**Customer Agent**

Data

String name;

String myChoice;

double cash;

double duesOwed;

double debt;

int hungerLevel = 10;

Timer timer = new Timer();

CustomerGui customerGui;

HostAgent host;

WaiterAgent waiter;

Menu menu;

int tableNumber;

enum AgentState

{DoingNothing, WaitingInRestaurant, BeingSeated, Seated,

Ordering, WaitingForOrder, Eating, DoneEating, Paying, DonePaying, Leaving};

AgentState state;

enum AgentEvent

{none, gotHungry, followWaiter, seated, decided, foodDelivered, doneEating, billDelivering, doneLeaving};

AgentEvent event;

enum CashierEvent

{ none, billDelivered, changeReceived }

CashierEvent cashierEvent

Messages

msgGotHungry() {

event = gotHungry;

cash += debt;

}

msgFollowMe(WaiterAgent waiter, Menu menu, int tableNumber) {

this.waiter = waiter;

this.menu = menu;

this.tableNumber = tableNumber;

event = followWaiter;

}

msgWhatDoYouWant() {

state = Ordering;

makeDecision(); \*\*stub

}

msgOutOfChoice(Menu menu) {

this.menu = menu;

state = Ordering;

makeDecision();

}

msgHereIsYourFood(String choice) {

if (myChoice = choice) {

state = Eating;

event = foodDelivered;

}

}

msgHereIsYourBill(double dues) {

cashierEvent = billDelivered;

duesOwed = dues;

}

msgHereIsYourChanged(double change) {

cashierEvent = changeReceived;

cash = change;

}

msgInDebt(double debt) {

this.debt = debt;

}

msgNotInDebt() {

debt = 0;

}

Scheduler

if (state = DoingNothing and event = gotHungry) {

state = WaitingInRestaurant;

goToRestaurant();

}

if (state = WaitingInRestaurant and event = followWaiter) {

state = BeingSeated;

SitDown();

}

if (state = BeingSeated and event = seated){

state = Seated;

CallWaiter();

}

if (state = Ordering and event = decided) {

state = WaitingForOrder;

GiveOrder();

}

if (state = Eating and event = foodDelivered) {

EatFood();

}

if (state = Eating and event = doneEating) {

state = DoneEating;

}

if (state = DoneEating and cashierEvent = billDelivered) {

payCashier();

}

if (state = Paying and cashierEvent = changeReceived) {

leaveRestaurant();

}

if (state = Leaving and event = doneLeaving) {

state = DoingNothing;

// do nothing

}

Actions

goToRestaurant() {

host.IWantFood(this);

}

SitDown() {

DoGoToSeat(1, tableNumber);

}

CallWaiter() {

timer.schedule(new TimerTask() {

run() {

waiter.ReadyToOrder(CustomerAgent.this);

}

}, time);

}

GiveOrder() {

waiter.HereIsMyOrder(myChoice, this);

}

EatFood() {

event = AgentEvent.none;

timer.schedule(new TimerTask() {

run() {

event = AgentEvent.doneEating;

}

}, hungerLevel \* 1000);

}

payCashier() {

DoGoToCashier();

goingToCashier.acquire();

state = Paying;

waiter.DoneandPaying(this);

}

leaveTable() {

DoExitRestaurant();

waiter.DoneAndLeaving(this);

}

Stub

makeDecision() {

if name = menu.food type

myChoise = name;

else

while cash < menuprices[decision]

decision = random \* menu.size;

myChoice = menu.at(decision);

event = AgentEvent.decided;

}