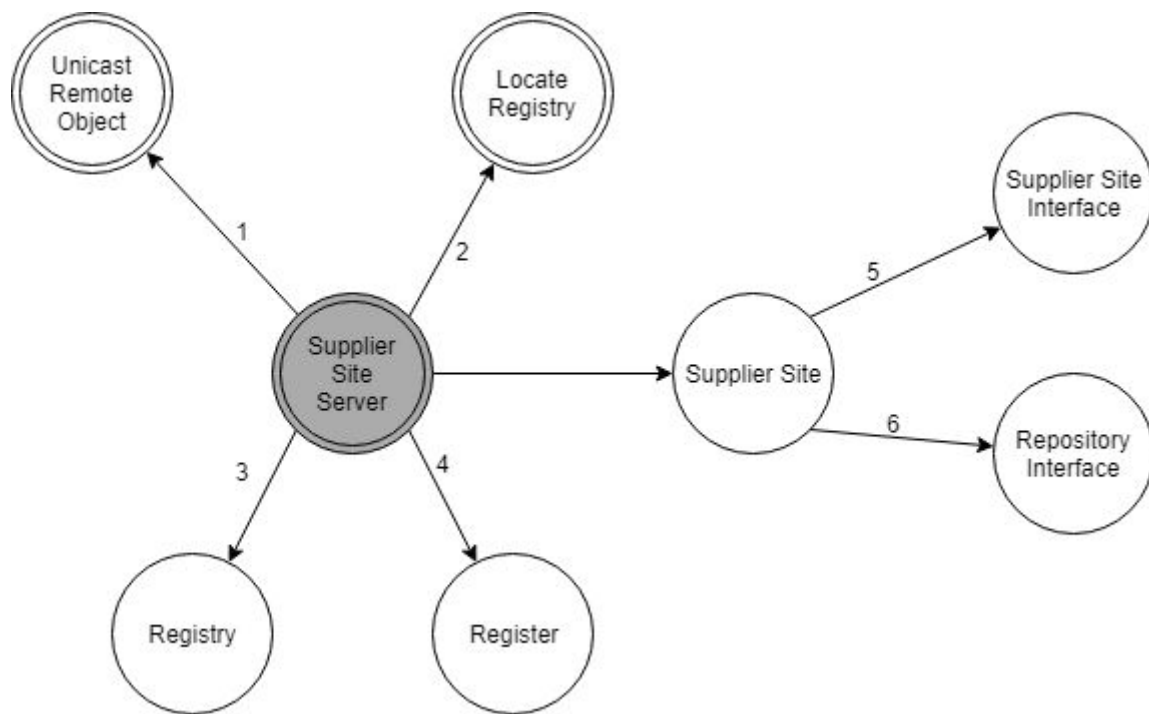
REPAIR AREA

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) getPieces(), readThePaper(int idMechanic), startRepairProcedure(), fixIt(int id, Piece piece), getRequiredPart(int id), partAvailable(Piece part, int idMechanic), letManagerKnow(Piece piece, int idCustomerNeedsPiece), registerService(int idCustomer), storePart(Piece part, int quant), getPiecesToBeRepaired(), enoughWork(), signalShutdown()
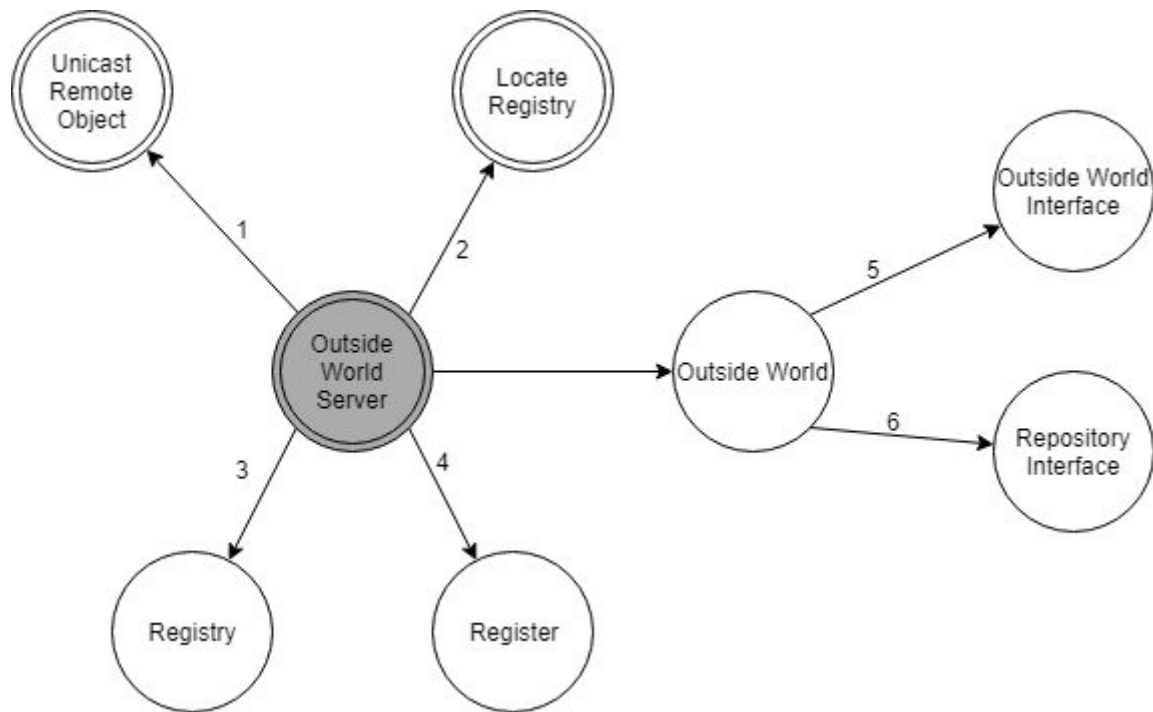6) setManagerState(), setMechanicState(), setPiecesStock(), setRequests(n), setManagerNotifed()

PARK

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) parkCar(int id), findCar(int id, int replacementCarId), getVehicle(int idCar, int idMechanic), returnReplacementCar(int idCar, int idCustomer), returnVehicle(int id), replacementCarAvailable(int idCustomer), reserveCar(int id), signalShutdown()
6) setCustomerState(), setMechanicState(), setCustomersParked(numcars), setReplacementParked(numrepcars)
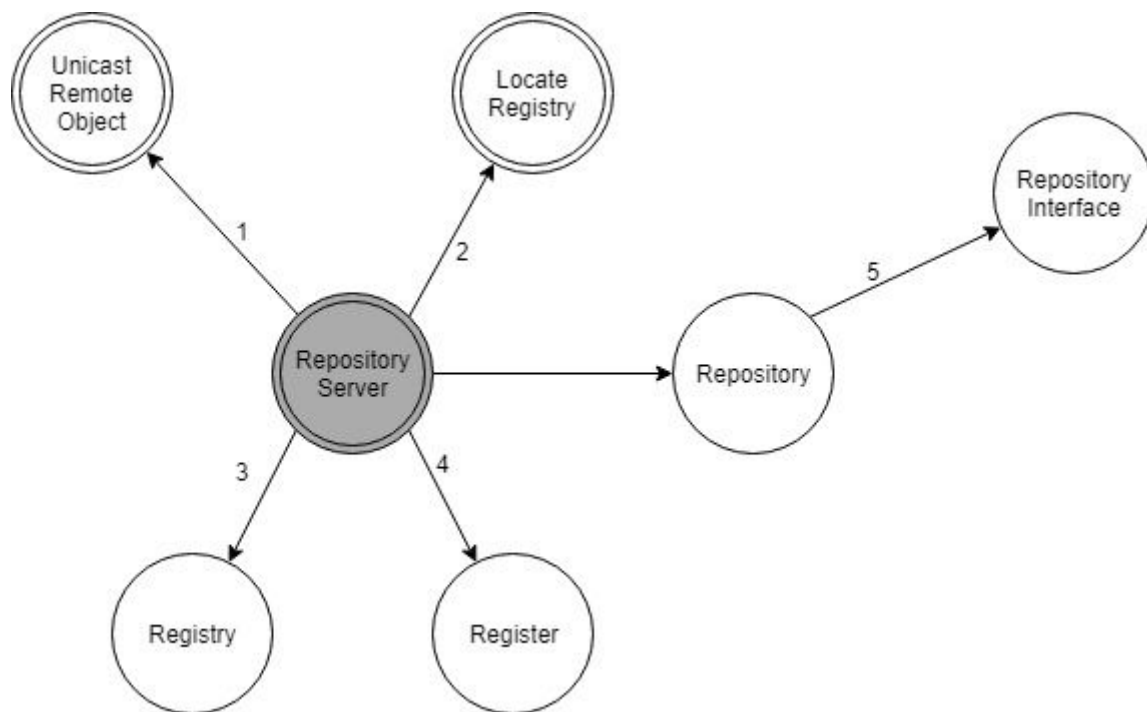
SUPPLIER SITE

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) goToSupplier(Piece partNeeded), signalShutdown()
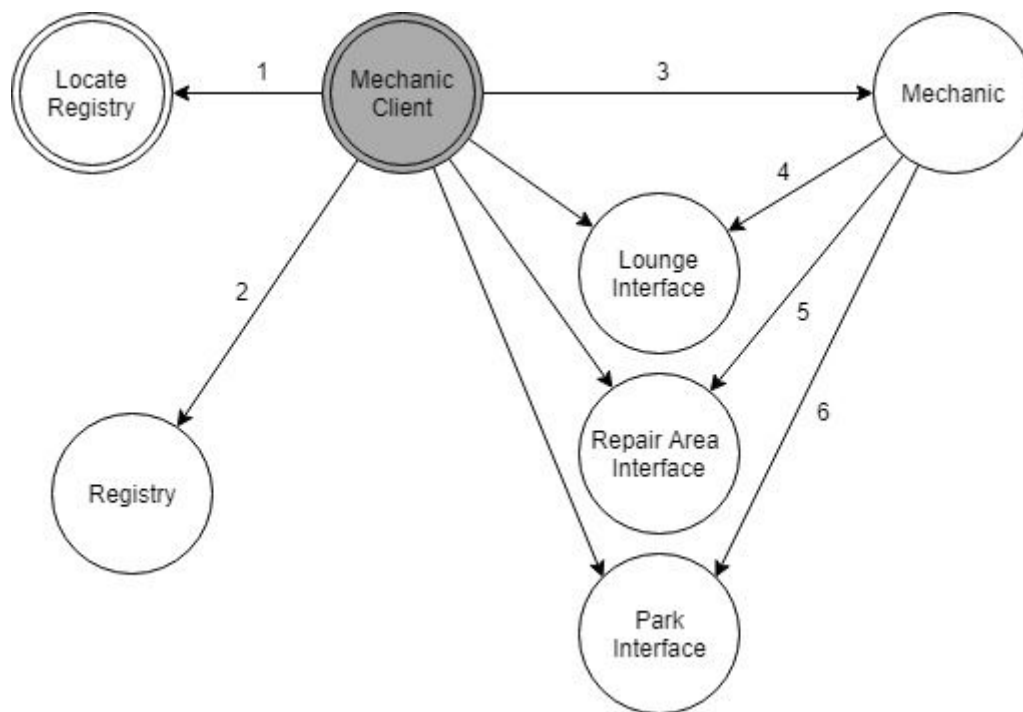6) setBoughtPieces(piecesBought)

OUTSIDE WORLD

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) decideOnRepair(int id), backToWorkByBus(boolean carRepaired, int id),
   backToWorkByCar(boolean carRepaired, int replacementCar, int id),
   phoneCustomer(int id), signalShutdown()
6) setCustomerState(), setVehicleDriven(), setRequiresCar(), setVehicleRepaired()

REPOSITORY
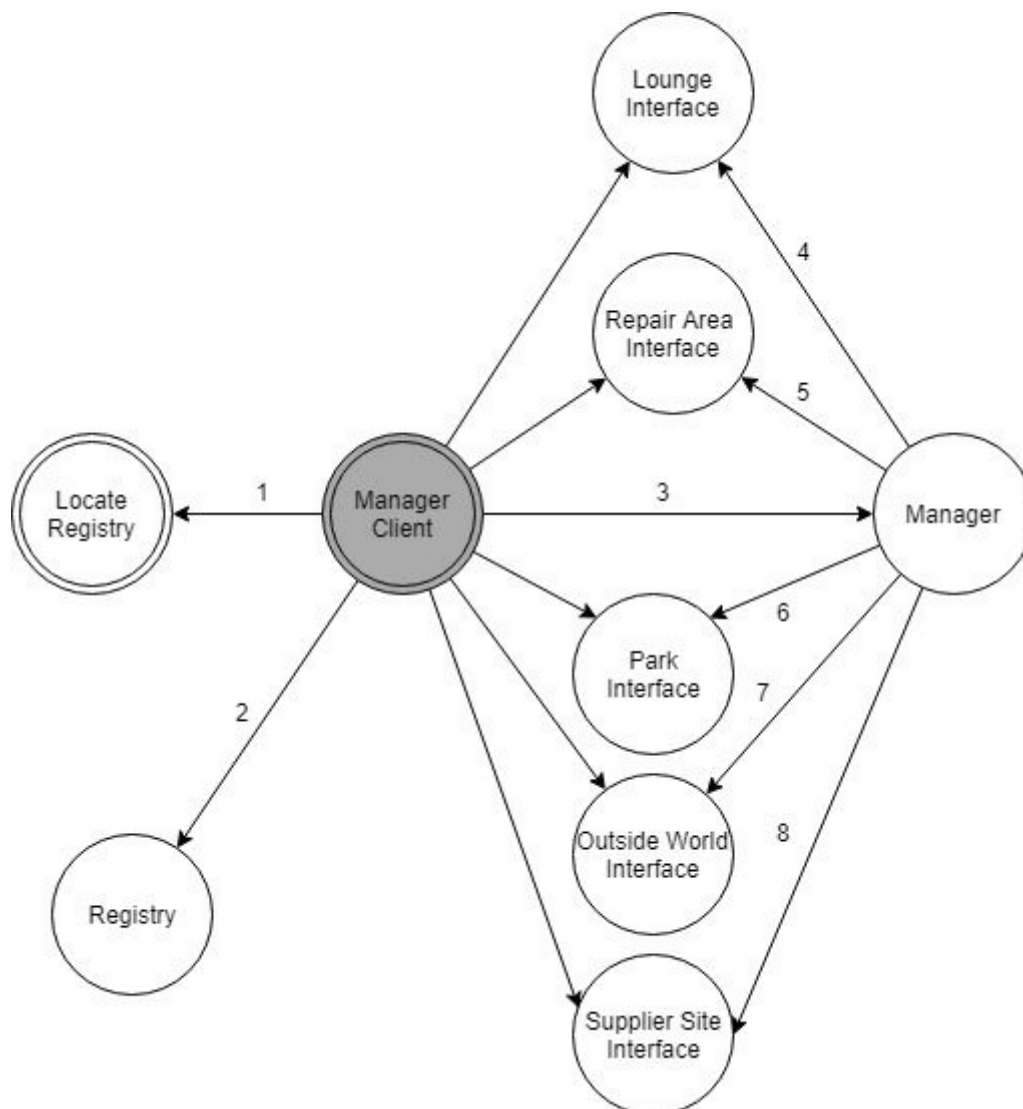

1) exportObject()
2) getRegistry()
3) lookup()
4) bind()
5) updateLog(), setManagerState(String state), setCustomerState(String state, int i), setMechanicState(String state, int i), setCustomersQueue(int size), setReplacementQueue(int size), setNumCarsRepaired(int size), setCustomersParked(int n), setReplacementParked(int n), setRequests(int n), setPiecesStock(HashMap<EnumPiece, Integer> pieces), setPiecesRequired(int[] pieces), setManagerNotifed(String[] not), setBoughtPieces(int[] pieces), setRequiresCar(String s, int i), setVehicleDriven(String s, int i), setVehicleRepaired(String s, int i), finished()
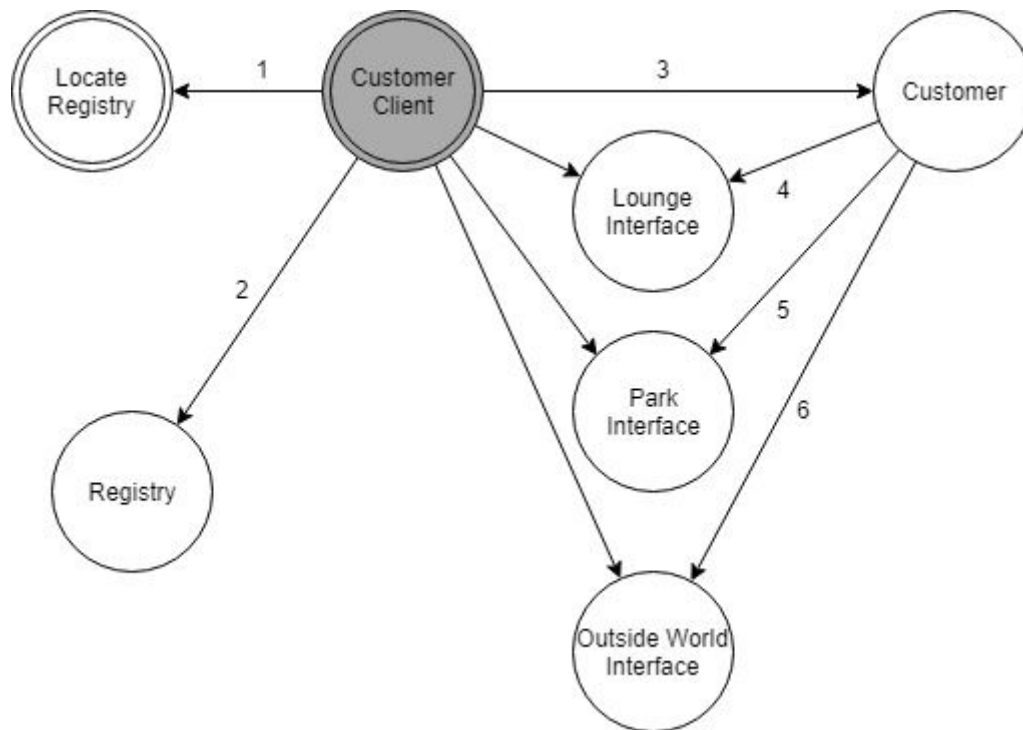
MECHANIC

1) getRegistry()
2) lookup()
3) start(), join()
4) alertManager(Piece piece, int cust_id),
5) readThePaper(), partAvailable(Piece piece), startRepairProcedure(), getPiecesToBeRepaired(), getRequiredPart(int car), fixIt(int car, Piece piece), letManagerKnow(Piece piece, int car_id)
6) getVehicle(int car), returnVehicle(int car)

MANAGER

1) getRegistry()
2) lookup()
3) start(), join()
4) checkWhatToDo(), getNextTask(), appraiseSit(), currentCustomer(),
   talkWithCustomer(), handCarKey(int car_id, int cust_id),
   addToReplacementQueue(int cust_id), receivePayment(), getPieceToReStock(),
   getIdToCall(), alertCustomer(int cust_id), goReplenishStock(),
5) enoughWork(), registerService(int cust_id), storePart(Piece partNeeded, int quant)
6) replacementCarAvailable(int cust_id), reserveCar(int cust_id),
7) phoneCustomer(int cust_id)
8) goToSupplier(Piece partNeeded)

CUSTOMER

1) getRegistry()
2) lookup()
3) start(), join()
4) collectKey(), getCarReplacementId(int id), queueIn(), talkWithManager(boolean carRepaired, boolean requiresCar), payForTheService()
5) parkCar(), returnReplacementCar(int replacementCar), findCar(int replacementCar),
6) decideOnRepair(), backToWorkByCar(boolean carRepaired, int replacementCar), backToWorkByBus(boolean carRepaired)

Legenda:

Main Program

Not instantiated data type; passive entity

Instantiated data type; passive entity

Instantiation, field access or method invocation