Software Engineering

SBERA GEORGETA

Professional Activity

Product Manager / Product Owner at NTT Data Romania

Experience:

20+ years of experience in software design, development and deployment of multi-tier applications, leading development teams

georgeta.sbera@yahoo.com

Software engineering

Scope and objectives

SCOPE

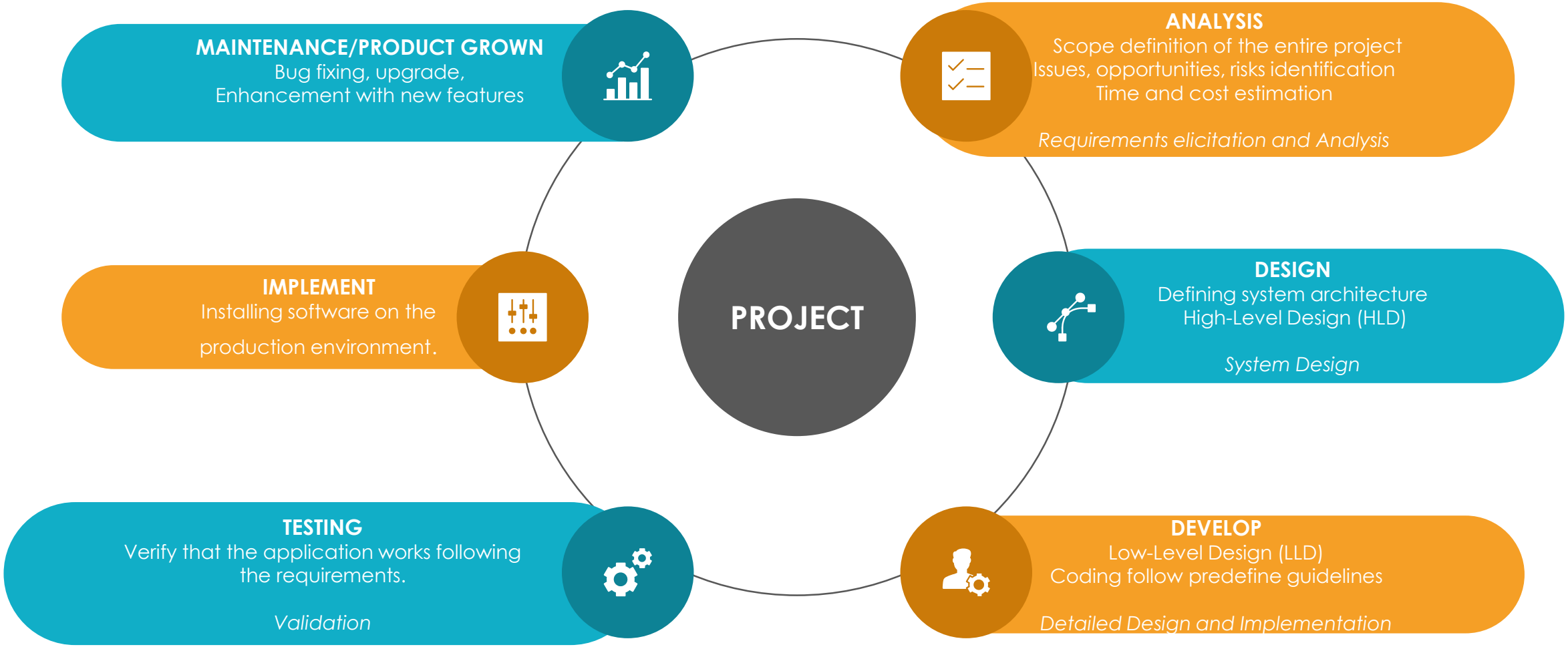
Develop an application following SDLC (software development lifecycle) process, using models for understanding the problem, identifying solutions, using proper architectures and patterns

OBJECTIVES

- Learning by doing:
  - SDLC stages
  - State and understand client requirements
  - Design system and object models for the solution
  - Implementation
  - Testing
  - Documenting

Software engineering

Software development lifecycle stages



AGILE Manifesto and Principle

AGILE – a way to run projects focusing on people

AGILE MANIFESTO

12 PRINCIPLES

- is an approach to project management that centers around incremental and iterative steps to completing project
- is a set of methods and practices where solutions evolve through collaboration between self-organizing, cross-functional teams.

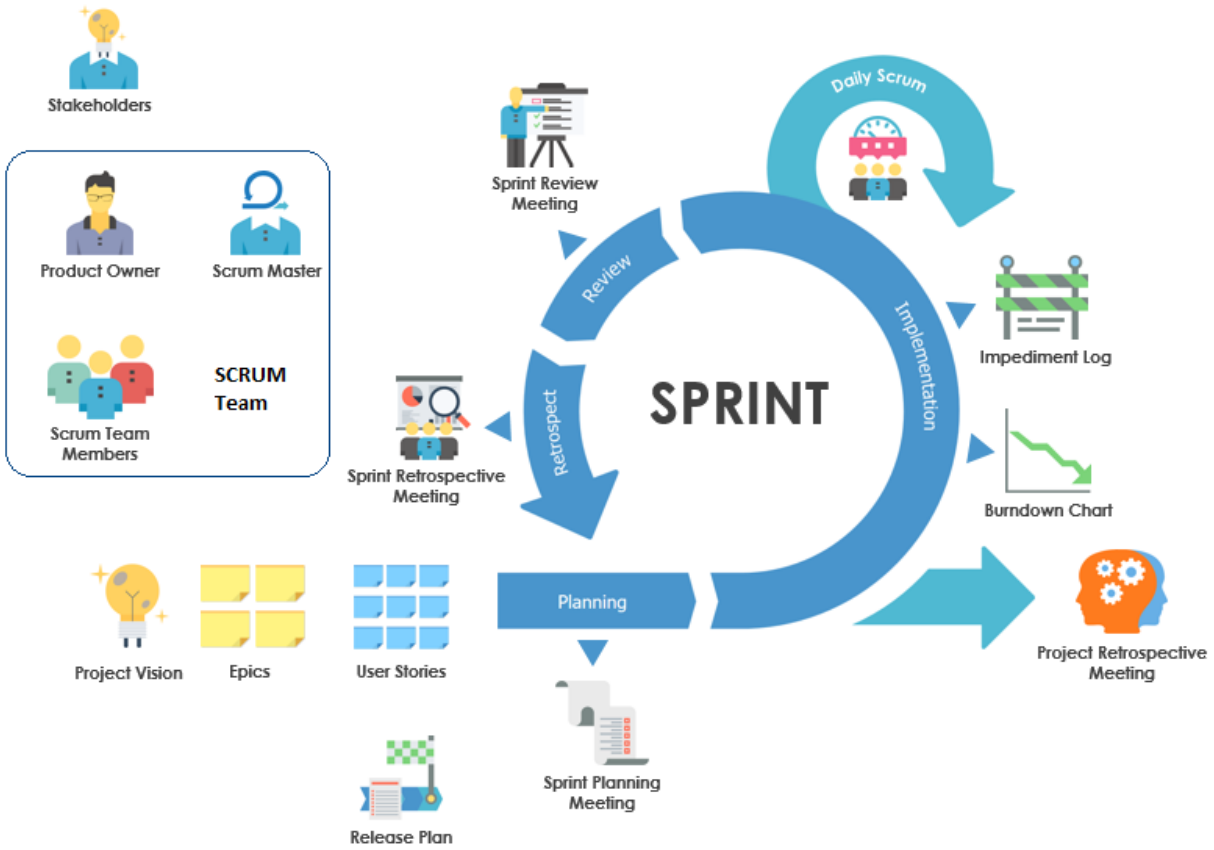
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12.. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

SCRUM Framework

What is SCRUM ?

- is a framework that helps teams work together, encourages teams to learn through experience, **self-organize cross-functional**, while working on a problem, and reflect on their wins and losses to continuously improve
- provides a structure for fast-paced Agile teams to prioritize, manage, and execute work



Software engineering

Product vision

IDEATION

product idea – a business product including functionalities that will attract customers & business partners

E-commerce website

PRODUCT VISION

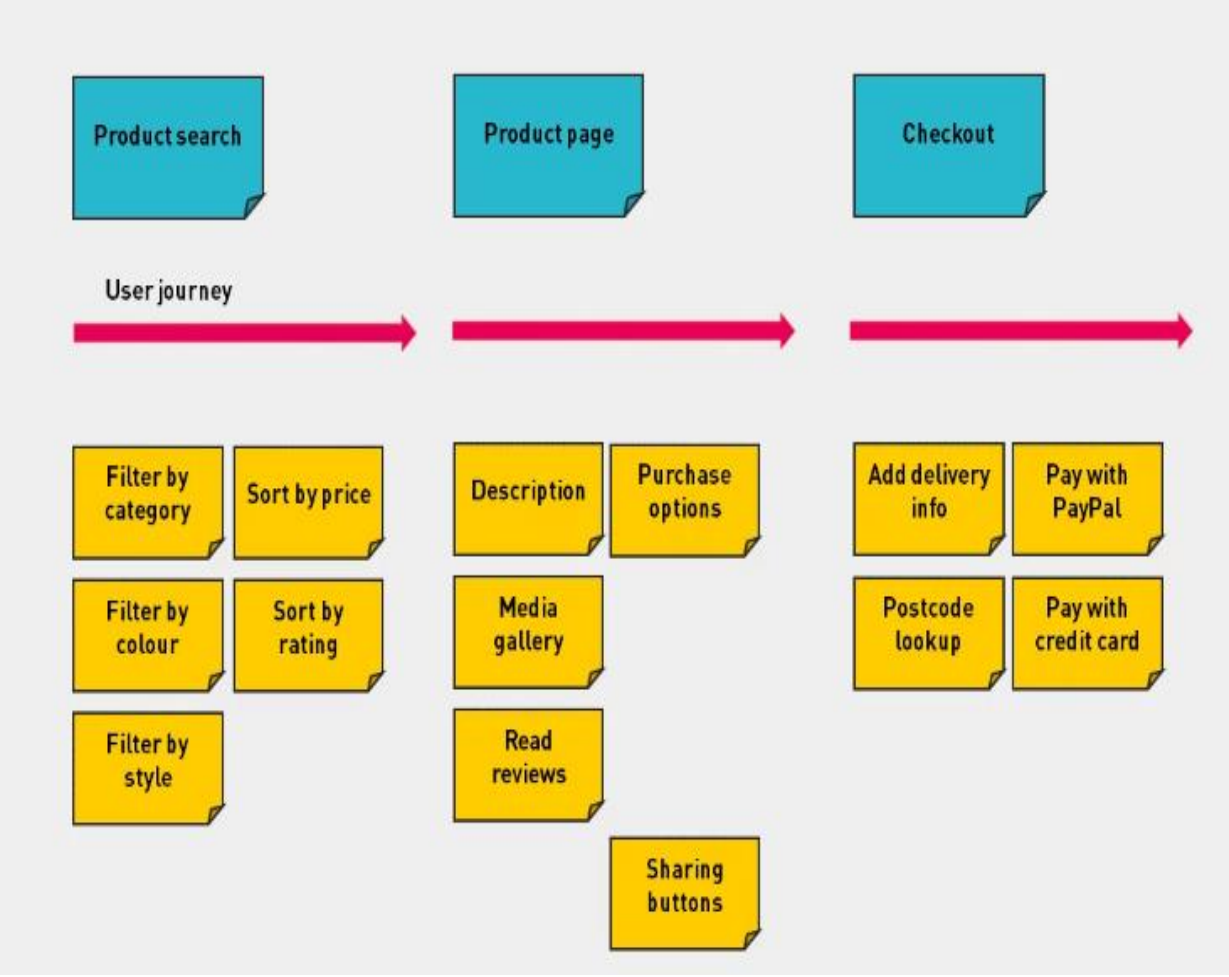
the baseline, the common goal in what we want to achieve with the product.

Product search
Product page
Checkout

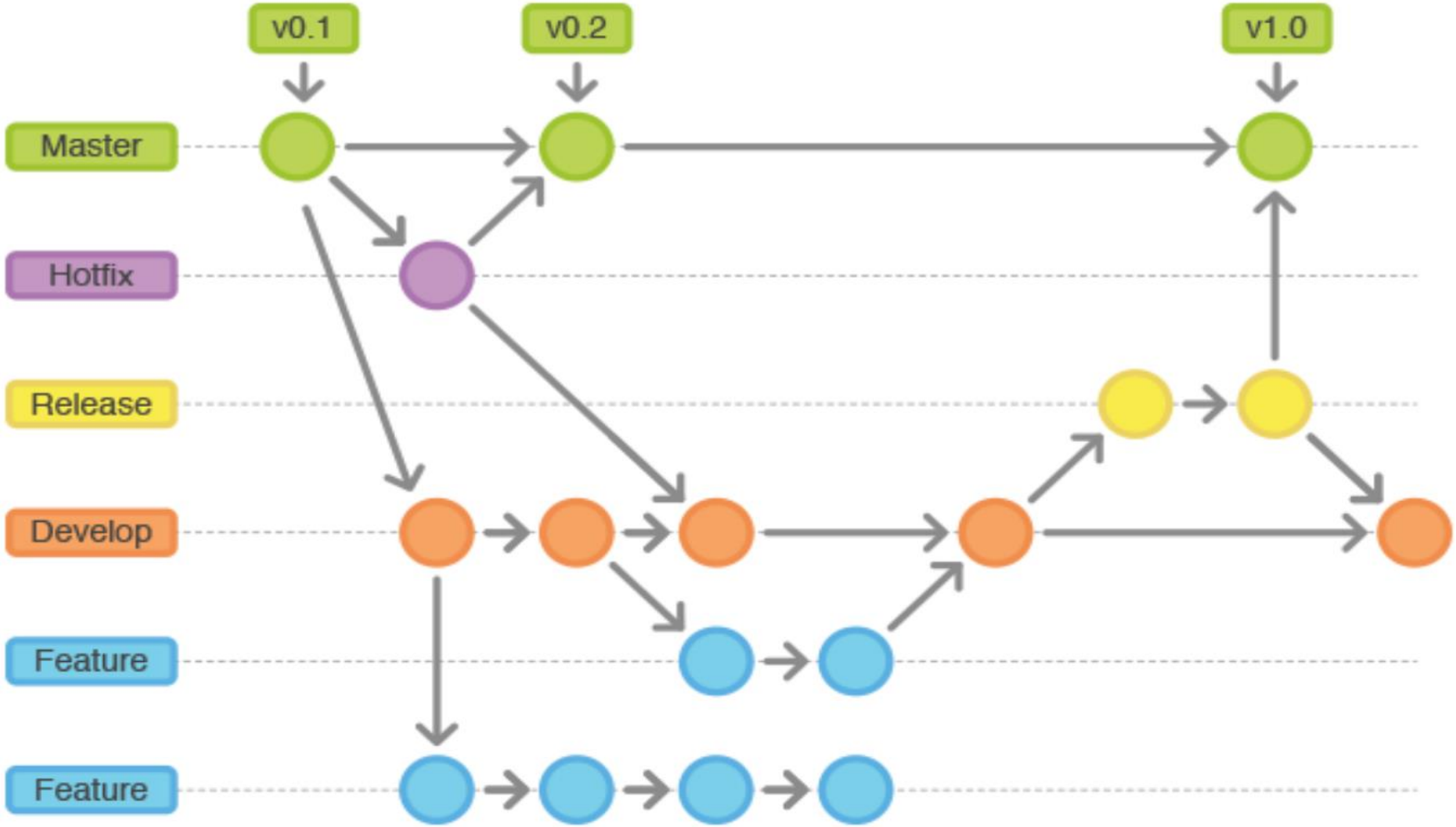
PERSONAS

for whom the product is built considering needs, pain-points, behaviors

Regular customers
Decisions based on reviews
PayPal and credit card holder



Branching strategy



Requirements Elicitation

Specification

describes the system from the user's point of view

"***TicketDistributor*** *is a machine that distributes tickets for trains. Travelers have the option of selecting a ticket for a single trip or for multiple trips, or selecting a time card for a day or a week. The* ***TicketDistributor*** *computes the price of the requested ticket based on the area in which the trip will take place and whether the traveler is a child or an adult. The* ***TicketDistributor*** *must be able to handle several exceptions, such as travelers who do not complete the transaction, travelers who attempt to pay with large bills, and resource outages, such as running out of tickets, change, or pow"*

Requirements

specify a set of features that the system must have

functional requirement is a specification of a function that the system must support
"***TicketDistributor*** must enable a traveler to buy weekly passes."

nonfunctional requirement is a constraint on the operation of the system that is not related directly to a function of the system (usability, performance, supportability)
"***TicketDistributor*** *must be written in Java*."
"**TicketDistributor** must be easy to use"

Actors

represent the external entities that interact with the system, with unique name and description

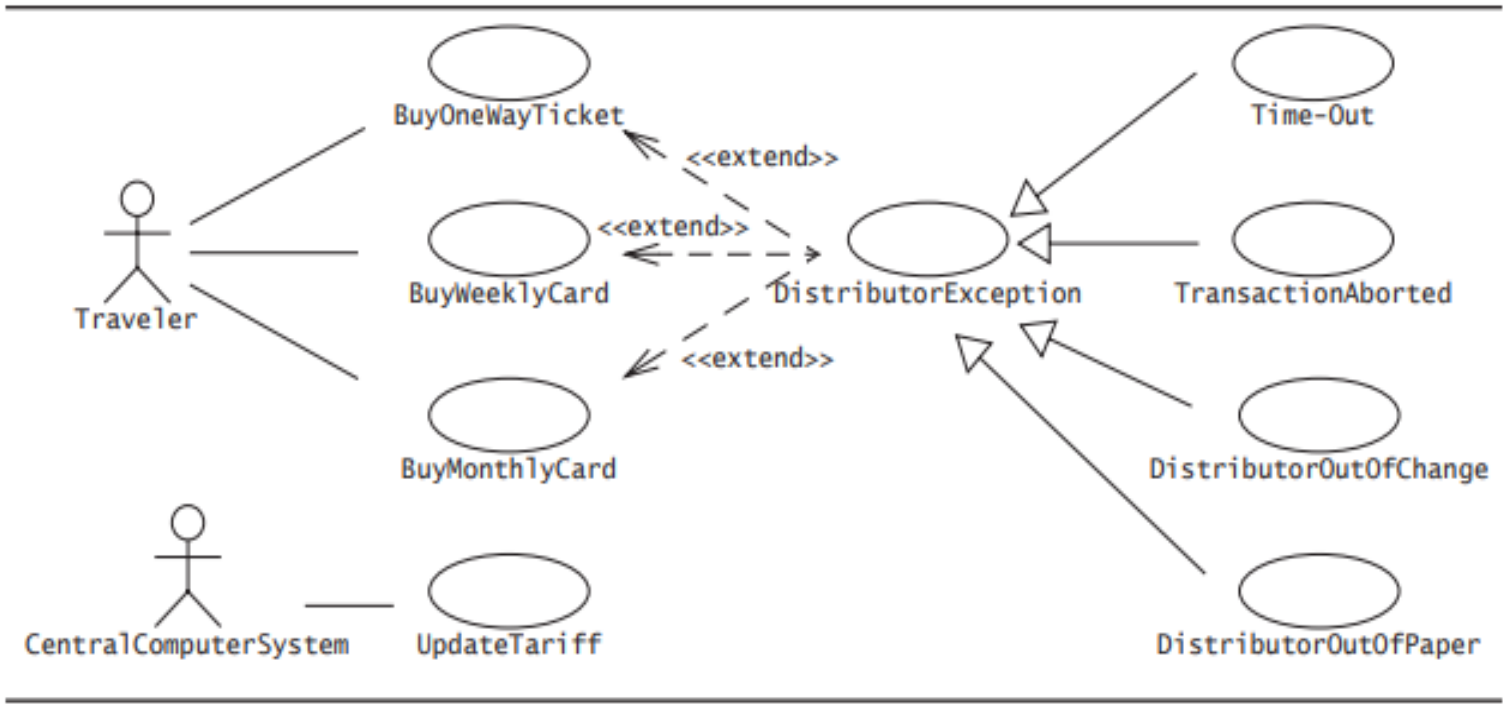
a user role (e.g., a system administrator, a traveler)
another system (e.g., a central database for train routes and tariff).

Requirements Elicitation

Use cases

reprezent the functionality of the system, describing the behavior seen from an actor's point of view and specifying all possible scenarios for a given piece of functionality

Ex: Use case diagram for TicketDistributor



Requirements Elicitation – Use case example

Use case name	`UpdateTariff`
Participating actor	Initiated by `CentralComputerSystem`
Flow of events	1. The `CentralComputerSystem` activates the "UpdateTariff" function of the ticket distributors available on the network. 2. The ticket distributor disables the traveler interface and posts a sign indicating that the ticket distributor is under maintenance. 3. The ticket distributor waits for the new database from the `CentralComputerSystem`. 4. After waiting a minute for the ticket distributors to reach a waiting state, the `CentralComputerSystem` broadcasts the new database. 5. The ticket distributor system receives the new database of tariff. Upon complete, the ticket distributor sends an acknowledgement to the `CentralComputerSystem`. 6. After acknowledgment, the ticket distributor enables the traveler interface and can issue tickets at the new tariff. 7. The `CentralComputerSystem` checks if all ticket distributors have acknowledged the new database. If not, the CentralComputerSystem invokes the `CheckNonRespondingDistributors` use case.
Entry condition	• The ticket distributor is connected to a network reachable by the CentralComputerSystem.
Exit condition	• The ticket distributor can issue tickets under the new tariff, OR • The ticket distributor is disabled and displays a sign denoting that it is under maintenance.
Quality requirements	• The ticket distributor stays offline at most 2 minutes and is considered out-of-order otherwise.

Software engineering

Specification

DestinationBucketList *– is an app that allows people to create a personal private vacation destinations bucket list.*

A vacation destination is a place with a geolocation, title, image, description, stay dates;
There is a public list of destinations managed by an admin.
Bucket list is private, each user should be able to register and add public items or create new ones private.
User should be able to manage items, including removal.
Desktop app or web app;
Propose friendly display of items and ways to manage them;
Users can cancel their account (GDPR) or modify it.

Requirements

Actors

References:

Bernd Bruegge & Allen H. Dutoit – Object-Oriented Software Engineering Using UML, Patterns, and Java – Third Edition – Prentice Hall 2010