**1. Order a hotle online before a trip**

Identify Objects and Behaviors:

Objects:

* Traveler:

Data: Name, Check-in Date & Check-out Date, Phone, E-mail Address, Preference

Behaviors: Search, Review, Compare, Reserve, Pay, Cancel

* Internet:

Data: Booking, Airbnb, TripAdvisor, Agoda

Behaviors: Search For Hotel-booking Websites

* Hotel-booking Website:

Data: URL, Available Rooms, Comments, Payment Method

Behaviors: Search, Sort, Filter, Display, Compare, ReserveTheRoom

* Hotel:

Data: Address, Environment, Transportation, Price, Room Type, Available Dates, Comments, Utilities, Discounts

* Payment Method:

Data: Credit Card, Debit Card, PayPal, AliPay

Sequence of invoking behaviors on Objects:

OrderAHotelOnline:

Traveler Alice,

Internet internet,

Hotel-booking Website Booking,

Hotel Hilton,

Payment Method credit card

Booking Confirmation Response

if Internet.isAvailable

Alice.searchOnline -> internet, search: Collection of HotleBookingWebsite

//Alice.findDesirableHotelBookingWebsiteInSecondOption -> Collection of Websites: website

optionNumber = 1;

Loop

if Alice.findsNoOptions

break

end

Alice.findDesirableWebsite -> internet, Input, Option: Website

Website = Booking;

if Booking is not empty

Break

Else

pageNumber = pageNumber + 1

End

if Booking is not empty or Booking != null

Booking.searchForAvailableRoom -> Price, Utilities, AvailableDate, RoomType: Collection of Hotel Room

Alice.reserveHotel -> Name, Check-inDate, Check-outDate, CreditCard, Booking: BookingConfirmation

Else

Alice.cantReserveHotel

Else

Alice.browseInternetAfterAWhileBack

**2. Design an app for calling taxis (e.g. Uber)**

Identify Objects and Behaviors

Objects:

* Developer

Data: plan, frame, prototype

Behavior: demandAnalysis, plan, check, modify, design, writeCode, debug, test, improve, implement

* User

Data: demand, characteristic, type

* Language

Data: C++, java, C#

Behavior: implementApp

* App

Data: name, image, features, functions, interface

Sequence of Flow – Invoke Objects with Behaviors

DesignAnApp:

Developer lu;

User passenger;

Language java;

App uber;

lu.analyzeUserDemand - > program: plan

if(plan is true)

lu.plan - > program, Uber : worked

if(worked is true)

lu.useLanguageToDesign - > java: run

if(run is true)

lu.implement - > uber: interface, functions, features

else

lu.check - > code: debug, modify, improve

end

else

lu.check - > program: change, improve

end

else

lu.rewrite - > plan: problem, change

end

**3. Design a job searching and posting platform**

Identify Objects and Behaviors

Objects:

* Developer:

Data: name, plan

Behavior: demandAnalysis, plan, check, modify, design, writeCode, debug, test, improve, implement

* User1:

Data: demand, name, phone, email address, experience, preference, status

Behavior: search, filter, sort, apply

* User2:

Data: name, address, description, requirement, applyLink

Behavior: edit, post

* Platform:

Data: name, website, interface, functions, database

Sequence of Flow – Invoke Objects with Behaviors

DesignAJobPlatform:

Developer Jack;

User1 jobSeeker;

User2 company;

Platform linkedIn;

jake.analyzeUserDemand - > plan: adopt

if(adopt is true)

jack.designPlatform - > linkedIn: worked

if(worked is true)

jobSeeker.search - > jobs: apply

company.use - > jobs: posted

else

jack.check - > code: debug, rewrite

end

else

jack.redesign - > plan: change

end

**4. Order food in a restaurant**

Identify Objects and Behaviors

Objects:

* Customer:

Data: name, preference, payment

Behavior: lookAtMenu, order, check

* Waiter:

Data: name, service

Behavior: ask, introduce, serve

* Restaurant:

Data: name, menu, specialty, food

Sequence of Flow – Invoke Objects with Behaviors

OrderFoodInRestaurant:

Customer daisy;

Waiter mike;

Restaurant PokeBowl;

daisy.lookAtMenu - > food: like

if(like is true)

daisy.order - > waiter: mike record

else

daisy.leave - > restaurant: change

end

**5. Design a course registration platform**

Identify Objects and Behaviors

Objects:

* Developer:

Data: name

Behavior: demandAnalysis, plan, check, modify, design, writeCode, debug, test, improve, implement

* Student:

Data: name, major, student number, semester

Behavior: filter, select, register

* Platform:

Data: interface, function, course

Sequence of Flow – Invoke Objects with Behaviors

DesignCourseRegistrationPlatform:

Developer Lee;

Platform courseRegistration;

Lee.analyzeUserDemand - > Plan: adopt

if(adopt is true)

Lee.designPlatform - > courseRegistration: work

if(work is true)

student.logIn - > course: register

else

Lee.check - > code: modify, debug

end

else

Lee.rewrite - > plan: change

end